ISBN 978-82-326-5390-4 (printed ver.)
ISBN 978-82-326-6966-0 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

NTNU
Norwegian University of
Science and Technology

Lars Bungum

Doctoral thesis

Doctoral theses at NTNU, 2021:179

NTNU
Norwegian University of
Science and Technology
Thesis for the degree of
Philosophiae Doctor
Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

NTNU

Doctoral theses at NTNU, 2021:179

Lars Bungum

Unsupervised Clustering of Structured and Unstructured Text Collections

with applications in Domain Adaptation in Machine Translation

NTNU
Norwegian University of
Science and Technology

Lars Bungum

Unsupervised Clustering of Structured and Unstructured Text Collections

with applications in Domain Adaptation in Machine Translation

Thesis for the degree of Philosophiae Doctor

Trondheim, May 2021

Norwegian University of Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

© Lars Bungum

ISBN 978-82-326-5390-4 (printed ver.)
ISBN 978-82-326-6966-0 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

Doctoral theses at NTNU, 2021:179

PRINTED IN
NORWAY
NO - 1598

NORDIC SWAN ECOLABEL
Printed matter
2041 0731

Printed by Skipnes Kommunikasjon AS

LARS BUNGUM

UNSUPERVISED CLUSTERING OF STRUCTURED AND UNSTRUCTURED TEXT COLLECTIONS

Lars Bungum: *Unsupervised Clustering of Structured and Unstructured Text Collections,* with applications in Domain Adaptation in Machine Translation, © March, 2019

SUPERVISORS:
Björn Gambäck (main)
Jon Atle Gulla (cosupervisor)
Tor Anders Åfarli (cosupervisor)

LOCATION:
Trondheim

mina thanda wena

FOREWORD

This thesis is submitted in partial fulfillment of the requirements for the degree of Philosophiae Doctor (Ph.D.) at the Norwegian University of Science and Technology (NTNU). The work was done at the Department of Computer Science at the Faculty of Information Technology and Electrical Engineering.

Some notes on typesetting: Graphics not created by me are printed in a frame. Emphasis on words is typeset with *italics*, direct quotes in "quotation marks". Explicit references to software packages or frameworks such as `Python` are typeset with a typewriter font. The numbering of linguistic examples will start with 101 to avoid confusion with other enumerated items in the text.

Some much-used terms, central to the thesis, will be capitalized, e. g., Domain Adaptation, Statistical Machine Translation, and Self-Organizing Map. Such terms are spelled out on their first use in a paragraph but abbreviated thereafter. Other, more rarely used terms, such as **cross-entropy** are printed in bold.

Trondheim, March 2019.

[illegible]

ABSTRACT

This thesis explores the application of unsupervised clustering for domain adaptation of machine translation systems. As in many artificial intelligence areas, creating a system that generalizes to any domain is a hard problem in machine translation. Domain adaptation, in contrast, aims to *specialize* a generic (or otherwise intended) system for a particular domain and translate text within that domain better. In this thesis, experiments on using unsupervised learning as a first step in solving this problem are explored, posing the research questions a) how unstructured data could be used for domain adaptation and b) how a bespoke translation of an input document could be provided.

In the first part of the thesis, background theory is presented and related work reviewed. In the second, experimental part, preliminary experiments on building n-gram models and multiword expression detection are presented before experiments into clustering of structured and unstructured document collections are conducted. Finally, the parts are brought together in experiments on using these input factors for domain adaptation of machine translation systems, with end-to-end evaluation.

Some of the clusters identified in the clustering experiments on unstructured web collections were used as auxiliary language models in machine translation, in the experiments on domain adaptation. Self-Organizing Maps are used in the first phase of unsupervised clustering before a hierarchical agglomerative clustering algorithm is applied to extract tangible clusters from the map, with the number of clusters determined by the knee method.

By creating a mapping between the input document and one of the auxiliary language models, translation is aided by this language model. Using the language model perplexity on the input documents to select the auxiliary language model for domain adaptation links the clusters to the translation process.

Results show that the performance according to metrics such as BLEU, TER, and Meteor were on-par, and in some cases better than the results from leveraging all the available supplementary text as an auxiliary language model. The difference when using different auxiliary LM could be up to 1 BLEU points and 0.9 Meteor points.

SAMMENDRAG

Denne avhandlingen undersøker bruken av ikke-veiledet klynging for å tilpasse maskinoversettelsessystemer til spesifikke domener. I likhet med andre problemer innenfor kunstig intelligens, er det vanskelig å lage systemer som generaliserer til et hvilket som helst domene. Domenetilpasning, derimot, har det til formål å *spesialisere* et generisk system (eller et system med annen spesialtilpasning) til et annet, spesifikt domene, og forbedre oversettelse av tekst innenfor dette.

I avhandlingen blir eksperimenter på bruk av ikke-veiledet læring utført som et første skritt til å løse dette problemet med utgangspunkt i forskningsspørsmålene a) hvordan ustrukturerte datasamlinger kan brukes til domenetilpasning, og b) hvordan oversettelse av dokumenter kan skreddersys til det enkelte dokument.

I avhandlingens første del blir bakgrunnsteori og andre relevante arbeider presentert. I den andre, eksperimentelle, delen, blir eksperimenter med å bygge n-gram-modeller og flerordskonstruksjoner vist, samt eksperimenter med klynging av strukturerte og ustrukturerte dokumentsamlinger. Self-Organizing Maps ble brukt til ikke-veiledet læring, før en hierarkisk agglomerativ klyngingsalgoritme ble brukt til å lage konkrete klynger ut av dette. Antallet klynger ble avgjort av kne-metoden.

Avslutningsvis blir eksperimentene forenet ved at eksperimentene nevnt ovenfor ble brukt til domenetilpasning av maskinoversettelsessystemer, med ende-til-ende-evaluering. Noen av klyngene som ble identifisert i klyngingseksperimentene på ustrukturerte tekstsamlinger fra internett, ble senere brukt til å bygge hjelpespråksmodeller i maskinoversettelsesystemer, brukt i domenetilpasningseksperimenter. Ved å tilordne et dokument som skal oversettes til en av hjelpespråksmodellene, ble oversettelsen assistert av denne språkmodellen. Språkmodellenes perpleksitet på inndatadokumenter ble brukt til å velge hjelpespråksmodell.

Resultatene viste at ytelsen ifølge metrikker som BLEU, TER og Meteor var på linje med, og i noen tilfeller bedre, enn resultatene fra å utnytte all tilgjengelig tilleggstekst som tilleggsspråksmodell. Forskjellen ved bruk av en annen tilleggsspråksmodell enn den beste ifølge algoritmen var i enkelte tilfeller så mye som 1 BLEU- og 0,9 Meteor-poeng.

les langues sont des organismes	languages are living organisms
vivants dont la vie,	the life of which,
pour être d'ordre purement intellectuel	even though it is purely intellectual,
n'en est pas moins réelle et peut	is in no way less real,
se comparer à celle des organismes	and may be compared to that of the organisms
du règne végeétal ou du règne animal	in the vegetable or animal kingdom

— Arsène Darmesteter [1887]

ACKNOWLEDGMENTS

As my research now has culminated in this thesis, it is my great pleasure to recognize and thank the many that have helped me along the way. Stumbling in the footsteps of those who know enabled me to find my feet, for which I am most grateful. First and foremost, to my main supervisor, Björn Gambäck, with whom I have collaborated and discussed extensively with over the years, but also to my co-supervisors Jon Atle Gulla and Tor Anders Åfarli for their valuable input, in addition to the rest of the research group Data and Artificial Intelligence at the Department of Computer Science at NTNU.

Researchers with whom I have co-authored papers have also been a great source of inspiration and learning; thank you, Mikael Breivik, Jørgen Faret, Valerij Fredriksen, Björn Gambäck, Brage Jahren, Torvald Lekvam, Erwin Marsi, Hans Moen, Øyvind Selmer, and Gleb Sizov. I would also like to thank Jan Christian Meyer for giving me instructions on the architecture of the HPC grid Vilje. Courses are integral parts to this education, and I would like to thank instructors Keith Downing, Joakim Nivre, Torbjørn Svendsen, and Pinar Øzturk, as well as the many behind the course KULT8850 at Dragvoll. Gratitude is also extended to the administrative and technical staff at the department. I would also like to thank the many colleagues and fellow students with whom I have interacted in various capacities over the years, at NTNU and other places. Thanks are also extended to my M.Sc. supervisor Stephan Oepen and the Research Group for Language Technology at the University of Oslo for introducing me to research, and the crew at Copyleft Software for enabling me to understand that engaging with this side of computer science was my desired path.

Gratitude also to my family for their patience and support. In reverse order of appearance in this world; my niece Hedvig, my son Matthew, my sister Birgit (you too, Thomas), my mother Helga, and my father Hilmar. Also, friends that have been on my side, oh, when times are rough, thank you, Living, Oskar, and Eli Iveland. Thanks also to Norstat Trondheim, as well as the city itself, where I fit in. Semper fidelis!

I would also like to thank my lawyers spanning four(!) jurisdictions; Halvor, Bianca, David, Sharon, Nicholas, and Susan.

This research was supported in part by the project PRESEMT from the European Community's 7th Framework Program under contract nr 248307, with computational resources at NTNU provided by NOTUR, `http://www.sigma2.no`, and the bilateral project HaBiT (MSMT-28477/2014) between the Norwegian Research Council (NFR) and the Czech Ministry of Education, Youth and Sports (MŠMT). Academic work contracts offered to me by Arne Dag Fidjestøl, Björn Gambäck, Jon Atle Gulla, Mads Nygård, Øystein Nytrø, and Alf Inge Wang have also made this work possible.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

Part I

PRELIMINARIES

The first part introduces the thesis, defines the problem, and provides the relevant background for the experiments in Part II. Three background chapters follow the introduction, viz.: Self-Organizing Maps and Cluster Analysis, Machine Translation, and Domain Adaptation and Machine Translation.

1 INTRODUCTION

Data sparsity is a problem in data-driven applications like Machine Translation (MT). The amount of online[1] data of all sorts increases rapidly year-on-year, but it is mostly unstructured. Unsupervised learning can be used to organize and understand such data, enabling its use, e. g., for making predictions or analyses. Specifically, unsupervised clustering algorithms can establish a structure in these document collections such that they can be leveraged, e. g., for MT.

As more people around the world gain access to the Internet and its vast resources, its share of English decreases. Consequently, the need for translation of documents from and into languages one does not master sufficiently well increases, e. g., for communication and knowledge extraction purposes. Coinciding with this development, advances in Machine Translation have made the use of foreign language resources easier, albeit often only by getting the gist of the document. This kind of low-quality MT, sufficient to acquire an understanding but hardly to publish, is popularly referred to as **gisting**.

Restricting the translation task to a **domain**, such as text about a certain topic, written in a certain style, or for a certain audience makes the task easier. However, such domains can vary greatly and are hard to separate. Moreover, systems created for one domain often perform poorly on text from another, e. g., Lu et al. [2007], Tiedemann [2010], and Pecina et al. [2015]. Domain Adaptation means to adapt a Machine Translation system such that it better translates text belonging to other domains than it was created for. Especially when the available data in the desired text domain is scarce, leveraging a larger body of general text is valuable.

This thesis will present work on using an unsupervised clustering method based on Artificial Neural Networks — the Self-Organizing Map (SOM) [Kohonen, 1982] — on monolingual data collected from the web, applied to this problem. Leveraging monolingual data is important for Domain Adaptation because it is as easy or easier to obtain, and often all that is available. In the remainder of this chapter, the thesis will be further introduced with a statement of motivation, research questions, and an outline of its structure.

1 Henceforth, *online* is understood as something published on the Internet, whereas *on-line* means something done while a system is running.

1.1 MOTIVATION

This section will motivate the experimental work by first addressing the complexity of the problem, then discussing how generic Machine Translation could be achieved through Domain Adaptation, and last addressing the relation between large data collections and DA. Together, these elements explain the motivation behind the research questions and the methods used for the experiments.

1.1.1 *AI-Completeness of Machine Translation*

Yampolskiy [2015] defined a problem C as AI-complete if it exhibits two properties; (i) it is in the set of problems solvable by the union of human minds, and (ii) any (other) such problem can be converted to C by some polynomial time algorithm. The notion of AI-completeness must be seen in light of the P vs. NP problem [Cook, 1971], and the impact this classification of problems had on Computer Science. P and NP reference *Polynomial* and *Non-deterministic Polynomial* time, respectively, i. e., categories of problems that can be solved or verified in polynomial time.

Yampolskiy argued that lacking a definition of the problem space was a shortcoming of the Artificial Intelligence (AI) field and proposed a theory of AI-completeness analogous to the classification of problems in computer science outlined in the previous paragraph. This formalism included a mechanism with which to *prove* the AI-completeness of problems by **reduction**. A problem can be reduced to another if all the information encoded in queries to the original problem can be put as queries to the other and if all solutions to that problem could be an answer for the original. After showing that a Turing Test (TT) is AI-complete, this problem was reduced to Question Answering (QA), as all queries to the TT could be put to a QA system, whose replies could be provided as answers during the administration of a TT. Thus, QA is also AI-complete according to the theory.

By categorizing AI problems as either AI-easy, AI-hard, or AI-complete, problems could then be *proven* to be AI-complete, as opposed to only *believed* to be AI-complete. Without reference to a similar formalism, Shapiro [1992] listed the following problems as AI-complete: (i) Natural Language (sic) (ii) Problem Solving and Search (iii) Knowledge Representation and Reasoning (iv) Learning (v) Vision, and (vi) Robotics. Moreover, Yampolskiy encouraged the research community to give formal proofs of the categorization of other problems and gave Machine Translation as an example.

Machine Translation must be in the family of problems solvable by the union of human minds as translation, by nature, depends on the human mind. Furthermore, it requires a full understanding of the reasoning behind the statement to be translated. If, as conjectured by Yampolskiy, Natural Language Understanding (NLU) is AI-complete, it can be reduced to MT by stopping at the level of representation where the input query is understood before its representation in a different language is generated. (If needed, the Machine Translation system could use the representation of the NLU system as a pivot language.) Thus, MT would also be AI-complete according to the above definition.

More loosely, solving AI-complete problems is equivalent to solving the strong AI problem, i. e., to make computers as intelligent as human beings. Perfect translation requires the ability to understand what the author argues, which requires unlimited world knowledge and the ability to reason about it.

1.1.2 *Machine Translation Through Domain Adaptation*

Machine Translation was an early problem in Computer Science, with research going back to the 1940s. Still, it is a problem far from *solved* in the broad sense that no human interaction would be needed to translate text with human-equivalent quality (Chapter 3 will define the term more rigorously). Since MT is both AI-complete and can involve NP-complete operations, it is not surprising that there is no panacea.

With some exceptions, such as Machine Translation systems for very limited purposes, MT output is not up to a publishable standard today. However, MT becomes an easier task as more restrictions are put on the specification of the problem. Definitions of MT often come as a set of constraints, which can be relaxed to arrive at a solvable problem. One of these constraints is the purpose of translation, such as processing text from a specific domain. Lehrberger [1982] argued that the solution to MT lies just in sublanguages (understood as equivalent to domains):

> "The solution in the case of automatic translation seems to lie in restricting one's attention to sublanguages." [Lehrberger, 1982].

in a period when the domain-specific, limited-vocabulary `TAUM-METEO` [Thouin, 1982] system attained considerable success.

Thus, if Domain Adaptation is possible for limited-domain contexts, general purpose Machine Translation can be recast as a multitude of limited-domain MT problems by treating every input document as if it were domain-specific. While an intriguing and highly motivating idea, the problem of *identifying* domains remains.

1.1.3 *Domain Adaptation in Machine Translation*

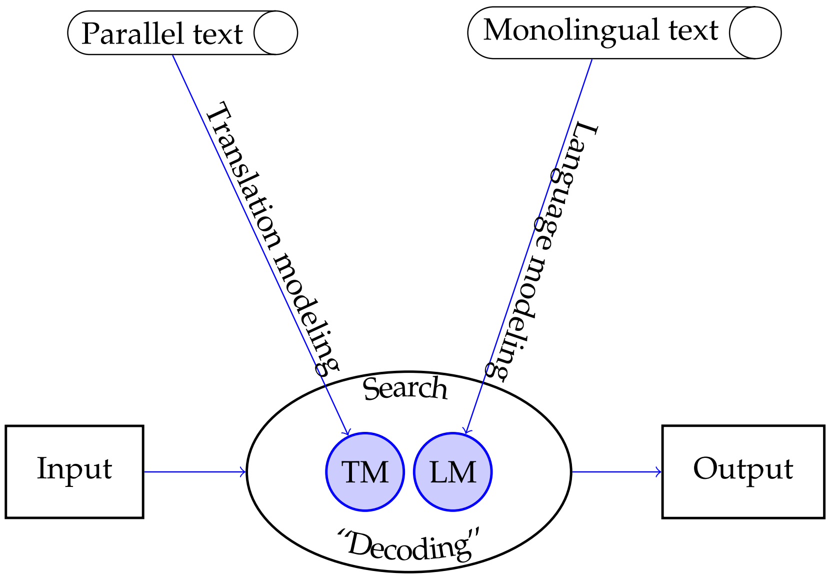


Figure 1: Overview of a data-driven Machine Translation system.

Figure 1 shows an overview of a data-driven Machine Translation system. A Translation Model (TM) is created by extracting bilingual relations from parallel text, and a Language Model (LM) is created by extracting relations from monolingual text. The TM models possible translations of source language terms into a target language, whereas the LM models the target language. These models are jointly searched for the best target language term given a source language input to provide a translation. For historical reasons called **decoding**, this search process produces the translation.

In a situation where two separate domain-specific Machine Translation systems are used for translation of documents from the other domain, the translation quality is likely to be poor. Figure 2 illustrates how two domain-specific test documents are cross-applied between two different, domain-specific systems. Results on translating **in-domain** data, i. e., documents from the same domain as the system was trained on, are relatively good, which is illustrated with green output boxes, while results on **out-domain** data are relatively poor, illustrated with red output boxes with sawed edges. The green core indicates that parts of the documents are translated correctly.

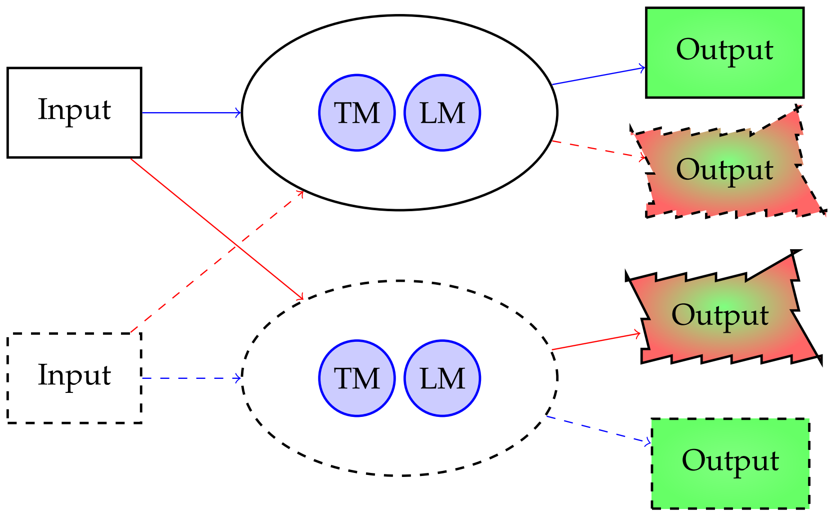


Figure 2: Two separate Machine Translation systems, where test input is cross-applied. System 1 with solid lines, and system 2 with sawed lines. When documents are applied to a same-domain system, lines are blue and in the opposite case red.

Figure 3 integrates the previous two figures and illustrates Domain Adaptation, i. e., that an existing Machine Translation system is *adapted* to another domain. Here, the top system from Figure 2 with solid lines is adapted to the domain of the bottom system with sawed lines. The magenta arrows indicate possible types of adaptation. The higher share of green in the output from running the adapted system on the in-domain document illustrates that it is more successful after DA.

A simple form of Domain Adaptation is to concatenate all domain-specific training data to the data already available and train a new model can on this concatenation. Alternatively, adaptation work can be done directly on the translation modeling, language modeling, or the decoding process. Each of the sub-processes in data-driven Machine Translation consists of many sub-tasks and parameters, and DA can be done on any of these. Successful adaptation should result in a model that provides better translations than either the (often larger) out-domain model or the (smaller) in-domain model. This thesis will address ways to achieve this, primarily by using Language Models created from clustering web data with Self-Organizing Maps.

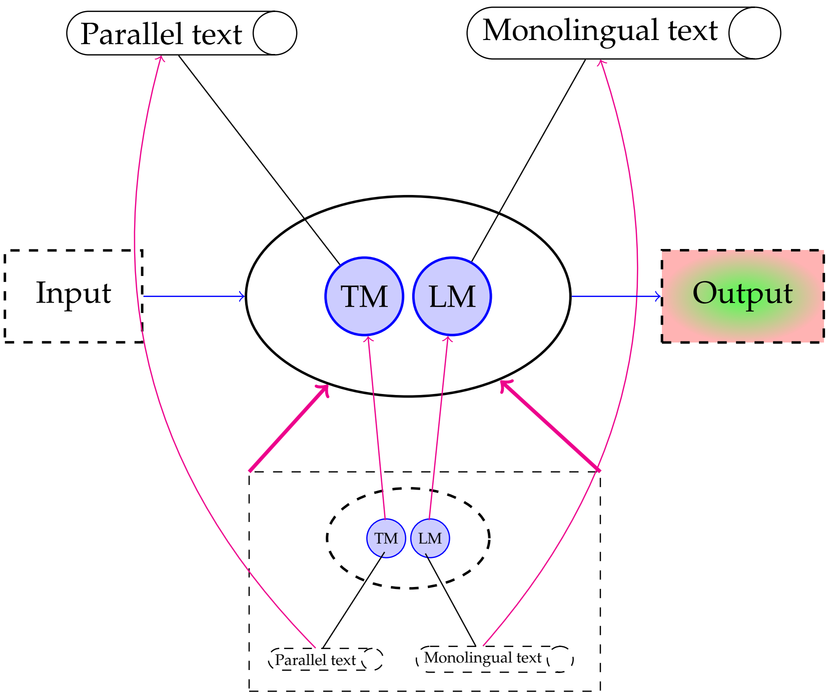


Figure 3: Strategies for Domain Adaptation of a Machine Translation system.

1.1.4 *Self-Organization and Large Data Collections*

With an abundance of web data in existence, an enormous body of text is available for research purposes, e. g., for use in Machine Translation or Computational Semantics. Advances in infrastructure have made crawling, storage, and access easier. A service like Common Crawl[2] makes Terabytes of crawled data available to everyone on a monthly basis. Similarly, large-scale computing resources are now easily available, making it possible to utilize this vast resource. These enormous data collections and their rapid updates create a potential, not only for finding large amounts of domain-specific segments for Natural Language Processing (NLP) applications but also for tracking changes in natural languages.

Self-Organizing Maps represent complex data with (very) high dimensionality in a lower-dimensional (mostly 2D) space [Floreano and Mattiussi, 2008]. Reduced dimensions mean that structure in the data can be visualized in this lower-dimensional space. Because of successes in data mining, non-dependence on a fixed set of clusters or partitions, and biological inspiration, unsupervised algorithms such

2 `commoncrawl.org` (Last visited: March 27, 2019.)

as SOMs are well-suited for finding domain-specific segments within large data collections with no provided structure. However, the question of how to utilize the structures found by unsupervised methods for Domain Adaptation remains open.

A Self-Organizing Map [Kohonen, 1982] is a topological mapping of objects of arbitrary complexity provided that a distance measure between objects exists. The maps provide cartographical insight into the relations between these objects by placing similar objects in the same area. The biologically inspired algorithm has its theoretical origins within the realm of Artificial Neural Networks, and it also has well-defined mathematical properties as a practical tool.

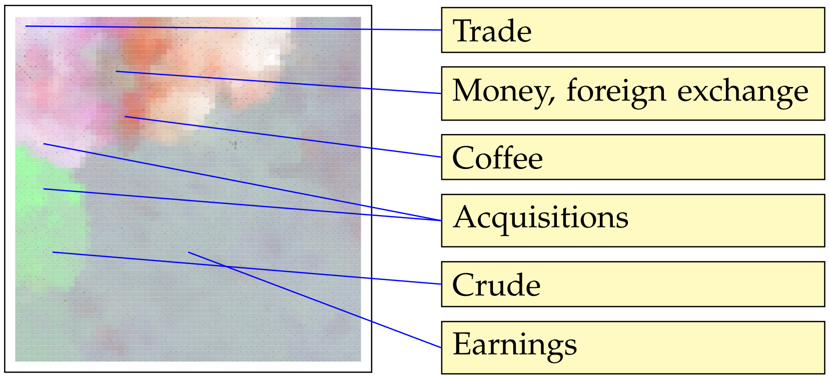


Figure 4: A Self-Organizing Map of the Reuters Corpus. Color-shaded areas contain documents with largely the same labels.

Figure 4 shows an example of a Self-Organizing Map of the Reuters Corpus [Lewis et al., 2004]. This map was trained on the text corpus (consisting of newspaper articles) by providing documents in random order for a given number of **Epochs** or iterations. (Chapter 7 presents the experiments that created this map.) The experiments on this small document collection effectively demonstrated that similar documents were placed in the same areas of the map since the document labels can verify their thematic similarity. The SOM algorithm can also be used for *unstructured* data, i. e., unlabeled documents, such as collections of text mined from the web. The proximity between objects in the low-dimensional space that visualizes similarity in higher dimensions is an important result of the algorithm.

General-purpose Machine Translation of high-quality has been notoriously difficult to achieve, but there is a solace to find in the methodological advances since the first applications of the 1940s. However, examples of special purpose systems for producing high-quality output do exist, e. g., Thouin [1982] and Palmer et al. [1998].

Using an unsupervised algorithm such as the Self-Organizing Map to leverage both the growth of the web as a corpus and the availability of increased computing power to identify pseudo-domain-specific corpora useful for translating arbitrary documents, is the primary motivation behind this thesis.

1.2 RESEARCH QUESTIONS

Machine Translation divides into different approaches, again divisible into algorithmic steps, which all contribute to the output translation. During a time-span of more than 60 years, significant advances have been achieved in MT at different increments. For the last three decades, data-driven approaches have achieved notable gains by methodological advances including the ability to process large amounts of training data. Recently, MT with Artificial Neural Networks has improved the state-of-the-art further. However, MT is not a solved problem, and several open questions remain [Lopez and Post, 2013] — Domain Adaptation being one. Presently, acquiring large collections of parallel text for every language pair is not possible, and certainly not for every domain. The following research questions specify the problem:

1. How can unstructured web data be used to facilitate the Domain Adaptation of translation into many different domains simultaneously?
2. How can a Machine Translation system be designed to translate an input document, adapted to the characteristics of that particular document?

1.2.1 *Facilitating Domain Adaptation with Unstructured Data*

The first research question pertains both to using the web as a corpus for Machine Translation and its application to the Domain Adaptation problem. Some *structure* in the data must be established to make use of previously unstructured data to translate between different domains within a language pair. With a structure, pseudo-domain-specific corpora can be identified. Using a fixed set of domains to select relevant portions of the data, and using unsupervised methods to identify inherent clusters, are possible ways to achieve this. If the set of target domains is presumed unlimited or unknown, methods that do not require a fixed set of domains are needed.

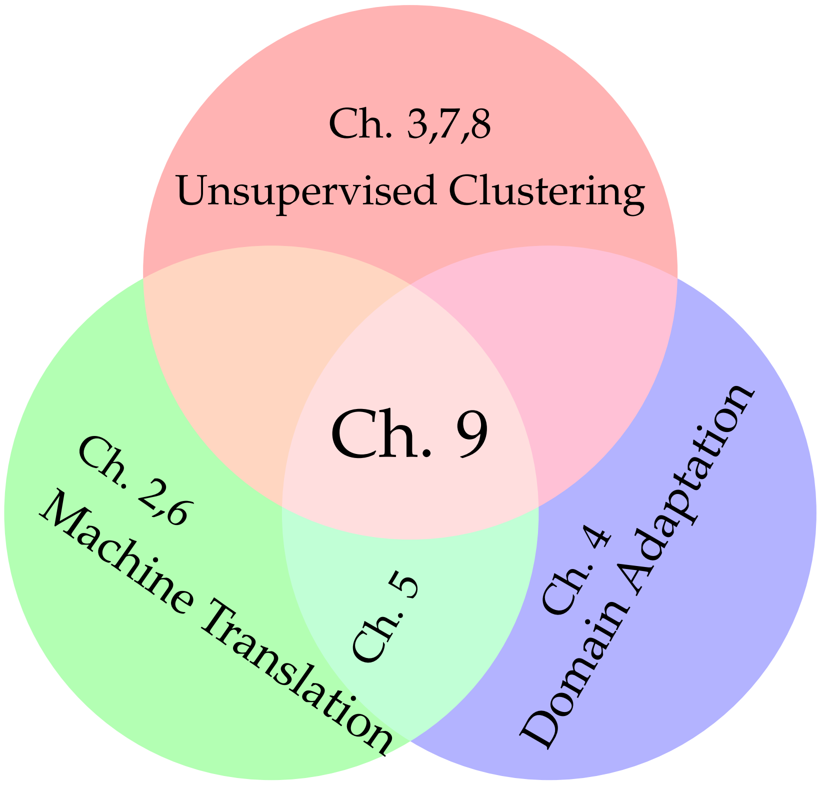


Figure 5: Venn diagram of thematic intersections. Chapter numbers refer to corresponding experiments.

1.2.2 *Adapting Translation to Input Documents*

The second research question relates to how Machine Translation systems can provide an adapted translation of each input document. A mapping between an input and the configuration of the system is required to provide a bespoke translation. The configuration of the system could then adapt to the properties of the input document. Concretely, a mapping could invoke the most relevant section of the available training data, when leveraging unstructured data collections as Language Models. However, LMs are in the Target Language while input documents are in the Source Language. Thus, this premise raises the question of how to map a Source Language document to a Target Language resource.

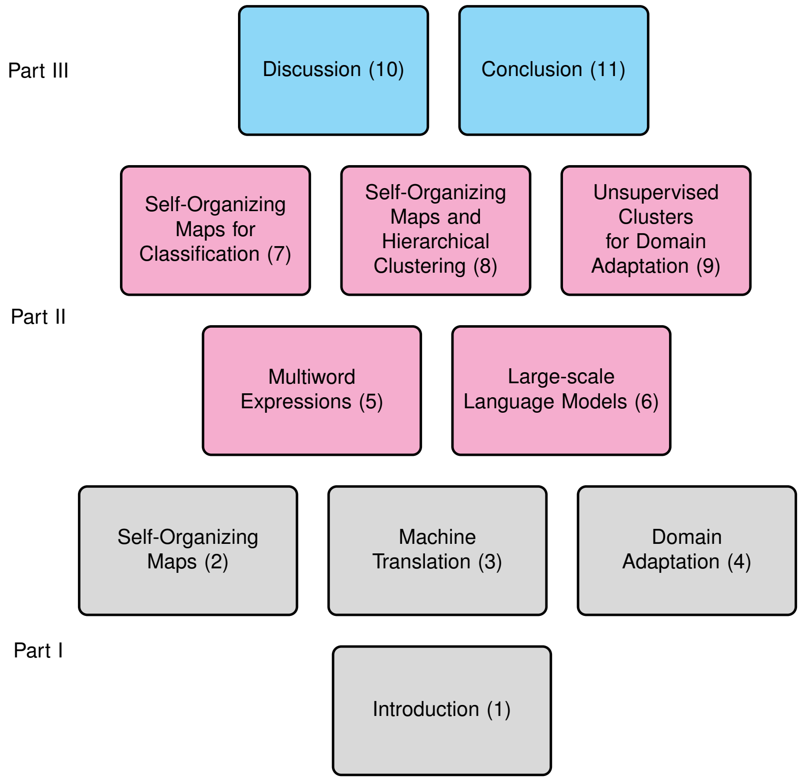

Part III
Discussion (10)
Conclusion (11)
Self-Organizing Maps for Classification (7)
Self-Organizing Maps and Hierarchical Clustering (8)
Unsupervised Clusters for Domain Adaptation (9)
Part II
Multiword Expressions (5)
Large-scale Language Models (6)
Self-Organizing Maps (2)
Machine Translation (3)
Domain Adaptation (4)
Part I
Introduction (1)

Figure 6: Chapter structure.

1.3 THESIS STRUCTURE

The core of the thesis is the intersection between three research areas; Machine Translation, unsupervised clustering, and Domain Adaptation, jointly comprising a novel approach to the problem of DA in MT. Figure 5 illustrates the thematic relationship between these areas, annotated with chapter numbers. The included chapters are preceded by an introductory chapter and succeeded by an analysis part.

Figure 6 presents an overview of the chapters in a 3-5-2 formation. Part I covers relevant background information in Chapters 2-4 and Part II presents experiments in Chapters 5-9. First, Chapter 5 presents experiments on identifying Multiword Expressions (MWEs) with dictionaries, and how MWEs can determine the proper translation of a word given its context. Next, Chapter 6 discusses work on constructing large-scale Language Models using grid computing.

The remainder of the experimental part covers experiments on Self-Organizing Maps and their application to Domain Adaptation in Machine Translation. First, Chapter 7 presents work on using the SOM algorithm on a structured collection of text. Comparing the performance of the SOM implementation to the state-of-the-art on classification of this corpus confirms the `MPI` implementation of the SOM algorithm. Next, Chapter 8 demonstrates how the SOM can combine with hierarchical clustering to produce concrete clusters, with exact demarcations. Finally, Chapter 9 describes how these clusters were for DA of MT systems by providing auxiliary training material (monolingual text).

Part III sums up the experiments with a discussion and presents ideas for future research. Additionally, Part IV contains appendices with more details on the experiments.

1.4 CONTRIBUTIONS

In addition to a number of scientific publications on core and related topics (listed in separate bibliographies in Part IV), the main contributions of this thesis are:

- A demonstration of a novel approach to Domain Adaptation integrated into a full Statistical Machine Translation pipeline and evaluated on end-to-end experiments.
- A dictionary mining method of resolving the translation of Multiword Expressions for use in Word Translation Disambiguation.
- Experiments on the Hierarchical Clustering of Self-Organizing Maps on unstructured document collections mined from the web with visualizations of results.
- Experiments on clustering a structured document collection using Self-Organizing Maps with experiments on large topologies.
- An implementation of the Self-Organizing Map algorithm in `MPI` and Python, and code to visualize the results and facilitate clustering of the SOM matrix with the `sklearn` package.
- A method for parallel building of `IRSTLM` Language Models with an `OpenPBS` scheduler.

1.4.1 *Scientific Publications*

During the work with this thesis, several intermediate results were published. Some of these papers include work related to the experiments in Part II or related to the analysis part (listed on Page 245), and some being peripheral to the thesis (listed on Page 247). All papers had some impact on Part I (introduction and background) and

Chapter	Title	Author(s)	Year
3	Word translation disambiguation without parallel texts.	EM, AL, LB, BG	2011
4	A survey of domain adaptation in machine translation: Towards a refinement of domain space.	LB, BG	2011
5	Disambiguating word translations with target language models.	AL, EM, LB, BG	2012
5	Improving word translation disambiguation by capturing multiword expressions with dictionaries.	LB, BG, AL, EM	2013
6	Efficient n-gram language modeling for billion word web-corpora.	LB, BG	2012
7	Self-organizing maps for classification of a multi-labeled corpus.	LB, BG	2015
8	Extracting and selecting relevant corpora for domain adaptation in machine translation.	LB	2014
9	Multi-domain adapted machine translation using unsupervised text clustering.	LB, BG	2015
1,2,10	Linguistic domains and adaptable companionable agents.	BG, LB	2016

Table 1: Paper relevance to specific chapters. LB=Lars Bungum, BG=Björn Gambäck, AL=André Lynum, and EM=Erwin Marsi.

Part III (analysis). Table 1 lists how selected papers relate to different chapters.

1.4.2 *Mining Multiword Expressions with Dictionaries*

Chapter 5 demonstrates a method for identifying Multiword Expressions using dictionaries. The method finds MWEs listed as entries in bi-directional dictionaries. The experiments were evaluated on a cross-lingual Word Translation Disambiguation task.

If a Multiword Expression was found in the context of the text examples, this was often the correct sense. However, experimental results showed that the method had high precision but low-recall. Thus, the method is well-suited for combination with methods with the inverse property. Moreover, it was included in a recent Springer book on the state-of-the-art in MWE processing [Ramisch, 2015].

1.4.3 *Parallel Building of IRSTLM Language Models on OpenPBS Schedulers*

Chapter 6 presents work on building large Language Models using the `IRSTLM` language modeling framework scripts modified to the `OpenPBS` scheduler. The `OpenPBS` scripts used the same methods as the scripts for the original scheduler for dividing LM creation into sub-tasks, which are distributed to an HPC grid and subsequently merged.

1.4.4 *Clustering a Structured Document Collection with SOMs*

Chapter 7 shows experiments on clustering the Reuters Corpus using Self-Organizing Maps. The experimental method largely replicates other SOM-based classification methods but also provides visualizations, which explore the multi-label characteristics of the task, i. e., how documents with multiple labels are similar. This similarity is relevant to the discussion on whether multi-label classification should be re-cast as a cascade of single-label classification tasks.

1.4.5 *Hierarchical Clustering on SOM-Structured Web Data*

Chapter 8 presents a method to cluster a trained Self-Organizing Map into concrete clusters. These resulting clusters consist of documents from areas on the map, which are concatenated into text corpora. The algorithm is aimed at transforming the separate areas in the visualizations of the SOMs into delineated clusters whose member documents can be output. The code relies on the `scipy` package for clustering methods, distance metrics, and intrinsic evaluation criteria.

1.4.6 *MPI Implementation of the SOM Algorithm*

An implementation of the Self-Organizing Map algorithm using a Message Passing Interface (MPI) was developed to conduct the experiments presented in Part II. While there are other SOM implementations available, e. g., `MATLAB`'s toolbox, an open source implementation of the algorithm in `MPI` (or similar) is not available to the best of found knowledge.

The implementation used for the experiments in Part II integrates the distributed calculation of vector comparisons and updates with software packages such as `scikit-learn` for distance metrics and tf-idf conversion, `Numpy` for matrix operations and `Matplotlib` for visualization. The use of these packages makes it easy to experiment with alternative distance metrics and vector representations. The code is released under the GNU GPL license and is available at `https://github.com/Pinkertoncito/som-mpi-py`.

1.4.7 *Integration of Unsupervised Clusters into an SMT Pipeline*

The final chapter in the experimental part provides an evaluation of the document collections resulting from the unsupervised clustering process in a specific setting, Domain Adaptation of Machine Translation systems. Language Models built on the document clusters are integrated into a Statistical Machine Translation pipeline as features in the log-linear framework of the `Moses` SMT system [Hoang et al., 2007].

A perplexity ranking was used to map input documents to the most relevant **auxiliary** Language Model built from supplementary data. The performance of the top-ranked LM was benchmarked against all other auxiliary LMs, and against one built on all of the available supplementary data. Some experiments showed that the top-ranked auxiliary LMs gave better results than using all the available supplementary data.

2 SELF-ORGANIZING MAPS AND CLUSTER ANALYSIS

The first background chapter will discuss the Self-Organizing Map algorithm in particular and cluster analysis in general. Section 2.1 describes the SOM algorithm, and Section 2.2 clustering and cluster analysis. Section 2.3 presents some categorizations of cluster analysis and Section 2.4 a selection of clustering algorithms. Section 2.5 addresses how SOMs apply to clustering, and finally, Section 2.6 ends the chapter with a discussion of cluster evaluation methods and metrics.

Self-Organizing Maps [Kohonen, 1982] are created by an *unsupervised* learning algorithm with its roots in Artificial Neural Networks (ANNs), which is mostly referred to as the Self-Organizing Map algorithm but also known as Kohonen Neural Networks, e. g., Lo et al. [1991]. The SOM algorithm has an interpretation as a special form of training ANNs, i. e., training their **neurons** and **synaptic weights**, with mathematically clear specifications how **nodes** and **vectors** are updated (the preferred vocabulary henceforth).

Unsupervised clustering, the algorithmic recognition of patterns and natural groups among a class of entities, has been applied to many areas of the natural sciences, e. g., biology, chemistry, and geophysics. These methods have also been applied to Language Technology going back at least to the early nineties, exemplified by Valbret et al. [1992] who clustered the acoustical space of each speaker into non-overlapping segments for use in speech synthesis. Furthermore, Self-Organizing Maps have been used in many Automatic Speech Recognition (ASR) applications, such as clustering a text corpus with SOMs to improve recognition by using multiple Language Models [Podder, 2004]. Shum et al. [2013] used unsupervised clustering for speaker detection and also for Domain Adaptation of speaker detection systems [Shum et al., 2014]. Unsupervised clustering has been applied to Machine Translation, e. g., by using mixture modeling of translation models [Sennrich, 2012b] or factored translation using word clusters [Rishøj and Søgaard, 2011].

By grouping semantically related documents together, Self-Organizing Maps achieve similar ends as Latent Semantic Indexing (LSI) [Deerwester et al., 1990, Hofmann, 1999] and Latent Dirichlet Allocation (LDA) [Blei et al., 2003]. LSI is based on Singular Value Decomposition (SVD), extracting the most salient terms using the left-singular vectors of the matrix decomposition. LDA draws multinomials from a Dirichlet distribution to organize a collection of documents based on a distribution of topics in documents and a distribution of words in topics. The SOM algorithm can also be used for dimensionality reduction, e. g., by inputting distances between clusters [Tsimboukakis and Tambouratzis, 2007] or cluster membership to other Machine Learning methods.

Vectorization refers to the representation of some object (such as a document) as a vector, such as by the occurrences of specific terms in each document, or by their tf-idf values. Widely used in Information Retrieval, tf-idf — the product of the term frequency and the inverse document frequency — offsets the number of occurrences of a term in a document with the number of times it occurs in the document collections with the aim of representing documents with more relevant terms. Wu et al. [2008a] and Roelleke and Wang [2008] formed theoretical bases to interpret the measure (as a relevance measure and probabilistically, respectively).

2.1 THE SELF-ORGANIZING MAP ALGORITHM

The Self-Organizing Map algorithm organizes nodes with a given topology, often two-dimensional with a square or hexagonal shape. Nodes consist of vectors of some higher dimensionality, directly comparable to the same-arity vectors that represent training samples. Node vectors are also known as **prototypes** in the SOM literature. The nodes' properties are gradually changed for a predefined number of **Epochs** (cycles), making the nodes more similar to the training samples as they are processed. Changing one data point (node) will also affect its neighboring nodes, inspired by biological systems in which neurons with similar functions organize in the same areas.

Figure 7 illustrates self-organization, where 16x16 nodes are represented as vectors with three dimensions, each with values between 0 and 1, making them directly plottable as red, green, and blue (RGB) colors. Training samples consist of a set of fixed colors (a red, green, and blue color) represented as vectors containing their respective RGB values. After training, the grid has *self-organized* into areas of similar colors based on the three training samples.

Specifically, the algorithm starts with an array of nodes and is repeated for a fixed number of Epochs. Sample vectors are successively, but in random order, compared to node vectors. The closest node vector to the training sample according to a distance metric is the winner (henceforth referred to as the Best-Matching Unit (BMU)). The vector values (interpreted as weights) of the BMU, as well as weights of the neighboring nodes, are updated to be more similar to the sample at a

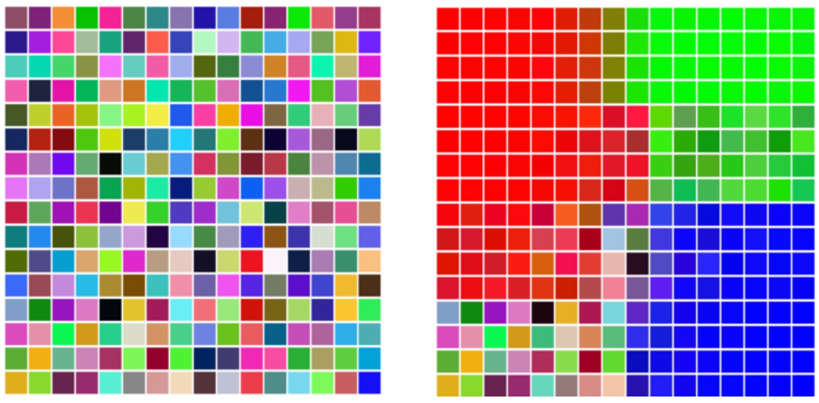

(a) First Epoch. (b) Last Epoch.

Figure 7: Self-Organizing Map of RGB color codes.

learning rate specified by a coefficient. Thus, each training sample is compared to every node, using a **distance metric** d, and drawing the BMU and its neighborhood towards the training sample.

Formally, a Self-Organizing Map consists of N nodes and S samples with each node vector $n_j \in \mathcal{R}^p$ and each sample vector $s_i \in \mathcal{R}^p$ representing nodes and samples, respectively, with vector dimension p. Training iterates for a number of Epochs, E.

For each $s_i \in S$, the Best-Matching Unit n_w at Epoch e is determined by:

$$n_w(t) = \operatorname*{argmin}_j d(n_j, x_i) \tag{1}$$

Next, the BMU n_w and neighbors update with the function:

$$n_k(t+1) \leftarrow n_k(t) + \lambda(t) h_{kw}(t)(s_i - n_k) \tag{2}$$

The learning rate λ is a function that can be defined over time (i. e., Epoch) and distance from the BMU in the low-dimensional space. The neighborhood function $h_{kw}(t)$ determines whether the current node n_k is in the neighborhood of the BMU n_w and optionally the proximity to the BMU. Neighborhood can be expressed as topological adjacency (i. e., bordering nodes) or as a circle around the BMU whose radius decays with time.

The resulting map has grouped (self-organized) similar high-dimensional vectors together in the low-dimensional topology. Hence, a feat of cartography (a map) in the landscape of input samples has been achieved, not only because similar samples attach to the same areas on the map, but also because the distances between nodes can be visualized with color shadings or contour plots. Algorithm 1 presents the Self-Organizing Map algorithm in pseudo-code.

Algorithm 1 SOM algorithm for E Epochs, a set of N nodes and a set of S training samples.

```
1:  procedure CREATE-SOM
2:      randomly initialize $|N|$ nodes
3:      for each Epoch $e \in E$ do
4:          for each sample $s \in S$ in pseudo-random order do
5:              for each node $n \in N$ do
6:                  distance[n] = distance(n,s)
7:              end for
8:              BMU = min distance[$n \in N$]
9:              update BMU + neighborhood towards $s$
10:         end for
11:         reduce neighborhood and/or learning rate
12:     end for
13: end procedure
```

The ability of Vector Space Models (VSMs) to represent real-world phenomena has been discussed by, e. g., Dubin [2004] and Clark [2015]. Machine Learning methods have been able to capture relations between the attributes of vectors, often referred to as latent. A wider discussion of the merits of vector representation as such, however, falls beyond the scope of this thesis.

2.1.1 *Computational Complexity of the Self-Organizing Map Algorithm*

Algorithm 1 shows that three loops are necessary for the computation of the Self-Organizing Map. Furthermore, the number of Floating Point Operations (FLOPs) will depend on the dimensionality of the vectors representing each sample and node as the distance is computed, the **numerical rank** of the matrix representing the SOM. An increase in either $|E|$, $|S|$ or $|N|$ will increase the number of distances to be computed, the product of the terms, $O(ExSxN)$. If two (or three) terms are related, an increase in one implies an increase of another, the number of comparisons will be quadratic (or cubic) to this term, otherwise linear. Practically, however, also linear growth of this magnitude matters.

Kaski [1997] argued that the complexity of the algorithm is quadratic to the number of nodes (map units) *N* because the number of Epochs should be a multiple thereof. Drigas and Vrettaros [2008] also claimed that the algorithm is quadratic to the number of map units.

Alternatively, Roussinov and Chen [1998] formulated complexity with regards to the growth of documents in a collection. The authors argued that the Self-Organizing Map algorithm is $O(|D|^2)$ where $|D|$

represents the size of the document collection. This argument is based on a formulation of time complexity, $O(VE)$, where *V* represents the length of the document vector and *E* the number of Epochs. Finally, Roussinov and Chen argued that both terms can be replaced by $|D|$, as the number of unique terms is proportional to the document collection.

2.1.2 *Alternative Formulations of the Self-Organizing Map Algorithm*

Owing to practical necessity as configurations grow, some alternative formulations have emerged to make processing computationally cheaper. Such include using a two-step approach (i. e., creating a Self-Organizing Map from the output of another SOM [Kohonen et al., 1996]), or formulating a **batch-SOM** algorithm. These methods contrast the *on-line* version by updating node weights in batches of samples, as opposed to individually for every sample.

After initialization of the node vectors, the method implements Vector Quantization (VQ) and smoothing of numerical values over a two-dimensional grid [Kohonen et al., 2000]. The **Voronoi set** V_i defines the set of samples that have node n_i as Best-Matching Unit. In a process resembling quantization, the average of the Voronoi sets for all nodes is calculated as:

$$\forall i, \qquad \bar{x}_i = \frac{\sum_{s \in V_i} s}{m_i} \tag{3}$$

where $m_i = |s \in V_i|$, i. e., the number of samples falling into V_i, and thereby representing the Voronoi set by its average. Subsequently, a node n_i is updated with:

$$n_i = \frac{\sum_j m_j h_{ji} \bar{x}_j}{\sum_j m_j h_{ji}} \tag{4}$$

where j iterates over the nodes in the neighborhood of the BMU and h is a neighborhood function determining the proximity between nodes. Thus, each node in the neighborhood of the updating node n_i is represented by the average of the samples with this BMU. The method iterates using the same input samples and recalculates $\bar{x}_i$. Kohonen and Honkela [2007] claimed that the **batch map** essentially gives the same results as the on-line SOM algorithm despite being an order of magnitude faster.

Roussinov and Chen [1998] proposed an alteration of the Self-Organizing Map algorithm, leveraging the sparsity in vector representations.

Fort et al. [2002] compared the approaches and noted some problems with the batch formulation, such as initialization sensitivity, while it had advantages regarding speed and efficiency. Furthermore, several implementations of the parallel batch-SOM algorithm exist, e. g., Lawrence et al. [1999] and Patel et al. [2015].

2.1.3 *Relation to Artificial Neural Networks*

Artificial Neural Networks (ANNs) constitute a branch of computational neuroscience where computational models mimic the behavior of biological systems to solve problems. These ambitions are traceable back to the early years of Artificial Intelligence [McCulloch and Pitts, 1943]. Both the complexity of Neuroscience and the rapid advances in the field [Marcus and Freeman, 2014] make it near-impossible for computational models to regard all aspects of this world of knowledge.

The term Artificial Neural Network makes a distinction between what are real neural networks wired in the biological brain, and those that are *artificial*, existing in computer models. Henceforth, this term is preferred, although the word *artificial* is often omitted in the literature with the same meaning. A rigorous treatment of the steadily increasing ecosystem of ANNs is beyond the scope of this thesis. (See Goldberg [2015] for a good primer on the use of "Neural Network Models" in Natural Language Processing.)

Artificial Neural Networks consist of neurons (nodes), organized in layers, connected by synaptic weights (edges). A function in each neuron sums the dot products of the neurons connected to it and their corresponding weights, and an activation function does a transformation of this value to determine its output value (whether the neuron fires). Connections can either be one-directional in the network (feed-forward) or be between neurons in the same layer and back (recurrent).

There is always an input- and an output layer. **Hidden layers** between input and output represent additional connections between input and output. **Deep learning** mostly refers to the utilization of numerous such hidden layers [Bengio, 2009], an approach also used for NLP tasks [Collobert and Weston, 2008, Collobert et al., 2011]. At the other end of the scale, the single-layer *perceptron* [Rosenblatt, 1962] is the simplest Artificial Neural Network.

For practical use cases like **classification** (the attribution of a class to an instance), output layers are given an interpretation. Connections are given weights, finally determining the values in the output layer, as the input values are fed forward. Setting these weights by trial-and-error is only feasible for very small architectures, and much work has been done on automatic training of these weights. Using variations of **Hebbian learning** [Morris, 1999] (strengthening weights if both

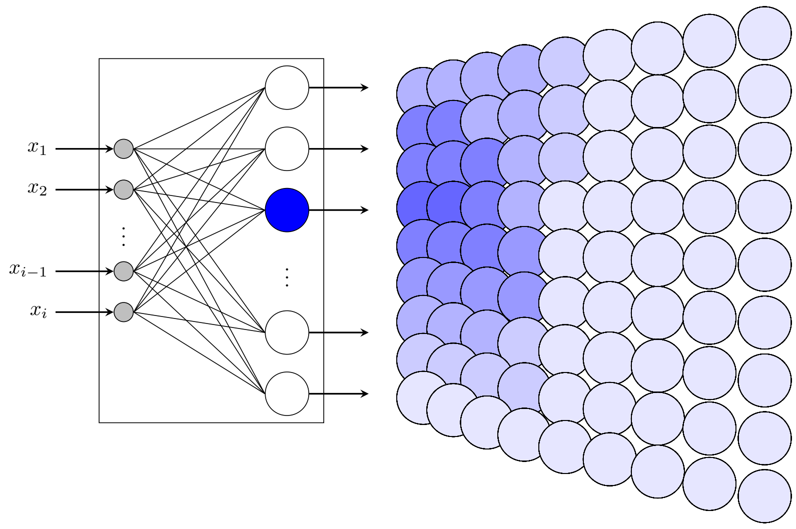


Figure 8: Self-Organizing Map viewed as connected neurons in an Artificial Neural Network.

connected neurons are firing), weights can be set using only local information.

In a supervised setting with desired (**gold standard**) values for the output layer, the error between the prediction of the Artificial Neural Network and the gold standard values can be used to set the weights. By defining a **loss** (**cost**) function, which quantifies the difference between the prediction and desired output, this difference can be propagated through the network and distributed on the weights. **Backpropagation** [Rumelhart et al., 1986] is done by computing the partial derivatives of the cost function with respect to the weights. Alternative methods for training network weights include neuro-evolution, which uses evolutionary algorithms to construct Artificial Neural Networks.

In a general discussion of biologically inspired systems, Floreano and Mattiussi [2008] categorized Self-Organizing Maps as an unsupervised learning approach to Artificial Neural Networks, without *desired output* when provided with training samples. When SOMs are viewed as ANNs, they consist of two layers; one is the input layer, corresponding to the dimensionality of the vectors in Algorithm 1, another being a competitive layer corresponding to the number of nodes chosen for the 2D space. Figure 8 illustrates how nodes in the input layer are fully connected to the nodes in the competitive layer.

Self-Organizing Maps are created using both **competitive** and **cooperative** learning strategies. Competitive, because neurons compete for each input vector to become its winning unit, i. e., the Best-Matching

Unit, cooperative because also the neighborhood of neurons around the BMU have weights set closer to the input sample. This neighborhood crucially relates to the topological space, and not the difference between vectors in their original dimensionality. Neurons bordering each other or residing close in 2D space are considered neighbors, regardless of the values of their vector weights. Floreano and Mattiussi [2008][p. 206] noted that such topological organization could also be found in the mammalian brain, with neighboring neurons affecting one another. Kohonen [1982] pointed out that this local feedback mechanism is key to the formation of the resulting maps.

2.1.4 *Self-Organizing Maps Applied to Natural Language Processing*

Kohonen et al. [1996] applied the algorithm to self-organize USENET (newsgroup) data in then-large simulations (with a 512-processor parallel architecture). A two-step approach was used to organize documents, so-called USENET posts[1]. First, each word in the corpus was represented by a 90-dimensional vector and organized in a small Self-Organizing Map. After the first step, each node represented clusters of words with related meanings. In a second step, a histogram of cluster memberships for each document was created, i. e., a vector with the dimensions corresponding to the number of nodes in the first SOM. By applying the SOM algorithm to these document vectors, a document map was created. Some heuristics were necessary to make training feasible, and the number of nodes in the final map was also expanded with interpolation. (The authors did not offer any detail on the node expansion step.) The final result was a granular map of nodes where similar documents were found in the same areas on the map.

Newsgroups are named according to a set of hierarchical categories, such as news, weather, and sports with both general groups and subcategories, specifying, e. g., location or topic. Thus, it could be verified that documents from related newsgroups ended up in the same area on the map.

This methodology was later used to cluster patent data [Kohonen et al., 2000] and the Encyclopedia Britannica [Lagus et al., 2004].

Hyötyniemi [1996] used Self-Organizing Maps to extract features for document representation, based on clustering character trigrams, arguing that this would account better for linguistic features. Eyassu and Gambäck [2005] used SOMs to classify Amharic news text. Categorized by experts, each document in the training corpus was associated with a query. A merged query and document matrix (i. e.,

1 Such posts are similar to emails but intended for many (unlimited) recipients.

the vector representation of the collection) was used for training the SOM. While the authors offered little detail on the SOM internals, such as topology and grid size and how class predictions were made, Eyassu and Gambäck did report using many Epochs (up to 20,000) and reported comparable classification accuracies to methods such as Latent Semantic Indexing.

Tambouratzis et al. [2012a] used Self-Organizing Maps for a specialized type of language modeling applied to Word Translation Disambiguation. Initially, terms were clustered with a SOM. Subsequently, the distances between these clusters were used to select word sequences using only monolingual resources.

Kaski et al. [1998] compiled a bibliography of SOM usage up to 1998, and Honkela [1997] surveyed the usage of SOMs within Natural Language Processing.

2.1.5 *Self-Organizing Maps as Classification Device*

Given a supervised classification task where a model first is fitted to a **training** corpus and subsequently tested on a held-out **test** portion, Self-Organizing Maps can be applied. Training samples have labels, which test samples will be attributed. A SOM is first created as described in Section 2.1 to train a classifier. After the last Epoch, the Best-Matching Units of all samples are stored.

For testing (classification), each test sample is first compared to the SOM matrix, returning its Best-Matching Unit. Because this BMU does not provide a prediction of a label directly, the BMUs from the training phase are consulted.

If each node in the Self-Organizing Map is the Best-Matching Unit of exactly one document with exactly one label, each node could predict this label for the test sample, and there would always be an available prediction. However, a given node can either be the BMU of multiple training documents with multiple labels or none at all. In the former case, a choice must be made from these labels. A natural solution is to pick the most frequent, i. e., **majority voting**. Another option is to attribute all found labels to the test sample, or all labels found within the neighborhood of the BMU. When the BMU of the test sample is void of training samples, labels can be found by searching the neighborhood of the BMU, either by moving on to the next closest node or gathering all labels within this neighborhood. These choices depend on how a task is defined, notably if one or more labels are to be requested for each test sample.

Figure 9 illustrates such a classification process, where two out of the four nodes in a 2x2 Self-Organizing Map are Best-Matching Units

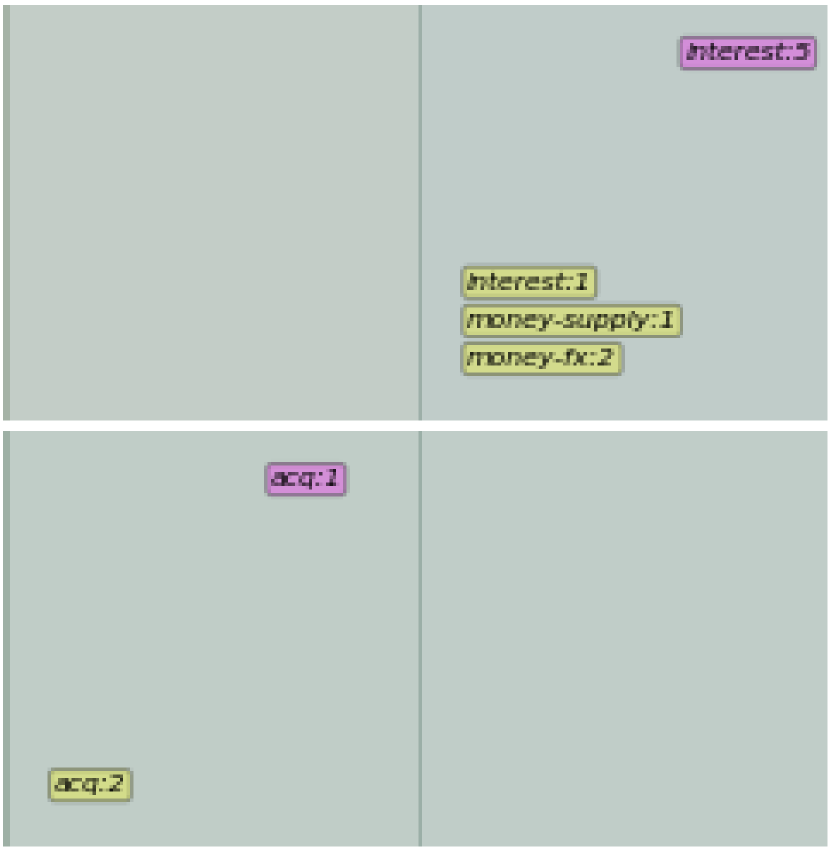


Figure 9: Example of clustering with a 2x2 SOM. The yellow labels are from the training corpus, and the magenta labels are from the test corpus.

for some documents. Yellow and magenta labels represent training and test samples, respectively.

In the bottom-left square of the grid, the label would be *acq* (acquisitions), and the single test document with that label would be classified correctly. In the top-right, however, the label of the node would be *money-fx*, and the five documents labeled *interest* would be incorrectly classified.

This example is from the Reuters Corpus [Lewis et al., 2004], which will be revisited in Chapter 7. The task can be defined as either single or multi-label by using subsets of the collection.

2.1.6 *Evaluation of Classification*

Accuracy is the ratio of correct predictions to the total number of predictions made. In binary classification, however, this measure can be misleading, as an unbalanced dataset with more positive than negative (or the opposite) instances can lead to a preference of a potentially useless classifier that classifies all instances as either positive or negative.

The measures **precision** and **recall** are evaluation metrics that provide alternative information on classification performance than accuracy. The following four outcomes of binary classification express the evaluation metrics well:

- True Positive (TP) = A positive instance classified as positive.
- False Positive (FP) = A negative instance classified as positive.
- True Negative (TN) = A negative instance classified as negative.
- False Negative (FN) = A positive instance classified as negative.

Precision is the proportion of correctly predicted positive instances to the number of predicted positive instances ($TP/(TP + FP)$). Recall is the proportion of correctly predicted positive instances to the number of positive instances ($TP/(TP + FN)$). Accuracy can be expressed as $(TP + TN)/(TP + FP + TN + FN)$.

F-score is a combination of precision and recall. Formally, F_β-score weights precision β times as much as recall:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} \tag{5}$$

With β set to 1, F_1-score is the harmonic mean of precision and recall:

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{6}$$

Henceforth, F-score is understood as this measure (Equation 6).

In a situation with more classes in a dataset for which instances can be positive or negative, calculating the **macroaverage** and **microaverage** over these classes are two ways of averaging the measures mentioned above. The microaverage averages over the classes by adding the TPs, FPs, TNs, and FNs for all categories and computing a sum.

Equation 7 shows microaveraged precision, where c denotes the number of correctly classified documents (summing TPs for all classes) and n (summing TPs and FPs for all classes) the total number of documents in that class, respectively:

$$Microaverage = \frac{c}{n} \tag{7}$$

This measure is skewed towards larger classes: Consider a classifier that classifies one large category with 90 documents 100% correctly, whereas ten other classes with one document each were all wrong. That would give a microaverage of 90%, even though most categories were completely wrong.

The **macroaverage** creates a simple average over classes, expressed formally for precision in Equation 8 where c_j and n_j are the numbers of correctly classified documents belonging to class j and the total number of documents in that class, respectively:

$$Macroaverage = \frac{\sum \frac{c_j}{n_j}}{|Classes|} \tag{8}$$

If a classification problem is cast as a **one-of** classification, i. e., when all documents in the test set belong to exactly one class, the microaveraged F-score will be the same as the accuracy. This because the number of False Positives and False Negatives will be equal. If a document is falsely classified to a label, it will be an FP in that class, but also an FN in the class it correctly belongs to [Manning et al., 2008].

2.2 CLUSTERING AND CLUSTER ANALYSIS INTRODUCED

Clustering is an important problem in Machine Learning, and key to unsupervised learning. Consequently, it is a topic in many books on statistical learning, e. g., Jain and Dubes [1988], Manning and Schütze [1999], Hastie et al. [2009], Everitt et al. [2011], and Han et al. [2011]; theses, e. g., van Dongen [2000] and Braeban [2011]; and reviews, e. g., Xu and Tian [2015]. Loosely, it pertains to the sectioning of objects into groups that are *close* to each other. A successful clustering algorithm establishes **natural groups** to which the various input objects belong. These groups should be as similar to each other as possible within each cluster (internal homogeneity), and as different as possible to the objects in other clusters (separation). Despite these criteria, a crisp definition has not been established. Everitt et al. [2011] highlighted the claim of Bonner [1964] that all that is required for a cluster to be a cluster is that this term is meaningful to the user. In the following, a **partitioning** refers to the specific division of a set of objects into a number of clusters.

Van Dongen (2000)[2] defined cluster analysis as follows:

> Cluster analysis is the mathematical study of methods for recognizing natural groups within a class of entities.

Tracing it back to the 1970s, van Dongen viewed cluster analysis as the result of cross-fertilization between mathematics and sciences such as biology, chemistry, medicine, and psychology. The author argued that mathematically, cluster analysis is the last step of sorting problems, progressing from classification through discriminant analysis.

Xu and Tian [2015] summed up the clustering process in four steps:

- Feature extraction and selection.
- Design of algorithms pertaining to the characteristics of the problem.
- Result evaluation: evaluate the clustering and judge the validity of the algorithm.
- Result explanation, give a practical explanation for the result.

Clustering algorithms need a **distance** score, or conversely a **similarity** score, between two data points to make assessments of these groups and the objects they contain. If a clustering is good, the intra-class similarity is high, whereas inter-class similarity is low.

Although there are many ways of establishing such natural groups, and often not possible to answer what the correct grouping is in every case (consider grouping people), there are patterns that a good algorithm should discover in some datasets. Consequently, an optimal clustering of objects largely depends on its intended use. A multitude of different clustering algorithms with different desiderata is available. Some categorizations of clustering are discussed in the next section.

2.3 categorizations of cluster analysis

Because there is no objectively correct clustering for many problems, the number of clustering algorithms is only limited by combinatorics, i. e., the theoretical bound on object groupings. Thus, categorizations of algorithms are helpful to analyze the various approaches.

Jain and Dubes [1988] noted an increased interest in cluster analysis, outlining the current methods and their mathematical foundations as well as their relation to application areas. Moreover, they provided a schematic overview of dichotomies in Pattern Recognition, presented

2 This thesis contains a very interesting chapter on the etymology of key terms.

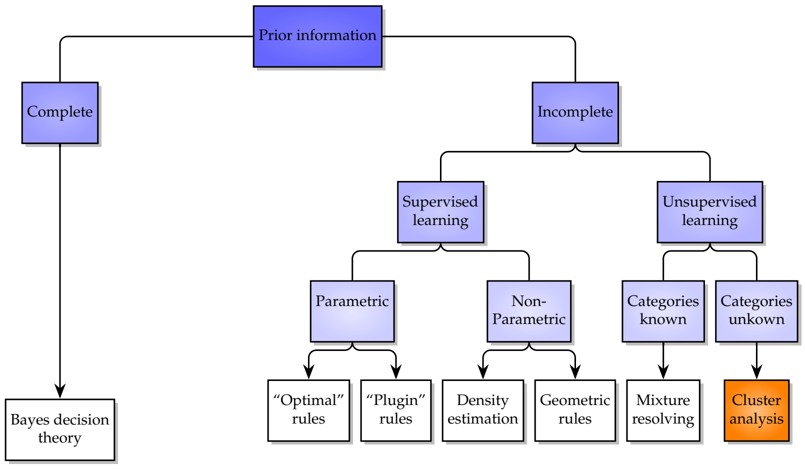


Figure 10: Dichotomies in Pattern Recognition. After Jain and Dubes [1988] (reprinted with permission from Anil Jain).

as a binary tree, shown in Figure 10. According to this figure, unsupervised learning recognizes patterns when prior information is incomplete, and as a branch thereof, cluster analysis finds patterns in data with unknown categories.

Manning and Schütze [1999] distinguished between clustering methods first based on their use, either for **exploratory data analysis** or **generalization**, understood as the induction of object bins from data. On the methodological side, Manning and Schütze focused on the structures produced by the clustering algorithms, either as **hierarchical** or **flat**, and whether the clustering is either **soft** (objects can be members of several clusters) or **hard** (objects can only be members of one cluster).

Han et al. [2011] summarized how clustering approaches are categorized, shown in Table 2. While the authors explained how distinctions are not crisply separatable, a categorization at this granularity was found meaningful. Other categorizations also exist, e. g., Braeban [2011] (from a discussion of **evolutionary** approaches to clustering). The **K-means** algorithm is an example of a **partitioning** method, and hierarchical agglomerative clustering algorithms are, as the name implies, hierarchical.

2.4 SOME CLUSTERING ALGORITHMS

A comprehensive presentation of clustering algorithms, of which there are many, e. g., [van Dongen, 2000, Lancichinetti and Fortunato, 2009],

Method	Description
Partitioning	Organizing objects into a set of several exclusive clusters.
Hierarchical	Organize objects into a hierarchy or tree, either **agglomerative** or **divisive**.
Density-based	Using connectivity and density functions, growing the clusters as long as the density is within some threshold. Density refers to the number of objects or data points in the neighborhood of a given radius.
Grid-based	Quantizing data objects into a structure of grid cells and doing the clustering operations on that.

Table 2: Different clustering approaches according to Han et al. [2011].

is beyond the scope of this thesis. Since, however, there is a relation between classification and clustering, i. e., that elements within the same class can be said to form a cluster, a selection of approaches will be discussed.

As shown above, the Self-Organizing Map algorithm is also applicable to clustering, as is the Expectation-Maximization (EM) algorithm [Dempster et al., 1977], e. g., to estimate the parameters of Gaussian Mixture Models. The EM algorithm loops over expectation and maximization steps, calculating expected values and maximizing parameters on those values until convergence.

In the remainder of this section, hierarchical agglomerative clustering will be presented, as well as the (also partitioning) K-means algorithm. The Self-Organizing Map can be seen as a constrained version of the K-means algorithm, which eventually will stabilize at one of its local optima [Hastie et al., 2009].

2.4.1 *Hierarchical Agglomerative Clustering*

Hierarchical agglomerative clustering algorithms, which were used for the clustering stage in the experiments in Part II, successively merge clusters from the bottom up according to some distance metric. In contrast, a top-down approach successively splits clusters.

Two elements are necessary for merging clusters; (i) a **similarity metric** to determine which clusters to merge and (ii) a **method** to select between which elements in the clusters to calculate this similarity. Distance metrics, such as Euclidean or Chebyshev distances are used for calculating a **distance matrix**, i. e., a matrix containing the distances between nodes. In a Self-Organizing Map, the grid of nodes is normally in 2D, while the vectors that represent them can be of any dimensionality (subject to hardware constraints). The intra-node distances are represented in a $\mathbb{R}^{nxn}$ matrix. Based on this matrix, a **linkage matrix** can be computed, consisting of the choice of points inside clusters to which the distance metric is applied.

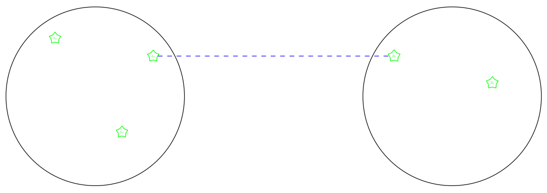
A linkage matrix is a representation of the links between the nodes. It is an (n-1)x4-matrix for n data points, with the first two columns being the IDs of the clusters to be merged, the third the distance between them, and the fourth the number of observations in the new cluster. The matrix can also be computed from raw observations directly, which is necessary for some methods, e. g., centroid, Ward's method, and median-based clustering. Using raw observations is more expensive since the intermediate distance matrix provides distance data that otherwise must be recalculated. In the bottom-up direction, nodes are merged with their closest neighbors, subsequently updating intra-cluster distances, such that clusters are formed, which, in turn, can be merged until all nodes are contained in one cluster at the top. Finally, the clustering height must be decided, which determines the number of resulting clusters.

2.4.2 *Clustering Methods*

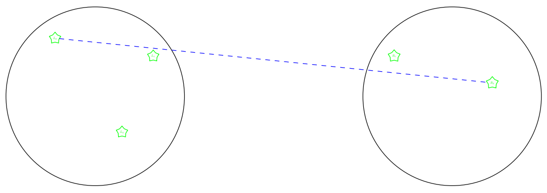
The clustering method determines how the distance between two clusters is measured, mainly a choice of what points to compare distances between, thereby establishing their degree of similarity. The methods used in this thesis are **single linkage**, **complete linkage**, **average linkage**, **weighted linkage**, **centroid**, **median**, and **Ward's clustering**. Figure 11 shows first three methods graphically, and Figure 12 summarizes the definitions of all methods mentioned above.

For the single linkage method, the distance between the clusters is determined as the distance between the two points in the two clusters that are closest. At the other extreme, the complete linkage method uses the maximum distance between two points in the clusters. The average method uses the distances between all points. The weighted linkage method takes the children of clusters into account, using the average of the distance between the two clusters that *formed* clusters A and B, as the distance between them [Han, 2005, Manning et al., 2008].

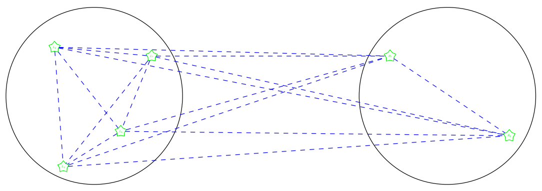
Ward's method, also known as the **minimum variance** method, minimizes the increase in within-cluster variance after merging. The



(a) Single Linkage; distance measured between the closest samples in candidate clusters.



(b) Complete Linkage; distance measured between the farthest samples in candidate clusters.



(c) Average Linkage; distance measured between all samples in candidate clusters.

Figure 11: Illustrations of three linkage methods.

merging cost is defined as the increase of the within-cluster variance as clusters A and B are merged and symbolized with Δ. The merging cost can be computed as the within-cluster squared difference of the merged cluster subtracted by the within-cluster sums of squares of clusters A and B. $\vec{m}_j$ is the center of cluster *j*.

$$\begin{aligned}\Delta(A, B) = &\sum_{i \in A \cup B} \|\vec{x_i} - \vec{m}_{A \cup B}\|^2 \\ &- \sum_{i \in A} \|\vec{x_i} - \vec{m}_A\|^2 \\ &- \sum_{i \in B} \|\vec{x_i} - \vec{m}_B\|^2 \\ &= \frac{|A||B|}{|A| + |B|} \|\vec{m}_A - \vec{m}_B\|^2\end{aligned} \tag{9}$$

As follows from Equation 9, the increase of within-cluster variance of the merged clusters A and B is the variance of the merged cluster subtracted by the variances of the two parts of which it consists. This

- single linkage (min) clustering:
  $d(A,B) = \min\{\, d(a,b) : a \in A,\ b \in B \,\}$
- complete linkage (max) clustering:
  $d(A,B) = \max\{\, d(a,b) : a \in A,\ b \in B \,\}$.
- average linkage clustering
  (Unweighted Pair-Group Method Using Arithmetic Averages):
  $d(A,B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a,b)$.
- weighted linkage
  (Weighted Pair-Group Method Using Arithmetic Averages):
  $d(A,B) = \frac{(d(S,B)+d(T,B))}{2}$ (A is formed by S and T)
- centroid clustering
  (Unweighted Pair-Group Method Using Centroids):
  The distance between two clusters is defined as the distance between their centroids $||c_s - c_t||$ where c_s and c_t are the centroids of the two clusters, calculated by the arithmetic mean, i.e., $c_s = \frac{1}{n} \sum_{x \in s} x$
- median clustering
  (Weighted Pair-Group Method Using Centroids):
  the weighted center of mass between the two clusters is the distance $d(A,B) = ||\tilde{x}_A - \tilde{x}_B||$ between their *weighted* centroids. $\tilde{x}_t$. If A was formed by clusters p and q, then $\tilde{x}_A$ is defined recursively as $\frac{\tilde{x}_q + \tilde{x}_q}{2}$
- Ward's minimum variance method [Ward, 1963]:
  The method minimizes the total within-cluster variance. At each step, it finds the two clusters with the minimum within-cluster variance after merging.

Figure 12: Summary of hierarchical clustering methods. d(a,b) is the distance metric between two nodes and d(A,B) is the distance between the corresponding clusters. The names in parentheses are alternative names for the methods.

calculation can be reduced to the expression in the final line, i. e., expressed via the centroids of clusters A and B.

2.4.3 *Determining the Number of Clusters*

With an agglomerative clustering algorithm, data points are recursively merged into clusters until all points are contained within one. While the different methods above perform the actual merging of the clusters through the establishment of their linkages, the resulting number of clusters depends on the height limit at which point branches are cut. Setting the height to the very top result in one cluster only. In contrast, setting it to the very bottom, i. e., before any merging is done, results in as many clusters as data points.

Since the optimal number of clusters depends on the task for which the clusters are used, there is no analytical way of finding this number. Methods for approximating this number include the **knee** (elbow) method, which finds the point with the maximum curvature on the line of within-cluster distances, information theoretic methods that quantify the fit of cluster members to their centroids and optimize the number of clusters on this basis, and experimental exploration according to some evaluation criterion [Salvador and Chan, 2004].

A dendrogram[3] shows how the agglomerative clustering algorithm successively merges all objects to clusters until only one cluster on top contains all objects. It is created from the linkages between nodes in the clusters as they are successively merged, i. e., by which nodes connect to each other. Figure 13 shows a dendrogram from clustering a Self-Organizing Map.

A visual interpretation of the dendrogram shows that a vertical bar can be placed anywhere in the figure to determine the number of clusters, cutting off the clusters at some height, illustrated with horizontal lines. In the extremes of top and bottom, the clustering process would add nothing, and the question is where between the two to place the vertical bar.

2.4.4 *The K-means Algorithm*

The K-means algorithm is an unsupervised clustering algorithm which requires a pre-set number of clusters (also referenced in Chapter 4) with appealing qualities, such as O(n) time complexity and conceptual simplicity. This algorithm was categorized as flat by Manning and Schütze [1999] and as a partitioning method by Han et al. [2011]. After initializing k means, each data point is assigned to its closest

3 From Greek *déndron* ("tree").

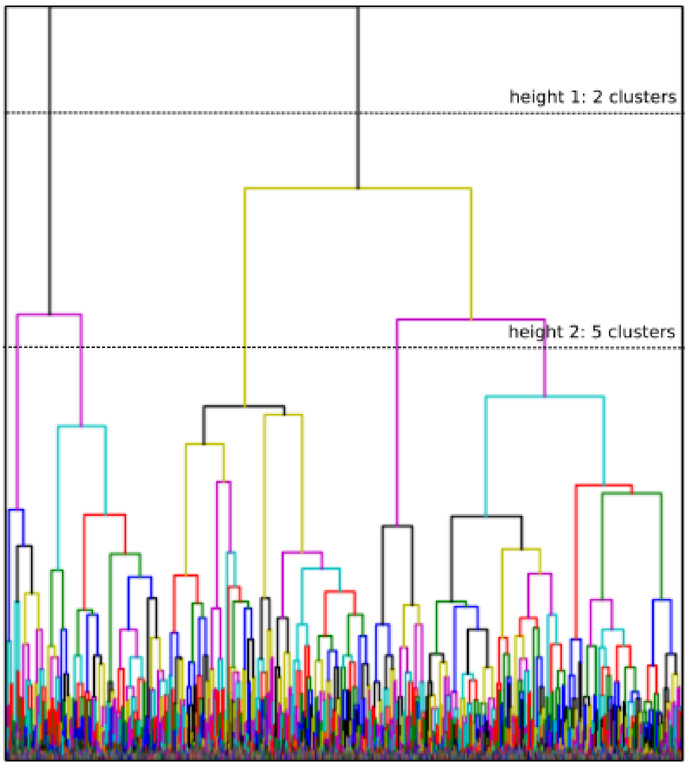
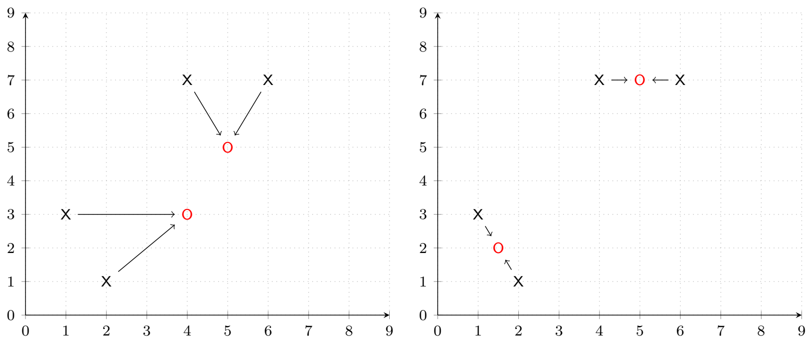


Figure 13: Dendrogram of hierarchical clustering of a Self-Organizing Map. Corpus: enTenTen, grid size: 64x64, clustering method: Ward's method.



(a) Nodes attaching to random means. (b) New means calculated based on *members*.

Figure 14: Sample iteration of the K-means algorithm. X and O represent the samples and the K means, respectively.

mean, after which new, true means are computed, and the process is repeated until the stopping criterion is met, such as a certain minimum of change in the means. Figure 14 illustrates one such iteration [Manning and Schütze, 1999].

2.5 SELF-ORGANIZING MAPS AND CLUSTERING

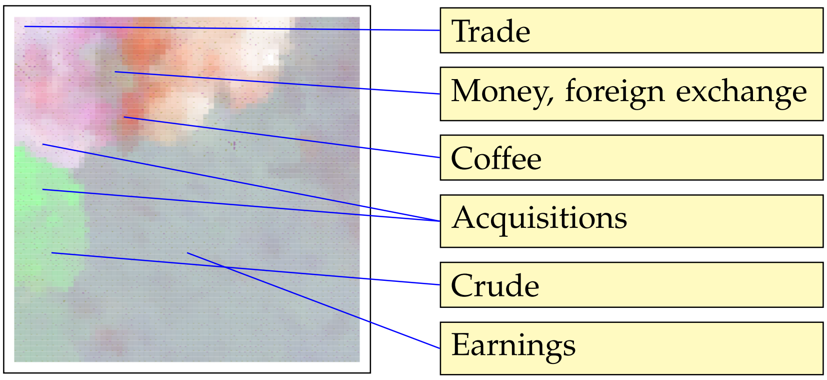


Figure 15: A Self-Organizing Map of the Reuters Corpus. Color shaded areas contain documents with the largely the same labels. A repeat of Figure 4 (example SOM from Reuters Corpus).

Figure 15 (repeat of Figure 4) visualizes each node in the grid as colors, annotated with labels indicating the classes of the documents belonging to the designated areas (after manual inspection). These colors are created by first reducing the vectors representing each node to four dimensions using Principal Component Analysis (PCA) [Jolliffe, 1986] (a method based on Singular Value Decomposition) and then rendering these vectors as colors. If two feature vectors are close to each other in the high-dimensional space, they will also be close to each other in the topological space of the Self-Organizing Map. Hence, the visualization of the SOM shows the underlying structure in the training data. However, it is the feature vectors in each node that are displayed, not the documents themselves.

The Best-Matching Units after the last training Epoch are stored, e. g., for labeling purposes (see Section 2.1.5) or dumping documents contained in a cluster. In the Self-Organizing Map displayed in Figure 15, there were 4096 nodes. While each node could be seen as a separate cluster, this number of clusters is high and not necessarily useful for an application. Thus, when the number of nodes is higher than the desired amount of clusters, node vectors must be grouped. The aim is to identify the borders of the areas discernible by the color shadings in Figure 15. In effect, that is the same as clustering the

SOM matrix after training. After clustering, areas in the SOM become concrete clusters, which group objects (documents in this example) with these node vectors as BMUs. Chapter 8 presents a method of clustering SOMs hierarchically (see Section 2.4.1) with experimental evaluation.

2.6 EVALUATION OF CLUSTERING

Evaluation of clustering splits in two directions; (i) **intrinsic** evaluation (according to the mathematical properties of relations between objects) or (ii) **extrinsic** evaluation (according to the performance on a task for which the clustering was done). Examples of extrinsic and intrinsic evaluation metrics will be presented in the remainder of this section. The list is not intended to be exhaustive, numerous other measures based on Information Theory (to which Section 3.2.5 returns) and Information Retrieval (i. e., obtaining relevant resources to an information need from a larger source) can be found. Rendón et al. [2011] conducted a study of the relationship between extrinsic and intrinsic metrics.

2.6.1 *Extrinsic Evaluation of Clustering*

In place of human evaluation, labeled data is necessary for automated extrinsic evaluation. The appropriate kind of extrinsic evaluation is heavily task-dependent, such as the automated metrics for Machine Translation that will be used for the end-to-end evaluation of the experiments in Part II, of which clustering is an integral part. With a labeled corpus, several automated metrics can also be applied to the clustering itself by measuring how well the proposed clustering matches the gold standard reference clusterings.

2.6.1.1 *Purity*

When a set of objects is labeled and partitioned into some number of clusters, purity is a measure of the proportion of data points correctly put into the same cluster, according to some **ground truth** t. Poor clusterings have purities close to 0, whereas perfect clusterings have purities of 1.

Which label in the ground truth that occurred most frequently in a cluster (hence, the $\max$) is determined for each cluster in a proposed partitioning. Ideally, each cluster would contain only elements from one label in the ground truth (being pure), but often elements also from other ground truth labels appear in a cluster. Equation 10 expresses this as follows:

$$\mathrm{Purity} = \frac{1}{N}\sum_{i=1}^{K} \max_{j} |c_j \cap t_i| \tag{10}$$

where c_k is a cluster, t_j a true classification (i. e., a group of objects with the same label). N is the number of data points, and K is the number of clusters.

The purity metric is not affected by which objects that are assigned to respective clusters, but expresses how pure clusters are, i. e., to what degree same-label objects are found therein. The highest purity is attained by mapping true clusters (t) to classifications (c) with the highest number of objects, as any other mapping would reduce the numerator in Equation 10, which explains the max operation. It should be noted that having many clusters is not penalized. Hence, putting each data point in a separate cluster would resulting in a purity of 1.

2.6.1.2 *Rand Index*

The Rand Index is based on typical Information Retrieval performance metrics (see Section 2.1.6) on the combinations of pairs of clustered objects based on the status of these pairs, viz. True Positive, True Negative, False Positive or False Negative. Of $\binom{N}{2}$ pairings in the set of objects, the status of each pair is assigned as follows:

- True Positive (TP) = Two similar objects assigned to the same cluster.
- False Positive (FP) = Two dissimilar objects assigned to the same cluster.
- True Negative (TN) = Two dissimilar objects assigned to different clusters.
- False Negative (FN) = Two similar objects assigned to different clusters.

From these statistics, the Rand Index is calculated similarly to accuracy, i. e.:

$$\mathrm{RandIndex} = \frac{TP + TN}{TP + FP + TN + FN} \tag{11}$$

2.6.2 *Intrinsic Evaluation of Clustering*

Intrinsic evaluation metrics rely on the properties of the clusters, without regard to their final application. These metrics can indicate how well-suited a partitioning of a set of objects is, but a good score does not guarantee the correspondingly good performance of an application. Three prominent intrinsic evaluation metrics will be described in the remainder of this section. More metrics can be found in the above-cited works on cluster analysis and also, e. g., in Legány et al. [2006] and Baarsch and Celebi [2012].

2.6.2.1 *The Davies-Bouldin Index*

The Davies-Bouldin index [Davies and Bouldin, 1979] measures the intra-cluster compactness relative to the inter-cluster separation. Formally expressed as:

$$DB = \frac{1}{K}\sum_{i=1}^{K} \max_{i \neq j} \left(D_{i,j}\right) \tag{12}$$

K is the number of clusters and $D_{i,j}$ is the ratio of the within-cluster distances of clusters i and j to the distance between them. $D_{i,j}$ is derived as follows:

$$D_{i,j} = \frac{\overline{d}_i + \overline{d}_j}{d_{i,j}} \tag{13}$$

$\overline{d}_x$ is the average distance between each object in the cluster and its centroid, and $d_{i,j}$ the Euclidean distance between the centroids. A lower Davies-Bouldin index value represents a better clustering.

2.6.2.2 *The Dunn Index*

The Dunn Index [Dunn, 1973] measures the ratio between the smallest inter-cluster distance and the largest intra-cluster distance, respectively. Equation 14 expresses this formally as:

$$DI = \min_{1 \leqslant i \leqslant K} \left\{ \min_{1 \leqslant j \leqslant K, i \neq j} \left\{ \frac{d_{inter}(C_i, C_j)}{\max_{1 \leqslant l \leqslant K} d_{intra}(C_k)} \right\} \right\} \tag{14}$$

The numerator represents the highest inter-cluster distance after minimization, and the denominator represents the cluster with the largest intra-cluster distance after maximization. This index can be used with any distance function, e. g., the type of linkage to determine the inter- and intracluster distances, respectively. A higher Dunn index value indicates a better clustering.

2.6.2.3 *The Silhouette Coefficient*

The Silhouette Coefficient (SC) is a measure of how well the clusters are separated and how compact they are. The number of clusters could also be determined using this score. The SC is calculated by first taking the average distance between each object belonging to a cluster and all other objects in the cluster to which this object belongs, and then the minimum average distance from the object to all other clusters to which it does not belong.

Formally[4], if an object $o \in C_i (1 \leqslant i \leqslant K)$, then

$$a(o) = \frac{\sum_{o' \in C_i, o \neq o'} d(o, o')}{|C_i| - 1} \tag{15}$$

The term $a(o)$ is an indication of **compactness**. Thus, a lower value indicates a more compact cluster, i. e., small distances between the objects inside a cluster. and

$$b(o) = \min_{C_j : 1 \leqslant j \leqslant K; j \neq i} \left\{ \frac{\sum_{o' \in C_j} d(o, o')}{|C_j|} \right\} \tag{16}$$

$b(o)$ describes the separation from other clusters. The larger it is, the more separated the cluster is from the others. Finally, the Silhouette Coefficient is determined by

$$s(o) = \frac{b(o) - a(o)}{max\{a(o), b(o)\}} \tag{17}$$

The value of the Silhouette Coefficient ranges from -1 to 1. When the SC reaches 1, the clusters are compact (low $a(o)$) and far away from the other clusters (high $b(o)$). A higher number indicates a better clustering.

Petrović [2006] compared the Davies-Bouldin measure to the Silhouette Coefficient on an intrusion detection task and found the latter to correspond better to performance on that task.

4 Definitions from [Han, 2005, pp 489-490].

3 MACHINE TRANSLATION

This second background chapter will further introduce Machine Translation by discussing major challenges, components, and methodologies. First, Section 3.1 outlines the main approaches to Machine Translation. Next, Section 3.2 takes up language modeling because of its importance for the experiments in Part II before Section 3.3 discusses the role of Artificial Neural Networks in MT. Finally, Section 3.4 addresses evaluation.

The linguist Garvin [1956] defined Machine Translation as using logical machines to perform translation, understood as:

> "The transference of meaning from one patterned set of symbols occurring in a given culture ... into another set of patterned symbols in another culture."

Garvin divided translation into two processes; (i) the selection of the appropriate translation unit and (ii) the arrangement operations ensuring that the translations of these units appear in the output in an order such that the text as a whole is correctly translated.

Commonly thought of as *the use of computers in translating from one natural language to another*, unassisted from input to output, Machine Translation is also understood as any use of computers along this path [Maegaard, 1999]. If defined as any use of technology in translation such as electronic dictionaries, most forms of translation would fall in this category. This definition would include Computer-Assisted Translation (CAT), translation aided by tools such as dictionaries, thesauri, and spell checkers [Dunne, 2012].

Other ways to divide the translation task between automated and manual systems involve pre- or post-processing of data. Because it requires less training than translation proper, post-editing by humans [Carl et al., 2015] is an alternative for systems that can produce reasonable, but not publishable quality. Systems like `Unbabel` [Graça, 2014] use crowdsourcing to leverage post-editing to ensure high-quality Machine Translation output.

An in-production example of Machine Translation is *La Vanguardia*, a newspaper that credits[1] MT for being able to publish in both Spanish and Catalan simultaneously [Martín and Serra, 2014]. However,

1 `http://www.lucysoftware.com/english/news-events/143-la-vanguardia-5-years-anniversary-of-catalan-edition.html` (Last visited: March 27, 2019.)

this system *does* require post-editing by a human, which means the output is not published verbatim, and the creators are clear that the output is not good enough for direct publication. Nonetheless, the newspaper considers the dual-language system as a product of MT.

While the details of the methodologies used in the corporate world often remain commercial secrets, fully automated translation in *production* is mostly associated with recommender websites like Tripadvisor[2] whose Machine Translation output is published to the user without post-editing[3] or the shared economy giant Airbnb[4], which machine translates customer reviews.

Henceforth, Machine Translation is understood as fully automated unless otherwise stated, i. e., without human intervention until the result is evaluated.

3.1 MACHINE TRANSLATION APPROACHES

With precursors and roots dating further back, Machine Translation (MT) using computers as we know them today, surfaced in the wake of World War II [Hutchins, 2010]. Since then, methodologies have progressed both regarding computational resources and linguistic analysis.

Coarsely typologized, Machine Translation systems are either rule-based (RBMT) or statistical (SMT), rationalist and empiricist paradigms, respectively. Hybrids slide between these distinct approaches. Some division of labor between them is widely accepted as necessary. The etymological roots of the word *translate* come from Latin and Old French, and roughly it means to *carry across*. Abstractly, what separates RBMT and SMT is how the meaning content is *carried across*, either using hand-crafted linguistic rules or statistically inferred relations based on data analysis.

Since 2016, Neural Machine Translation (NMT) is established as the state-of-the-art on many Machine Translation tasks. As Statistical Machine Translation, it is a data-driven approach that learns rules from processing data. However, using Artificial Neural Networks to learn these relations deviates significantly from SMT in methodological terms, such that NMT represents a salient shift and a new approach.

Henceforth, a translation of a **language pair** is referred to as translating Source Language (SL) text into a Target Language (TL) representation, i. e., **Source** and **Target** Languages refer to what is translated from and into, respectively.

2 `http://www.tripadvisor.com/` (Last visited: March 27, 2019.)
3 `http://www.promt.com/media/news/44383/` (Last visited: March 27, 2019.)
4 `http://www.airbnb.com` (Last visited: March 27, 2019.)

3.1.1 *Rule-Based Machine Translation*

Rule-Based Machine Translation dominated the early decades of Machine Translation. Hutchins [2005] argued that the Georgetown Experiment of 1954 marked the beginning of MT as a research field worthy of funding. In this experiment, a system comprised of a vocabulary of 250 words and six grammar rules was able to produce translations of text in the chemistry domain. Some of the sentences looked like general statements. Despite the limited scope, the press reported the program as an "electric brain," and the public reception of the program showed most interest for automatic translation of the general-sounding sentences.

The coverage of rule-based systems improved with time, both regarding linguistic phenomena accounted for by grammatical rules and semantic entities covered by increased vocabulary. The Canadian `TAUM-METEO` system [Thouin, 1982], which translated weather forecast between the two official languages (English and French) and the `Systran` system used at the European Commission are prominent use cases. In recent years, the open-source `Apertium`[5] framework [Forcada et al., 2011] has been widely used to create Rule-Based Machine Translation systems, offering translation between many language pairs.

Figure 16 shows the Machine Translation triangle, which explains the difference between different types of Rule-Based Machine Translation [Vauquois, 1976]. This triangle appears in many forms, and the differences mostly owe to the vertical granularity, as intermediate levels of analysis are not always made explicit, e. g., between word forms and tokens. Figure 16 shows five levels of analysis. From the bottom up:

1. token — here understood as the raw input to the system, such as inflected forms.

2. lemma — the canonical form of words that can be looked up in traditional dictionaries.

3. syntax — analyzed (parsed) text, which is attributed with syntactic information.

4. semantic — analyzed (parsed) text, which attributed with semantic information.

5. interlingua – text is converted to a representation shared between two or more languages.

5 `http://www.apertium.org` (Last visited: March 27, 2019.)

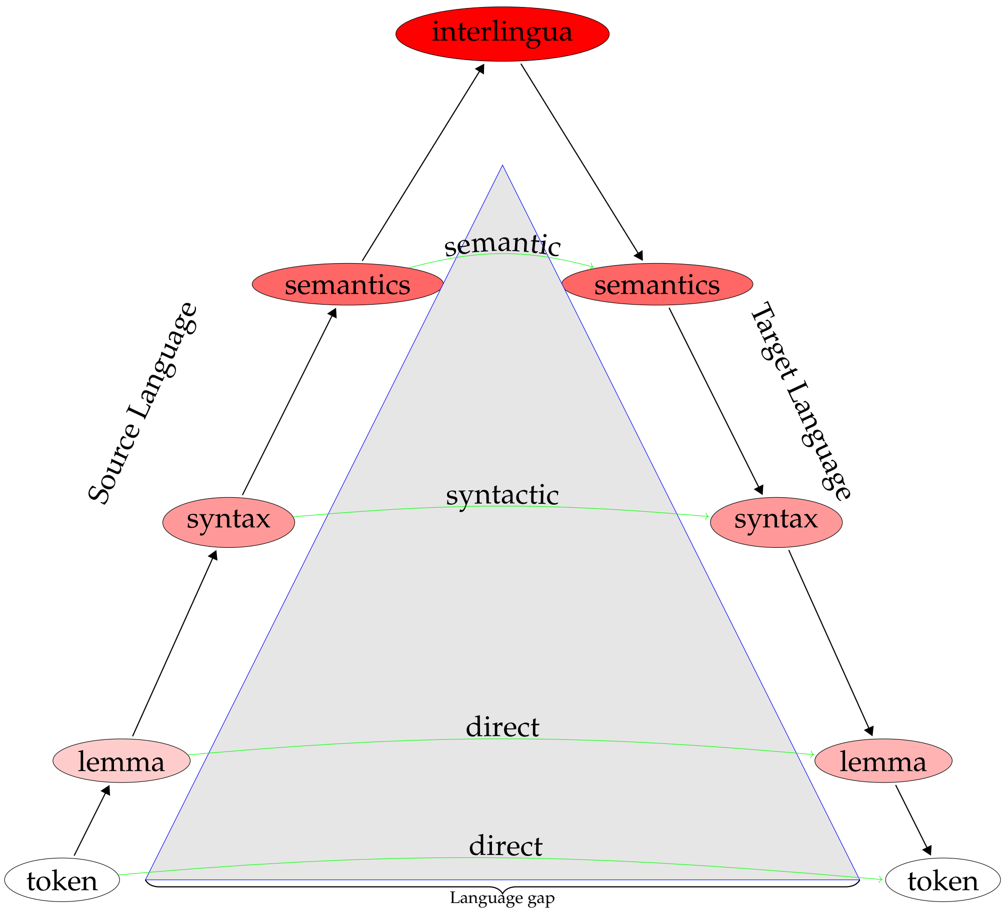


Figure 16: Machine Translation Triangle. Source Language text is analyzed at various depth levels, and translated into the Target Language by generating text at levels closer to surface form. The green arrows show the types of transfer rules required to translate at each level.

Translation is done by first *analyzing* the Source Language input (left side of the figure), then looking up the representation of this analysis in translation rule tables, and finally *generating* the Target Language output with corresponding grammars in that language (right side of the figure). Thus, the transfer rules on different levels work like bridges, over which the meaning can be *carried* across from Source to Target Language. The triangular shape captures how the distance between the languages decreases as the levels of representation deepen, illustrated with deeper shades of red.

Rule-Based Machine Translation has the advantage that complex linguistic phenomena such as anaphora resolution can be addressed directly in the analysis phase. Such phenomena can be idiosyncratic to each language pair, and difficult to capture by data-driven methodologies, e. g., owing to data sparsity. Conversely, complex rules have challenges with robustness and coverage. Furthermore, the maintenance of such rules is resource-demanding, as adding all new rules and exceptions needed would be an endless task [Attnäs et al., 2005].

3.1.2 *Example-Based Machine Translation*

Example-Based Machine Translation (EBMT) systems originated in Japan in the early 1980s. By reusing examples in the form of already translated text, such examples can improve Machine Translation systems, similar to how translation memories offer help to human translators [Somers, 2003, Reinke, 2013]. EBMT is separated into three phases, viz.: **matching**, **alignment**, and **recombination**. First, an input sentence is matched to the database of translation examples. Next, the portions (as a full match is unlikely) that are matches between the sentences are identified. Finally, these relevant portions are recombined. Thus, EBMT has similarities to Case-Based Reasoning [Aamodt and Plaza, 1994], a data-driven AI method that learns from examples for problem-solving.

3.1.3 *Statistical Machine Translation*

Statistical Machine Translation is a data-driven method for inferring relations between source and target languages. The so-called IBM models, whose implementations were published in the early 1990s [Brown et al., 1990, 1993] were especially influential. Data-driven approaches have established themselves as the state-of-the-art, and a myriad of different approaches have emerged. Surveys by Lopez [2008] and Koehn [2010] offer comprehensive overviews. A “survey wiki," i. e., a continuously updated online survey is also maintained at `http://statmt.org/survey/`[6].

Statistical methods estimate parameters from occurrences of events in **parallel corpora**. Parallel corpora consist of **sentence-aligned** data structures, which means that the algorithms are processing sentences that are translations of each other to infer statistical relations between the languages.

By gathering statistics from these corpora, a model can be queried for the probability of a Target Language string being a translation of the Source Language input. Thus, translation becomes a search for the TL string with the highest probability of being its translation.

$$\underset{T}{\operatorname{argmax}} P(T|S) = ? \tag{18}$$

Equation 18 expresses the conditional probability of seeing T given S. T and S and represent the Target and Source Language strings, respectively. The modeling of such translation probabilities will be explained below.

3.1.3.1 *Statistical Modeling*

Counting the possible translations of each Source Language sentence in a given parallel corpus would create a valid probability mass function since all members of the sample space have above-zero probabilities, and the probability of the samples would sum to 1.

However, such a model would only be able to translate the exact sentences appearing in a parallel corpus. It is necessary to model translation probabilities compositionally to extrapolate the statistical relations to unseen data successfully. Neither the set of source language sentences nor the set of their translations is finite. Consider how easy it is to construct a perfectly valid, syntactically and semantically, English sentence yielding zero Internet search results, despite the size of the indexed web.

Because of innumerability, there is no such thing as a probability of a given string's existence in or translation into a natural language. Nonetheless, the coverage of models estimating it will increase as the available training data increases. Still, to be operational, statistical models of translation must assign some probability to a given string and its translation, also if it contains elements not seen in training. Concretely, Equation 18 is decomposed into many other equations

6 Last visited: March 27, 2019.

that represent phenomena in the corpus, which provide a translation in combination.

$$P(T|S) = \frac{\overset{\text{Translation model}}{P(S|T)} * \overset{\text{Language model}}{P(T)}}{\underset{\text{Constant during search}}{\cancel{P(S)}}} \tag{19}$$

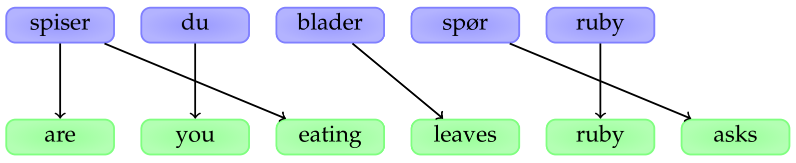
Equation 19 shows the first step in decomposing the argument of the $argmax$ function in Equation 18, which is its Bayesian inversion. The search for the best translation is done by querying a model of how likely the Target Language string is, given the Source Language input multiplied by the probability of the TL string itself. When searching, the denominator $P(S)$ can be disregarded, as the SL input — whose translation is being searched for — is constant during the search.

The probability mass function $P(S|T)$ is referred to as the Translation Model and $P(T)$ as the Language Model. The former models the probability of a Source Language string being the translation of a Target Language string — a model of translation — whereas the latter models the probability of seeing that particular string in the TL — a model of language.

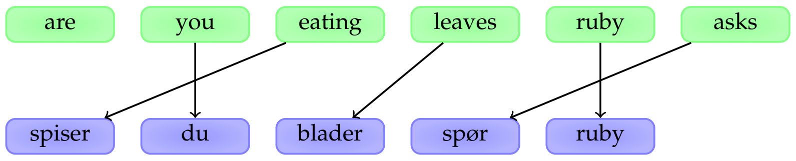
While Equation 19 retains equality with Equation 18, these probability mass functions must be decomposed further such that they can be queried for probabilities of individual words and aggregated. Thus, the equation must be modified to provide translations for entire Source Language strings from its parts, such that translations for unseen events can be provided.

The IBM models decomposed the formula for the Translation Model with transformations that did not maintain equivalence, but resulted in a model that looks up translation probabilities of single words and combines them. As a result of these transformations, the Translation Model formula was transformed into a combination of generative models of separate aspects of translation.

IBM models 1 to 5 introduced concepts such as **alignment**, alignment probabilities, **fertility**, and **deficiency**. Fertility pertains to the probability of a Source Language word becoming translated into multiple words in the Target Language. Deficiency is a less intuitive term relating to the wasted probability mass on impossible alignment probabilities. Alignments, having a different meaning than in Example-Based Machine Translation, will be explained in the next subsection.



(a) Direction: Norwegian-English.



(b) Direction: English-Norwegian.

Figure 17: Example word alignments between sentences in Example (101) (alignments provided by author). Punctuation removed.

3.1.3.2 *Alignment*

(101) Spiser du blader? spør Ruby.
eat you leaves? asks Ruby
Are you eating leaves? Ruby asks.

Consider Example (101)[7], a Norwegian sentence translated into English. The gloss outlines some syntactic differences between the languages, such as differences in word order (last two words), and the use of an auxiliary verb in the English question.

Figure 17 shows an example of aligning the English sentence to the Norwegian original. Each Target Language word aligns to a maximum of *one* Source Language word, which is not required in the opposite direction. The **alignment** between the sentences represents this mapping. While the words are not spatially *aligned* in the figure as the gloss in Example (101) is, the alignment rather models what words are translations of each other. This way, translation probabilities of individual words based on their alignment can be modeled and used in a generative model. Translation Models and alignment probabilities are estimated with the Expectation-Maximization algorithm in the IBM models, from parallel corpora.

Determining the correct alignment of a sentence pair is difficult, also for humans. Valid arguments for different alignments can often be made, much like how multiple translations can be equally correct. Since the Target Language token *spiser* can only be aligned to one Source Language word, a decision must be made on aligning either

7 From Lars Saabye Christensen's novel *Herman*.

are or *eating* to the Norwegian word *spiser*, which is a translation of both in Figure 17b.

Reports on the effect of alignment on overall Statistical Machine Translation performance vary. Lambert et al. [2012] studied different alignment characteristics (such as precision, recall, and the number of links) and concluded that the impact of these characteristics depends on the particular Machine Translation system and the training corpus.

Alignment may also refer to sentence alignment, i. e., which sentences that are translations of each other in a parallel corpus. Henceforth, alignment refers to word alignment unless otherwise stated.

3.1.3.3 *Phrase-Based Statistical Modeling*

Since relying on modeling translations of whole sentences is infeasible due to sparsity, it is necessary to model translation compositionally for a model to be useful. However, reducing translation to a multiplication of word translations has caveats. The problems posed by alignment above show that reducing translation to a translation of individual words is problematic. Often, the units of equivalence are sequences of words between languages. When phrase-based Statistical Machine Translation was introduced [Koehn et al., 2003], it achieved significant performance improvements over word-based models. The phrase-based models reduce the complexity of the more elaborate generative models by reducing the Translation Model to translation probabilities between phrases and their *reordering*.

Extraction of these phrases from parallel data represents an increase in complexity, however. Koehn et al. [2003] did this by computing alignments in both directions and subsequently extracting phrases from these alignments based on their commonalities. First, these alignments are expanded according to a heuristic somewhere between the intersection (high precision) and the union (high recall) of the two alignment sets. From these expanded alignments, **consistent phrases** are extracted, i. e., phrases that are exclusively aligned with each other, and no words outside [Och et al., 1999].

Figure 18 shows a matrix of both the intersection and the union of the two alignments presented in Figure 17, handmade for illustration purposes. Phrase pairs such as Example (102) are extracted from this matrices, such that their frequencies can be computed.

(102) spiser du
are you eating

The green ellipse in Figure 18 marks the phrase, whereby all words in either language are aligned to words in the same phrase. On the other hand, the red ellipse encloses the phrase pair "are you/*spiser*

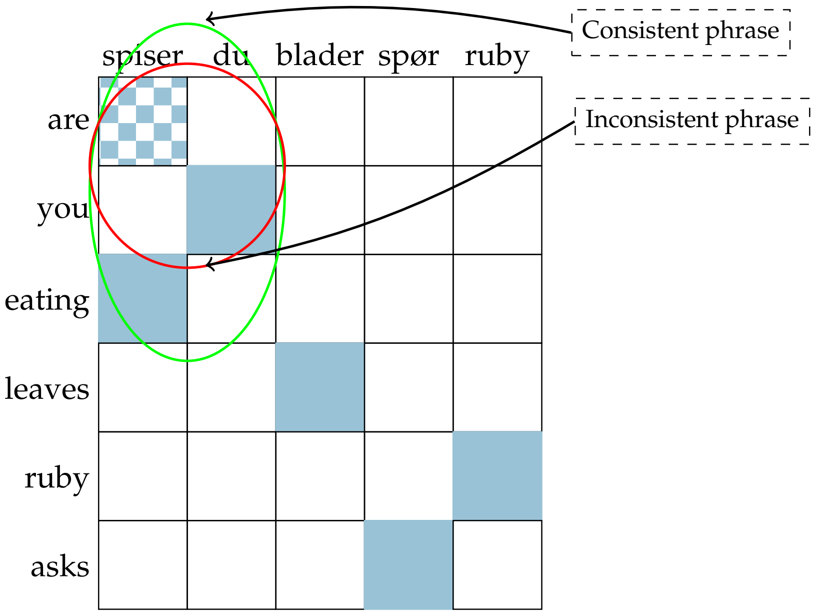


Figure 18: Alignment example for phrase extraction. Intersections of alignments in solid pattern, union in checkered.

du", which is inconsistent, as the word *spiser* is aligned to "eating", outside of the phrase pair. Other methods for phrase extraction include syntax-based (dependencies and treebanks) [Srivastava et al., 2009] or information theoretic measures, such as multivariate mutual information [Nasri et al., 2013].

$$\prod_{i=1}^{I} \phi(\bar{s}_i|\bar{t}_i)^{\lambda_\phi} d(\mathtt{start}_i - \mathtt{end}_{i-1} - 1)^{\lambda_d} \prod_{i=1}^{|t|} p_{LM}(t_i|t_1 \cdots t_{i-1})^{\lambda_{LM}} \quad (20)$$

In its basic form, the search for the best translation finds the Target Language phrase that maximizes Equation 20, where $\bar{s}_i$ and $\bar{t}_i$ denote source and target language phrases, $\phi(\bar{s}_i|\bar{t}_i)$ is the phrase translation probability, d the reordering model, and p_{LM} the language model probability. I and $|t|$ denote the number of phrases and the length of the Target Language string, respectively. If two phrases are translated in succession, the current phrase, $\mathtt{start}_i$ equals the end of the previous phrase, $\mathtt{end}_{i-1} + 1$. Thus, the reordering model applies a reordering cost to differences from this state if the starting position is elsewhere. A feat of the phrase-based models is the log-linear formulation where the score of a given translation candidate is represented as a combination of features with weights λ_ϕ, λ_d, and λ_{LM}, which can be extended to more features.

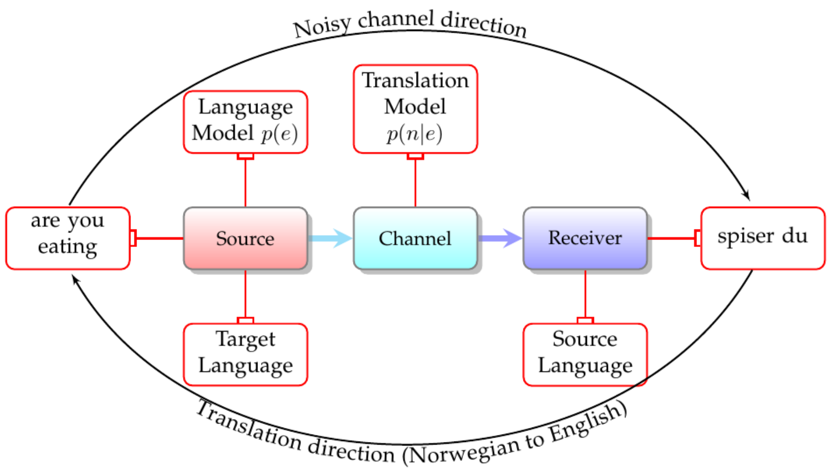


Figure 19: Noisy channel interpretation of Machine Translation. Noisy channel direction: left to right. Translation direction: right to left.

3.1.3.4 *Decoding of SMT Models*

Section 1.1.3 introduced **decoding** as a search for the optimal Target Language string given the input to the statistical models. The term for what really is a search is attributed to Weaver [1947]:

> "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode."

Shannon's theorem [Shannon, 1948] models how signals are transmitted from a source, through a noisy channel, are received at the other end. These received signals can be *decoded*, such their original state is recovered. The noisy channel model is used as an analogy for many NLP problems such as Automatic Speech Recognition (ASR), Optical Character Recognition (OCR), as well as Statistical Machine Translation.

Figure 19 shows the noisy channel model applied to Machine Translation. The source signal (here: an English string) is passed through the noisy channel, distorting the signals and dispatching them to the receiver (here: a Norwegian string). By modeling the noisy channel as a Translation Model and the source signal as a Language Model, it is possible to recover the *original* string that was sent through this noisy channel. Hence, the model assumes that the received (Norwegian) signal was *intended* to be an English string and searches the models for the optimal English string given the received signal. This is consistent with the formulation in Equation 19 if English is the Target and Norwegian is the Source Language. The first letters of the languages are used to describe the models in the figure to avoid confusion between the source of the noisy channel and the SL of translation.

Going backward through the noisy channel means searching for the Target Language string that has the highest probability of the Translation Model and Language Model combined. Because TL word-order can change for word-based models, the search space is exponential. Knight [1999] proved that decoding is NP-complete.

Log-linear models have the general form:

$$\exp\left(\sum_i \lambda_i f_i(X)\right) \tag{21}$$

where λ contains the weights and f_i are real-valued feature functions on the variables X. The combination of weights listed in Equation 20 generalizes to a log-linear model combining **feature functions** (corresponding to the log of what is being raised to an exponent) with **weights** (the exponents). With a log-linear model, decoders can integrate knowledge external to the models, such as preliminary evaluation scores, auxiliary Language Models, or other sources of knowledge, as long as they produce real-valued output.

Costa-Jussà and Farrús [2014] surveyed the integration of linguistic knowledge into SMT models, describing feature functions as one way of achieving this (in addition to, e. g., to pre- and post-processing and syntax-based models). Feature weights can be tuned in a phase between learning the featured models and decoding using various strategies utilizing a development corpus and error metrics to optimize weights [Cherry and Foster, 2012]. (Neubig and Watanabe [2016] surveyed optimization methods for SMT systems and concluded that good evaluation metrics, stable optimization of millions of features, and better utilization of non-linear scoring methods remained open questions.)

Decoders such as the `Moses` decoder [Hoang and Koehn, 2008], the decoder of the `Joshua` Machine Translation System [Post et al., 2015], and the `cdec` decoder [Dyer et al., 2010] offer well-known search and pruning strategies, such as A* search, greedy hill climbing, and stack decoding, as well as histogram and cube pruning.

Data structures representing partial translations, **hypotheses** are created by ticking off Source Language words in any order, creating the Target Language string left-to-right. Heafield et al. [2014] attained performance gains with comparable output quality by sorting hypotheses into equivalence classes, avoiding unnecessary queries to the Language Model by exploiting shared words between hypotheses that only differ in SL coverage.

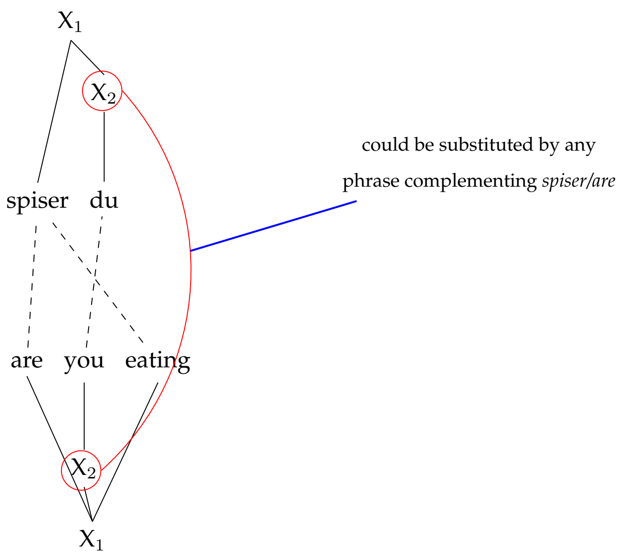

X_1
X_2
could be substituted by any
phrase complementing *spiser/are*
spiser
du
are
you
eating
X_2
X_1

Figure 20: Example (102) annotated with syntactic levels.

3.1.3.5 *Hierarchical Phrases / Synchronous Grammars*

Motivated by leveraging the strengths of the phrase-based approach in reordering words to reordering of clauses, Chiang [2007] presented a framework for mapping **hierarchical phrases** (phrases containing other phrases) to each other. Formally, the methods learn synchronous context-free grammars, where syntax trees of translation pairs form the basis for inducing rules for translation between Source and Target Languages. Weighted rules are learned from parallel corpora and applied to translation as the models are decoded.

Figure 20 shows Example (102) amended with trees for both languages. The example shows how creating a synchronous rule such as $S \Rightarrow < spiser, X_2, are, X_2, eating >$ can deal with the reordering and use of an auxiliary verb in this phrase. Furthermore, it can account for similar syntactic constructions, and not just that particular realization of the phrase.

While syntactically enhanced methods are competitive with, and also have outperformed phrase-based translation [Zollmann et al., 2008, Williams et al., 2015] on some translation tasks, they are still burdened with the complexity of learning rules [Cohn et al., 2010] and decoding [Sennrich, 2014].

The hierarchical phrase-based systems do not use syntactic trees with linguistic annotations as they are known from treebanks such as the Penn Treebank [Marcus et al., 1994], or indeed, classical linguistics. Syntax has also been used for Statistical Machine Translation, such as by using linguistic syntax as input or output [Neubig and Duh, 2014]. Combinations of the words *tree* and *string* denote whether syntax was used on either the source or target sides of translation, e. g., tree-to-string and string-to-tree.

3.1.4 *Hybrid Machine Translation*

If "hybrid" is understood as a combination of something heterogeneous in origin, the Hybrid Machine Translation idea presupposes a dichotomy of methods. While heated exchanges of words in early years of Statistical Machine Translation may have given rise to such a dichotomy [Jelinek, 2005, Hajič and Hajičová, 2007] between Rule-Based Machine Translation and SMT systems, elements of both are often included in either approach today. For instance, in the documentation of the `Moses` MT system [Hoang et al., 2007], rule-based preprocessing steps such as tokenization [Dridan and Oepen, 2012] are included in the walk-through of example systems. Costa-Jussà and Fonollosa [2015] separated hybrid approaches into three; (i) multi-engine approaches combining the output of various MT systems, (ii) the incorporation of linguistic information into SMT pipelines [Costa-Jussà and Farrús, 2014], and (iii) systems that combine MT architectures into single architectures.

The PRESEMT MT system [Marsi et al., 2011, Tambouratzis et al., 2012b] is an example of a Hybrid Machine Translation system which processes a small bilingual corpus for structural correspondences offline, later to be used in a translation engine. Aimed at providing translation capabilities between scarcely resourced language pairs bilingually (e. g., Greek-Italian and Norwegian-German), it was conjectured that a bilingual corpus of only a few hundred sentences is necessary to encompass the salient contrastive differences between language pairs. The parallel corpus was manually revised by linguists to account for as many structural correspondences as possible. Accompanied by much larger monolingual corpora collected from the web, only these resources were used for Machine Translation.

The method matches a Source Language sentence against a database of sentences in example-based fashion during translation, and the highest-ranking match provides a Target Language structure. This structure is filled with lemma translations provided by a dictionary, and the word order in the phrases therein is resolved by querying monolingual resources for co-occurrence information. Tambouratzis et al. [2013] reviewed the performance of PRESEMT.

3.2 LANGUAGE MODELING

A Language Model is a computerized model of *a* particular language and can be queried with strings, scoring them with numbers that can be interpreted as probabilities that the string is part of that language. Presently, this is mostly done with data-driven methods, but in principle, such models could be created with any knowledge source. As data-driven methods gained popularity in the 1980s and 1990s, this was also reflected in data-driven Language Models, then called Statistical LMs, as they were applied, initially, to Automatic Speech Recognition. When the term Language Model is used henceforth, Statistical Language Models are meant. Such models have also found use in other Natural Language Processing tasks, e. g., Information Retrieval, spell checking, and Optical Character Recognition (OCR).

In formal language theory, the members of a language have an exact definition, as they either can or cannot be generated by the grammar and vocabulary. In contrast, natural languages are moving targets, and it is impossible to ascertain which strings are valid members in the same way. When a language is modeled from a corpus, probabilities are attributed to strings according to their distributions in this training corpus. If a Language Model is queried with a set of strings, strings that were seen during training must be a part of that language according to the model. Unseen strings, may or may not be a part of the language, and a higher score should ideally be given to the strings that are. Thus, LMs must have a strategy for differentiating between such unseen events to model a language properly.

Consider the Norwegian string in Example (103).

(103)	Lisa	gikk	til	skolen
	Lisa	walked	to	school

Table 3 lists the 4! permutations of the four words, ranked after Language Model[8] score. Of these 24 permutations, only a few are grammatical and give semantical meaning. The LM can say something about how likely it is that one of these strings is a valid member of the Norwegian language. Syntactically and semantically valid strings should, therefore, rank above invalid strings.

Permutations 1,2 and 6 (as a question) are normal Norwegian utterances. Permutations 9 and 23 are possible if the two first words are interpreted as a topicalized preposition phrase. These sentences have correspondences where *Lisa* and *skolen* are interchanged, which is syntactically possible, but semantically questionable. An exception

8 Tested on a 4-gram model built on the noTenTen corpus (`https://www.sketchengine.co.uk/notenten-corpus/` (retrieved: 28th of January, 2016)).

ID	Permutation	LM score	Syntax	Semantics
1	Lisa gikk til skolen	0.92	yes	yes
2	skolen Lisa gikk til	0.46	yes	yes
3	Lisa skolen gikk til	0.0013	no	no
4	Lisa gikk skolen til	0.0002	yes	no?
5	gikk Lisa skolen til	3.4e-05	no	no
6	gikk Lisa til skolen	2.4e-05	yes	yes
7	til skolen Lisa gikk	9.2e-06	no	yes
8	gikk skolen Lisa til	7.5e-06	no	no
9	skolen gikk Lisa til	7.5e-06	yes?	yes
10	skolen til Lisa gikk	2.2e-06	no	no
11	til Lisa skolen gikk	9.8e-08	no	yes?
12	Lisa til skolen gikk	4e-08	no?	yes
13	til gikk Lisa skolen	3.3e-11	no	no
14	Lisa til gikk skolen	2.1e-11	no	no
15	gikk til Lisa skolen	8e-12	no	no
16	skolen Lisa til gikk	6e-12	no	no
17	til Lisa gikk skolen	3.4e-12	yes?	yes?
18	Lisa skolen til gikk	2.3e-12	no	no
19	gikk skolen til Lisa	2e-13	yes	yes?
20	skolen gikk til Lisa	1.3e-13	yes	yes?
21	til gikk skolen Lisa	2.3e-16	no	no
22	skolen til gikk Lisa	1.3e-16	no	no
23	til skolen gikk Lisa	6.5e-17	yes?	yes
24	gikk til skolen Lisa	5.6e-20	no	no

Table 3: LM scores for permutations in decreasing order. The columns Syntax and Semantics represent an informal assessment of their validity according to the author (native speaker). Question marks signal readings where the school did the walking.

is permutation 3, which is not syntactically valid, as the noun phrase *Lisa* is indefinite as opposed to *skolen*. While being an unrealistic example in the context of Machine Translation (i. e., that n-gram models rank utterances alone without translation probabilities), Table 3 shows how an n-gram model can capture normal utterances, but also miss other, possible but peripheral (thus, less frequent) utterances.

For polysemous words, one (or more) sense(s) should be transferred to the Target Language, out of the potential senses of the Source Language string during translation. A Language Model queried with candidate translations does not hold any information about which sense is the correct and will attribute the highest score to the most likely string according to the model. It could, however, filter out candidates that are very unlikely, e. g., if the permutations in Table 3 were translation hypotheses.

3.2.1 *Estimation of N-gram models*

N-grams, defined as contiguous sequences of n items from a given sequence of text or speech, are statistically modeled by their **counts**, i. e., their occurrences in a body of text. The counted items are normally tokens (mostly corresponding to words [Grefenstette and Tapainen, 1994]) but alternatives are individual characters, phrases, or, in principle, sentences. N-grams are discerned by their order *n*, such as **unigrams** modeling unique tokens, **bigrams** two successive tokens, **trigrams** three, and **quadgrams** four consecutive tokens, as in Example (103).

N-gram models model languages as sequences of words, where the next token in the sequence is predicted based on the relative number of times that word has followed the preceding words in the sentence. In Example (103), the conditional probability that *skolen* follows the sequence *Lisa gikk til*, i.e., $P(\text{skolen} \mid \text{Lisa gikk til})$ is estimated from data by counting relative occurrences.

Formally, $p(w_1, w_2 \cdots w_n)$ is a joint probability of n occurrences. This joint probability can also be transformed using the chain rule of probabilities, into a product of conditional probabilities $p(w_1) * p(w_2|w_1) \cdots * p(w_n|w_1, w_2 \cdots w_{n-1})$. Under the Markov Assumption, this is modeled as a Markov chain where transitions depend only on a few, previous steps. For bigram models, the probability of a word sequence depends only on the previous word.

$$p(w_1, w_2 \cdots w_n) = p(w_1) * p(w_2|w_1) \cdots * p(w_n|w_{n-1}) \tag{22}$$

Equation 22 shows how this probability is calculated given this assumption for a bigram model (sequences of two tokens).

$$p(w_1, w_2 \cdots w_n) = \prod_i p(w_i | w_{i-(n-1)} \cdots w_{i-1}) \tag{23}$$

Equation 23 shows the formula in general form (for n-grams of any order *n*).

Modeling of the conditional probabilities of n-grams models is done by counting occurrences in a text corpus. The probability of a unigram is the occurrences of that unigram divided by the total number of unigram occurrences. The probability of a bigram is the occurrences of that particular bigram (w_1 followed by w_2) divided by all bigram occurrences starting with the same history, expressed formally as:

$$p(w_2 | w_1) = \frac{\text{count}(w_1, w_2)}{\sum_w \text{count}(w_1, w)} \tag{24}$$

3.2.2 *Problems with N-gram Models*

It is well established that the Markov assumption does not hold for natural language. Consider long-distance dependencies such as the stranded prepositions found in Norwegian and English:

(104)	Mannen	skrev	jeg	etter	lang	tid	et	brev	*med*
	The-man	wrote	I	after	long	time	a	letter	*with*

where the preposition *med* relates to the noun *Mannen*. If the noun were different like the inanimate *traktoren* (*the tractor*), a different preposition would be needed for the utterance to be sensible[9] (but not grammatical). Intuitively, these long-distance dependencies cannot be captured by the previous few words. Furthermore, it is not obvious that the sentence was constructed left to right, or even sequentially. Some of the sentence structure could be chosen initially with long-distance dependencies at each end, filling the middle portion as it was formed.

Another weakness with n-gram models is the difficulty with modeling higher-order n-grams (say, an order of 25 or 30) as the models would become too large. Thus, they could not model all the relevant context for a token, even if a corpus with sufficient occurrences of all long-distance dependencies were available. Such context can contain relations such as stranded prepositions as above, sentence-end word

9 In most, but not all models of the world, such as imaginary worlds where tractors do write letters or are written letters with.

particles as seen in, e. g., German, or domain-specific word distributions.

As the n-gram order increases, the number of n-grams increases as long as n-grams of that order are represented in the data, and pruning (i. e., the systematic trimming-down of models) becomes necessary. Higher order n-grams increase the likelihood that n-grams in the test data will not be observed in training. Because the number of different word forms would be so high for morphologically rich languages, amassing training data that accounts for the relative occurrences of these all word forms is difficult. Data sparsity also weakens the ability of n-gram models to account for long-distance relationships.

Nonetheless, n-gram models remain popular. The Markov Assumption makes them easy to compute, and they scale well to large datasets. Being conceptually simple and computationally inexpensive to train are practical advantages. Training mostly consists of counting occurrences of n-grams, which can be done in parallel, although some smoothing techniques introduce added complexity. The resulting n-gram model is a lookup table of (log)-probabilities for each n-gram.

3.2.3 *Handling Unseen Events*

What is the probability of a random person uttering *jibberish*? Alternatively, something that really is, such as *blarghsnarksnackishly*, or perhaps the famous sentence:

> "Colorless green ideas sleep furiously" [Chomsky, 1956]

This sentence was constructed to demonstrate that deriving meaning also from syntactically valid phrases can be difficult. Ironically, while true at its inception, now, this is a well-known reference to a famous piece of literature, which is uttered all the time at universities across the world — with an intended meaning.

The true probability of new words is unknown, beyond the grasp of current models. Still, by virtue of the productivity and evolution of language, novel utterances are created all the time. Also, Language Models trained on finite resources will contain many strings that are neither syntactically nor semantically valid.

Since Equation 23 is the product of several multiplications depending on the length of the string, if any factor is zero, the product is zero. Thus, the probability of such a word sequence would also be zero, which could make little sense. An unseen event in any n-gram would render a zero probability, and the model would fail at its task. Such a model would not be useful for Natural Language Processing tasks unless the vocabulary of the queries was strictly controlled.

Some of the probability mass must be transferred from seen to unseen events in the training corpus to avoid zero probabilities, and unseen events are expected to occur during the application of a Language Model. **Back-off**, **smoothing**, and **interpolation** (and combinations thereof) are methods that address this task. A back-off method would back off, literally, to lower-order n-grams if they were not found in the model at full length, e. g., if the quadgram *skolen | Lisa gikk til* was not found, the method would recursively back off to the trigram *skolen | gikk til* (and further on to bi- or unigrams if needed).

Smoothing is a statistical method, where the Probability Mass Function is smoothed to assign som probability mass to unseen events. Because the sum of the probability of all possible events must be 1, the probability mass must be reduced accordingly for the tokens observed 1 or more times to remain a probability distribution.

Interpolation refers to a weighted (e. g., linear or estimated by a held-out corpus) average of n-gram probabilities of different orders, and can be applied to all or only the unseen n-grams.

Chen et al. [1998] empirically evaluated many different varieties of back-off and smoothing techniques.

3.2.3.1 *Katz Back-off*

A much-used implementation is Katz back-off [Katz, 1987].

$$
\begin{aligned}
&P_{bo}(w_i \mid w_{i-n+1} \cdots w_{i-1}) \\
&= \begin{cases} d_{w_{i-n+1} \cdots w_i} \dfrac{c(w_{i-n+1} \cdots w_{i-1} w_i)}{c(w_{i-n+1} \cdots w_{i-1})} & \text{if } c(w_{i-n+1} \cdots w_i) > k \\ \alpha_{w_{i-n+1} \cdots w_{i-1}} P_{bo}(w_i \mid w_{i-n+2} \cdots w_{i-1}) & \text{if } c(w_{i-n+1} \cdots w_i) \leqslant k \end{cases}
\end{aligned} \tag{25}
$$

Equation 25 shows how the back-off probability of an n-gram is derived. If the n-gram count *c* is above *k* (normally 0), the probability is calculated as its count divided by its full history, then discounted by a parameter *d*, to have some probability mass to send to the lower-order models. If it is not, the model will instead use the probability of the n-gram with a history of one less word. This estimate is normalized with a parameter, α, such that only the probability left over from the discounting is used for the n-grams that are estimated by backing off.

3.2.3.2 *Absolute Discounting Interpolation*

The easiest way of doing smoothing is to add counts to all unobserved n-grams, and discounting the observed accordingly. Another simple method is absolute discounting interpolation, which subtracts probability mass from higher-order n-grams by subtracting a constant from their counts and adding probability mass to lower order n-grams.

$$P_{abs}(w_i|w_{i-1}) = \frac{\max(c(w_{i-1}w_i) - \delta, 0)}{\sum_{w'} c(w_{i-1}w')} + \alpha p_{abs}(w_i) \tag{26}$$

Equation 26 formulates absolute discounting interpolation for bigrams. The parameter δ, by which counts are discounted, can be estimated, e. g., by a held-out dataset. α is a normalization term.

3.2.3.3 *Kneser-Ney Smoothing*

Kneser-Ney smoothing [Kneser and Ney, 1995] performed best in the experiments by Chen et al. [1998] and still is considered a state-of-the-art smoothing method. This method is based on **continuation probabilities**. For the bigram case, this means the count of bigrams that each word completes as its continuation. A much-used example is "San Francisco" because the word *Francisco* is unlikely to appear in many other bigrams than this (with exceptions, e. g., religious texts and texts about Spanish names). While absolute discounting would give a high probability of *Francisco* because of frequency, the Kneser-Ney method would lower it, as its ratio to the number of possible n-grams is low.

$$\begin{aligned} P_{KN}(w_i|w_{i-1}) = & \frac{\max(c(w_{i-1}w_i) - \delta, 0)}{\sum_{w'} c(w_{i-1}w')} \\ & + \alpha \frac{|\{w_{i-1} : c(w_{i-1}, w_i) > 0\}|}{|\{w_{j-1} : c(w_{j-1}, w_j) > 0\}|} \end{aligned} \tag{27}$$

Equation 26 formulates Kneser-Ney smoothing for bigrams. *j* is a free term in the denominator of the right-most term, which consists of the number of possible n-grams. The terms α and δ are the same as for absolute discounting.

Brants et al. [2007] argued that the importance of refined smoothing techniques diminishes as the training material increases, and introduced a simple back-off scheme, nearing the performance of the Kneser-Ney method at a lower computational cost.

3.2.4 *Alternative Models of Language*

Several ideas have been put forward to address the above problems with n-gram models. An example is exponential models, such as Maximum Entropy (logistic regression) modeling presented in [Rosenfeld, 2000], which model the conditional probability of one word being the next in a sequence, given the history. An arbitrary number of features f are combined with the formula:

$$\frac{1}{Z(h)} * \exp\left[\sum_i \lambda_i f_i(h, w)\right]$$

where Z is a normalization term, and λ is the weight vector. In this framework, language is modeled as a sequence of words. The models can include features that account for long distance and domain specificity. Such feature-based representations can also be used for whole-sentence models, with which the probabilities of whole sentences are modeled based on the same feature-based approach [Rosenfeld et al., 2001].

Class-based models first address sparsity by separating words into classes and then modeling the sequences of class patterns, similar to how n-grams are modeled, often interpolated with n-gram models. Brown et al. [1992] separated words into classes based on the minimal loss of Mutual Information, an information-theoretic measure of the association between variables. If variables are independent, Mutual Information is zero. Finding words classes also can be achieved by clustering [Goodman, 2001].

Using a cache — a dynamic window of words — to train a specialized model for interpolation with a larger, static n-gram model has been used to enable Language Models to account for phenomena such as that the probability of seeing rare words increases if seen in recent history [Louis and Webber, 2014].

Other methods for modeling language include decision tree models [Rosenfeld, 2000], where a sequence of binary questions model the history of each word. However, the storage space of such histories will grow large as the vocabulary increases, which poses practical problems. Parsing text with formal grammars (e. g., context-free or link grammars [Sleator and Temperley, 1993]) for inclusion in **structured** Language Models has also been done, e. g., Chelba and Jelinek [2000], facing many of the same problems with complexity. Since then, Neural Network Language Models (NNLMs) were introduced [Bengio et al., 2003] (see Section 3.3).

3.2.5 *Evaluation of Language Models*

How useful a particular model of a language is, ultimately depends on its intended use, which can only be evaluated in an end-to-end setting with a Language Model embedded, i. e., evaluated **extrinsically**. However, extrinsic evaluation is often resource-demanding and not possible to rerun for every adjustment of an LM.

Language Models can also be evaluated **intrinsically**, which is a lighter process that offers an approximation of how the evaluated differences ultimately impact a given application. Such intrinsic evaluation can be done using metrics relating to information entropy, expressing the surprisal, or **perplexity** of an LM when seeing some test data. While this approximation can be useful, it is not always the case that the text it is tested on is distributed equally to the training corpus. Moreover, Mikolov [2012] argued that reporting Language Model improvements as perplexity reduction can obscure results because constant *relative* reductions in perplexity will not result in the same reduction of cross-entropy (explained below).

3.2.5.1 *Information Entropy*

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_b p(x) \tag{28}$$

Equation 28 defines Information Entropy, a measurement of uncertainty in a random variable from Information Theory. Borrowed from thermodynamics where it pertains to how evenly energy is distributed in a system, it is used to capture how evenly distributed a probability mass is.

A distribution where all events are equiprobable will have the highest entropy because the unpredictability of the information content is the highest. Low probabilities give low values in the $\log_b p(x)$ term in Equation 28 as they are canceled out by the $p(x)$ term contributing less to lowering the sum, thereby increasing the Entropy as the sum is negated. Correspondingly, the very high probabilities would give $\log_b$ terms very close to zero, and the contribution towards Entropy would be small. Jelinek [1997, pp. 121-126] discussed the mathematical properties of Entropy more rigorously.

When modeling language as a set of word sequences, the entropy *H* of a variable ranging over all these sequences is:

$$H(w_1, w_2 \cdots w_n) = -\sum_{W_1^n \in \mathcal{L}} p(W_1^n) \log_b p(W_1^n) \tag{29}$$

where n is the maximum length of the sequence W in the language L.

3.2.5.2 *Entropy Rate*

Dividing Equation 29 by n arrives at the **entropy rate** or entropy per word of such a *sequence*. The Entropy rate is the limit of the per-word entropy, as n goes to ∞:

$$\begin{aligned} H(L) &= -\lim_{n\to\infty}\frac{1}{n}H(w_1, w_2\cdots w_n) \\ &= -\lim_{n\to\infty}\frac{1}{n}\sum_{W_1^n\in L} p(w_1, w_2\cdots w_n)\log p(w_1, w_2\cdots w_n) \end{aligned} \tag{30}$$

Through simplifying assumptions, e. g., that the language is a **stationary** and **ergodic** process, the Shannon-McMillan-Breiman Theorem [Algoet and Cover, 1988, Cover and Thomas, 1991] states that the Entropy rate is:

$$H(L) = -\lim_{n\to\infty}\frac{1}{n}\log p(w_1, w_2\cdots w_n) \tag{31}$$

A statistical process is ergodic if its statistical properties can be deduced from a large enough random sample, meaning that it does not change erratically. It is stationary if the statistical properties do not change over time. This assumption does not hold for natural language, as the probability of upcoming words can depend on arbitrarily time distant events [Jurafsky and Martin, 2017].

3.2.5.3 *Cross-Entropy*

$$H(p, q) = -\sum_{x\in\mathcal{X}} p(x)\log_b q(x) \tag{32}$$

Equation 32 defines the cross-entropy between two probability distributions p and q over the same events with base b. It is a measure of the number of bits needed to identify an event from the set if optimized for an approximate distribution q, rather than the true distribution, p. Cross-entropy will find its minimum when the distributions p and q are equal.

Similarly, the cross-entropy per word is derived by:

$$H(p, q) = -\lim_{n\to\infty}\frac{1}{n}\sum_{W_1^n\in L} p(w_1, w_2\cdots w_n)\log q(w_1, w_2\cdots w_n) \tag{33}$$

and by the Shannon-McMillan-Breiman theorem, this equals:

$$H(p, q) = -\lim_{n\to\infty}\frac{1}{n}\log q(w_1, w_2\cdots w_n) \tag{34}$$

The true entropy is a lower bound on the approximate entropy, i. e.:

$$H(p) \leqslant H(p, q) \tag{35}$$

Equation 35 implies that the cross-entropy is a good measure of how the approximate distribution q (e. g., a Language Model) fits the true distribution p (e. g., the true entropy of a language) as it will be lower the better the approximation is.

3.2.5.4 *Derivation of Perplexity*

Perplexity has origins in Information Theory, mainly related to the concept of Information Entropy [Shannon, 1948]. Crooks [2016] defined perplexity as the exponentiation of the cross-entropy with the base used to derive it. With base 2, perplexity is 2 raised to the cross-entropy:

$$PP = 2^{H(P_{LM})} \tag{36}$$

where $H(P_{LM})$ is the cross-entropy of the Language Model. Raising 2 to the average amount of bits necessary to encode the words, can be interpreted as the number of choices available at each point. This corresponds to the 7-bit American Standard Code for Information Interchange (ASCII) character set having $2^7 = 128$ options at each choice point. When normalizing perplexity by the number of words (tokens), perplexity per word is reported, making figures smaller and easier to handle, as well as having intuitive appeal.

The cross-entropy between the true probability distribution p and the approximate distribution q from Equation 34 is then estimated for a sequence of words $W = w_1 w_2 \cdots w_n$ with a fixed length n:

$$H(W) = -\frac{1}{n} \log(q(w_1 w_2 \cdots w_n)) \tag{37}$$

Substituting the right side of Equation 36 with the formula from Equation 37 and replacing the approximation q with a Language Model probability P_{LM} gives the following:

$$PP(W) = 2^{-\frac{1}{n}\log(P_{LM}(w_1 w_2 \cdots w_n))} = \left[\frac{1}{P_{LM}(w_1 w_2 \cdots w_n)}\right]^{\frac{1}{n}} \tag{38}$$

Thus, perplexity can also be expressed as the geometric average of the inverse of the conditional word probabilities, when the formula for Language Model probabilities in Equation 23 is used:

$$PP(W) = \sqrt[n]{\prod_{i=1}^{n} (\frac{1}{P_{LM}(w_i | w_{i-(n-1)} \cdots w_{i-1})}} \tag{39}$$

3.3 ARTIFICIAL NEURAL NETWORKS AND MT

Feed-forward Artificial Neural Networks are universal function approximators, and recurrent ANNs are universal approximators for discrete sequences [Goodfellow et al., 2016, Chapters 6 and 10]. Thus, ANNs can be used to predict an output from an input, with or without regard to their biological inspiration and plausibility. Consequently, ANNs may be used in place of other Machine Learning methods for predictions, integrated into Statistical Machine Translation frameworks or for the entire modeling of translation. When ANNs are used for the entire translation process, the process is referred to as Neural Machine Translation.

The next subsection will present the use of Artificial Neural Networks for constructing Language Models, followed by a subsection on applications of ANNs to Machine Translation.

3.3.1 *Artificial Neural Networks and Language Models*

Section 3.2 mentioned Neural Network Language Models, an early application of ANNs in NLP and Machine Translation based on feed-forward networks [Bengio et al., 2003, Schwenk, 2007]. As output, such Language Models compute a probability for every word in the vocabulary given the input using the Softmax function:

$$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^T \mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^T \mathbf{w}_k}} \tag{40}$$

which gives a probability for the output class y (word) being the class j for vocabulary $j = 1, \ldots, K$. Hence, these models also rely on the Markov Assumption (see Section 3.2.1).

The computation of the Softmax is expensive, which creates problems for large vocabularies. Vocabularies can be trimmed by automatic methods, such as regarding more words as one token if they are fixed expressions or given names.

3.3.1.1 *Recurrent Neural Network Language Models and Embedded Features*

Later, other types of Artificial Neural Networks have been used for language modeling, such as Recurrent Neural Networks (RNNs) [Elman, 1990, Mikolov et al., 2010, Mikolov, 2012], a by-product of which has received much attention — Word Embeddings. Such models update the recurrent states one word at the time, and thereby do not rest on the Markov Assumption, conditioning predictions on arbitrarily long context.

Word Embeddings represent discrete words in continuous space, with easy-to-handle vectors in a lower dimension than the vocabulary size. The embeddings are derived from an intermediate representation of the discrete word occurrences, by projecting them onto a lower dimensional space, implemented as another hidden layer with the same weights for all words. Hence, these embeddings have computational advantages (e. g., caching) and can also be used to measure word similarities in applications outside language modeling and Machine Translation [Mikolov et al., 2013b, Goldberg and Levy, 2014].

Mikolov et al. [2013c] demonstrated that these vector representations encompass both syntactic and semantic (demonstrated on a SemEval 2012 test set) regularities, such as the vector subtractions of $x_{apple} - x_{apples} \approx x_{car} - x_{cars}$. Furthermore, Mikolov et al. [2013a] found that the similarities are partially consistent also across languages, which could be used for Machine Translation purposes, e. g., by augmenting dictionaries and phrase tables.

3.3.2 *Neural Machine Translation Applications*

Different versions of Recurrent Neural Networks are used for sequence modeling, such that weights in an Artificial Neural Network layer are updated depending not only on the input layer but also on values from the current layer. Thus, information from used training material is retained as more is processed.

The models use an **encoder-decoder** architecture where a Recurrent Neural Network encodes the information in the Source Language string, such that last hidden state in the encoder encompasses information about the entire input string as a real-valued vector. Next, another RNN is trained to predict (decode) the Target Language string based on that vector. The output layer is interpreted as a probability distribution over the words in the vocabulary. Methods such as **random sampling** or **beam search** can be used to produce the TL string.

This idea for modeling translation goes back the 1980s, e. g., Allen [1987] who conducted translation experiments with as little as 30-40 words. Hardware constraints made experiments with data large enough for real-world applications difficult. Decades later, advances such as the ability of Graphics Processing Units (GPUs) to efficiently do matrix calculations, made increasingly large models feasible, and in 2013 Kalchbrenner and Blunsom [2013] presented a purely neural Machine Translation system.

The following year, Neural Machine Translation systems were competitive also for larger-scale tasks, and by 2016 they were outperforming Rule-Based- and Statistical Machine Translation on most tasks [Bojar et al., 2016]. Moreover, commercial systems such as SYSTRAN's pure NMT system were launched the same year [Crego et al., 2016].

Britz et al. [2017] experimented with the hyperparameters of Neural Machine Translation systems, using 250,000 GPU hours on exploring embedding sizes, Recurrent Neural Network variants, and beam sizes during decoding.

3.3.2.1 *Neural vs. Statistical Machine Translation*

While Neural Machine Translation is widely considered to be state-of-the-art, the picture is nuanced. Still, the vocabulary sizes of NMT systems are limited (up to 50,000 tokens +/- 10%), which could impact performance when large datasets are available for training. Also, for the smaller translation tasks used, e. g., in shared tasks, the picture is nuanced.

Toral and Sánchez-Cartagena [2017] made a systematic comparison of the outputs of the Workshop on Statistical Machine Translation 2016 tasks and found that Neural Machine Translation outperformed Statistical Machine Translation for 7 of 9 language pairs. They summarized their findings as follows:

- NMT output was considerably more varied.
- NMT output was more fluent.
- NMT systems introduced more changes to word order (except a hierarchical SMT system).
- NMT performed better for most sentence lengths unless sentences were very long (over a threshold).

These findings mostly corroborated a case study by Bentivogli et al. [2016]. Additionally, Farajian et al. [2017] found that Statistical Machine Translation outperformed Neural systems in multi-domain scenarios (see Subsection 4.5.5).

3.4 MACHINE TRANSLATION EVALUATION

Different aspects like purpose, sender, recipient, style, and information content must be taken into account to provide a good translation, which makes professional translation resource- and knowledge intensive. Disagreement on the *correct* translation of a given input can also arise between highly skilled translators. Similarly, as the recipient of translations of cultural items, one might also object to translations. Likewise, human evaluation of Machine Translation performance varies among annotators, which can be quantified by the

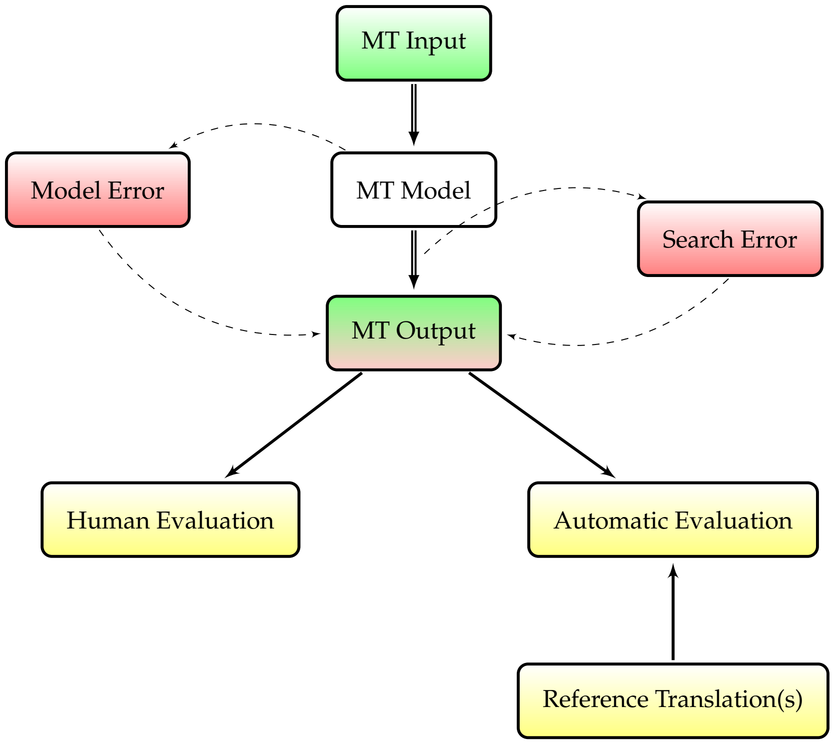


Figure 21: Summary of Machine Translation evaluation.

level of **inter-annotator** agreement. However, to automatically evaluate MT output, it is necessary to have a specific notion of what a correct translation *is*.

Figure 21 highlights two aspects of Machine Translation Evaluation: the source of error and the methods that assess errors. Finding the source of errors, i. e., error analysis, is instrumental in improving systems. Furthermore, an actual assessment and a subsequent ranking of output are necessary to account for improvements in quality after system modifications.

3.4.1 *Model Error vs. Search Error*

Model error is the translation error of the best translation found in the model and **search error** is the failure to find the best translation in the model [Germann et al., 2001]. Heuristics such as pruning are necessary for decoding Statistical Machine Translation models. Hence, better translations could, in principle, be present in the model with a better decoder. Post and Gildea [2008] did experiments on selected sentences with few words to cut down on pruning, reducing search error and isolating the model error.

Auli et al. [2009] refined model error with the introduction of **induction errors**, i. e., the possibility of the rule set of the Statistical Machine Translation to find the/a correct translation, in the context of log-linear models. By restricting model error to the parameterization of the models and the search errors to the heuristics (the approximations in search), the induction errors pertain to the actual rules being weighted. If a Target Language string is not present in the rule tables of the model, it is impossible to find this translation even with optimal weights and search algorithms. Comparing phrase-based and hierarchical models, Auli et al. found that the main difference between the models stemmed from parameterization.

3.4.2 *Assessment of Machine Translation Output*

Evaluation of Machine Translation is a contentious and much-debated issue. Broadly, MT output is evaluated either by human annotators directly or by automatic metrics based on reference translations. A reference translation is a translation against which candidate translations are measured. Such references are mostly created by humans, but could also come from another MT system.

There is no consensus on which metrics correlate the best with human assessment, and a wide variety of measures exist. Tuning the parameters of Statistical Machine Translation systems to the same metrics that later will be used to evaluate them raises the question of whether said criteria fairly rank them against Rule-Based Machine Translation systems not tuned to these metrics. Some evidence suggests that the widely-used automatic metric BLEU (defined below) is more favorable to statistical than rule-based systems, e. g., [Koehn and Monz, 2006, Callison-Burch et al., 2006]. Koehn [2010, Chapter 8] presented evidence for both camps and other principled points from this debate.

Human evaluation of Machine Translation output usually involves more evaluators, who rank systems after a set of criteria. The two dimensions **adequacy** and **fluency** were established as the primary means of human evaluation in MT by the DARPA Machine Translation initiatives in the 1990s [Graham et al., 2012]. These dimensions were also used by the Linguistic Data Consortium when creating human evaluated datasets, expanded with detailed guidelines on how to use them for ranking [Ma and Cieri, 2006]. Adequacy captures whether the meaning in the Source Language is properly carried across and preserved in the Target Language. Fluency measures whether the output is fluent and proper language, which flows freely in the TL with well-formed sentences.

Translation is a science in its own right, for which training can take years. Thus, the supply of professional translators, especially for rare language pairs is limited and using such to evaluate output every time a Machine Translation system is run is practically impossible. Using human evaluators, with or without formal training, is expensive nonetheless, whereas automatic evaluation metrics output results almost instantly.

Automatic evaluation metrics require one or more reference translations and score Machine Translation output on the difference to these references. Some metrics do alterations of the output before reference comparison, e. g., by expanding synonyms. In principle, any distance metric between output and reference sentences can be used to provide a score, which makes the correlation between automatic metrics and human evaluation important.

Much work has gone into researching the various automatic metrics' correlation with human evaluation. Machine Translation evaluation has been reviewed frequently, e. g., Euromatrix [2007], Lopez [2008], and Koehn [2010]. Song et al. [2013] deconstructed the BLEU measure and experimented with changing parts of it, such as weighting arithmetically instead. They reported better correlation with human evaluation using arithmetic means.

Custom measures have been created for some language pairs in addition to work on what evaluation metrics best provide information about specific aspects of Machine Translation. Irvine et al. [2013] proposed new measures for MT errors for statistical systems moving into a new domain. These measures rely on word alignments, on which errors are marked as either:

1. Seen. An attempt to translate a source word unseen to the model.

2. Sense. An attempt to translate a word in a sense unseen to the model.

3. Score. A translation the model could have successfully produced but did not because an incorrect translation hypothesis scored higher.

4. Search. Errors due to approximations during searching.

Crowdsourcing leverages, e. g., services and content from a large number of mostly online contributors. Because it is a cost-effective form of human evaluation, it has become an option for MT Evaluation [Zaidan and Callison-Burch, 2011].

3.4.3 *Automatic Metrics*

This subsection will present a selection of automatic evaluation metrics often used in Machine Translation evaluation. Automatic metrics have the advantage that they produce results quickly.

3.4.3.1 *BLEU*

BLEU [Papineni et al., 2002] is a highly influential and widely used automatic metric, motivated by a need for quick and frequent translations. Fundamentally precision-based, the metric counts the n-gram overlaps between candidate and reference translations. The method is also augmented with a **brevity penalty** to penalize too short translation candidates and **count-clipping**. Count-clipping reduces the number of correct n-grams in a sentence to the maximal number of occurrences in any reference translation to avoid translations with only one repeated word getting perfect precision. N-gram precisions up to N (normally set to 4) are averaged geometrically.

$$\mathrm{BLEU} = \mathrm{BP} * \sqrt[N]{\prod_{n=1}^{N} p_n} \tag{41}$$

$$\mathrm{BP} = \begin{cases} 1 & \text{if } |c| > |r| \\ \exp(1 - \frac{|r|}{|c|}) & \text{if} |c| \leqslant |r| \end{cases} \tag{42}$$

$$p_n = \frac{\sum_{C \in Candidates} \sum_{\text{ngram} \in C} Count_{\text{clip}}(\text{ngram})}{\sum_{C' \in Candidates} \sum_{\text{ngram}' \in C} Count(\text{ngram}')} \tag{43}$$

Equations 41-43 show the derivation of the BLEU metric.

3.4.3.2 *NIST*

Doddington [2002] launched NIST, another n-gram-based metric based on n-gram co-occurrences (present in both candidate and reference translations), and suggested two variations on BLEU scoring. First, since n-grams of higher orders co-occur less frequently, the geometrically averaged BLEU metric has the potential of counterproductive variance and should be replaced, e. g., by an arithmetic average. Second, since n-grams that occur less frequently might be more *infor-*

mative and are more difficult to anticipate, lower-frequency n-gram co-occurrences should get increased weight.

$$\begin{aligned} \text{Score} = \sum_{n=1}^{N} \left\{ \frac{\sum_{wc} \mathtt{Info}(w_1 w_2 \cdots w_1)}{\sum_{wo} (1)} \right\} \\ \cdot \exp\left\{ \beta \log^2 \left[\min\left(\frac{L_{sys}}{L_{ref}}, 1 \right) \right] \right\} \end{aligned} \tag{44}$$

$$\mathtt{Info}(w_1 \cdots w_n) = \log_2 \left(\frac{\mathtt{count}(w_1 w_2 \cdots w_{n-1})}{\mathtt{count}(w_1 w_2 \cdots w_n)} \right) \tag{45}$$

Equations 44 and 45 show the weighting formulas. wc and wo refer to the string that co-occurs in the string length up to *n* in the set of reference translations and the count of the same string in the system output. L_{sys} and L_{ref} refer to the average number of words in the system output and the average number of words in the reference translations, respectively.

3.4.3.3 *Meteor*

`Meteor` [Lavie and Denkowski, 2009] included more concepts in evaluation while maintaining speed and ease of use. Computing word alignments between candidate and reference translations and counting matches on aligned words are fundamental to the metric. When more references are available, the highest scoring is used for matching.

$$F_{mean} = \frac{P \cdot R}{\alpha \cdot P + (1 - \alpha) \cdot R} \tag{46}$$

$$\text{Penalty} = \gamma \cdot \left(\frac{\#\text{ chunks}}{\#\text{unigram matches}} \right)^{\beta} \tag{47}$$

$$\text{Meteor} = (1 - \text{Penalty}) \cdot F_{mean} \tag{48}$$

Equation 46 shows how a parameterized harmonic mean of precision and recall of unigram matches is computed, optionally accepting synonyms as correct based on WordNet synsets [Miller, 1995]. Equation 47 introduces a **fragmentation** metric, designed to penalize systems outputting several chunks of contiguous and identically ordered words. Equation 48 combines the two as a final score. Chunks refer to consecutive unigrams in the system output that are also consecutive in the reference, and the number of unigram matches is the raw count of said unigrams.

3.4.3.4 *Word Error Rate*

Word Error Rate (WER) is a distance metric first used in Automatic Speech Recognition, but also used in Machine Translation.

$$\text{Word Error Rate} = \frac{\text{insertions} + \text{deletions} + \text{substitutions}}{\text{reference-length}} \tag{49}$$

Equation 49 shows this formally. This score, computing the Levenshtein distance [Levenshtein, 1966] for words, not characters, can be efficiently computed with dynamic programming.

3.4.3.5 *Translation Edit Rate*

Translation Edit Rate (TER) [Snover et al., 2006] is defined as the minimum amount of edits necessary to morph the system output into the reference translations.

$$\text{Translation Edit Rate} = \frac{\text{\# Edits}}{\text{Average \# of reference words}} \tag{50}$$

Equation 50 shows this formally. Edits include insertions, deletions, substitutions (as in WER), and crucially shifts, i. e., movements of words and phrases, making the calculation NP-complete [Shapira and Storer, 2007]. The first three types of edits are computed with dynamic programming, whereas a greedy search is done for the shifts that subsequently determine these edit distances. Snover et al. [2008] published a refined version of TER, incorporating stem matches, synonyms, and phrase substitutions.

3.4.4 *Validity and Significance of Results*

Reported differences in automatic metrics are often small. These differences could be caused by random error, e. g., as a consequence of how the test data was sampled. Some methods have been developed to assess the validity of such small differences between Machine Translation systems.

Riezler and Maxwell [2005] cautioned about the multiplicity problem in significance testing of Machine Translation systems. As the number of pairwise comparisons grows, the chance of any of them falsely rejecting the null hypothesis (no difference) H_0 increases. The risk p_e of such an error occurring in *k* pairwise comparisons, depending on the risk of falsely rejecting the H_0 in a given experiment, is:

$$p_e = 1 - (1 - p_c)^k \tag{51}$$

For a p-level of 0.05, a 50% chance of falsely rejecting H_0 occurs already at 14 comparisons.

Clark et al. [2011] studied the impact of parameter optimization methods and their impact on results. The authors claimed that instability in the optimizer could significantly affect results and experiments should, therefore, be rerun such that spurious differences (or non-differences) between systems attributed to this instability can be uncovered.

Berg-Kirkpatrick et al. [2012] empirically investigated significance testing in Natural Language Processing on some tasks, e. g., Machine Translation. While the authors strongly advocated the use of significance testing based on their findings, their experiments on the WMT 2010 data suggested that the rule-of-thumb BLEU score differences (0.5) deemed as significant were not far from their empirical results. When plotting differences in BLEU score against confidence levels $(1 - p)$, Berg-Kirkpatrick et al. found that the 0.05 threshold was met with 0.28 BLEU points for same-system scores and 0.37 for different systems, respectively.

3.4.5 *Bootstrap Resampling*

Koehn [2004] proposed a method using bootstrap sampling, a method whereby a test set is resampled n times, *with* replacement. The idea behind the bootstrap method [Efron and Tibshirani, 1993] is to use observed data, a sample, to construct an estimated population distribution, which in turn is used to figure out a statistic of interest. For Machine Translation evaluation, this statistic is often the difference between systems, which may or may not be present in the population that was sampled for evaluation.

As the number of samples *n* increases, the estimate of the significance level will improve. However, there is no analytical way to determine an optimal, or required, *n*, but it must be reasonably large (sample sizes in the thousands) to avoid Monte Carlo errors. A high number is rarely a problem, as average computers unproblematically calculate evaluation metrics with bootstrap samples in the tens of thousands, which is considered sufficient [Clark et al., 2011].

The bootstrap samples can be used for hypothesis testing. When assessing the difference between two systems, the hypothesis H_0 (no difference) can be rejected if the difference is present in sufficiently many of the bootstrap samples.

Of the existing randomized significance tests, the simplest is the one-sided bootstrap resampling test. If the system hypothesized as better performs worse for a sample, a counter is incremented. The p-value is then determined by the quotient of the counter divided by the number of bootstrap samples.

Formally, for B bootstrap samples with mean τ_B, the test statistic is:

$$S_{X_b} - S_{Y_b} - \tau_B \geqslant S_X - S_Y \tag{52}$$

The absolute value is used for the two-sided test:

$$|S_{X_b} - S_{Y_b} - \tau_B| \geqslant |S_X - S_Y| \tag{53}$$

where S_{X_b} and S_{Y_b} represent the scores on the individual bootstrap samples and S_X and S_Y the scores on the actual data, respectively.

3.4.5.1 *Approximate Randomization for p-values*

Approximate Randomization [Noreen, 1988] reshuffles sentences between the compared systems. If sentences are randomly exchanged between the (result of) using two Machine Translation systems, the score will be equal if the systems are equal. It differs from bootstrap sampling regarding replacement as shuffling is done without. Otherwise, the score of the reshuffled data is compared to the actual score differences:

$$S_{X_r} - S_{Y_r} \geqslant S_X - S_Y \tag{54}$$

Because Machine Translation evaluation metrics are not, most often, comparable across datasets, samples are *stratified* in that only translations of the same sentences are exchanged. **Stratification** is understood as dividing samples into homogeneous groups before sampling, strata here being the translations of the unique Source Language sentences.

Graham et al. [2014] compared paired bootstrap sampling, bootstrap resampling, and Approximate Randomization to human evaluation and found that all methods reached the same conclusions.

4 DOMAIN ADAPTATION AND MACHINE TRANSLATION

This final background chapter will present Domain Adaptation (DA) in a Machine Translation context. First, Section 4.1 discusses some definitions, and then Section 4.2 addresses the linguistic background of domains before Section 4.3 thematizes statistical modeling. Section 4.4 is aimed at categorizing Domain Adaptation approaches, and Section 4.5 presents relevant applications.

Chapter 1 stated that Domain Adaptation of Machine Translation systems pertains to the adaptation of a system either built for general purpose or some other domain, to a **target** domain. Thus, DA is also relevant to many other Natural Language Processing and Artificial Intelligence tasks.

Whether rules are hand-written or statistically learned, to **adapt** a system to new input, these rules must be changed from their initial state to another state that performs better on the target domain. Consequently, Domain Adaptation is both a theoretical problem for statistical learning, as well as a practical problem for application areas like Machine Translation.

As the focus of Machine Translation gradually shifted from rule-based to data-driven (and later hybrid) approaches in the wake of the data-driven IBM models [Brown et al., 1990, 1993] (see Chapter 3), Domain Adaptation received more attention as a particular problem. Since Rule-Based Machine Translation systems are better at precision but worse at coverage (recall) in general, it follows that they are adapted to the particular types of text they translate well. In practice, DA of RBMT systems has been done by manual encoding of domain-specific terminology [Wolf and Bernardi, 2013]. Lagarda et al. [2009] and Wolf and Bernardi argued that Statistical Machine Translation systems are more sensitive to out-domain input than rule-based systems. Training material is a scarce resource for data-driven approaches; often the available data is not particular to a domain at all, let alone the domain in demand. Thus, leveraging the little in-domain data available as effectively as possible is essential, either by altering the models directly or acquiring more in-domain training material based on the available corpora.

Machine Translation systems consist of a cascade of pre-processing, processing, and post-processing steps, each of which can be adapted to a particular domain and evaluated either in unison (intrinsically) or plenum (extrinsically). Hence, Domain Adaptation applies to the different, potentially nested, **levels** of this cascade. Gambäck and Bungum [2016] divided DA at any such level into three steps for dialog agents, also a complex problem with many subparts. Analogously, a similar distinction applies to MT:

1. **Identification** of the structures at a given level.
2. **Mapping** of these structures to a set of distinct domains.
3. **Leveraging** this domain knowledge to the task at hand.

4.1 DEFINITIONS AND PROBLEM FORMULATIONS

In Psychology, a domain pertains to the way knowledge is stored in the human mind. As in Natural Language Processing, there is no sharp definition of what domains are; it is rather an assumption that it is common knowledge, also known under names such as **domain-specific knowledge**, **subject-matter-knowledge**, and **content-specific knowledge** [Alexander, 1992].

In Mathematics, the domain of a function is the set of argument values, i. e., the set of legal inputs, for which the function is defined.

Chapter 1 introduced Domain Adaptation of Machine Translation as the adaptation of an MT system created for one domain (or a general system) for use in another domain. This definition will be expanded upon below, both concerning the meaning of **domain** in general, and how it is understood for DA in MT in particular. Some of the analyses referenced below will have a broader scope than MT, i. e., apply to more Natural Language Processing applications.

> "Since there is no accepted, clear definition of what a *domain* is with regard to Machine Translation, there is no accepted definition of the term Domain Adaptation either."
> Carpuat [2014]

This statement by Carpuat is typical, and also reiterated by Cuong and Sima'an [2017] who considered the definition question open. Often, a *notion* of domains is referenced without specific criteria for how they are separated, other than that a text corpus is attributed to some domain, such as news, medical text or subtitles. While consensus on a precise definition is missing, this does not entail that the existing definitions are conflicting. Despite being hard to define and delimit, domains do exist, much like other Artificial Intelligence terms like **intelligence** and **creativity**.

Mahajan et al. [1999] used the terms **domain** and **topic** interchangeably and made the following observation:

> "Topic of discussion is a dynamic concept. It can change over time within the same document and new topics of discussion can also get created with new developments."

From a Machine Learning perspective, Jiang [2008] defined Domain Adaptation as:

> "exploiting labeled data from the source domain to help train classifiers in the target domain..."

in an analysis of Domain Adaptation in Natural Language Processing. In that study, the experimental tasks were formulated as classification problems. Furthermore, Jiang confronted a major point observing (author's emphasis) the following:

> "Although the focus on domain adaptation in natural language processing in this thesis, most of the analysis of the problem and the proposed domain adaptation techniques are not restricted to natural language processing problems but *can be generally applied to most classification tasks when the training and the test domains differ*."

From a statistical learning point of view on Domain Adaptation, the findings are relevant to all such tasks. Fundamentally, a statistically learned model assumes that the evaluation data on is distributed identically to the training data. Practically, this is never the case, which gave rise to the questions studied by Jiang.

Blitzer [2008] emphasized the data sparsity of target domains in the definition:

> "Domain adaptation methods provide a way to alleviate the problem of creating training sets for different domains by generalizing models from a resource-rich **source** domain to a different resource-poor **target** domain."

without going into the specifics of what domains are, other than different probability distributions. The focus was on how sparse in-domain training data is a practical problem that is relevant for Machine Translation.

Plank [2011] defined a domain as a hypernym over any types of text variability, such as topic, genre, style, medium, and vocabulary, in a thesis on Domain Adaptation of parsing systems. Furthermore, Plank provided empirical evidence on the variability of sentence length across domains, and stated the following concerning the goal of DA:

> "Therefore, the goal of **domain adaptation** is to develop algorithms that allow the adaptation of NLP systems to new domains without incurring the undesirable costs of annotating new data."

This description is also centered around the cost of annotation, another practical problem in applications, which is related to having little in-domain data as more annotation means more domain-specific training data.

Chen et al. [2013] understood the term **domain** as a particular combination of facors along which domains vary, such as genres, topics, and dialects.

Sennrich [2013b] considered Domain Adaptation as important for Statistical Machine Translation, due to linguistic phenomena such as homonyms and polysemous words. While the study included an empirical assessment of the differences between the two considered text domains, the discussion of what a domain *is* was limited to subject matter (following Lee [2001]). However, that domains vary over other dimensions was mentioned.

Hasler [2014] saw Domain Adaptation and Topic Modeling as subfields of Machine Translation, which concern building systems that are suited to translate either a given type of input text or a specific text instance. Thus, the author emphasized the design of adapted systems, in contrast to the alteration of a present system to an adapted system. Moreover, Hasler defined the domain as the source of a text corpus (such as text about news), and topic as the latent variables in a probabilistic model of topics. According to this view, Topic Adaptation is fine-grained DA under the assumption that there can be multiple distributions of domains on the same dataset, as opposed to datasets being domain-homogeneous.

Corpora that separated topic and genre differences were created by van der Wees et al. [2015] for experiments to investigate their relative impact on Domain Adaptation of Machine Translation systems. The authors found that genre-specific errors were more attributable to model coverage than topic-specific errors.

4.2 DOMAINS IN LINGUISTICS

In Linguistics, the understanding of domains is tied to Lexical Semantics, i. e., the study of the word meaning, not grammatical function. Carpuat et al. [2012] argued that errors due to unseen words and word senses account for most errors when moving into a new domain in Machine Translation. Thus, word meaning is a vital aspect of Domain Adaptation.

The history of Lexical Semantics (LS) since 1830 can be divided into three phases, viz.: (i) Historical-Philological, (ii) Structuralist, and (iii) Post-Structuralist Semantics [Geeraerts, 2010]. In the second of these phases, **Lexical Field Theory** (also known as Semantic Fields) [Trier, 1973] emerged as a way of explaining the structure of the lexicon and the relations between words.

Trier characterized such lexical (semantic) fields as mosaics of sets of related lexical items with interdependent meanings. These delineated lexical fields consist of sense-related words. The theory was not uncontroversial, e. g., because the mosaic metaphor is suggesting hard borders between fields exist. Nonetheless, a relationship can be inferred between this theory and modern ideas like semantic nets. A similarity between Self-Organizing Maps such as the ones presented in Figures 4 and 7 on Pages 9 and 19 and a mosaic is apparent.

4.2.1 *A Theory of Semantic Domains*

Gliozzo and Strapparava [2009] developed a theory of Semantic Domains based on Lexical Field Theory. The hypothesis of **lexical coherence** conjectures that a large part of the lexical concepts in a specific text belongs to the same domain, and constitutes the founding idea behind the theory. Additionally, **language games** [Wittgenstein, 1953] were used as the theoretical basis for delineating these Semantic Domains by inspecting real data in such a way that Semantic Domains arise from words as they are used (i. e., meaning-is-use). Each of the texts from which Semantic Domains are induced comprises language games of their own. While the "theoretical void" of delineation in Lexical Field Theory is filled by positing that Semantic Domains are defined by the properties of domain-specific corpora, the delineation of *those* texts was not theoretically addressed.

Nonetheless, the theory established a computational model for Semantic Domains and a way of characterizing them. Such **domain models** are tabular expressions of the **relative importance** of terms and domains. Gliozzo and Strapparava [2009, p.20] defined Semantic Domains as:

> "Semantics Domains are common areas of human discourse, such as Economics, Politics, Law, Science, etc., which demonstrate lexical coherence."

Table 4 shows an example of a domain model. The numbers indicate the **domain relevance**, for which a function $R(D_z, O)$ is defined to indicate the relevance of a domain D_z to a linguistic object, O. Domain relevance could be established either from hand-made sources such as WordNet Domains [Magnini and Cavaglià, 2000] or via unsupervised algorithms such as Latent Semantic Indexing.

Gliozzo and Strapparava [2009] applied domain models to several Natural Language Processing tasks, such as Word Sense Disambiguation and (Cross-lingual) Text Categorization, and reported consistent

	Medicine	Computer Science
HIV	1	0
AIDS	1	0
virus	0.5	0.5
laptop	0	1

Table 4: Example of a Domain Model.

performance improvements. By building Multilingual domain models from parallel corpora, comparable corpora, and dictionaries, such models could be used to quantify the similarity between documents from different languages.

4.2.2 *Domains and Sublanguages*

The term **sublanguage** [Lehrberger, 1982, Luckhardt, 1991] is closely related to domains. Constituting subsystems of language that, e. g., arise in a subject-matter domain, sublanguages are potentially *infinite* in number. Despite observing that:

> "Sublanguages have been characterized in various ways, but there is no widely accepted definition of them."

Kittredge [1983] considered the following factors as present when a subset of natural language was restricted enough for efficient semantic processing: (i) restricted domain of reference, (ii) restricted purpose and orientation, (ii) restricted mode of communication, and (iv) community of participants sharing specialized knowledge. Regarding the term sublanguage as relatively new, Kittredge and Lehrberger [1982] recognized the need for sharpening its definition.

Lehrberger [1982] presented a descriptive study of a text corpus comprised of about 70,000 words regarding instructions for aircraft maintenance and analyzed it as a sublanguage in connection with studies of translation. Figure 22 on Page 85 lists factors that help to characterize sublanguages, as a result of the study. The syntactic and semantic restrictions in point 2 are exemplified by, e. g., non-use of questions in instruction manuals, or words like *air* and *flap* only being used concretely, or verbs being restricted in their complements. Deviant rules of grammar are exemplified by sentences that are normal in the sublanguage but would be considered ungrammatical in the standard language. Finally, particularities in text structure (point 5) are exemplified by numbered sections and linking devices between neighboring sentences.

1. Limited subject matter.
2. Lexical, syntactic and semantic restrictions.
3. *Deviant* rules of grammar.
4. High frequency of certain constructions.
5. Text structure.
6. Use of special symbols.

Figure 22: Factors which help to characterize sublanguages [Lehrberger, 1982].

However, the term *sublanguage* itself has not been uncontroversial, as the connotations to set theory presupposes that there exists a superlanguage from which sublanguages can be separably cut out [Karlgren, 1993]. Karlgren argued that this could make the concept seem unrealistically simple, and preferred the term **register** [Zwicky and Zwicky, 1982]. However, the disagreement appears largely about semantics, as Lehrberger [1982] stated:

> "Furthermore, sublanguages overlap and their interrelations form a part of the description of the language as a whole. A language is not simply a union of sublanguages ..."

Moreover, Lehrberger argued that a sublanguage is not merely a subset of the set of sentences of a language, and could have a grammar of its own, not restricted to a subset of the grammar of the standard language.

Melby [1997] distinguished between the terms in the following definition:

> "A sublanguage could be considered to be a case of domain-specific language that is naturally rather than artificially controlled"

Kittredge [2003] further refined the notion of sublanguage by introducing the phenomenon of *natural sublanguage*, contrasting it with *controlled language* as two types of sublanguages, arising spontaneously within a domain or imposed by conscious design, respectively.

Additionally, Lippincott et al. [2011] explored subdomain variation in biomedical literature (considering sublanguages as associated with domains) and Temnikova et al. [2014] created a tool for assessing the sublanguage characteristics of corpora automatically.

4.2.3 *Sublanguages and the Feasibility of Machine Translation*

Kittredge and Lehrberger [1982] argued that similarity between sublanguages across languages should make translation easier due to the similarity between **linking devices**, i. e., the way sentences are interconnected (such as repetition of words or sharing pronouns).

Relaxing the constraints on what successful Machine Translation *is*, makes the task easier. In fact, MT is trivial if the input is completely controlled, e. g., to one particular sentence. The (in)feasibility of MT was debated since its early stages as a function of such requirements. Yehoshua Bar-Hillel claimed in the 1950s that ambiguity resolution required a "Universal Encyclopedia" [Hutchins, 1999] and that this entails that Fully Automated High-Quality Translation (FAHQT) is not feasible.

Refinements of what the feasibility of Machine Translation requires ensued. Kay [1997][1] maintained that MT only could produce useful results under very special circumstances, repeating the concerns of Bar-Hillel 20 years earlier, and extending the above abbreviation with **General Purpose** (GP). "Purpose" could mean text within a specific domain, thereby integrating Domain Adaptation in the high-level description of translation.

Melby [1997] plotted the feasibility of Machine Translation in a 2x2 matrix with domain-specific language and dynamic general language along one axis and high-quality and indicative translation (i. e., **gisting**) on the other, and claimed that MT did well in the high-quality, domain-specific box.

4.3 DOMAIN ADAPTATION AND STATISTICAL MODELING

As seen in some of the definitions in Section 4.1, Domain Adaptation can be regarded as the adaptation between two probability distributions, i. e., the source, and target domains. Hence, DA considered at this level would apply to any statistical learning problem.

Jiang [2008] analyzed Domain Adaptation from a theoretical Machine Learning perspective, addressing the following subproblems:

- The difference between the joint distributions (intrinsic problem).
- The difference between the optimal classification functions for the two domains.
- Whether the domain difference comes from special characteristics in the target domain or those in the source domain.

1 First appeared as a Xerox PARC Working paper in 1980.

The analysis points to differences between the source domain(s) and the target domain according to properties of the respective statistical distributions. Adaptation targets pick up novel characteristics in the target domain, such that the resulting, adapted model can correctly classify instances that belong to it.

Blitzer [2008] also analyzed making changes to distributions to fit the data in the target domain better. In a binary classification scenario, a domain was considered a pair consisting of a distribution $\mathcal{D}$ on $\mathcal{X}$ and a labeling function $f : \mathcal{X} \to [0, 1]$. $\mathcal{X}$ is a set of instances (such as words or Part-of-Speech tags) and their contexts, and f a labeling function (here, attributing one of two possibilities). The value of $f(x)$ corresponds to the probability that the label of x is 1.

Two such pairs were considered, the **source domain** $\langle \mathcal{D}_S, f_S \rangle$ and a **target domain** $\langle \mathcal{D}_T, f_T \rangle$. A **hypothesis** $h : \mathcal{X} \to \{0, 1\}$ is a predictor that indicates to what degree the instance belongs to a label, and the ideal hypothesis minimizes the probability of disagreeing with both $\langle \mathcal{D}_S, f_S \rangle$ and $\langle \mathcal{D}_T, f_T \rangle$. If this ideal hypothesis performs badly, little gain from using source domain data could be expected in the target domain.

4.4 CATEGORIZATIONS OF DOMAIN ADAPTATION METHODS

This section includes some categorizations of Domain Adaptation methods before the next sections will review specific research efforts. Broadly, the approaches can be categorized according to the level they target, whether the methods focus on the corpora or the Machine Translation methods, or on the availability of data.

4.4.1 *Domain Adaptation on Different Levels*

Domain Adaptation is a general problem for statistical modeling, and some general methods used in statistical learning were discussed above. However, not all DA methods found in general Machine Learning can be directly applied to Statistical Machine Translation [Carpuat et al., 2012]. Due to the log-linear combination of Translation Model and Language Model features in decoding, it is not possible to cast Machine Translation directly as a classification problem, although some sub-processes can be.

Domain Adaptation applies to any of the sub-processes in a Machine Translation pipeline that deviate with domains. Figure 23 diagrams how different DA methods relate to their placement in a pipeline. The diagram is not intended as a flowchart but instead divides the process into key components. Sub-processes and methods

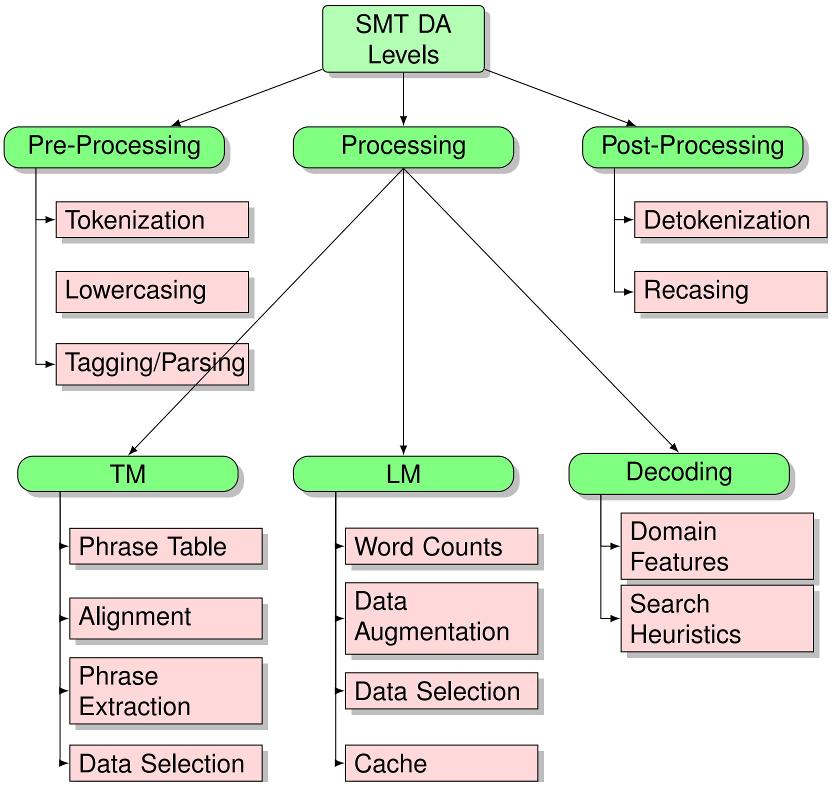


Figure 23: Overview of Domain Adaptation levels in Machine Translation.

are mixed, with the sub-processes in a pink shade and the adaptation methods in blue. Several of the methods are relevant to more levels, such as data selection that applies to Language and Translation Models.

4.4.2 *Corpus and Model-Focused Adaptation*

Domain Adaptation can also be categorized along other dimensions, such as the distinction between **corpus**- and **model**-focused adaptation. There is a conceptual difference between methods that use different portions of training data, but the same procedure for creating the models, and other methods that make alterations to an already built model. Since it has been shown that adding more data has consistently improved Statistical Machine Translation performance, e. g., Brants et al. [2007] and Dyer et al. [2008], simply adding more training data to the data-driven model will increase coverage, and thereby *domain* coverage, as domains' representation in the training data will increase. However, in some DA experiments, adding all data has not been necessary to acquire the same increase in performance. Much work has gone into identifying relevant portions of an available text corpus to attain this performance gain with a selection of data.

Active Learning strategies have also been applied to Machine Translation, e. g., Haffari et al. [2009] and Bloodgood and Callison-Burch [2010], where the former also applied the method to Domain Adaptation. Active Learning strategies select salient sentences for human evaluation and feed them back into the MT pipeline.

Wang et al. [2014] and Joty et al. [2015] divided Domain Adaptation into a) corpus level methods and b) model level methods. Corpus level methods combine and select among available training material, while the model methods combine the resulting models, e. g., mixture modeling.

Alternatively, the models can be manipulated, e. g., by altering Language Model counts directly [Iyer et al., 1997] or adding Translation Model and LM entries, [Wang et al., 2016]. In the following, methods that make alterations to how models are created are also considered model-focused.

4.4.3 *Adaptation and Data Availability*

Pecina et al. [2015] focused on the availability of data. The authors divided the literature into three according to data availability; either (i) available, and could be directly used in training (often by interpolating in and out-domain models), (ii) existing, but not readily available and needing to be acquired by data mining, or (iii) unidentifiable sources of in-domain data such that pseudo-domain must be used. Furthermore, they argued that there are three ways to use either monolingual or bilingual data to adapt Language or Translation Models; (i) simple concatenation and retraining, (ii) linear interpolation, and (iii) log-linear interpolation.

4.4.4 *Adaptation and the Specificity of Target Domains*

Foster and Kuhn [2007] separated Domain Adaptation approaches on whether the target domain was specific or unknown. A situation where the target domain is known beforehand with some in-domain material available was termed **cross-domain adaptation**. In contrast, **dynamic adaptation** assumes no knowledge about the target domain.

4.4.5 *Induction and Combination of Models and Features*

In a survey of Domain Adaptation of Statistical Machine Translation, Cuong and Sima'an [2017] separated the work into sections on the induction and combination of phrase translation tables, lexical weights, and reordering probabilities. Methods such as instance weighting and data selection fall under induction, and linear and log-linear combinations of models fall under combination. Additionally, Cuong and Sima'an had a separate section for "other trends", which include methods such as multi-domain adaptation and cache-based systems. These methods will be discussed in the next section.

4.5 REVIEW OF SELECTED DOMAIN ADAPTATION METHODS

Considering the divisions of Domain Adaptation approaches outlined in the previous section, some examples from the literature follow below. This review is not intended to be exhaustive but rather to highlight some relevant methods to put the work presented in this thesis into perspective. The continuously updated `StatMT.org` survey wiki (see Section 3.1.3) has a separate section on DA.

This chapter introduced the structure of Domain Adaptation as a three-fold process directed at some level of analysis. While this is helpful in surveying related work, there are overlaps between methods and sub-processes, e. g., data selection methods used for both language and translation modeling or data selection criteria used for domain identification and mapping instances to domains. Hence, the review in the remainder of this section is structured as follows: First, DA directed at particular levels is discussed. Next, methods for identification, mapping, and leveraging are presented, before the section finishes with a discussion of multi-domain adaptation. All methods target at least one level of representation, and the structure is aimed at highlighting the salient parts of the surveyed work.

4.5.1 *Adaptation Specific to Levels*

The work presented in this subsection is directed at specific levels, notably Language Models, and also the alignment process that can be evaluated standalone. Some of this work also targeted Translation Models. That work was mostly evaluated extrinsically and will be described in the following sections.

4.5.1.1 *Language Model Domain Adaptation*

Early work on the Domain Adaptation of Machine Translation models focused on Language Model adaptation, which was already well underway in the Automatic Speech Recognition community. Such research efforts include Carter [1994], who clustered input sentences according to the best Language Model by linear combination of these clusters, Iyer et al. [1997] who interpolated LM counts, Rosenfeld [1996] who used **trigger** words (i. e., domains are triggered by words) in Maximum Entropy modeling, Mahajan et al. [1999] who retrieved similar documents to the input and built new, termed dynamic Language Models, on the retrieved documents, and Janiszek et al. [2004] who retrieved similar documents using Singular Value Decomposition and augmented the corpus counts with the corpus of retrieved documents. Also, Gao et al. [2005] compared linear interpolation to a selection of discriminate methods (optimizing a feature weight vector) based on an in-domain development corpus, applied to a specialized task of converting Japanese signs into string form, of which Language Models are integral parts. Adaptation of Language Models was surveyed by Bellegarda [2004].

Applied to Machine Translation, Eck et al. [2004] retrieved documents based on their similarity with an initial translation of the input document. Similarity was defined as the cosine of the angle of the vectors, vectorized with tf-idf counts. Subsequently, domain-adapted Language Models were created with these documents. The input document was then translated again, using this Language Model in decoding. The most similar documents (10, 100, 1000) and sentences (100, 1000, 10k, and 100k) were retrieved from a larger document collection. Eck et al. concluded that sentence level retrieval performed better in terms of perplexity reduction, but extrinsic evaluation showed that the correlation between improvement of the Language Model and the overall Statistical Machine Translation task was weak.

4.5.1.2 *Translation Model Domain Adaptation*

Hildebrand et al. [2005] expanded their earlier work on Language Model Adaptation [Eck et al., 2004] to Translation Models, by similarly using Information Retrieval techniques to select the sentences in the training material that were similar to the test set. Adapted TMs were created by selecting extra training material more similar to each sentence in the test corpus and re-training a TM on that selection. This work was combined was with an adapted LM, showing small changes in areas such as language pairs and selection criteria. This method is based on training a new adapted TM for each input sentence, which raised questions about scalability. Translation would potentially be very slow because of re-training, or the TMs would have to be small.

4.5.1.3 *Domain Adaptation of Alignment Models*

Civera and Juan [2007] applied mixture modeling to the **alignment** process in Statistical Machine Translation (see Section 3.1.3.2). Mixture models model an overall population probabilistically by representing the presence of subpopulations therein, each modeled by separate probability distributions. When these distributions are weighted, they form a mixture distribution. Experiments used 1 to 4 components in the mixtures with parameters estimated with the Expectation-Maximization algorithm. Two corpora (Europarl Koehn [2005] and News Commentary [Tiedemann, 2012]) were used for the language pair English-Spanish, bi-directionally. An extrinsic evaluation was done with a Statistical Machine Translation system using these alignments. Some improvement of BLEU score was reported for the News Commentary corpus for the direction English-Spanish.

4.5.2 *Identification of Structures*

The initial step of Domain Adaptation, the identification of structures, establishes the idiosyncrasies of each domain. When using labeled corpora, it is (normally) assumed that a corpus belongs to the domain corresponding to its label. Otherwise, such structures can be identified through the acquisition of more training material, or the selection of data from a larger, general corpus. This subsection is ordered correspondingly.

4.5.2.1 *Data Acquisition*

Wu et al. [2008b] used an in-domain dictionary and a monolingual corpus to improve a Statistical Machine Translation system. A second phrase table was created from the in-domain dictionary, giving the phrases from the dictionary probabilities according to their importance as translations, measured as $1/n$ where n is the number of translations for that entry. The two phrase tables were then combined using linear and log-linear interpolation, as were the Language Models built from in- and out-domain data. The method gave absolute improvements of 8.16 and 3.36 BLEU points for the language pairs English-Chinese and English-French, respectively. The best results came with linear interpolation.

Bertoldi and Federico [2009] used monolingual in-domain data by translating a monolingual corpus into the target language (running the experiment in both directions), thereby synthesizing a parallel corpus. The experiments were done on the United Nations[2] and Europarl corpora. A performance gain was found when parallel corpora were synthesized by translating Target, but not Source Language data.

Daumé and Jagarlamudi [2011] addressed unseen words in Statistical Machine Translation by mining dictionaries for such. The authors argued that unseen words represent a significant problem when moving between domains, and searched dictionaries for tokens missing from the phrase table. Later, these entries were used in the SMT model by including them as a quasi-parallel corpus. Results were reported to improve from 0.5 to 1.5 BLEU points.

4.5.2.2 *Data Selection*

Zhao et al. [2004] retrieved similar sentences from a document collection based on initial translations. A new, *specialized*, Language Model was built on these documents, subsequently interpolated with the original, assumed *general* LM. Tf-idf weighting was used to retrieve sentences in this project, building queries based both on bag-of-words and Structured Query Models, which enable the inclusion of syntactic and semantic information with the use of proximity operators. The authors cited better results using this retrieval method than the bag-of-words method. While the Gigaword[3] corpus was used, the largest experiments retrieved only 4000 sentences.

Moore and Lewis [2010] proposed an alternative method for selecting relevant sentences from an out-domain corpus to build an auxiliary Language Model. The experiments were done on the Europarl and Gigaword corpora[4], where the former was used as in-domain text. This method selected sentences from the Gigaword corpus by measuring the difference in cross-entropy from an LM built on the English side of the English-French Europarl corpus and a random selection of N sentences from Gigaword corresponding in size:

$$H_{EU}(s) - H_{N}(s) \tag{55}$$

The Gigaword corpus was segmented into eight equal portions, ranked after this difference, from which new LMs were built. Subsequently, the models were tested on a held-out portion of Europarl. Moore and Lewis compared the method with other selection criteria, such as ranking the sentences only on the perplexity score from the Europarl corpus alone (corresponding to Gao et al. [2002] and Lin et al. [1997]), and scored each Gigaword sentence on the log-likelihood of

2 `https://conferences.unite.un.org/uncorpus` (Last visited: March 27, 2019.)
3 `https://catalog.ldc.upenn.edu/LDC2003T05` (Last visited: March 27, 2019.)
4 Third edition.

the tested Europarl corpus on unigram models built with and without this sentence, as well as to a random selection of Gigaword sentences.

Results using all selection methods converged as more and more text was added, until all available text was selected, at which point the auxiliary Language Models were equal. The method used the difference in cross-entropy, and it reported to have lower perplexity than all of the other methods until all data was added. More importantly, reached lower perplexity than adding the whole corpus, also after using only a small portion of the data.

Axelrod et al. [2011] did a similar extraction of **pseudo-in-domain** sentences based on cross-entropy measures — *pseudo* because they are similar, but not identical to in-domain data. Expanding the method proposed by Moore and Lewis [2010] by measuring the difference in cross-entropy between the in-domain and out-domain models on the source level, they proposed a measure motivated by the bilingual nature of translation:

$$H_{I-src}(s) - H_{O-src}(s) + H_{I-tgt}(t) - H_{O-tgt}(t) \tag{56}$$

which includes the corresponding difference on the Target Language side[5].

These pseudo-in-domain corpora were used to train smaller Statistical Machine Translation models adapted to the target domain, and performing better than a model built on the entire material. Scores improved more when the models were combined. Discarding as much as 99% of the original, general-purpose corpus, Axelrod et al. achieved better (1.8 BLEU points) results with the pseudo in-domain corpus.

Axelrod [2017] proposed an idea of adding sentences based on their marginal contribution to cross-entropy (not yet empirically evaluated) consisting of both a penalty and a **gain** term. Axelrod argued that this would not treat the in- and out-domain Language Models as opposing ends, taking into account that portions of them can be similar.

Duh et al. [2013] experimented with a similar selection criterion as Axelrod et al. [2011] but using Neural Network Language Models, not n-gram models. Sentences selected by the criteria were compared in a Statistical Machine Translation translation task. The authors stressed that the difference between the language modeling frameworks was not dramatic. Still, the overlap between the selected sentences was reported to be in the range of 60-75%. This method achieved improvements from 0.1 to 1.7 BLEU points on the translation task.

5 Axelrod et al. [2011] used s, not t also on the target language side, but this was changed for clarity.

Wang et al. [2014] introduced an edit-distance approach to selecting relevant corpora. The edit-distance was based on the normalized Levenshtein [1966] score:

$$FMS = 1 - \frac{LD(S_G, S_R)}{Max(|S_G|, |S_R|)} \tag{57}$$

S_G and S_R denote sentences from the general and in-domain corpora, respectively. The *general* corpus consisted of more than 1M sentences from various sources (mainly from the Linguistic Data Consortium), and the in-domain corpus came from the Hong Kong Law Corpus[6].

The experiments mainly targeted exploring differences between selection criteria. Linear interpolation both of corpora directly and models was done, but without a thorough investigation of the interpolation parameters. First, sentences were retrieved based on the various criteria, and then their impact on BLEU score was measured. Differences were small, but combined models did best, followed by perplexity, IR, and edit-distance selection criteria.

Banerjee et al. [2015] used Quality Estimation (QE) of translated sentences as the basis for data selection, i. e., selecting out-domain sentences similar to those with the worst translations according to this QE, such that the weakness of the Statistical Machine Translation model is targeted in the Domain Adaptation process. First, the quality of translations was estimated, using only monolingual data by training a classifier. Next, relevant sentences from the out-domain corpus were selected, and finally leveraged as linearly interpolated (weights estimated with Expectation-Maximization) Language and Translation Models in the `Moses` Statistical Machine Translation system. Banerjee et al. found that this selection method performed better on the overall translation task than either using all supplementary data or other selection strategies.

Biçici [2015] experimented with instance selection based on a Feature Decay Algorithm (FDA), and Dice's coefficient [Dice, 1945] which provides an association score for each word position pair. The features used were n-gram overlaps and the output of the **common cover link** algorithm [Seginer, 2007], an unsupervised parsing algorithm that provides links between head and base words. Experiments compared the algorithms after how instances were selected. Two domains were used, and a parameter α determined their relative share of the sampling pool. The best method produced almost as good BLEU results as a baseline system with 2M sentences using only 10,000 selected instances.

6 Linguistic Data Consortium: LDC2004T08.

4.5.2.3 *Unsupervised Methods*

Sennrich et al. [2013] used an adaptation of K-means clustering on a concatenated corpus of in- and out-domain data, with 10 and 100 means, respectively, both on document and sentence level. Two distance functions were applied; (i) using Language Models as centroids and their scores as distances and (ii) using Word Sequence Kernels [Cancedda et al., 2003]. Furthermore, exponential decay was used with the distance function to capture that words that are close to each other in the text belong to the same domains. Separate models were built from these clusters, interpolated similarly as by Foster and Kuhn [2007], using perplexity minimization on a separate development corpus to determine the interpolation weights. In contrast, the baseline methods used sentence or document level distances to predict a domain and combining the domain-specific model with a general one using fixed weights. These experiments also showed that perplexity optimization performed as well as — or better than — competing methods.

Hasler et al. [2014] combined Topic Modeling with Domain Adaptation using interpolation as per Sennrich [2012a]. Topic Models are built using Latent Dirichlet Allocation, which estimates the parameters for a set of **multinomials** sampled from a Dirichlet distribution, resulting in a distribution over words for each topic, and a distribution over topics for each document. In place of documents, a context was created for each **phrase pair**, all words in the sentence contexts of each such pair.

The domain-specific information was implemented as a set of features with information specific to each topic from the model, e. g., how many times the source and target pairs were seen in the contexts. The models were also used as domain classifiers, by comparing cosine similarities between input documents and topics to determine their provenance. Reported performance gains were modest (0.82 and 1.67 BLEU points over unadapted and mixture models, respectively), but the method was a template for how Topic Modeling could offer finer granularity on the document level on top of broader text categories, as well as an unsupervised method for domain classification.

4.5.3 *Mapping of Structures to Distinct Domains*

The mapping of the structures at a given level is a prerequisite for its subsequent use in an application. Often, the same criteria as used for data selection in the previous subsection can also be used to attribute a domain to an input document or string. Otherwise, this can be cast as a classification problem.

4.5.3.1 *Domain Identification*

Xu et al. [2007] built separate Statistical Machine Translation models for each domain based on little in-domain data for, building both Translation and Language Models. These models were combined with a larger, general SMT system, to mitigate data scarcity. Domain Adaptation was implemented as a classification task; a document was classified before translation into domains, and the corresponding SMT model was used for translation. Classifying the right domain for an input document was reported as more accurate when data selection was done with perplexity (LM score) than with a similarity score between a document and the in-domain collection based on the number of shared words (termed an Information Retrieval method).

4.5.4 *Leveraging Domain Knowledge to Translation*

There is a variety of options for leveraging the obtained structures for translation. Many methods are based on mixture modeling or interpolation of more Language or Translation Models, evaluated jointly or separately. Moreover, experiments have been done using, e. g., caches, dynamic phrase tables, Artificial Neural Networks, and also manual adjustment of parameters.

4.5.4.1 *Cache Models*

Tiedemann [2010] used caches for Domain Adaptation, either implemented directly in the Language Model software using interpolation or as a feature in log-linear decoding. The cache contained recently seen events, either in the Language Model or the Translation Model. Without needing domain-specific training data (other than for interpolation weights), this method could be applied to any new domain. Furthermore, a decay factor was introduced to diminish the impact of the cache proportional to the time since the last observation. Improvements in BLEU score of 0.78 (2.6% relative) were reported.

Louis and Webber [2014] included domain-specific information in cached Language Models as features in log-linear decoding. This work leveraged the structure of Wikipedia documents and the domain characteristics of biographies to create specialized caches for the general domain and sub-topics, respectively. The sub-topic caches were obtained by topic modeling with Latent Dirichlet Allocation. The authors reported improvements of 0.5 BLEU points.

4.5.4.2 *Dynamic Phrase Tables*

Dynamic phrase tables are auxiliary phrase tables in Statistical Machine Translation systems that are created especially for the current input and combined with an existing, primary phrase table. Thus, source-side information that is relevant within a certain time window is retainable, such as domain provenance affecting lexical choice on phrasal or word level. Feeding post-editing results being back into a live system [Germann, 2014] is also possible.

Sennrich [2011] implemented dynamic phrase tables using an alternative scoring mechanism. In contrast to Maximum Likelihood Estimation (MLE) scoring, the proposed method factored in the frequencies used to make MLE scores in the primary corpus when scoring the dynamic phrase table. That increased coverage (recall) but still took the in-domain characteristics of the dynamic corpus into account (precision). The method was evaluated on the bilingual (French-German) Swiss Alpine Club Annals [Volk et al., 2010].

The alternative scoring was evaluated using two approaches to Machine Translation, both implementable with dynamic phrase tables; multi-engine learning (i. e., training dynamic phrase tables with the output of multiple MT systems) and on-line learning (i. e., training dynamic phrase tables with corrections of translation output). Reference translations of already translated sentences were used to simulated post-editing. The alternative scoring attained a gain of 0.73 and 0.78 BLEU points for the two approaches, respectively.

4.5.4.3 *Interpolation*

Lu et al. [2007] adapted a Machine Translation system to three different domains from the Linguistic Data Consortium: the Hong Kong Laws (LDC2004T08), Hong Kong Hansards (LDC2004T08), and the FBIS News corpora (LDC2003E14). The authors reported that concatenating all the available data did not yield better results while experimenting with better ways of utilizing the available parallel data. A distinction between **off-line** and **on-line** approaches was made. Lu et al. used the concatenation of all three sources as the baseline corpus.

In the off-line phase, documents were represented by tf-idf-transformed vectors and retrieved ranked by cosine difference with the input query, where the target domain was given. Each sentence in the training data was treated as a document. The adaptation was implemented as altered sentence weights *of* the original training data *with* the retrieved sentences by using them in `GIZA++` word alignment (see Section 3.1.3.2). Counts from the retrieved sentences, close to the target domain, were added before computing alignment.

In the on-line phase, log-linear interpolation of sub-models, i. e., models built from the data from each of the domains mentioned above, was done. For each input, similar sentences were retrieved and used to determine the weighting of the sub-models. Four weighting schemata based on different weightings of the proportion of each retrieved document coming from each component were explored, but results suggested that different weighting did not markedly impact results.

Foster and Kuhn [2007] applied mixture modeling to both Language and Translation Models. First, a corpus was clustered into components; next models were built on each component before they were weighted according to distance criteria. Finally, the models were combined as a global model linearly and log-linearly. The crux of this approach is the weighting of the models. For **cross-domain** adaptation, weights were set using the available in-domain data. For **dynamic** adaptation, weights were a function of distance from the components and the input document. Several distance metrics were used, e. g., tf-idf, LSI-reduced word and document frequencies, Language Model scores, and weights obtained by using Expectation-Maximization on a word-level mixture model. Foster and Kuhn found LM-based metrics to work better, and LM adaptation to yield better results than TM adaptation.

Sennrich [2012a] experimented with different methods for interpolating Translation Models, either by weighting counts when estimating translation probabilities, or two ways of linearly interpolating the scores. This work moved away from the in-domain or out-domain dichotomy and experimented with four and ten domains simultaneously. Interpolation weights were set by minimizing TM perplexity on the in-domain development data. While results were modest, Sennrich argued that perplexity minimization is a robust baseline for TM interpolation. Furthermore, the expected gains of custom interpolation gains depend on how close the interpolated domains are. Uniform weights (corresponding to concatenation) were expected to work better for closer domains.

4.5.4.4 *Manual Alterations*

Offersgaard et al. [2008] studied the use of Translation Model interpolation for Machine Translation systems in production. The study was conducted in connection with a comparison of Statistical Machine Translation and Translation Memory for a Linguistic Service Provider. For domains as close as *camcorders, software, DVDs, printers* and *mobile phones*, interpolation increased BLEU scores in the range of 2.2 to 3.7 with a zero gain outlier and reduced TER rates in the range of 1.8

to 2.8 points (with the same outlier). The interpolation weights were manually set.

4.5.4.5 *Continued Training of Neural Machine Translations Systems*

Neural Machine Translation has been reported to perform either worse than or on par with Statistical Machine Translation for low-resource domains [Zoph et al., 2016]. Zoph et al. used transfer learning, i. e., using parameters learned from high-resource language pairs to improve results on other tasks, but Domain Adaptation methods would also have been applicable.

For Domain Adaptation of Neural Machine Translation systems, **continued training** is a method used, e. g., by Sennrich et al. [2016], where a generic system is trained on out-domain data, and parameters are later fine-tuned on a smaller corpus by **continued training**. Both parallel and monolingual data (e. g., by back-translating a synthetic corpus) can be used for this. Chu et al. [2017] expanded this idea by proposing mixed-fine tuning, where the tuning is done on a mixture of out- and in-domain data to reduce the problem of overfitting.

4.5.4.6 *Neural Network Joint Models*

Joty et al. [2015] trained Neural Network Joint Models (NNJMs) on in-domain and out-domain data and later combined them into adapted models referred to as Neural Domain Adaptation Models (NDAMs). NNJMs model Target and Source Language information jointly, integrated as an additional Language Model feature in log-linear decoding [Devlin et al., 2014]. The NNJMs were trained on the whole data but regularized with the in-domain data (by an alternative definition of the loss function). The data selection methods were similar to Axelrod et al. [2011]. However, there are clear differences, as Joty et al. did scoring at the bilingual level and also retained coverage by not discarding unused out-domain data. Instead, the weights of data similar to in-domain instances were altered.

4.5.4.7 *Phrase-Sense Disambiguation*

Carpuat and Wu [2007b] introduced Phrase-Sense Disambiguation (PSD), which is Word Sense Disambiguation-style disambiguation extended to the phrase level, i. e., that entire phrases (in the Statistical Machine Translation sense) are disambiguated based on Source Language information. Carpuat et al. [2012] applied standard Domain Adaptation techniques like interpolation and data selection to PSD, implemented as features engineered with the Vowpal Wabbit Machine

Learning system [Langford et al., 2007]. The reported results were modest for the method over the baselines.

4.5.5 *Multi-Domain Adaptation*

Multi-Domain Adaptation systems refer to Machine Translation systems that are adapted to more domains at the same time, in contrast to adaptation to one specific target domain. In a Multi-Domain Adaptation setting, different input documents will be given different translations depending on domain provenance. Strategies include using different configuration parameters for the decoder [Huck et al., 2015] or using separate models altogether [Cui et al., 2013]. Domain membership can be assumed to be known, or established by a statistical classifier [Banerjee et al., 2010].

4.5.5.1 *Sentence Clustering and Bi-and Monolingual Domain-Specific Models*

In the most similar work to the experiments presented in Part II, Yamamoto and Sumita [2008] used unsupervised methods to cluster both bilingual and monolingual data, subsequently using these clusters for multi-domain Domain Adaptation. While the paper did not use that term, it did take a distributional view of domains (i. e., without pre-defined labels).

The authors argued for their method probabilistically, by making several transformations of the Statistical Machine Translation objective function (Equation 18 on Page 48). A variable D for domain was introduced, on which both Language and Translation Models depend:

$$\underset{T}{\operatorname{argmax}}\, P(T|S, D) \tag{58}$$

which, in turn, is transformed into a domain-conditioned formulation interpretable as separated into Language and Translation Model probabilities (the Bayes' Rule denominator $P(S|D)$ is constant during search and is left out):

$$\underset{T}{\operatorname{argmax}}\, P(T, D) * P(S|T, D) \tag{59}$$

On this background, and through a series of algorithmic operations and approximations, Yamamoto and Sumita ended with a two-part setup. The probability of each translation unit (word sequence, or sentence) is first modeled to belong to some domain and later translated with a Machine Translation model particular to this domain.

Clustering of the documents was done off-line. Domain adapted translation, where a domain is attributed to the input and later translated with the corresponding model was done on-line. The algorithm required a preset number of clusters and assigned sentence pairs to these clusters randomly. Language Models were built from these clusters, and the entropy (equations were not provided) for the documents within the clusters was calculated. The entropy was summed for both languages and all clusters and termed the **total entropy**. Bilingual sentences were reassigned such that total entropy was reduced. This process was repeated until reduction met a certain threshold, and the clusters were returned. Bilingual corpora were clustered according to the above procedure.

Sentences were evaluated on the above-mentioned Language Models as well as a general LM in the monolingual case. If any of the domain-specific models had lower entropy than the general model, the sentence was added to it. The system was evaluated both using domain-specific Language and Translation Models separately, as well as their combination. All systems improved results over the baseline, with the largest improvement in BLEU terms for the combination (2.7 points). Results improved with 0.56 BLEU and 0.5 `Meteor` points over the baseline with the method proposed by Yamamoto and Sumita.

4.5.5.2 *Multitask Learning*

Motivated by the heterogeneity of input, Cui et al. [2013] created a multi-domain system tailoring several domain-specific models. The authors used Multi-Task Learning (MTL) [Caruana, 1997], a method which trains more Machine Learning problems jointly to leverage commonalities. In a complicated setup using an in-house hierarchical phrase-based Machine Translation system, several domain adapted models were created using a three-phase method; (i) selecting in-domain bilingual data similar to Axelrod et al. [2011], (ii) training in- and general domain models, and (iii) tuning the models jointly using MTL.

Luong et al. [2016] used Multi-Task Learning for a translation task, citing improvements of up to 1.5 BLEU points over single-task baselines. Martínez Alonso and Plank [2017] argued that MTL is not always effective but works well for auxiliary tasks with more compact and uniform label distributions.

4.5.5.3 *Multi-Domain Versus Mixed-Domain Translation*

Huck et al. [2015] contrasted the multi-domain approach to a **mixed-domain** approach, where all domains are mixed, i. e., concatenated into one training corpus, which is used for training and tuning of the system. Besides the domain-specific models of translation themselves, multi-domain approaches require a classification of domains, such that the input can be translated with the optimal domain model. The authors created three domain-specific models for a pre-defined set of domains (Europarl, TED-talks, and News), by tuning the log-linear weights of the model to the respective development sets. Further, the general Language Model from the baseline system was replaced with an interpolation of domain-specific LMs, as well as incorporating **provenance** features binary signaling whether the input sentence was seen in a given corpus.

Huck et al. explored several classification algorithms and concluded that the multi-domain method was more reliable across language pairs, while the mixed-domain approach worked well for half of the pairs. Resulting differences in BLEU score were small ($\leqslant 0.9$), however.

Part II

EXPERIMENTS

The second part of the thesis contains experiments. First, Chapter 5 presents experiments on identification of Multiword Expressions, followed by Chapter 6 with experiments on building large-scale Language Models using HPC resources. Next, a series of experiments on Self-Organizing Maps and Machine Translation follows. Chapter 7 shows a SOM implementation used for a standard classification task, followed by Chapter 8 using SOMs with hierarchical clustering to produce tangible clusters. Finally, Chapter 9 describes experiments that used these clusters for Domain Adaptation of Machine Translation systems.

5 MULTIWORD EXPRESSION IDENTIFICATION

The first experimental chapter will present a computationally inexpensive method for the identification and translation of Multiword Expressions (MWEs). Section 5.1 discusses related work before Section 5.2 describes the datasets. Section 5.3 explains the methods used in the experiments, and Section 5.4 presents the experimental results before Section 5.5 sums up.

Multiword Expressions are sequences of words that exhibit some kind of idiosyncratic interpretations on, e. g., syntactic, semantic, or pragmatic levels [Yazdani et al., 2015]. **Idiomaticity** refers to the deviation of an MWE from the basic properties of component lexemes, and **compositionality** refers to the degree to which the features of an MWE combine to predict the features of the whole [Baldwin and Kim, 2010]. Baldwin and Kim defined MWEs as:

> Lexical items that: (a) can be decomposed into multiple lexemes; and (b) display lexical, syntactic, semantic, and/or statistical idiomaticity.

Multiword Expressions cause particular lexical choice problems in Machine Translation, but can also be an opportunity both to generalize outside the bilingual corpora often used as training data in Statistical Machine Translation, and as a vehicle to adapt to specific domains. Identification of MWEs is important for many Natural Language Processing tasks [Sag et al., 2002], but can be crucial in Word Sense Disambiguation and MT. Because the semantics of many MWEs are sufficiently non-compositional, a suitable translation cannot be guaranteed by translating the words in isolation. Identifying MWEs can help identify idioms or domain specific language use (leading to better translations) and potentially reduce the amount of lexical choice an MT system faces during Target Language generation.

The method proposed in this chapter uses bilingual dictionaries as a source of Multiword Expressions. Relationships are induced between the source sentence and candidate translation lexical items based on their corresponding entries in the dictionaries. The approach was combined with a data-driven baseline for evaluation on an adaptation of the SemEval'10 cross-lingual Word Sense Disambiguation task [Lefever and Hoste, 2010a]. Concretely, a deterministic Multiword Expression disambiguation procedure based on translation dictionaries in both directions (from Source to Target Language and vice versa) is used, in comparison to a baseline system that ranks target lexical items based on their immediate context and an n-gram Language Model. The n-gram model is a high-coverage, low-precision method, which complements the dictionary approach.

Results showed that the dictionary method improves the baseline system. Experiments also showed that when Multiword Expression dictionary entries are identified, they tend to be particularly strong translation candidates, suggesting that features based on MWE dictionary entries have the potential to improve the performance of data-driven Machine Translation methods. Furthermore, the method is computationally cheap, and also expandable to any Source Language words in an MT pipeline (only nouns were considered in these experiments). When domain-specific dictionaries are available, as they are, e. g., for law and medicine, these could be leveraged for Domain Adaptation.

5.1 RELATED WORK

Nagy T. et al. [2011] discussed how methods for discovering Multiword Expressions (exemplified by nouns and light verb constructions) could be adapted to different domains. The authors observed domain-related differences in the frequency of MWE types, which impacted the quality of Machine Translation.

In any translation effort, automatic or otherwise, the selection of Target Language lexical items to include in the translation is a crucial part. In Rule-Based Machine Translation systems, lexical choice is derived from the semantics of the source words, a process that often involves complex semantic composition. Data-driven systems, on the other hand, commonly base their translations almost exclusively on co-occurrences of bare words or phrases in bilingual corpora. This approach leaves the responsibility of selecting lexical items in the translation entirely to the local and bare context found in phrase translation tables and Language Models without an explicit notion of either source or target language semantics. Systems of this type have been shown to produce reasonable translation quality without explicitly considering Word Translation Disambiguation. Carpuat and Wu [2007a] found that the incorporation of Word Sense Disambiguation systems into Statistical Machine Translation consistently improved results, although earlier work suggested the contrary [Carpuat and Wu, 2005].

The translation of Multiword Expressions can be a significant source of errors for phrase-based Statistical Machine Translation systems [Pal et al., 2013], despite that those systems explicitly recognize and use alignments of sequential chunks of words. Several researchers approached this problem by adding MWE translation tables to the systems, either through expanding the phrase tables [Ren et al., 2009] or by injecting the MWE translations into the decoder [Bai et al., 2009]. Furthermore, there has been some interest in automatic mining of MWE pairs from bilingual corpora as a task in itself: Caseli et al. [2010] used a dictionary for evaluation of an automatic MWE extraction procedure using bilingual corpora. The authors recommended stopword filtering, which was done in the experiments presented below. Sharoff et al. [2006] extracted MWE pairs from comparable monolingual corpora in place of a parallel bilingual corpus. Semmar and Laib [2017] mined dictionaries of MWEs from parallel corpora and used them in Domain Adaptation of an Example-Based Machine Translation system, and reported significant performance improvements for out-domain translation.

5.2 DATA AND RESOURCES

The 2010 Semantic Evaluation exercise (SemEval'10) featured a Cross-Lingual Word Sense Disambiguation (CL-WSD) task where systems were given an English word in its context and asked to produce appropriate substitutes in another language [Lefever and Hoste, 2010a]. The task was created to facilitate the integration of general or domain-specific WSD into Natural Language Processing applications like Machine Translation.

5.2.1 *The Cross-Lingual Word Sense Disambiguation Datasets*

The CL-WSD datasets were constructed by making a sense inventory of all possible translations of a given Source Language word based on word-alignments in Europarl [Koehn, 2005]. Alignments that involved the SL words were manually checked. The aligned target words were manually lemmatized and clustered into translations with a similar sense; see Lefever and Hoste [2010b] for details. Trial and test instances were extracted from two other corpora, JRC-Acquis [Steinberger et al., 2006] and British National Corpus (BNC) [Burnard, 2007]. Four human translators picked the contextually appropriate sense for each SL word, preferring 0–3 translations. Since translations were restricted to those appearing in Europarl, a slight domain bias was introduced. Each translation has an associated count indicating how many annotators that considered it to be among their top-3 preferred translations in the given context.

The trial data for each language consists of five nouns (20 sentence contexts per noun), and the test data of twenty nouns (50 contexts per

*(a) AGREEMENT in the form of an exchange of letters between the European Economic Community and the **Bank** for International Settlements concerning the mobilization of claims held by the Member States under the medium-term financial assistance arrangements*
{bank 4; bankengesellschaft 1; kreditinstitut 1; zentralbank 1; finanzinstitut 1}
*(b) The Office shall maintain an electronic data bank with the particulars of applications for registration of trade marks and entries in the Register. The Office may also make available the contents of this data **b**ank on CD-ROM or in any other machine-readable form.*
{datenbank 4; bank 3; datenbanksystem 1; daten 1}
*(c) established as a band of 1 km in width from the **banks** of a river or the shores of a lake or coast for a length of at least 3 km.*
{ufer 4; flussufer 3}

Table 5: Examples of contexts for the English word *bank* with possible German translations.

word, 1000 per language). The CL-WSD data covers Dutch, French, Spanish, Italian and German; however, the experiments presented below were only evaluated on the German data. Table 5 provides examples from the trial data of contexts for the English word *bank* and its possible translations into German.

The Cross-Lingual Word Sense Disambiguation task involves two subtasks:

1. Finding translations candidates.

2. Ranking and filtering translation candidates.

Section 5.4 presents experiments that addressed Word Sense Disambiguation (Subtask 2). As Subtask 1 was skipped, it was assumed that a perfect solution to finding translation candidates is available. This assumption implies that all possible translations are in the dictionary, which is accomplished by extracting all possible translations from the gold standard. For the English lemma *bank*, for example, the CL-WSD trial gold standard for German contains the word *Bank* itself, together with 40 other translation candidates, as shown in Table 6. Eight of those are related to river banks (*Ufer*, but also, e. g., *Westbank* and *Westjordanland*), three to databases (*Datenbank*), and one to blood banks.

bank, bankanleihe, bankanstalt, bankdarlehen, bankengesellschaft, bankensektor, bankfeiertag, bankgesellschaft, bankinstitut, bankkonto, bankkredit, banknote, blutbank, daten, datenbank, datenbanksystem, euro-banknote, feiertag, finanzinstitut, flussufer, geheimkonto, geldschein, geschäftsbank, handelsbank, konto, kredit, kreditinstitut, nationalbank, notenbank, sparkasse, sparkassenverband, ufer, weltbank, weltbankgeber, west-bank, westbank, westjordanien, westjordanland, westjordanufer, westufer, zentralbank

Table 6: All German translation candidates for *bank* as extracted from the gold standard.

The rest are connected to financial institutions (e. g., *Handelsbank* and *Finanzinstitut*) or by association to finance, e. g., *Konto, Weldbankgeber, Banknote, Geldschein, Kredit*).

5.2.2 *Dictionaries*

Two English-German dictionaries were used in the experiments, both with about 900,000 entries. One was obtained by reversing an existing proprietary German-English dictionary made available by its owners, while the other is a free online resource:

- The GFAI dictionary (called D1 below) is a proprietary and substantially extended version of the Chemnitz dictionary providing over 900,000 entries. The Chemnitz electronic German-English dictionary[1] contains over 470,000 word translations and is available under a General Public License (GPL).
- The freely available CC dictionary[2] (called D2 below) is an Internet-based German-English and English-German dictionary based on user-generated word definitions with 879,456 dictionary entries (83,781 MWEs in the German-English direction and 83,613 in the English-German direction).

The actual dictionaries are irrelevant to the discussion at hand. Since determining the strengths or weaknesses of either dictionary was not an objective of this research, neither was indicating a bias towards a particular resource.

1 `http://dict.tu-chemnitz.de/` (Last visited: March 27, 2019.)
2 `http://www.dict.cc/` (Last visited: March 27, 2019.)

5.3 METHOD

This section describes the complete disambiguation model that consists of both Language Model-based context matching and dictionary-based Multiword Expression extraction. Using a dictionary to identify MWEs after translation had low recall of Target Language MWEs, but high precision. Often the dictionary did not have an entry for a would-be-identified MWE, or there were no MWEs to discover, such as in Example (1). The precision was high, however, when an MWE was identified.

(1) ...1 km in width from the `banks` of a river...

Extraction of Multiword Expressions was done in both a direct and indirect manner: *Direct extraction* used adjacent words in the Source Language in combination with the word to be translated, if the combination had an entry in the source-to-target language (SL–TL) dictionary. *Indirect extraction* searched the target-to-source (TL–SL) dictionary, looking up translation candidates for the combined words.

Due to low recall, relying on Multiword Expressions from dictionaries would not be sufficient for the Cross-Lingual Word Sense Disambiguation task. As a consequence, the method was combined with an n-gram model built on a Target Language corpus. The Language Model was used to rank translation candidates according to the score of the n-gram best matching the context around the translation candidate, i. e., the LM score of the n-gram with context. This approach is more robust but less precise, and it was used as a baseline for, and also in combination with, the high-precision but low-recall dictionary approach. The n-gram-matching method is described before the dictionary-based methods in the following subsection.

5.3.1 *N-gram Context Matching*

The n-gram matching procedure is used to produce a ranked list of translation candidates and their context, providing robustness as well as a baseline measure. The procedure consists of two steps:

1. An n^{th} order source context is extracted, and the translation candidates for each source word in this context are retrieved from the dictionary. (The context is stopword-filtered.)

2. All relevant n-grams are inspected in order from left to right and from more specific (5-grams) to least specific (single words).

For each part of the context with matching n-grams in the target Language Model, the appropriate target translation candidates are extracted and ranked according to their LM score. The result is an n-best list of translation candidates.

Since dictionary entries are lemma-based, lemmatization is necessary to use this approach in combination with the dictionary enhancements. The source context is formed by the lemmata in the sentence surrounding the **focus word** — the word to be disambiguated — by a window of up to four words in either direction, limited by a 5-gram maximum length. Stopwords are removed to extract the semantically most relevant content. For each of the five lemmata in the window, the relevant translation candidates are retrieved from the bilingual dictionary. The candidates form the ordered translation context for the source word window.

The following example from the trial data illustrates how the contexts are created. First, the relevant part of the Source Language sentence with the focus word in boldface:

(2) The BIS could conclude stand-by credit agreements with the creditor countries' central **bank** if they should so request.

The following Source Language context is obtained after a preprocessing involving lemmatization, stopword removal, and insertion of sentence start (<s>) and end markers (</s>):

(3) country central **bank** request </s>

From this window, the possible n-grams in the target side context are generated by assembling all ordered combinations of the translations of the Source Language words for each context length: the widest contexts (5-grams) are looked up first before moving on to narrower contexts and ending up with looking up only the translation candidate.

Each n-gram is looked up in the Language Model, and for each context part, the n-grams are ordered according to LM score. Table 7 shows a few examples of such generated n-grams with their corresponding scores from the LM. The target candidates (*italics*) are then extracted from the ordered list of Target Language n-grams. Note that there are no scores for the 4- and 5-grams here; an expected outcome of generating TL n-grams with direct translation. This results in a ranked list (n-best) of translation candidates. Duplicate translation candidates are filtered from the n-best list.

How stopword-filtered n-grams that correspond to the lemma-based LM are created can be explained by introducing **slots**. A not stopword-filtered model would, given a 3-gram order (slots $[0, 3]$, $[-1, 2]$, and $[-2, 1]$) generate the n-grams created by combining the translations of words:

n	n-gram	LM score
5	land mittig *bank* nachsuchen </s>	Not found
4	mittig *bank* nachsuchen </s>	Not found
3	mittig *bank* nachsuchen	Not found
3	*kredit* anfragen </s>	-0.266291
2	mittig *bank*	-3.382560
2	zentral *blutbank*	-5.144870
1	*bank*	-3.673000

Table 7: Target language n-gram examples from look-ups of stopword-filtered lemmata *country central bank request* reported in log scores. The first three n-grams were not found in the Language Model.

bank if they, central bank if, country central bank

that subsequently are listed and looked up in the Language Model. For a stopword-filtered model, however, the corresponding bag of n-grams is the exhaustive list of translations of lemmata:

bank request </s>, central bank request, country central bank

some translations of which are shown in Table 7. The slot is the sequence of words determined by the indices by the tuple in its specification[3], with the focus word at slot $[0, 1]$.

A priority list of slots can be configured, such that the algorithm prioritizes the n-grams of various sizes. Thus, if $[-4, 1]$ is the top priority, n-grams generated with four words of context before the focus word are ranked highest until the specified number of results is reached. If asking for five answers (and the $[-4, 1]$ set of n-grams could only yield two suggestions), the algorithm would have to move on to the next slot in the priority list. In practice, many instances have to revert to n-grams comprised by the slot $[0, 1]$ — the direct translation of the focus word.

5.3.2 *Using Dictionaries*

After creating an n-best list of translation candidates with the n-gram method, more candidates are generated by looking up multiword entries in bilingual dictionaries. The existence of multiword entries in the dictionary corresponding to adjacent lemmata in the source context or translation candidates in the target context is taken as a strong indicator of the suitability of a particular translation candidate. Such

3 Inspired by Python-style indexing.

Algorithm 2 SL algorithm to rank translation candidates (tcands) for SL lemma b given list of tcands

```
1: procedure FINDCAND(list $\mathrm{rlist}$,SL-lemma $\mathrm{b}$, const $\mathrm{tcands}$)
2:     $\mathrm{comblemmata} \leftarrow \mathrm{list(previouslemma(b) + b, b + nextlemma(b))}$
3:     for $\mathrm{lem} \in \mathrm{comblemmata}$ do
4:         $\mathrm{c} \leftarrow \mathrm{sl\text{-}dictionary\text{-}lookup(lem)}$
5:         if $\mathrm{c} \in \mathrm{tcands}$ then $\mathrm{rlist} \leftarrow \mathrm{list(c + rlist)}$
6:         end if
7:     end for
8:     return $\mathrm{rlist}$
9: end procedure
```

entries are added to the top of the n-best list, which represents a strong preference in disambiguation. An example is how the translation candidate *Commerzbank* is ranked first because it is found as a translation of the dictionary entry *Community Bank*. (The n-gram matching method also used dictionaries to look up translation candidates and Target Language translations of the context words.)

Two separate procedures are used to find such indicators, a direct procedure based on the source context and an indirect procedure based on the weaker Target Language context, as follows:

SOURCE LANGUAGE (SL) METHOD: If there is a dictionary entry for the source word and one of its adjacent words, search the set of their translations for any of the provided translation candidates for the word (see Table 5 for examples). Specifically, the translations of the combination of the source word and an adjacent word in the context are matched against the translations candidates for the word. Algorithm 2 shows this method is pseudocode.

TARGET LANGUAGE (TL) METHOD: If a translation candidate looked up in the reverse direction matches the source word along with one or more adjacent words, it is a good translation candidate. The target translation candidates are looked up in a TL–SL dictionary and multiword results are matched against SL combinations of disambiguation words and immediate context. Algorithm 3 shows this method in pseudocode.

The dictionary entry for either the target word or translation candidate is matched against the immediate context in both methods. Such lookups are done exhaustively for all combinations of translations of the words each slot. A translation candidate is only added once if it generates hits with either (or both) methods.

Algorithm 3 TL algorithm to rank translation candidates (tcands) for SL lemma b given list of $tcands$

```
1: procedure FINDCAND(list $rlist$,SL-lemma $b$, const $tcands$)
2:     for $cand \in tcands$ do
3:         $translist \leftarrow list(cand, tl\text{-}dictionary\text{-}lookup(cand)) + translist$
4:     end for
5:     for $cand, trans \in translist$ do
6:         if $previouslemma(b) + b || b + nextlemma(b) \in trans$ then
7:             $rlist \leftarrow list(cand) + rlist$
8:         end if
9:     end for
10:    return $rlist$
11: end procedure
```

(4) country central **bank** request

Example (4) revisits the running example, stopword-filtered, and with lemmatized context. The example generates two Source Language Multiword Expressions; *central bank* and *bank request*. With the SL method, these word combinations are looked up in the dictionary where the *zentralbank* entry is found for *central bank*, which is also found as a translation candidate for *bank*. The Target Language method works in the reverse order by looking up the translation candidates in the TL–SL direction and creating a list of translations in the SL. If a lemma combination is found in this list, the corresponding translation candidate is ranked on top. Since the entry *zentralbank:central bank* is found in the dictionary, with a translation matching the Source Language context, *zentralbank* is assumed to be a correct translation.

5.3.3 *Evaluation*

Two evaluation measures were used for the CL-WSD shared task: the **Best** and **Out-of-Five** scores [Lefever and Hoste, 2010a]. The Best criterion was intended to measure how well a system succeeded in delivering the best translation, i. e., the one preferred by the majority of annotators. The Out-of-Five (OOF) criterion measures how well the top five candidates from the system match the top five translations in the gold standard:

$$\mathrm{OOF}(i) = \frac{\sum_{a \in A_i} freq_i(a)}{|H_i|}$$

where H_i denotes the multiset of translations proposed by humans for each test data sentence t_i ($1 \leqslant i \leqslant N$, with N being the number of test items). A_i is the set of translations produced by the system for test item i. Since each translation has an associated count of annotators that selected it, there is for each t_i a function $\mathtt{freq}_i$, which returns this count for each term in H_i (0 for all other terms), and a function $max\ \mathtt{freq}_i$, which returns the maximal count for any term in H_i.
For the first example in Table 5, the cardinality of the multiset is: $|H_1| = 8$:

$$\begin{cases} H_1 = \{\text{bank, bank, bank, bank, zentralbank,} \\ \qquad \text{bankengesellschaft, kreditinstitut, finanzinstitut}\} \\ \mathtt{freq}_1(\text{bank}) = 4 \\ \mathtt{freq}_1(\text{zentralbank}) = 1 \\ \mathtt{freq}_1(\text{bankengesellschaft}) = 1 \\ \mathtt{freq}_1(\text{kreditinstitut}) = 1 \\ \mathtt{freq}_1(\text{finanzinstitut}) = 1 \\ max\mathtt{freq}_1 = 4 \end{cases}$$

This equates to the sum of all top-3 preferences given to the translation candidates by all annotators.

For the Out-of-Five evaluation, systems are allowed to submit up to five candidates of equal rank. OOF is a recall-oriented measure with no additional penalty for precision errors, so there is no benefit in outputting less than five candidates. Concerning the previous example from Table 5, the maximum score is obtained for system output

$$A_1 = \{\text{bank, bankengesellschaft, kreditinstitut, zentralbank, finanzinstitut}\}$$

is

$$\text{OOF}(1) = (4+1+1+1+1)/8 = 1$$

whereas

$$A_2 = \{\text{bank, bankengesellschaft, nationalbank, notenbank, sparkasse}\}$$

would give

$$\text{OOF}(1) = (4+1)/8 = 0.625$$

Note that the maximum OOF score is not always 1 (i. e., it is not normalized) since the gold standard sometimes contains more than five translation alternatives.

	Source Language			Target Language		
Dictionary	D1	D2	comb	D1	D2	comb
Mean	3.25	1.5	3.25	**12.65**	11.45	14.20
Total	65	30	65	253	229	284

Table 8: Number of instances with a translation candidate over test set (20 words).

	MF	MFA	5-gram	5-gram + Dict	All Dict Comb	VSM Model
Top	51.77	68.71	52.02	52.74	24.67	**55.92**
Low	1.76	9.93	14.09	**15.40**	0.00	10.73
Mean	21.18	34.61	30.36	**36.38**	10.13	30.30

Table 9: F1-scores on SemEval data across methods.

For evaluation of overall system performance, the average of OOF scores across all test items for a single source word was used, with F1-score reported as a harmonic mean of the precision and recall of the OOF scores.

5.4 EXPERIMENTS

Experiments were carried out both on the trial and test data described in Section 5.2 (5 trial and 20 test words; with 20 and 50 instances of each word, respectively; in total 1100 instances in need of disambiguation).

Baseline scores from the SemEval dataset were used in the evaluation. A stopword-filtered 5-gram model built with the IRSTLM language modeling toolkit [Federico and Cettolo, 2007, Federico et al., 2008] was used as a benchmark.

Figure 8 shows how many instances that produced translation candidates with the Source and Target Language methods, respectively.

5.4.1 *Results*

Table 9 shows a result overview for the different methods on the dataset. For each method, the three lines give both the best (*Top*) and worst (*Low*) scoring terms, as well as the mean value for all terms in the dataset.

	Source language			Target language			All
Dictionary	D1	D2	comb	D1	D2	comb	comb
Top	*8.89*	6.99	*8.89*	22.71	24.43	25.34	24.67
Low	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Mean	2.71	0.99	*3.04*	8.35	7.10	*9.24*	**10.13**

Table 10: F1-scores for individual dictionaries.

The maximum attainable score for each of those would be 99.28, 90.48 and 95.47, respectively, but those are perfect scores not reachable for all items due to the way scores are calculated (revisit OOF-scoring in Section 5.3.3). Instead, the columns Most Frequent (MF) and Most Frequently Aligned (MFA) give the baseline scores for the SemEval dataset. MF is the translation most frequently seen in the corpus and MFA the translation most frequently aligned in a word-aligned parallel corpus (Europarl [Koehn, 2005]). The next columns show results for using only a stopword-filtered *5-gram* model, and when combining the 5-gram model with the dictionary approach (*5-gram + Dict*).

The next-to-last column (*All Dict Comb*) shows how the dictionary methods performed on their own, i. e., without the support of a high-recall method. The combined dictionary approach has low recall (see Table 8) and does not provide a good solution to the overall problem by itself. Due to high precision, however, the method can enhance the n-gram method, which already produces acceptable results. Finally, the column *VSM Model* as comparison gives the results obtained when using a Vector Space Model for Word Sense Disambiguation [Marsi et al., 2011].

A comparison of the dictionary approach to state-of-the-art monolingual solutions to the Word Translation Disambiguation problem on this dataset shows that the method scored better for the terms' lowest and mean scores, but not for the top scoring [Lynum et al., 2012]. The Top result row shows that the Vector Space Model produced the overall best score for a single term. However, the method combining a 5-gram Language Model with the dictionary approach was best both at avoiding too low scores for any single term and when comparing the mean scores for all the terms.

5.4.2 *Dictionary Results*

A more fine-grained analysis of the precision scores of the dictionary experiments could help explain the improvements. Table 10 shows

	Source language			Target language			All
Dictionary	D1	D2	comb	D1	D2	comb	comb
coach	1.00	0.00	1.00	0.21	0.00	0.21	0.21
education	0.83	0.67	0.83	0.47	0.62	0.54	0.53
execution	0.00	0.00	0.00	0.17	0.22	0.17	0.17
figure	1.00	0.00	1.00	0.51	0.57	0.55	0.55
job	0.88	0.80	0.94	0.45	0.78	0.46	0.44
letter	1.00	0.00	1.00	0.66	0.75	0.62	0.66
match	1.00	1.00	1.00	0.80	0.50	0.80	0.80
mission	0.71	0.33	0.71	0.46	0.37	0.36	0.36
mood	0.00	0.00	0.00	0.00	0.00	0.00	0.00
paper	0.68	0.17	0.68	0.53	0.35	0.55	0.55
post	1.00	1.00	1.00	0.39	0.48	0.45	0.48
pot	0.00	0.00	0.00	1.00	1.00	1.00	1.00
range	1.00	1.00	1.00	0.28	0.37	0.30	0.30
rest	1.00	0.67	1.00	0.60	0.56	0.56	0.58
ring	0.09	0.00	0.09	0.37	0.93	0.38	0.38
scene	1.00	0.00	1.00	0.50	0.42	0.44	0.50
side	1.00	0.00	1.00	0.21	0.16	0.23	0.27
soil	1.00	0.00	1.00	0.72	0.58	0.66	0.69
strain	0.00	0.00	0.00	0.51	0.88	0.55	0.55
test	1.00	1.00	1.00	0.62	0.52	0.57	0.61
Mean	0.84	0.74	0.84	0.50	0.56	0.49	0.51

Table 11: Custom precision scores for all terms in test data.

results for each dictionary approach on the test set: Target Language look-up contributes more to providing good translation candidates than the Source Language methodology. At least one guess was required to make the scoring script provided with the SemEval data work. As a consequence, the results in this table were calculated as though the methods guessed a *wrong* candidate word if they provided none. This gauges how well the approach performs overall on the task without combination with methods that offer higher recall.

Table 11 lists the results of filtering out the instances for which no candidate translation was produced and taking the average precision scores only over these, for each term in the test data. Table 12 summarizes the results. In the last rows, markedly different mean precision

	Source language			Target language			All
Dictionary	D1	D2	comb	D1	D2	comb	comb
Top	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Low	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Mean	0.84	0.74	0.84	0.50	0.56	0.49	0.51

Table 12: Custom precision scores summarized on dictionary and method use.

scores appear: the Source Language method has higher precision on the suggestions it makes than its Target Language counterpart. (Table 8 on Page 118 showed how this higher precision is offset by lower coverage, with far fewer instances producing a translation candidate with the dictionary lookup methods.)

Furthermore, Table 11 shows a difference in the precision of the SL and TL approaches, coinciding with more candidates produced by the latter. Several words give 100% precision scores for at least one dictionary, while a few give 0% precision for some dictionaries. The word *mood* has 0% precision for both dictionaries in both directions.

5.5 DISCUSSION

This chapter presented experiments into Cross-Language Word Sense Disambiguation and Multiword Expression identification using dictionary lookups based on the concatenation of a focus word with adjacent words in both the Source Language text and Target Language candidate translations. The top-ranked translation is identified by disambiguating the available translation candidates. Generating Multiword Expressions lookup strings by using both neighboring words improved disambiguation performance on the SemEval 2010 English-German Cross-Lingual Word Sense Disambiguation task datasets.

Multiword Expressions are often missing from dictionaries, potentially resulting in contorted Machine Translation output. The experiments in this chapter used general-purpose dictionaries, but domain-specific dictionaries and word lists could also have been applied to translate particular MWEs. Since the use of specialized terms is a feature of domain-specific language (see Chapter 4), the treatment of such is relevant for Domain Adaptation. Sennrich [2013b] also considered the treatment of MWEs important for Domain Adaptation.

Sennrich [2013a] noted the importance of Multiword Expressions for Statistical Machine Translation in general, due to the propensity of SMT models to overgeneralize translations that are used in only specific contexts. The English expression *of course*, whose idiomatic French correspondence, *bien sûr*, could lead to *sûr* (literal translation: *sure*) being learned as the translation of *course* was provided as an example. Sennrich used a measure of phrase **flexibility** to provide better translations, which refers to the set of different phrases an MWE may translate into, similar to different **continuations** in Kneser-Ney smoothing (see Section 3.2.3.3). Thus, incorrect translations could also appear in other contexts, due to the frequency of the expression. The phrase flexibility measure, quantifying the propensity to occur also in other contexts, was integrated as log-linear features.

The experimental results presented above were compared to the baselines from the SemEval data and an n-gram-based method. However, restricting the task to only picking among a predefined set of translation candidates makes the task easier than the full task of determining the sense inventory. Thus, results are not directly comparable to the SemEval results [Lefever and Hoste, 2010a, 2013].

Additionally, differences between the Source and Target Language methods of using dictionary lookups were addressed, where the former has very high precision (0.84) but low coverage, while the TL method compensates lower precision (0.51) with markedly better coverage. These results mean that the errors using the first method are profound; without a hit on a candidate, no solution will be provided, but with one, this candidate is likely to be correct. Consequently, this method pairs well with other, recall-strong methods such as the baseline n-gram-based method.

The SL dictionary method provided answers to only between 1.5 and 3.25 of 50 instances per word on average, depending on the dictionary. The difference owes mainly to the TL method matching any adjacent lemma to the focus word with the translation of the predefined translation candidates, whereas the SL method matches dictionary entries of the lemmata of the focus word and its adjacent words combined, to the same list of translation candidates. False positives (identifying dictionary entries for incorrect translation candidates) must be expected for the TL method. The use of specialized dictionaries or data mining to extend dictionaries by leveraging resources such as Wikipedia or Wiktionary could mitigate this.

Using extended dictionary lookups provides valuable information for disambiguating translation. In future work, this could be integrated into existing feature-based Machine Learning approaches, such as the Vector Space Model approach [Lynum et al., 2012]. The extended use of dictionaries proves a valuable source of information for disambiguation and can introduce low-cost phrase-level translation to quantitative Word Sense Disambiguation approaches such as n-gram or Vector Space Model methods.

Searching the entire context of the Source Language instances using this method — and not just a focus word as provided by test data — could generate translations of Multiword Expressions elsewhere in the sentence, for a better translation of the sentence as a whole. Integration into a Statistical Machine Translation pipeline would be necessary to test this idea in practice. Differences between dictionaries could also be explored, e. g., by giving more weight to translations found in two or more dictionaries. Additionally, differences between the Source and Target Language approaches should be explored in further detail, to see if the higher precision of the SL method is possible to retain with the added coverage of the TL method. Moreover, the method should be tested on more language pairs, e. g., to assess the potential impact of less related languages on results.

6 BUILDING LARGE LANGUAGE MODELS

This chapter will present experiments into building large Language Models with parallel processing. First, Section 6.1 presents related work and Section 6.2 the data collection. Section 6.3 explains the method used, and Section 6.4 shows the experiments. Finally, Section 6.5 discusses the findings and sums up.

Broadly, in a Statistical Machine Translation context, a Language Model (LM) contributes to the fluency of the Target Language string (conversely, a Translation Model contributes to the adequacy [Banchs et al., 2015], see Section 3.2). An LM will assign a higher probability to strings it has seen compared to those observed less frequently during training or not at all. A well-formed sentence will include longer passages of text found in the LM, whereas the same sentence in random order is unlikely to do so and be assigned a lower score. Thus, during decoding of SMT models, the LM will contribute to well-formed translation hypotheses being ranked higher.

Language Models are used in many Natural Language Processing tasks, such as Machine Translation, Automatic Speech Recognition, and Optical Character Recognition. In MT, one or more LMs can be used during decoding in a standard Statistical Machine Translation configuration. Moreover, they can also be used in separate disambiguation or Domain Adaptation modules, during decoding or reranking stages. In commercial systems, LMs can be as large as 1 Terabyte (TB) [Lopez and Post, 2013]. Also for creating LMs not nearly that large, building them in a reasonable amount of time requires resources beyond single machines.

6.1 RELATED WORK

Brants et al. [2007] built large Language Models using distributed techniques. The authors reported decreased perplexity and better n-gram coverage with increasing numbers of tokens, and also that larger n-gram models improved BLEU scores for Machine Translation tasks. The Map-Reduce framework [Dean and Ghemawat, 2008] was used for distributed compilation of LMs, parallelizing the counting step of the n-gram model creation. This method has an initial **mapping** step where words (keys) and values are gathered on different parts of the data, followed by a **reduce** step aggregating the values of the same keys on the same nodes. The result is a key/value list

(i. e., n-gram counts) for the entire corpus. LMs with Kneser-Ney (see Section 3.2.3.3) smoothing were built in two days for a 30G corpus.

Storing and processing large n-gram Language Models have been dealt with using different methods; Talbot and Osborne [2007] used a Bloom filter with logarithmically quantized n-gram frequency counts, i. e., a **lossy** randomized representation efficiently encoding the n-grams with their corresponding frequency information. This randomized language modeling (commonly referred to as `RandLM`) can give significant storage space reductions, at the cost of additional false positives (the filter may report that an item not in the set is a member) and speed.

In contrast, Pauls and Klein [2011] presented some compact **lossless** implementations based on tabular tries storing only the suffix of the n-gram (the last word) together with an offset encoding the context (the remaining words). Working on the Web1T 5-gram corpus [Brants and Franz, 2006], Pauls and Klein encoded each n-gram in only 23 bits, in the best case reducing storage requirements to only 1/4 and also improving on the best previous lossy representations. Encoding the context also gives faster processing (since there is no need to look up the context again when moving on to the next word). Combined with a direct-mapped cache, the method gained substantial speed-ups (up to 300%).

Heafield [2011] introduced the language modeling library `KenLM` and compared using regular hash tables to tries. Results showed that a linear probing hash table method gave significantly faster processing while tries produced smaller data structures. Furthermore, Heafield addressed how a lossy compression of the trie pointers further reduces necessary storage space but concluded that linear probing hash tables are preferable if processing speed is more important than reduced memory usage. On the other hand, `RandLM` is potentially the most memory efficient approach, even though lossless compression can further optimize the memory allocation needed by the tries. Raj and Whittaker [2003] reported an 86% reduction in Language Model size for a marginal increase in word error rate.

6.2 DATA

During development of the language modeling scripts, three corpora of English (enTenTen, with 3.5Bn Words), German (deTenTen, 3.2Bn Words), and Italian (itTenTen, 2.2Bn Words) were used, all originating in the previous "Web as Corpus" corpora known as UKWac, DeWac and ItWac [Baroni and Kilgarriff, 2006], respectively. The TenTen corpora were mined from the web and provided in a "vertical" corpus format, with one word per line [Kilgarriff et al., 2010, 2011]. On each line, words presented in three forms (tab-separated): original form, lemma, Part-of-Speech (POS), and (lempos) (lemma + POS). Section 8.2 will return to the enTenTen corpus, and Listing 3 on Page 164 shows an excerpt.

6.2.1 *Corpus Noise*

The corpora contained noise also after tagging, which was removed before building the models, because certain high UTF-8 characters could break the language modeling software. Many of the problems stemmed from the web-corpora being encoded in a mixture of character sets. Scripts were written to handle these errors, as well as to unify different representations of dates and numbers into the collection tokens `@date` and `@card`. The most notable types of noise in the German corpus were:

- Words beginning with special characters (e. g., *-Bus*).
- Higher order special UTF characters, e. g., different newline characters. With the `Unix` command `less`, these characters are rendered as, e. g., *U+0084*, but with `cat` and `more` they are invisible. In the LM software, they appear as **token ghosts**, i. e., tokens comprised of empty strings.
- Umlauts being rendered differently (from various character sets).
- Incorrectly split words (e. g., *Bewaff- net*).
- Repeated words (often multiple times) as tokens.
- Very long words (usually between 50 or 100 characters, possibly created by keyboard hammering).

6.2.2 *Preprocessing*

Each corpus was tokenized and Part-of-Speech-tagged with the `Tree-Tagger` [Schmid, 1994]. Preprocessing included stripping higher-order UTF characters that would cause the `IRSTLM` software to crash or give undesired output, e. g., when tokens rendered as spaces would produce spurious n-grams. Before building the Language Models, the corpora were transformed from the vertical format to a horizontal format with one *sentence* per line encapsulated in `<s> </s>` sentence boundary markers. This format was a requirement of the language modeling software.

The deTenTen corpus produced a large number of unique tokens. The highly compounding nature of the German language resulted in many tokens ending with "–", as compounds were rendered in *split* mode (i. e., by enumeration or line breaks) in the web material. Date formats also varied extensively.

The corpora used for the experiments were sectioned into Training and Test corpora with a `Perl` script, randomly sampling 10% of the lines for the latter. Language Model perplexity was calculated on these test corpora.

6.3 METHOD

The n-gram models were built with the standard tool `IRSTLM`, the IRST Language Modeling Toolkit [Federico and Cettolo, 2007, Federico et al., 2008]. The `IRSTLM` framework was adapted to the `OpenPBS`[1] queue handler to distribute the task to a cluster of machines[2].

The alternative to adapting the parallelization scripts from `IRSTLM`, the similar `SRILM` [Stolcke et al., 2011], or other already existing, openly available language modeling toolkits, e. g., `RandLM` or `KenLM` (see Section 6.1) would have been to implement an n-gram modeling tool in a parallel programming framework such as `MPI` or `OpenMP`. This approach was discarded because it was unlikely to result in any significant gain in performance over the chosen PBS alternative, which produced acceptable results. Language Models of various types (lemma-based, word-based, POS-based and combinations thereof) were built.

The `IRSTLM` software package already had scripts for parallel treatment of data developed for a proprietary implementation of the PBS system, and these were adapted to the slightly different syntax of `OpenPBS`. Such job schedulers receive jobs that are submitted to a queue and handle their execution depending on access rights and system load.

Figure 24 lists the processing steps of the Language Model creation. The **dictionary** (i. e., list of unigrams) was sectioned into lists that were frequency-balanced: a list of unigrams for each section, with a similar amount of total frequency counts. More unigrams are needed to reach approximately the same total, as unigram frequencies go down. Top unigrams alone could have a frequency well above the average per section, but, obviously, a one-entry list cannot be split.

Subtasks were submitted to a queue handler, enabling them to be done in parallel (except the first step, dictionary collection, and the last step, merging) as processing resources became available on the grid. `IRSTLM` provides similar scripts for diving the building of LMs

1 `http://www.mcs.anl.gov/research/projects/openpbs` (Last visited: March 27, 2019.)

2 A 96 node cluster consisting of equal amounts of nodes with 48G and 24G RAM was used.

1. Compile a dictionary for the entire corpus.
2. Section the corpus into n parts according to word frequency.
3. Count n-grams for each of these parts.
4. Compute (sub-)LM scores for each part.
5. Merge all counts into one LM.

Figure 24: Overview of the processing steps in parallel Language Model building.

into smaller steps to overcome memory constraints, also for serial architectures, such that the parallelizable steps 3 and 4 are processed separately, but serially, before the parts are merged. The parallel processing reliably gave the same output as serial processing.

The changes required to modify the `Sun Grid PBS` script to the `OpenPBS` format mostly related to the control of workflow, i. e., submission of jobs. Specifying that some jobs submitted to the queue should halt until the successful execution of other jobs on which they depend finished was necessary, such as n-gram counting depending on the completion of the dictionary creation. Also, to avoid submitting too many jobs at once, the shell script waited for the dictionary compilation before the n-gram counting and sub-LM jobs were submitted to the queue.

Steps 3 and 4 can be done in parallel by distributing jobs to a queue handler. A shell script submits the jobs to the PBS queue and delays merging until all have finished successfully. When submitting the jobs in steps 3 and 4, each sub-LM job depended on the corresponding job for n-gram counting (did not start before it was finished), as the n-gram counts for each section needed to be compiled before their respective sub-LM (LM for that section) could be computed.

The `IRSTLM` framework can output LMs in the ARPA LM format, an intermediate, internal format (iArpa), as well as a compiled version for quicker access with `IRSTLM` tools. The (D)ARPA format is a text file with of a header specifying the number of n-grams of each order, which are ordinally listed below, in sections with lines containing log-probabilities and n-grams[3].

3 The SRILM [Stolcke, 2002, Stolcke et al., 2011] documentation has a good explanation of the format: `http://www.speech.sri.com/projects/srilm/manpages/ngram-format.5.html` (last visited: March 27, 2019).

English	Training		Test	
	Lines	Words	Lines	Words
Lemma	108.4	3150.5	12.0	350.3
Fullform	107.4	3122.6	11.9	347.0

(a) Size of the English Corpora

German	Training		Test	
	Lines	Words	Lines	Words
Lemma	141.8	2837.0	15.8	315.3
Fullform	141.1	2809.0	15.7	312.2

(b) Size of the German Corpora

Italian	Training		Test	
	Lines	Words	Lines	Words
Lemma	78.5	2913.4	8.7	323.9
Fullform	77.9	2851.2	8.7	317.0

(c) Size of the Italian Corpora

Table 13: Sizes of Training and Test corpora.
Figures reported in millions of lines and words.

6.4 EXPERIMENTS

Two corpora were extracted for each language; (i) a corpus comprised of words represented by lemmata and (ii) a corpus with words in full forms. Henceforth, these corpora are referred to as the Lemma and Fullform, respectively. Differences in the extraction methods from the original format of the corpora explain the minor differences in size. Table 13 shows the size of the training and test corpora in the different languages.

5- and 7-gram models were built with Kneser-Ney smoothing for all extracted corpora. For the Fullform corpora, 5-gram models with and without pruning of singleton n-grams (i. e., n-grams occurring only once in the corpus) were built as well. The number of words reported in Table 13 is lower for the Fullform corpora because some tokens from the Fullform corpora were discarded in preprocessing, but retained in the lemmatized versions.

6.4.1 *Corpora Statistics*

Evaluating Language Models extrinsically would have complemented the information from intrinsic measures. However, for these experiments, no specific task was intended, and the held-out test corpora were used to compute perplexity and Out-of-Vocabulary (OOV) statistics. Perplexity statistics on the Test corpora were computed for the pruned models, but for the unpruned models, this was deferred due to memory constraints.

The `IRSTLM` [Federico et al., 2010] language modeling software's standard functionality offers computation of such corpus statistics, which were collected for all three languages and corpus types.

Table 14 shows results on the Test corpora for twelve Language Models. The differences in word numbers in the Test corpora compared to the dictionary sizes presented in Table 13 are explained by how sentence boundary markers are counted. In Table 14, they are counted once per line, whereas all markers are counted in Table 13, which means that the differences equate to the line numbers of the models.

The 7-gram models had lower perplexity than the 5-gram models for lemma-based Language Models, whereas the opposite was true for the for English and Italian Fullform models (although barely). This result was observed both with and without the effect of the OOV words taken into account. On the German corpus, however, the 7-gram model had markedly lower perplexity than the 5-gram model also for the Fullform corpus.

The difference in perplexity between the Lemma and Fullform-based models were larger for the deTenTen corpus, suggesting a higher degree of noise in this corpus (see Section 6.2.1).

6.4.2 *Dictionary Growth Curves*

In addition to the statistics above, a Dictionary Growth Curve (DGC) was created, i. e., a curve showing the number of n-grams above the orders 0–9, with the OOV frequency in each category on the held-out Test corpus. Tables 15 and 16 on Pages 133 and 134 show these DCGs for the Lemma and Fullform corpora, respectively. The first three columns of each table show the percentage of words in the training corpus whose frequencies are over 0 (all of them, 100%), those having a frequency over 1 (40%), and so on.

The number of dictionary entries (unigrams) should be the same as the number of unigrams in an unpruned Language Model. However, due to the implementation of Kneser-Ney smoothing, the singleton

English	Nw	PP	PPwp	Noov	OOV
5-Lemma	338M	66,467	1,483	517K	0.15%
7-Lemma	338M	**65,721**	**1,466**	517K	0.15%
5-Fullform	335M	**59,606**	**1,432**	564K	0.17%
7-Fullform	335M	59,813	1,437	564K	0.17%

(a) EN Test Corpus

German	Nw	PP	PPwp	Noov	OOV
5-Lemma	299M	29,740	2,088	1347K	0.45%
7-Lemma	299M	**29,264**	**2,054**	1347K	0.45%
5-Fullform	296M	62,139	4,606	1,423K	0.48%
7-Fullform	296M	**60,933**	**4,516**	1,423K	0.48%

(b) DE Test Corpus

Italian	Nw	PP	PPwp	Noov	OOV
5-Lemma	315M	81,825	1,682	434K	0.14%
7-Lemma	315M	**79,060**	**1,625**	434K	0.14%
5-Fullform	308M	96,281	**2,208**	483K	0.16%
7-Fullform	308M	**99,458**	2,280	483K	0.16%

(c) IT Test Corpus

Table 14: Experimental results on the Test corpora. N_w is the total number of words in the evaluation corpus (reported in millions of words), PP is the perplexity, and PP_{wp} reports the contribution of out-of-vocabulary (OOV) words to the perplexity. The out-of-vocabulary word term OOV is defined as $(N_{oov}/N_w) * 100$, with N_{oov} being the number of OOV words (reported in thousands of words).

Freq	Entries	Percent	Freq	OOV
>0	19,300,334	100.00%	<1	0.45%
>1	7,404,928	38.37%	<2	0.64%
>2	4,841,598	25.09%	<3	0.76%
>3	3,706,439	19.20%	<4	0.86%
>4	3,042,511	15.76%	<5	0.94%
>5	2,606,399	13.50%	<6	1.01%
>6	2,293,273	11.88%	<7	1.07%
>7	2,056,786	10.66%	<8	1.13%
>8	1,870,568	9.69%	<9	1.18%
>9	1,719,753	8.91%	<10	1.22%

(a) Dictionary Growth Curves for German Lemma Corpus

Freq	Entries	Percent	Freq	OOV
>0	7,507,448	100.00%	<1	0.15%
>1	3,072,234	40.92%	<2	0.22%
>2	2,074,727	27.64%	<3	0.26%
>3	1,628,826	21.70%	<4	0.29%
>4	1,362,639	18.15%	<5	0.32%
>5	1,185,035	15.78%	<6	0.35%
>6	1,054,988	14.05%	<7	0.37%
>7	956,259	12.74%	<8	0.39%
>8	877,378	11.69%	<9	0.41%
>9	813,647	10.84%	<10	0.43%

(b) Dictionary Growth Curves for English Lemma Corpus

Freq	Entries	Percent	Freq	OOV
>0	6,475,359	100.00%	<1	0.14%
>1	2,778,546	42.91%	<2	0.20%
>2	1,903,603	29.40%	<3	0.24%
>3	1,511,911	23.35%	<4	0.27%
>4	1,275,415	19.70%	<5	0.30%
>5	1,116,931	17.25%	<6	0.32%
>6	1,000,423	15.45%	<7	0.34%
>7	911,485	14.08%	<8	0.36%
>8	840,714	12.98%	<9	0.38%
>9	782,787	12.09%	<10	0.40%

(c) Dictionary Growth Curves for Italian Lemma Corpus

Table 15: Dictionary Growth Curve for the Lemma corpora.

Freq	Entries	Percent	Freq	OOV
>0	8,203,706	100.00%	<1	0.17%
>1	3,335,431	40.66%	<2	0.24%
>2	2,280,889	27.80%	<3	0.28%
>3	1,801,269	21.96%	<4	0.32%
>4	1,516,574	18.49%	<5	0.35%
>5	1,325,480	16.16%	<6	0.38%
>6	1,185,591	14.45%	<7	0.40%
>7	1,079,341	13.16%	<8	0.43%
>8	994,459	12.12%	<9	0.45%
>9	925,466	11.28%	<10	0.47%

(a) Dictionary Growth Curves for English Fullform Corpus

Freq	Entries	Percent	Freq	OOV
>0	20,775,474	100.00%	<1	0.48%
>1	8,163,441	39.29%	<2	0.68%
>2	5,463,441	26.30%	<3	0.81%
>3	4,253,230	20.47%	<4	0.92%
>4	3,543,995	17.06%	<5	1.01%
>5	3,071,589	14.78%	<6	1.09%
>6	2,731,179	13.15%	<7	1.15%
>7	2,470,988	11.89%	<8	1.21%
>8	2,263,809	10.90%	<9	1.27%
>9	2,096,181	10.09%	<10	1.32%

(b) Dictionary Growth Curves for German Fullform Corpus

Freq	Entries	Percent	Freq	OOV
>0	7,365,655	100.00%	<1	0.16%
>1	3,169,535	43.03%	<2	0.22%
>2	2,222,344	30.17%	<3	0.27%
>3	1,786,668	24.26%	<4	0.31%
>4	1,525,238	20.71%	<5	0.34%
>5	1,348,829	18.31%	<6	0.37%
>6	1,219,203	16.55%	<7	0.39%
>7	1,119,484	15.20%	<8	0.41%
>8	1,040,111	14.12%	<9	0.44%
>9	974,612	13.23%	<10	0.46%

(c) Dictionary Growth Curves for Italian Fullform Corpus

Table 16: Dictionary Growth Curves for the Fullform corpora.

n-grams for orders 1 and 2 were pruned. Markedly higher number of dictionary entries and OOV rates for the deTenTen corpora stand out.

6.4.3 *N-gram Counts*

N-gram level counts were extracted from the corpora by looking at the header of the output Language Model files. Table 17 on Page 137 shows the number of n-grams in the models for all three languages. For the English and Italian corpora, 4-grams was the most frequent n-gram order in the LMs, whereas the highest amount of non-singleton n-grams was found in the bigram category for the German corpus.

Unpruned models were built for the Fullform corpora. Comparing those to the singleton-pruned Language Models shows that the number of n-grams grows quickly. These unpruned models required about 31G of storage (in the intermediate iArpa format), compressed with `bzip`[4].

6.4.4 *Computation Times*

Because of variations in cluster load owing to other users, it is difficult to report comparable computation times for the different models. However, as an indication, the whole process of building any one of these Language Models could take 8–12 hours, using the hardware configuration described in Section 6.3.

On average, the parallelized jobs took about 1.5 hours, except the jobs counting n-grams based on the most frequent unigrams that could take up to 5 hours to finish. The merging of the sub-LMs would have to wait for the creation of all the sub-models to finish. Hence, it was necessary to wait for these initial jobs to start merging and exit the script.

It would be possible to find an ideal number of jobs to minimize the total computation time, where the smaller jobs were big enough to correspond in time-usage to the jobs counting n-grams for the most frequent unigrams. If the jobs performing the counting step required a similar amount of time, fewer resources would be constrained by waiting jobs. With an ideal number of jobs, the total computation time would be smaller, requiring fewer resources from the cluster. However, the total time the script needs to return with a Language Model would not be changed as the counting of the most frequent n-grams would still be the lower bound.

4 `http://www.bzip.org/` (Last visited: March 27, 2019.)

6.5 DISCUSSION

Experimenting with building and rebuilding n-gram models from large corpora requires efficient computation. The above experiments show how Language Models can be efficiently built using the `IRSTLM` framework, adapted to the `OpenPBS` job scheduler. The Language Models created in the above experiments could not fit in memory on a single machine (available at that time), and exploring alternative ways of building them was necessary. In addition to the proposed method, vector quantization could have been used.

With web corpora, noise can be a problem, and some steps that can be taken to reduce the number of unique tokens were identified through these experiments. Such tokens are not necessarily members of the language but have rather come about as the result of idiosyncrasies in the corpus processing.

The low degree of Out-of-Vocabulary words, also when using corpora that retain capitalization and inflected forms (such as the Fullform corpora), is an indication of the effect of adding more data. The experiments indicate differences in noise between the English, German and Italian corpora, but more experiments are necessary to establish such a relation.

English		Fullform	
Order	5 pruned	5 unpruned	7 pruned
1-gr	3.3	3.3	3.3
2-gr	135.3	135.3	135.2
3-gr	165.6	668.8	165.6
4-gr	**222**	1,451.9	**222.0**
5-gr	179.4	**2,026.2**	179.4
6-gr			115,4
7-gr			72,1
In total	705.7	4,285.5	893.3

(a) EN Corpus n-gram counts

German		Fullform	
Order	5 pruned	5 unpruned	7 pruned
1-gr	8.1	8.1	8.1
2-gr	237.0	237.0	**237.9**
3-gr	168.5	842.9	168.5
4-gr	**180.5**	1,493.3	180.5
5-gr	128.1	**1,844.2**	128.1
6-gr			79.6
7-gr			52.6
In total	722.4	4,425.5	854.7

(b) DE Corpus n-gram counts

Italian		Fullform	
Order	5 pruned	5 unpruned	7 pruned
1-gr	3.1	3.1	3.1
2-gr	131.2	131.2	131.2
3-gr	169.7	670.3	169.7
4-gr	**225.1**	1,454.8	**225.1**
5-gr	174.8	**1,991.3**	174.8
6-gr			114.2
7-gr			79.2
In total	704.1	4,250.9	897.6

(c) IT Corpus n-gram counts

Table 17: N-gram counts for pruned and unpruned 5 and 7-gram models from the Fullform en/de/itTenTen corpora. Figures reported in millions of n-grams.

7 SELF-ORGANIZING MAPS AND CLASSIFICATION

This chapter is the first of three chapters presenting experiments involving Self-Organizing Maps. In these experiments, SOMs are created from a structured document collection, as a precursor to the document clustering experiments in Chapter 8.

Section 7.1 presents related work and Section 7.3 the data collections and evaluation. Section 7.4 introduces the experimental methods. Next, Section 7.4 presents the experiments and Section 7.5 their visualization. Finally, Section 7.6 ends the chapter with a discussion.

Categorization of a text corpus where a set of categories (labels) is attributed to each article is a classical **supervised** classification task. For such classification tasks, parameters are learned from a training set of labeled instances, and the learned model is subsequently used to classify test instances. The Self-Organizing Map, in contrast, is an **unsupervised** method that clusters training instances without any prior knowledge of their categories. Using a two-step methodology, the labels in a training corpus first associate with areas of the map; areas that in turn can be used to classify previously unseen documents.

A key problem in document classification is *representation* — how documents are transformed into vectors. In principle, such vectors can be of any size, limited only by hardware constraints. Practically, the vector representation is constrained to some **arity**. When representing documents by their tf-idf or n-gram statistics (such as below), cut-off points as a minimum threshold for occurrences can be specified, indirectly limiting the vector dimensionality. Additionally, features can be engineered from other sources, either knowledge-based such as specialized dictionaries or thesauri, or machine-learned semantic representations.

Several experiments were conducted on two different portions of the Reuters Corpus, termed the "Top 10" and "Full" sets of categories. While the experiments follow a tradition of Self-Organizing Map-based classification, they provide more details on implementation, vectorization, and the parameters used to create the map. SOMs of different sizes were used to evaluate the benefit of increasing the number of nodes (grid sizes) in SOM-based classification versus the increased computational cost of a larger grid.

7.1 RELATED WORK

Wermter and Hung [2002] used Self-Organizing Maps with WordNet-based [Miller, 1995] semantic networks for classifying the Reuters Corpus (details follow in Section 7.2). Documents were represented by **significance vectors**, representing the degrees to which the documents belong to certain preassigned topics, which were calculated from the importance of words in each category. Tsimboukakis and Tambouratzis [2007] used SOMs in an ensemble method classifying Greek language documents. First, terms were grouped by interpreting each node in the SOM as one group. Next, the number of groups was further reduced before group membership was used as features in supervised classification.

Saarikoski [2009] experimented with using Self-Organizing Maps for Information Retrieval and document classification [Saarikoski et al., 2011], and compared the algorithm to other Machine Learning methods on classification of the Reuters Corpus. While Saarikoski et al. explained how the SOM was applied to the problem, some factors that affect classification performance were unaccounted for, such as the parameters for the SOM creation, notably grid size and learning rate. Restricting the classification task to the Top 10 categories (by frequency), their best experiments had micro- and macroaverages of 93.2% and 83.5%, respectively. Notably, a Naïve Bayes classifier outscored all other methods on the data (95.2% and 90.4%), results that also significantly beat the findings of Dumais et al. [1998], who reported only 81.5% accuracy for the Naïve Bayes method. This discrepancy is likely due to a difference in scoring, as Dumais et al. [1998] reported break-even scores (motivated by comparability with other research), as opposed to Saarikoski et al. [2011] who provided true accuracy scores without any adjustment for precision-recall trade-off. The break-even point is where the precision is equal to recall, the point at which false positive and negative misclassifications are done at the same rate.

Saarikoski et al. [2011] noted how costly training of Self-Organizing Maps was, in contrast to the cheap testing of new instances *during* classification, and suggested using multiple maps for multi-label classification, or using the three nearest labels for a new instance. This approach, however, would imply that all documents have the same amount of labels in multi-label classification.

7.2 DATA

The Reuters Corpus [Lewis et al., 2004] has been extensively used for text classification research. Because it makes the comparison of results easier, some conventions on the corpus are established, such as the ModApté split [Lewis, 1997] into training and test sets with designated sections.

Other much-used splits are: to divide the ModApté into either the ten most frequent categories, the categories with at least one positive training and one test example (90), or the categories with at least one training example (115). The split with the 90 categories with at least one positive training and test example is somewhat confusingly also called ApteMod [Yang and Liu, 1999], and the *R(90)* subset of the ModApté split [Debole and Sebastiani, 2004]. Elsewhere, the R(90) split with 10,789 documents is also called ModApté [Yang et al., 2009, Chen et al., 2004, Saarikoski et al., 2011].

The Reuters Corpus consists of text documents with varying numbers of labels per instance. Sebastiani [2002] noted a fundamental distinction between the **single-label** and **multi-label** classification tasks. In the former, only one label is attributed to each document, whereas sets of labels are attributed to the documents in the latter. Hence, the Reuters Corpus is a multi-label dataset.

Zhang and Zhou [2013] reviewed multi-label classification, and highlighted two transformations of the problem; (i) it can be recast as a cascade of single-label (binary) classification problems and evaluated together, thus disregarding the dependence between labels, and (ii) it can be transformed into a **multiclass** (one-of) problem, which treats each labelset as a separate class and ensures that all documents are attributed one, and only one, of the labelsets.

The Natural Language Toolkit (NLTK) interface [Loper and Bird, 2002] to the Reuters Corpus, provides the ApteMod split as comprised of 7,769 training and 3,019 test examples. The frequencies of category counts are 9160, 1173, 255, 91, 52, 27, 9, 7, 5, 3, 2, 1, 0, 2, 1, i. e., 9160 documents with just one category, 1173 with two categories, etc. The category counts follow Zipf's law [Zipf, 1935] with an exponent of 3.44[1]. Figure 25 is a plot of the log-frequency of category counts. The *Top 10* categories refer to the ten categories with the highest frequency of documents.

7.3 METHOD

The method used to classify the Reuters Corpus is two-phased; (i) a Self-Organizing Map is created from the training portion of the corpus and (ii) test instances are labeled according to their placement on this map.

1 The exponent was found by optimizing the squares of differences between the log of the frequency probabilities and the log-theoretical probabilities, which was 1.35 at its optimum.

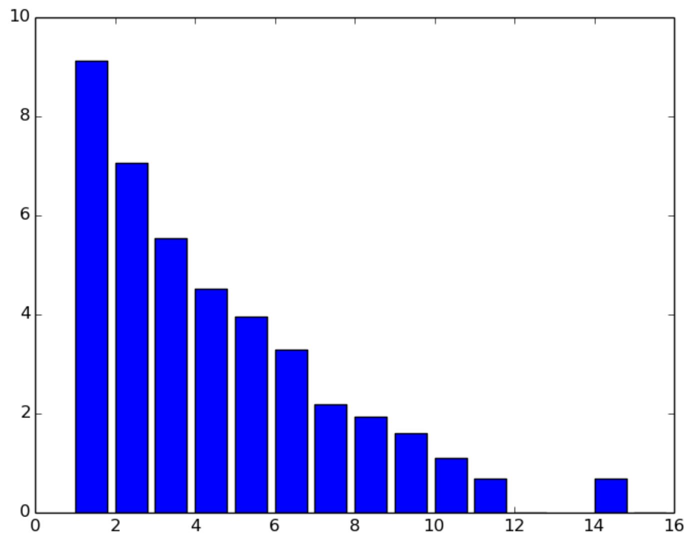


Figure 25: Number of categories per document (x-axis) plotted against log-frequencies of category counts (y-axis) in the Reuters Corpus. Log of 0 set to 0.

Before creating the Self-Organizing Map, the document collection was vectorized for both the training and test parts. Documents were converted into vectors consisting of tf-idf or raw word counts, or a combination thereof (details follow in Section 7.3.2). A parallel version of the SOM algorithm (see Section 2.1) was implemented, and a quadratic node topology was used for the Self-Organizing Maps.

In the subsequent classification phase, the methodology presented in Section 2.1.5 was applied. Majority voting was used for labeling test samples, and if no training samples were found in the Best-Matching Unit of the given test sample, an n-best list was searched. This follows Saarikoski et al. [2011] and largely also Isa et al. [2009].

7.3.1 *Implementation*

The Self-Organizing Map algorithm was implemented in MPI (using `mpi4py` bindings)[2] and `Python` [Dalcin et al., 2011], and run on a Portable Batch System (PBS)[3] scheduler. Samples were processed serially, but for each sample, the vector comparisons and updating of nodes were done in parallel. Hence, the algorithm was still **on-line** as

2 `https://mpi4py.readthedocs.io/en/stable/` (Last visited: March 27, 2019.)

3 Both `OpenPBS` and PBS Professional were used.

Algorithm 4 Parallel, vectorized on-line SOM algorithm for E epochs, a set of N nodes, and a set of S training samples and P processes.

```
1: procedure CREATE-SOM-PARALLEL
2:     randomly initialize $|N|$ nodes into a matrix $M^{|N|}$
3:     control node: scatter $M^{|N|}$ nodes to P processes
4:     for each epoch $e \in E$ do
5:         for each sample $s \in S$ do
6:             for each node $n \in N$ do
7:                 distance[n] = distance(s,n)
8:             end for
9:             BMU = min distance[$n \in N$]
10:            reduce BMU back to control node
11:            update BMU + neighborhood towards s
12:        end for
13:        reduce neighborhood and/or learning rate
14:    end for
15: end procedure
```

in Algorithm 1 (see Section 2.1). With the on-line formulation, vector comparisons and updates are run per training sample, increasing linearly with the number of training samples. Consequently, the number of vector comparisons increases with the number of nodes in the grid. Vector comparisons are costly, and therefore lend themselves well to parallelization. Hence, the parallel on-line implementation used in these experiments is sensitive to large numbers of training samples but well equipped to handle large Self-Organizing Map topologies. Algorithm 4 outlines the process in pseudo-code.

Equation 60 shows time complexity of a parallel SOM implemented on a hypercube grid [Hämäläinen, 2002].

$$O(m \log P + m(K/P)) \tag{60}$$

m represents vector length, K the grid size and P the number of processes. The two parts of the equation respectively represent the cost of communication and computation, where the latter term grows much faster than the former, which is reflected in the results in Table 19 on Page 149.

7.3.1.1 *Parallelization of the On-line Self-Organizing Map Algorithm*

Figure 26 explains Line 3 of Algorithm 4. Before nodes are distributed, the matrix of nodes is resized to a list format, thereby distributing the nodes fairly among processes with the strict requirement that the total number of nodes must be divisible by the number of processes. The current sample is also distributed to all processes, enabling them

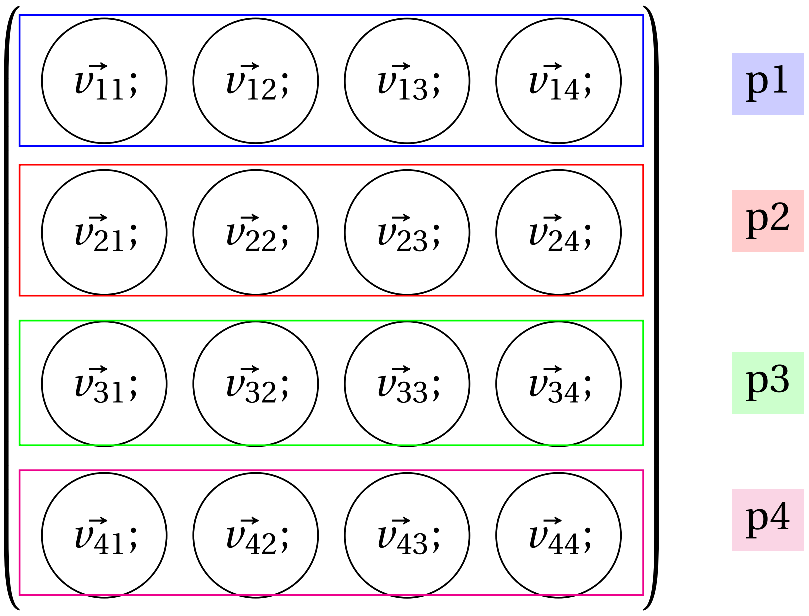

$\vec{v}_{11}$; $\vec{v}_{12}$; $\vec{v}_{13}$; $\vec{v}_{14}$;
$\vec{v}_{21}$; $\vec{v}_{22}$; $\vec{v}_{23}$; $\vec{v}_{24}$;
$\vec{v}_{31}$; $\vec{v}_{32}$; $\vec{v}_{33}$; $\vec{v}_{34}$;
$\vec{v}_{41}$; $\vec{v}_{42}$; $\vec{v}_{43}$; $\vec{v}_{44}$;
p1
p2
p3
p4

Figure 26: Distribution of 16 nodes to 4 processes.

to calculate distances for their nodes and return the minimum. In the update step, each process can calculate its nodes' locations in the original 2D grid by using its process ID, with which the distance from the Best-Matching Unit in Euclidean space can be determined. Thus, updating nodes can be done with matrix operations inside each process. The storage required for this matrix grows with vector dimensionality, and storing the total matrix — a matrix of all nodes in the grid inside one node — would exhaust the memory available for single processes even for small SOMs. At the end of a given Epoch, each process stores its local matrix in a file. Special high-memory nodes are then used to read all these matrices from file and recombine them into their original shape. Operations such as drawing graphics and calculating distances for hierarchical clustering are not parallelized and operate directly on the total matrix.

7.3.1.2 *Experiment Configuration*

Each experiment that will be presented in Section 7.4 was defined in a configuration file, where parameters such as the size of the grid, the number of iterations, the learning rate, and the size of the initial neighborhood radius were specified. Listing 1 contains an example configuration.

Listing 1: Example SOM configuration file. Selected parameters displayed.

```
[SOM]
dimension = 64 % quadratic layout side size
num_epochs = 100 % number of epochs
learning_rate = 0.20 % initial learning rate
start_radius_divisor = 4 % initial neighborhood radius size
input_vectorization = filename % option for loading
    vectorization from file

[Dataset] % options for loading datasets from files;
    collapsed

[Runtime] % runtime options controlling temporary storage
    and plot frequency; collapsed

[Logging] % logging options, notably log-file; collapsed
```

The neighborhood surrounding each Best-Matching Unit was defined by a radius diminishing with Epochs, with a configurable initial size. The radius was reduced by exponential decay, as was the learning rate, i. e., the degree to which BMUs were updated to be similar to training samples.

$$\sigma(t) = \text{start radius} * \exp(-\frac{t}{\lambda}) \tag{61}$$

Equation 61 shows the calculation of the decaying neighborhood with respect to time steps (Epochs). The starting radius is configurable, and λ is a time constant dependent on this initial radius. The learning rate was reduced similarly.

7.3.2 *Document Vectorization*

The NLTK interface provides a data structure where the raw text and categories of the documents in the Reuters Corpus are indexed with filenames. Documents were vectorized with `scikit-learn` [Pedregosa et al., 2011], retaining a vectorizer object such that the test corpus could be vectorized equivalently to the training corpus. In experiments below, the tf-idf transformer was used for vectorization alongside the Euclidean distance. The implementation is modular in the choice of vectorization method. In the experiments presented below, documents were transformed into a matrix of tf-idf values.

7.3.3 *Evaluation*

The Reuters Corpus is comprised of text documents with 0 to 15 category labels. The five most frequent labels of 90 categories in the

ModApté split have the following counts: 3923, 2292, 374, 326, 309, totaling 7224 of 9160 documents. The skewed distribution means that accuracy on the entire corpus is not necessarily a good measure of classifier performance, as the performance on less frequent categories may drown in the larger categories. The distinction between micro- and macroaveraging mitigates this problem (see Section 2.1.6). Both metrics are reported for the experiments below.

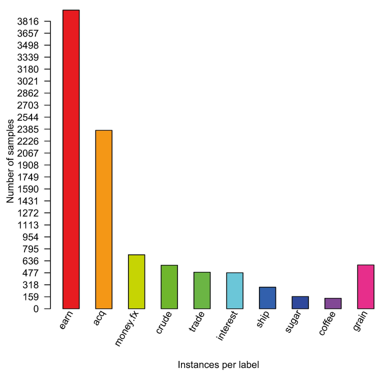
7.4 EXPERIMENTS

Two rounds of experiments were carried out: The first experiments were conducted with the same vectorization parameters on five grid sizes; 8x8, 16x16, 32x32, 64x64, and 128x128 (i. e., from 64 to 16,384 nodes) on the Top 10 categories of the Reuters Corpus. The computation times for different grid configurations were compared as the numbers of parallel processes were increased. The second round of experiments was done on the entire AptéMod split (90 categories), investigating the effect on classification performance of varying grid sizes by the same amount. In both rounds of experiments, the Self-Organizing Maps were configured with an initial learning rate of 0.10 and an initial neighborhood radius of a quarter of the grid dimension.

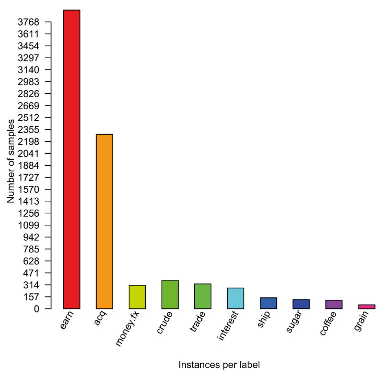
The training and test corpora were limited to documents with only one category when classifying the Top 10 categories to concentrate experiments on the *one-of* classification problem. When using majority voting for prediction, the method would not benefit from the added information that some documents have more classes, as less frequent classes in each node could be voted down. Using documents restricted to one label could, therefore, bring about better separation in the Self-Organizing Map.

Figures 27a and 27b plot the differences between the multi-label and multi-class versions of the ModApté split, respectively. They show how the labels that are concurrent with other labels shrink (such as *money-fx* and *interest*) whereas others, such as *acq* and *earn*, are largely retained in counts. Figure 27c partly explains the differences, by plotting the concurrence of the labels in the multi-label version of the dataset. Each arc represents a label and is proportional to the number of documents with more labels, e. g., documents labeled as *money-fx* share 1,2,35,184, and 2 labels with the labels *acq*, *crude*, *trade*, *interest*, and *grain*, respectively. The links between the arcs show the strength in the relationships. Furthermore, the plot shows how the shared instances between the most frequent labels, *acq* and *earn* are few, both between themselves and the other labels.

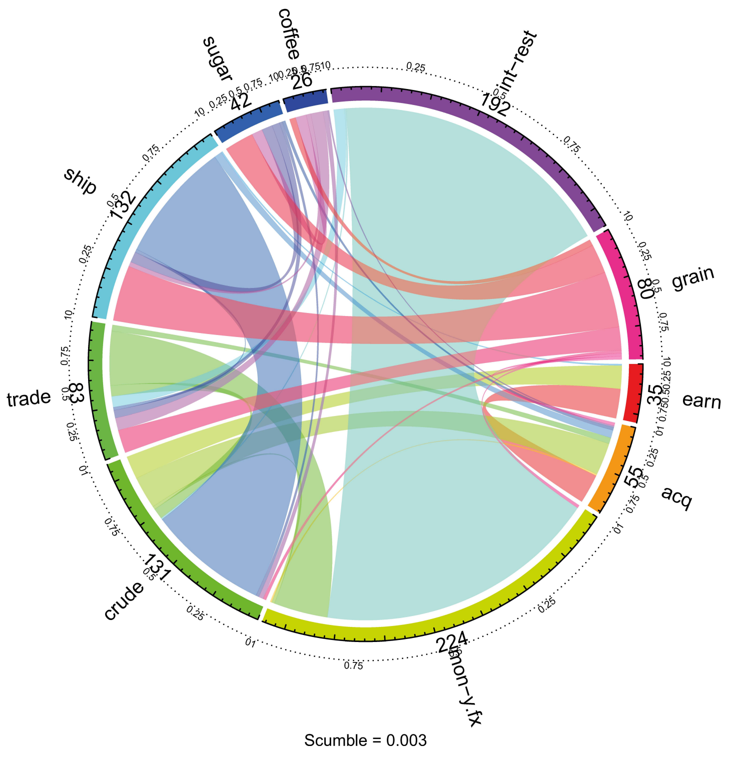
Charte et al. [2014] introduced the Score of ConcUrrence among iMBalanced LabEls (SCUMBLE) metric (provided by the R package



(a) Multiple labels



(b) One label



(c) Concurrences between labels

Figure 27: Histograms of labels in subsets (a and b) and an interaction plot of subset a (c).

Category name	Training	Test
earn	2840	1083
acq	1596	696
money-fx	222	87
crude	253	121
trade	250	76
interest	191	81
ship	108	36
sugar	97	25
coffee	90	22
grain	41	10

Table 18: Number of training/test samples in Top 10 categories of the AptéMod split of the Reuters Corpus when restricted to documents with only one category.

`mldr` [Charte and Charte, 2015]), which is a metric between 0 and 1 that denotes the concurrence levels between frequent and infrequent labels, designed at gauging the effectiveness of resampling algorithms on a multi-label dataset. Under the assumption that documents that are labeled with both frequent and infrequent (i. e., imbalanced) labels would cause problems for resampling algorithms, the metric is intended to predict whether such techniques would be helpful for a given dataset. A lower score (few concurrent, imbalanced labels) would indicate that resampling techniques work well. If in the extreme case, all minority labels are also labeled with the majority label, oversampling of these instances would not mitigate the dataset imbalance. The SCUMBLE score was 0.003 for the Top 10 categories on the ModApté dataset, and 0.03 for the full dataset. For the multi-class subset, the SCUMBLE is 0. Compared to the results presented by Charte et al. [2014], these values are low enough for resampling techniques to work well.

7.4.1 *Top 10 Categories*

In these experiments, documents were limited to those belonging to the Top 10 categories and vectorized into 33,120 dimensions. Each vector consisted of the tf-idf values for the n-grams ranging from 1 to 7, with a cut-off frequency of 5 (i. e., ignoring dictionary terms with a frequency below this threshold). The numbers of documents labeled with these categories are listed in Table 18.

Grid size	8x8	16x16	32x32	64x64	128x128
Microavg.	0.8	0.89	**0.89**	0.85	0.80
Macroavg.	0.52	0.79	**0.82**	0.78	0.75
HPC procs.	8	64	256	1024	2048
Time (hours)	0.5	1	2	4.5	11.5

Table 19: Classification scores for Reuters Corpus (Top 10) for different grid sizes.

Grid size	8x8	16x16	32x32	64x64	128x128
Microavg.	0.65	0.75	**0.78**	0.77	0.76
Macroavg.	0.07	0.13	0.25	0.25	**0.30**

Table 20: Classification scores for Reuters Corpus (all categories) for different grid sizes.

Table 19 summarizes the first experiments. The experiments did not exhaustively explore the optimal number of parallel processes for all experiments, i. e., finding the sweet spot beyond which new processes do not mean a speed-up due to overhead costs. However, one experiment was done on creating the same Self-Organizing Map (built with identical parameters) with a different amount of nodes processes in use on the HPC grid. This experiment compared creating a SOM with a 16x16 node grid, computed with 64 processes over four nodes (the same as in Table 19), to running it all on just one node, with 16 parallel processes. When using 64 parallel processes, the experiments took 1 hour to complete vs. 4 hours when running on only one node (16 processes). Otherwise, the speed-up potential was not explored.

7.4.2 *All Categories*

Table 20 summarizes the second group of experiments, conducted on all (90) categories. In these experiments, documents were vectorized into 19,092 dimensions. Documents were represented by tf-idf values for n-grams ranging from 1 to 7, using a cut-off frequency of 10. While documents both in training and test sets could have multiple labels, each label was separately treated during prediction and scoring, as a series of single-label classification problems.

7.5 VISUAL INTERPRETATION OF RESULTS

Since Self-Organizing Maps are *maps*, their visual interpretation is a source of information. This section will display and discuss different visualizations of SOM development and the resulting maps. For visualizing high-dimensional data, reduction of these dimensions is key [Grinstein et al., 2001]. Each node is represented by a vector with the same dimensionality as the vectorized documents; these node vectors were subsequently reduced to four dimensions using Principal Component Analysis (PCA) [Jolliffe, 1986].

The Principal Component Analysis algorithm implementation in `sklearn` uses `LAPACK` package, which has cubic time complexity for computing Singular Value Decomposition of a square matrix[4]. PCA is equivalent to SVD when the data points have been mean-centered and normalized by the standard deviation (see, e. g., Raiko et al. [2008]). The full SVD was calculated serially for the visualizations below. However, for matrices that are too large for cubic complexity, other algorithms to approximate the SVD exist. Tang et al. [2016] proposed a method based on constructing a **K Nearest Neighbor** graph from the data, projecting it to a lower-dimensional space. Their method is linear in the number of samples.

The four dimensions obtained through PCA were then used to encode color in the `Matplotlib` library [Hunter, 2007], displaying each node as a color grading. The library accepts vectors of both three and four dimensions to create colors, and vectors were reduced to four dimensions to retain more of the variance in the original node vectors. The similarity in colors visualizes the similarity between vectors in the same area.

For Self-Organizing Maps created from labeled data, labels from both the training and test corpora could be plotted on top of the color-coded node vectors. This color-shaded map could be used to verify that similar labels were attributed to nodes with similarly colored vectors.

7.5.1 *Visualization of Self-Organizing Map Development*

Figure 28 shows the development of a Self-Organizing Map with a grid size of 32x32 from the experiments on the Top 10 categories. During the first iteration steps, the radius of the initial neighborhood is visible before the grid self-organizes into a map. As the development of the SOM draws closer to the specified number of Epochs, the ra-

4 `http://www.netlib.org/lapack/lug/node71.html` (Last visited: March 27, 2019.)

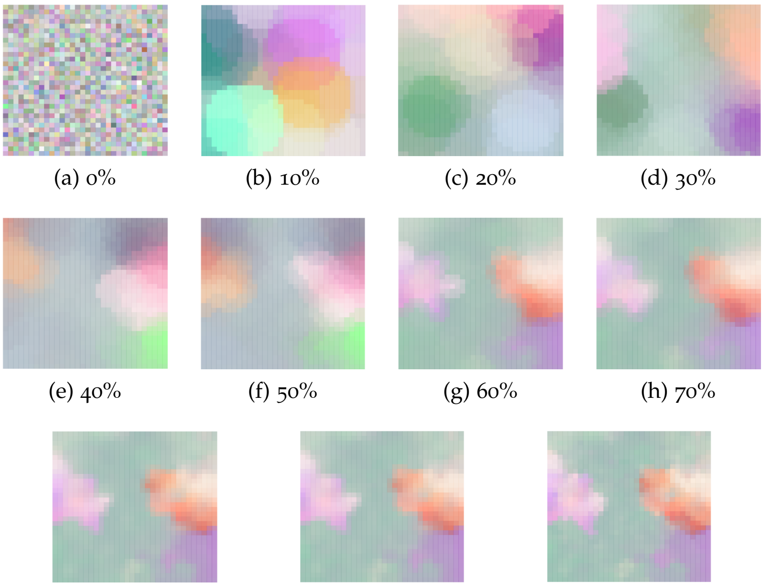


Figure 28: Development of SOM. Corpus: Reuters Corpus (Top 10 categories), grid size: 32x32, features: 33,120, iterations: 100.

dius around each Best-Matching Unit is reduced, as are the changes to the map.

7.5.2 *Visualization of Self-Organizing Maps and Document Labels*

Figure 29 shows a trained Self-Organizing Map annotated with the categories of the training (yellow) and the test (magenta) samples. An 8x8 map is used for illustration here since it is easier to show the contents of grid cells in this format. As the number of nodes grows large, it is difficult to fit the areas in one frame. The training labels (yellow) are sorted in decreasing order on the left side of the cells, and test labels (magenta) in decreasing order on the right side.

Figure 30 on Page 155 shows three magnified excerpts of the 8x8 SOM in Figure 29. Starting with Figure 30a, the cell in the top-left corner of the map has three training corpus labels attaching to it; *acq* (41 documents), *earn* (3), and *interest* (1), and 14 documents from the test corpus, all with the category *acq*. Majority voting among the training samples assigns the category *acq* to the node. In this case, all test samples are labeled correctly.

In Figure 30b, the four nodes have attracted documents with multiple labels. The labels with the highest frequency both in the training

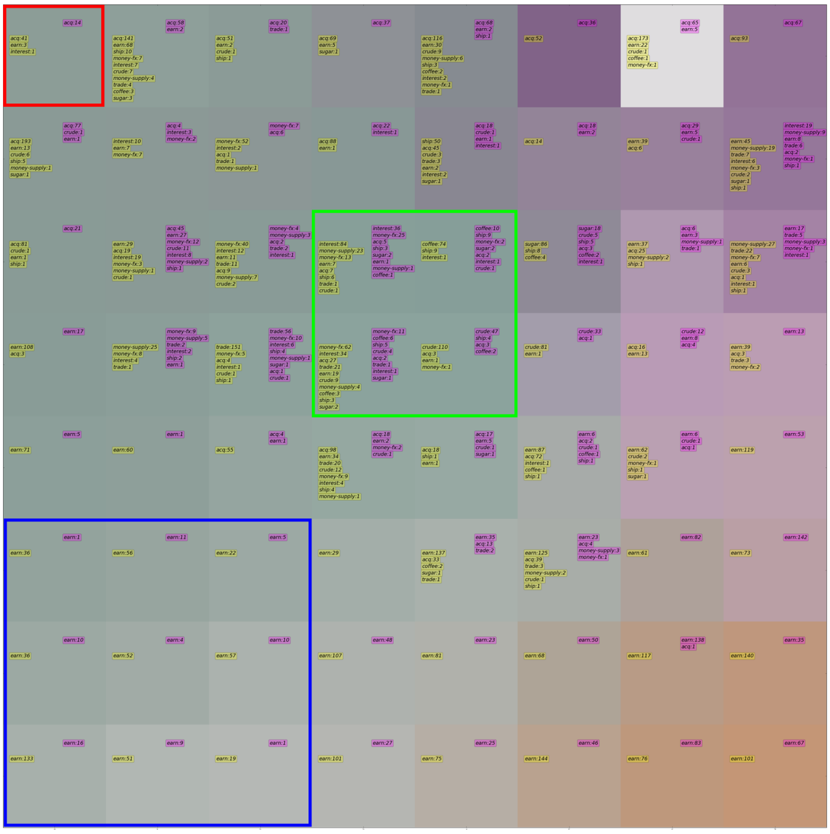


Figure 29: SOM annotated with categories. Highlighted excerpts are magnified in Figure 30. Training samples in yellow, test samples in magenta. Corpus: Reuters Corpus (Top 10 categories), grid size: 8x8, features: 33,120.

and test corpora match for all four cells (*interest*, *coffee*, *money-fx*, and *crude*). Thus, these labels from the training corpus represent the predictions for each node and the set of correctly predicted test samples, respectively. In addition to the most frequent labels, the lists share many other elements, indicating that the clustered documents have more properties in common. In Figure 30c, all documents — both from the training and test corpora — have the label *earn*.

7.5.3 *Visualization of Resulting Self-Organizing Maps*

This section refers to Figures 31-33 on Pages 156-158. Figures 31 and 32 show the final maps for the Top 10 category and all category runs. In the larger maps, fewer categories attached to each node, both from training and test corpora.

Figure 31d shows a Self-Organizing Map with grid size 64x64. On the left side of the plot, a light green area is visible. This green area is populated by documents labeled *crude*, going into an area labeled *earn* as the area turns gray in the vertical direction. In the middle of the figure, areas populated by documents labeled with *interest* neighbor areas with documents labeled with *money-supply*.

As Tables 19 and 20 showed, macroaveraged F-scores increase as the grid sizes increase for the experiments on all categories, but for the microaverages and the Top10 categories, the best score was found at size 32x32.

Figure 33 complements the information of the color shadings for the 64x64 SOM shown in Figure 31d. The **heatmap** shows the number of documents attached to each node (having the node as their Best-Matching Unit). This figure is not visible from the color gradings of map visualization and shows the relative size of these areas. Figure 33b shows a **win-plot**, a 3D bar chart of the number of documents in each node, i. e., a different plot of the same data as in the heatmap. A 2D snapshot of a 3D plot can be hard to read for some data, especially when the grid is large. It is easier to visualize with a 3D model on a computer screen, which can be turned around its axes.

The Unified Distance Matrix (UDM) is computed by taking the average distance between each node and its immediate neighbors in Euclidean space. This plot is also harder to read in a 2D screenshot than in a 3D model, but it does show how the regions are separated by high fences between clusters when the distance between nodes and their neighbors is high. Large distances between nodes become peaks in Figure 33c, and low areas represent clusters of similar vectors. Comparing Figures 31d and 33c, the flatter area on the UDM reflects the large area in the bottom right corner in the same gray shading. Together they form a comprehensive view of the clustering process.

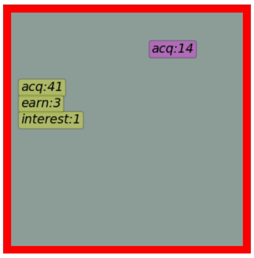
7.6 DISCUSSION

Confirming that the performance on the classification task on the labeled Reuters Corpus was close to the state-of-the-art performance on the dataset shows that the proposed method clusters the document collection into meaningfully separated sub-parts. While the settings of the Self-Organizing Maps were not fully optimized and tuned, the classification scores were near to, and in some cases better than comparable experiments.

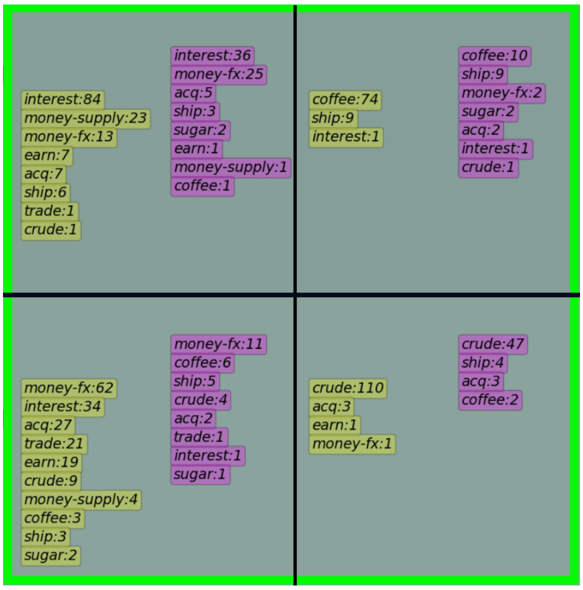
The experiments on classification of the Reuters Corpus suggest considerable overlap between labels. As shown in the plots, some nodes attracted many differently labeled documents both from the training and test corpora; suggesting that these documents did, in fact, belong to many different domains simultaneously.

Using the unsupervised Self-Organizing Map method, relations between categories become apparent. Both regarding how clusters of documents that belong to similar categories are placed next to each other on the map, and also how some nodes in the SOM attract separate documents with thematically adjacent labels as their Best-Matching Unit, such as *money-fx* and *interest*. This relation could reflect that these documents are in a thematical intersection between topics, or genuinely belong to both.

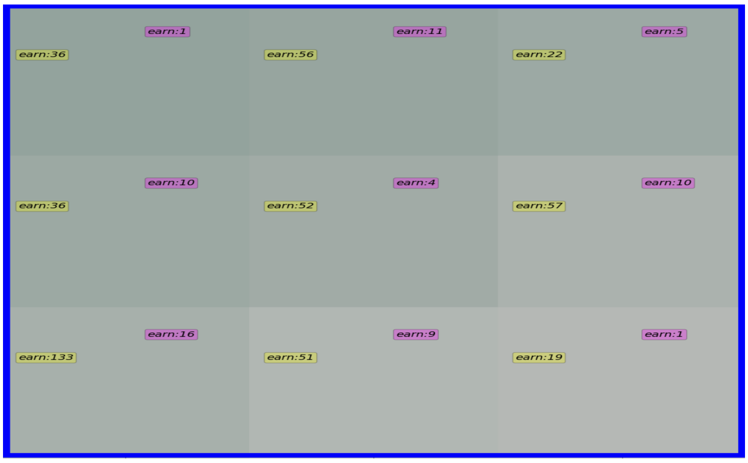
While an increase in the **one-of** classification performance when the Self-Organizing Map topologies went above 32x32 nodes was not observed for the Top10 experiments, it is likely that the multi-label problem benefits from a larger amount of nodes, offering finer granularity. With fewer categories per node, smaller categories have a better chance of being the majority category of a node, enabling the Self-Organizing Map to make predictions of that category. However, the parameter space for each SOM of every size used in the experiments was not explored. Thus, the variation in classification performance could be attributed to confounding variables.



(a) Top-Left (red)

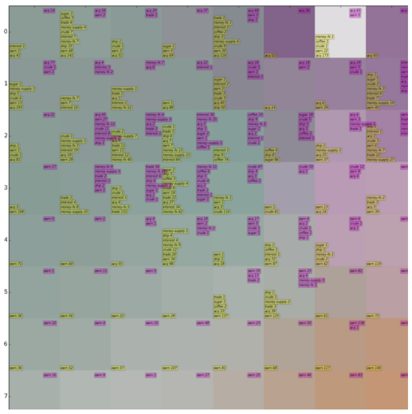


(b) Middle (green)

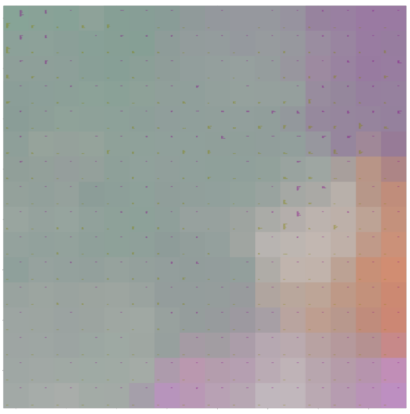


(c) Bottom-Left (blue)

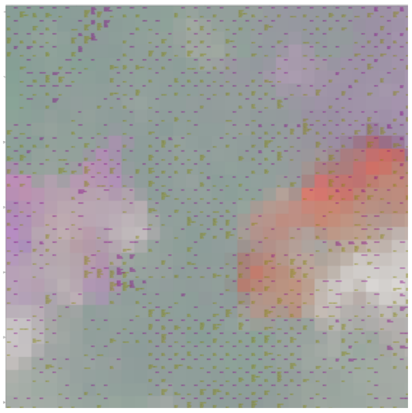
Figure 30: Excerpts from Figure 29.



(a) 8x8

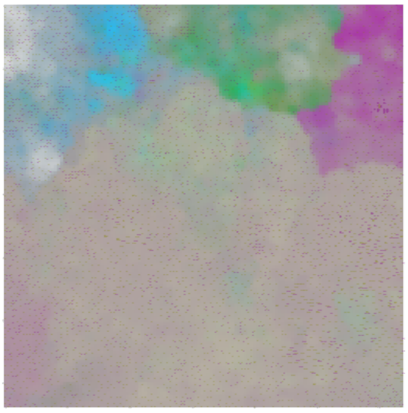


(b) 16x16



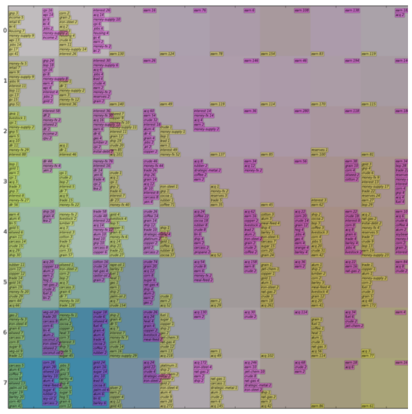
(c) 32x32

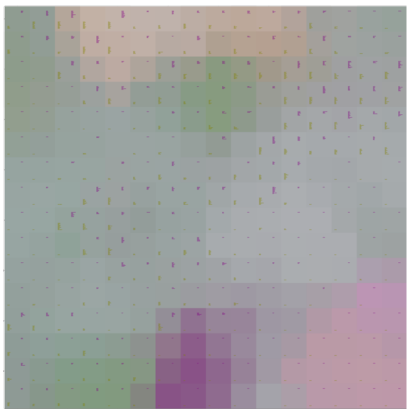
(d) 64x64

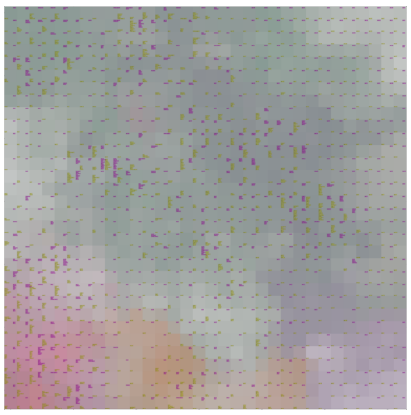


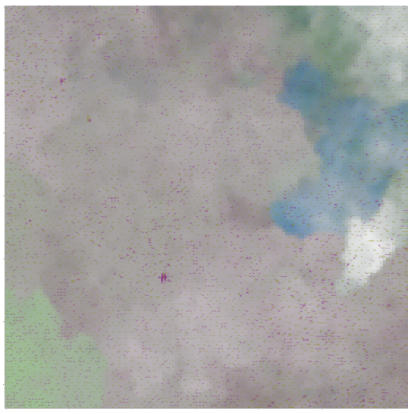
(e) 128x128

Figure 31: SOMs of various sizes. Corpus: Reuters Corpus (Top 10 categories), features: 33,120.


(a) 8x8


(b) 16x16


(c) 32x32


(d) 64x64

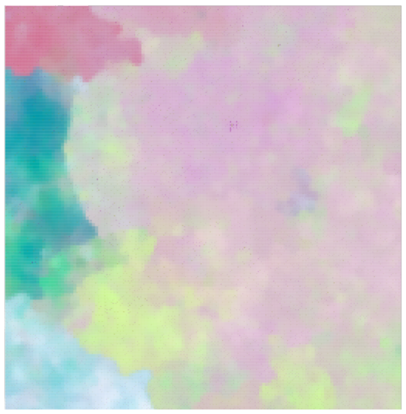
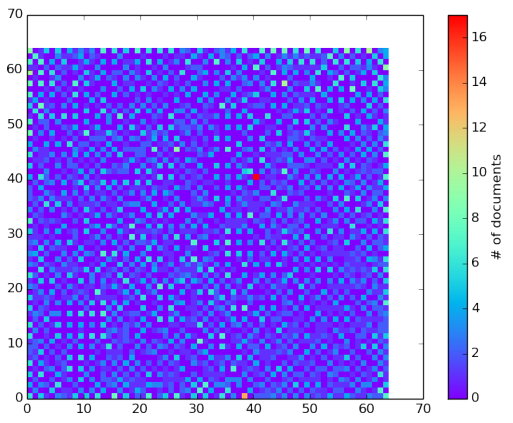
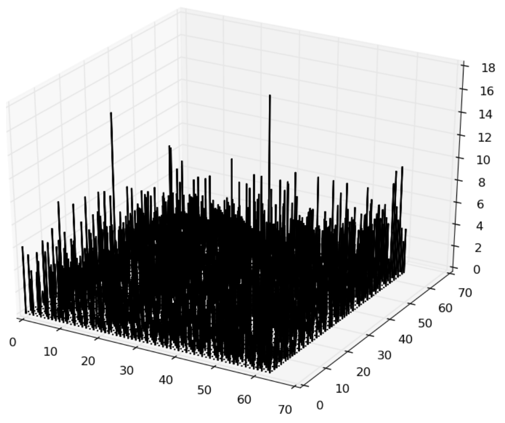

(e) 128x128

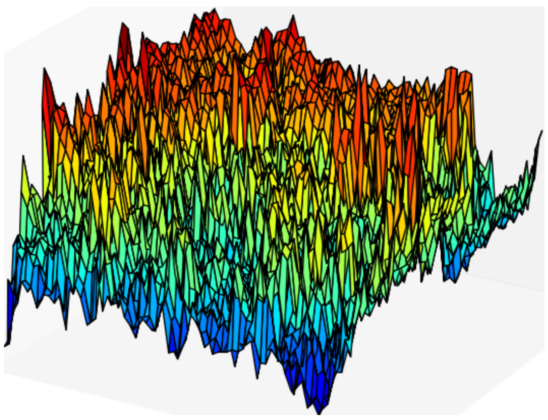
Figure 32: SOMs of various sizes. Corpus: Reuters Corpus (all categories), features 19,092.



(a) Heat map



(b) 3D bar chart



(c) Unified Distance Matrix

Figure 33: Distribution of documents per node (a and b) and the difference between node vectors (c) after final run. Corpus: Reuters Corpus (Top 10 categories), grid size 64x64, features: 33,120.

8 SELF-ORGANIZING MAPS AND HIERARCHICAL CLUSTERING

Chapter 7 presented a Message Passing Interface (MPI) implementation of the Self-Organizing Map algorithm. The next step towards using the SOM clusters in Domain Adaptation in Machine Translation is to apply a clustering algorithm to segment the resulting map into delimited clusters, which in this case are collections of documents.

This chapter will present experiments using hierarchical agglomerative clustering for clustering SOMs. Section 8.1 presents related work, Section 8.2 the data collections used for the experiments, and Section 8.3 explains the methodology. Section 8.4 goes through the experimental results before Section 8.5 ends the chapter with a discussion.

8.1 RELATED WORK

Section 2.5 showed how large Self-Organizing Maps could be used for clustering by clustering its node vectors, for which any clustering algorithm that accepts multivariate data is applicable. Vesanto and Alhoniemi [2000] found that clustering the SOM gave better results than clustering the raw data (document representations) directly, as well as having computational advantages.

Kolehmainen [2004] experimented with using Self-Organizing Maps as a starting point for Sammon's mapping [Sammon, 1969]. This method maps a high-dimensional vector space to a lower dimension (often two), by minimizing (e. g., by gradient descent) an error term including the difference between the two distances:

$$E = \frac{1}{\sum\limits_{i<j} d^*_{ij}} \sum_{i<j} \frac{(d^*_{ij} - d_{ij})^2}{d^*_{ij}} \tag{62}$$

where d and d^* are the Euclidean distances in the high-dimensional and lower-dimensional space, respectively. Kolehmainen argued that the SOM is efficient as a starting point for Sammon's mapping by reducing its time complexity, and cited other studies with similar findings.

Wu and Chow [2004] also found a two-stage approach beneficial, because it could leverage the established relations in the Self-Organizing Maps. The authors argued that a hierarchical clustering method is preferable to a partitioning method, as a partitioning can result in incorrect clusters, and the number of clusters must be pre-defined. SOM clustering as per Wu and Chow [2004] included using a validity index — Composing Density Between and With Clusters (CDbw) [Halkidi and Vazirgiannis, 2008] as merging criterion and a pre-processing step, which reduces noise by removing outliers. CDbw is based on a mixture of inter- and intra-cluster densities (i. e., a density-based approach in Table 2 on Page 31). Additionally, Halkidi and Vazirgiannis found that the CDbw is maximized when the partitioning includes the optimal number of clusters. Thus, the measure can also be used to determine the optimal number of clusters.

Aroui et al. [2008] used hierarchical clustering of Self-Organizing Maps with Ward's method to diagnose rotor faults in machines, where each identified region in the SOM represented one type of fault. Aroui et al. achieved an accuracy of 96.66% on a labeled test set. Alternatively, Grieco et al. [2017] used K-means clustering to refine the results of a Self-Organizing Map, using a version of the knee method (will be explained in Section 8.3) to determine the number of means, *K*.

8.2 DATA

Three text corpora were used in these experiments, one structured (with labels) and two unstructured document collections from the web. Chapter 7 introduced the Reuters Corpus, and the remaining two corpora will be described below. While the documents in the web corpora do not come with labels suited for classification experiments, the originating URLs are provided for all documents.

8.2.1 *The SdeWac Corpus*

The Stuttgart DeWac (SdeWac) [Faaß and Eckart, 2013] corpus of German text, consists of parsable sentences from the DeWaC corpus from the WaCKy initiative [Baroni et al., 2009], subject to duplicate removal. Thus, the SdeWac corpus is a derivative of the TenTen family of corpora[1]. The DeWaC corpus was crawled using words from the Süddeutsche Zeitung[2] and basic German vocabulary lists as seeds. After crawling, the corpus was tagged with the TreeTagger [Schmid, 1995]. The corpus consists of 10,830 documents of which 9056 were used in the experiments presented below, in total 8.1G of text.

1 `https://www.sketchengine.co.uk/` (Last visited: March 27, 2019.)
2 `http://www.sueddeutsche.de/` (Last visited: March 27, 2019.)

Listing 2: Two first documents from the SdeWac raw data.

```
 <year="0"/><source="1403"/> <error="0.009"/>  Sie dürfen eine
    Kopie der Software auf dem Dateiserver Ihres
Computers installieren , (.............) oder die Verwendung der
   Software gestattet ist , die zulässige Anzahl nicht
überschreitet .
<year="0"/> <source="3708"/> <error="0.009/>  Die Henker der
   Geschichte , die Abkmmlinge Kains , machten sich über
die Kinder Adams (........),  , Zivilisierte und Unzivilisierte ,
   Araber und Nicht-Araber .
```

Listing 2 presents the two first entries of the raw data. A database of their origins is distributed with the raw files. The entries for the two documents are:

1403=http://www.adobe.de
3708=http://www.enfal.de

This is a mapping of the documents to their origin URLs. The first document is from the software company Adobe and the second from a website about Islam. At this granularity, these URLs are not sufficient to serve as class labels (e. g., denoting domain or topic), as one such URL can comprise infinitely many types of text; consider blogging or search sites.

8.2.2 *The enTenTen Corpus*

A selection of documents was made from the enTenTen corpus [Baroni and Kilgarriff, 2006] due to its considerably larger size (3.3Bn tokens, crawled in 2008). In contrast to SdeWac, the enTenTen corpus consists of raw text from the TenTen crawls. This corpus was also tagged with the `TreeTagger` and provided a URL for each document as well.

Listing 3: The first sentence from the first document in the enTenTen corpus.

```
 <doc xdedupl_id="100" lang_filter_score="0.684605"
(....)
url="http://www.bedfordshire.police.uk/?_id=343&showArticle=137"
(...)>
<p class="notbad" cfclass="notbad" heading="0" tokens_count="14">
<s>
BURGLARS        NNS     burglar-n
love    VVP     love-v
to      TO      to-x
see     VV      see-v
presents        NNS     present-n
under   IN      under-i
the     DT      the-x
Christmas       NP      Christmas-n
tree    NN      tree-n
just    RB      just-a
as      RB      as-a
much    JJ      much-j
as      IN      as-i
children        NNS     child-n
<g/>
.       SENT    .-x
</s>
```

Listing 3 shows an excerpt from the corpus. Each word is presented on one line in three tab-separated columns. From left to right: full form, Part-of-Speech, and lempos (lemma and POS combination).

8.2.3 *Pre-processing*

It was necessary to extract the body text from each document in the collections into separate text files (see Section 6.2 for details) such that documents could be vectorized. Additionally, enTenTen documents were indexed from 0 to keep track of the URL origin of each document, and the corresponding URLs kept in separate text files. The SdeWac corpus provided such unique identifiers.

8.3 METHOD

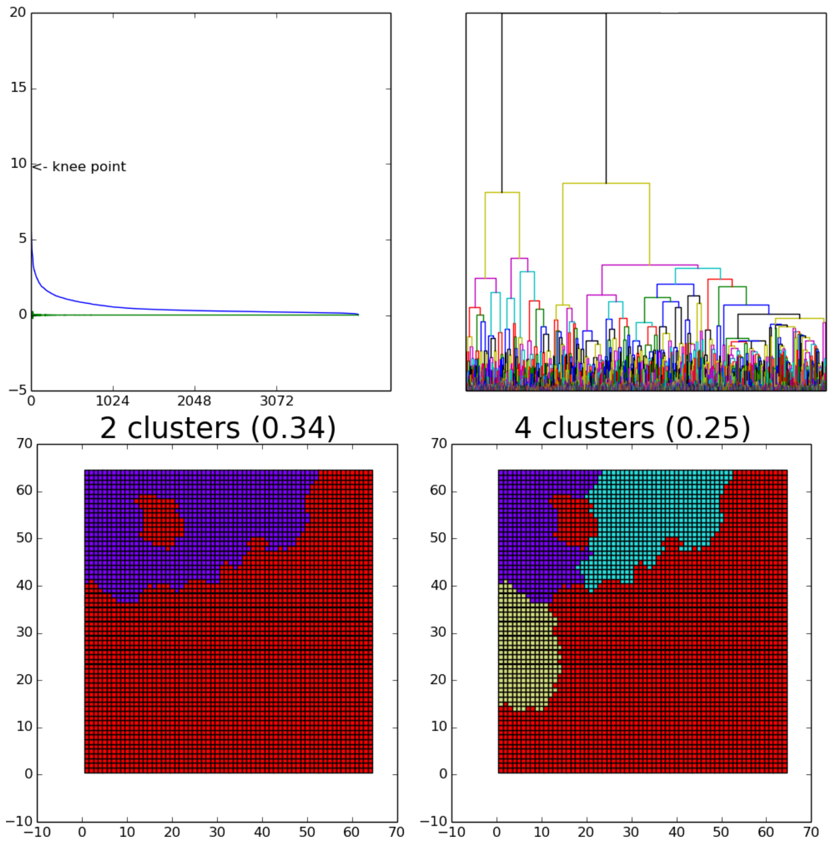
Clustering with Self-Organizing Maps was done with a two-stage approach, similar to methods used in related work. First, the SOMs were created with the same method as in Chapter 7. Next, agglomerative hierarchical clustering (see Section 2.4.1) was used to create explicit clusters, with Ward's clustering method and Euclidean distances. The Knee/Elbow (henceforth: **knee**) method was used to determine the number of clusters (details follow below). In addition to the knee method that finds the point with the maximum curvature, other information theoretic methods can be used, e. g., to quantify the fit of cluster members to the centroids and optimize the number of clusters on this [Salvador and Chan, 2004].

Loosely defined, the knee of a curve is the point on the curve with maximum curvature, where moving in either direction on the curve would mean the largest change in value on the y-axis (here; cluster distances). Thus, the marginal gain of merging in another cluster in terms of intra-cluster difference is low at this point.

A **linkage matrix** (see Section 2.4.2) contains the distances between clusters as they are merged. Finding the knee can be done, e. g., by computing the maxima of the forward, backward or central differences (which applied to the second degree can be used to estimate the second derivative). The Linkage Matrix is ordered in reverse, starting with the last merge into the all-containing cluster. The second degree forward differences of the linkages were computed from the linkages of the n nodes in the Self-Organizing Map. The first two knee points (maxima) in descending order were used to form partitionings.

Figure 34a shows a detailed analysis of the Ward clustering of a Self-Organizing Map of dimension 64x64 based on the Reuters Corpus. In the top-left plot, the bottom line is a plot (in green) of the intra-cluster differences against the number of merges (the number of clusters minus one), whose highest n values identify the knee points, where merging two clusters would add the most to the within-cluster distance. The forward difference of this curve of distances is plotted above it (in blue), whose points of maximum curvature are identified by the knee-points below. The numbers of knee-points were at 2 and 4 clusters, which is reflected in the partitionings shown in the bottom row of Figure 34a. The first two clusterings are plotted with their Silhouette Coefficients in parentheses, and alongside a dendrogram corresponding to the cluster relations in each clustering method in the upper right cell.

Plots of smaller node grids are better for visualizing the derivation of the knee point because of fewer merges, which makes the knee points in the curvatures more visible (see Appendices A and B). In the following experiments, two **partitionings** were used, corresponding to the first two knee points.



(a) Clustering details for hierarchical clustering (Ward method) of SOM.

(b) Plot of resulting SOM.

Figure 34: Juxtaposition of Ward clustering details and PCA-based plotting of SOM. Corpus: Reuters Corpus, dimension: 64x64, features: 33,120.

8.4 EXPERIMENTS

Two sets of experiments are presented below; (i) on the structured Reuters Corpus, and (ii) on the unstructured SdeWac/enTenTen corpora. The experiments are presented in that order in the remainder of this section.

With a labeled corpus as the Reuters Corpus, it is possible to display these documents labels found in the areas determined to be separate clusters by the hierarchical clustering algorithm. This algorithm works directly on the vectors representing each node in the 2D grid of the Self-Organizing Map. Hence, it does not consider the number of documents having each node as Best-Matching Unit after the final Epoch but only the distances between vectors in high-dimensional space.

When using unstructured web corpora such as SdeWac and enTenTen, it is not possible to reliably determine the thematic content of the clustered documents without labels other than URLs. Plots of the clusters in 2D space, as well as other quantitative information, will be displayed.

The Self-Organizing Maps were configured with an initial learning rate of 0.20 and an initial neighborhood radius of a quarter of the dimension of the SOM.

8.4.1 *Experiments on the Reuters Corpus*

Figure 35 shows a detailed analysis of the clustering of the Top 10 categories of the Reuters Corpus using each of the clustering methods listed in Figure 12 on Page 34. The single, average and median linkage methods produced one cluster that dominated the others totally, as exemplified by the average method displayed in the second row. Complete, weighted, and centroid linkage were only slightly more successful in achieving balanced clusters, as exemplified by the centroid method in the fifth row. Ward's clustering method produced the most evenly sized clusters with more clusters of substantial size, as shown in the sixth row. This finding was consistent for all clusterings experiments on SOM data, across map sizes and corpora.

Figure 34b (repeat of Figure 31d on Page 156) showed the resulting Self-Organizing Map on the Reuters Corpus, a magnified version of the Ward line (sixth from top) in Figure 35. Juxtaposed in the figure, the patterns are similar in the visualization of the SOM and the hierarchical clusterings (shown in the lower row of the clustering details in Figure 34a), which demonstrates how the SOMs are transformed into similar clusters.

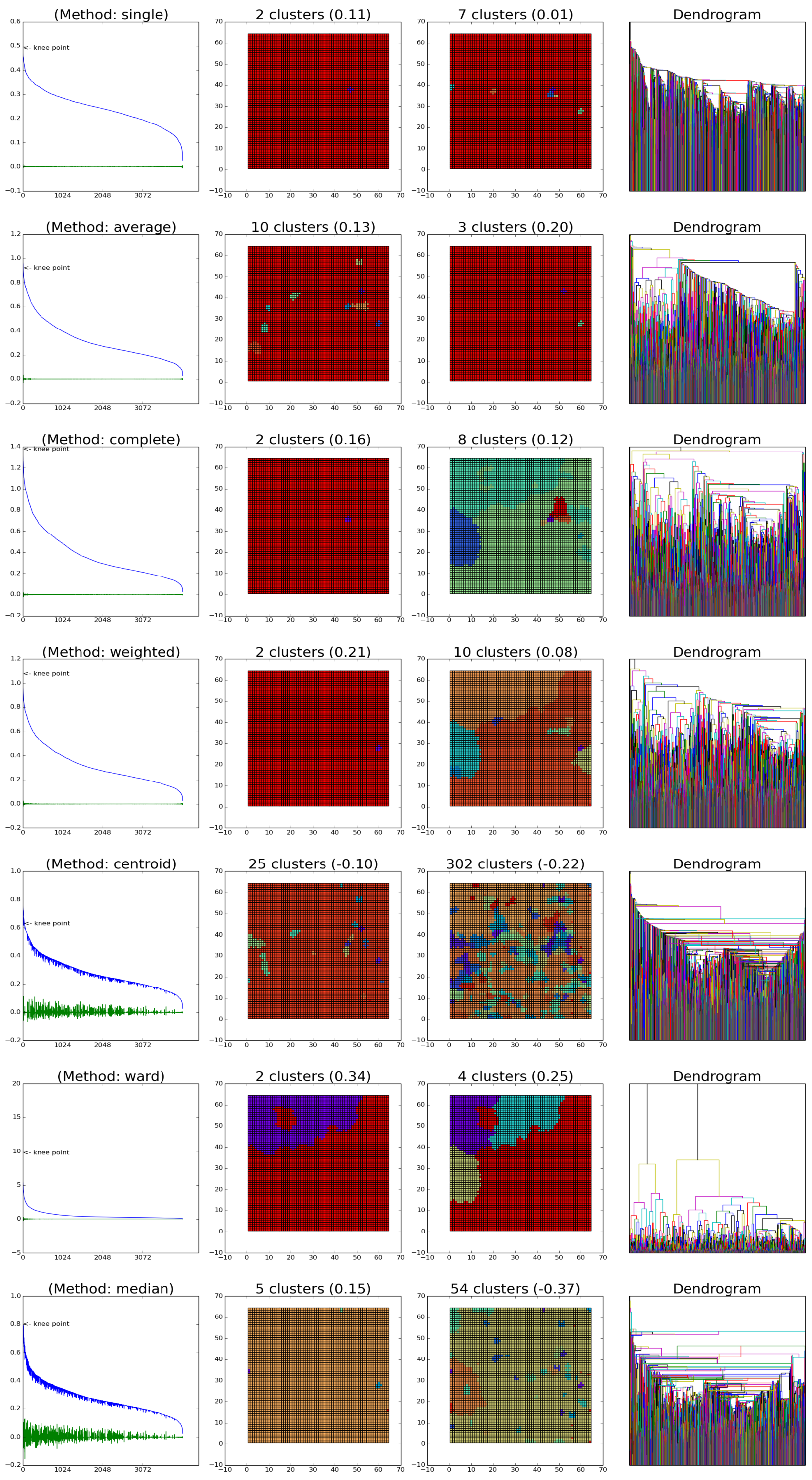


Figure 35: Detailed analysis of hierarchical clustering for SOM. Corpus: Reuters Corpus (Top 10 categories), grid size: 64x64, features: 33,120.

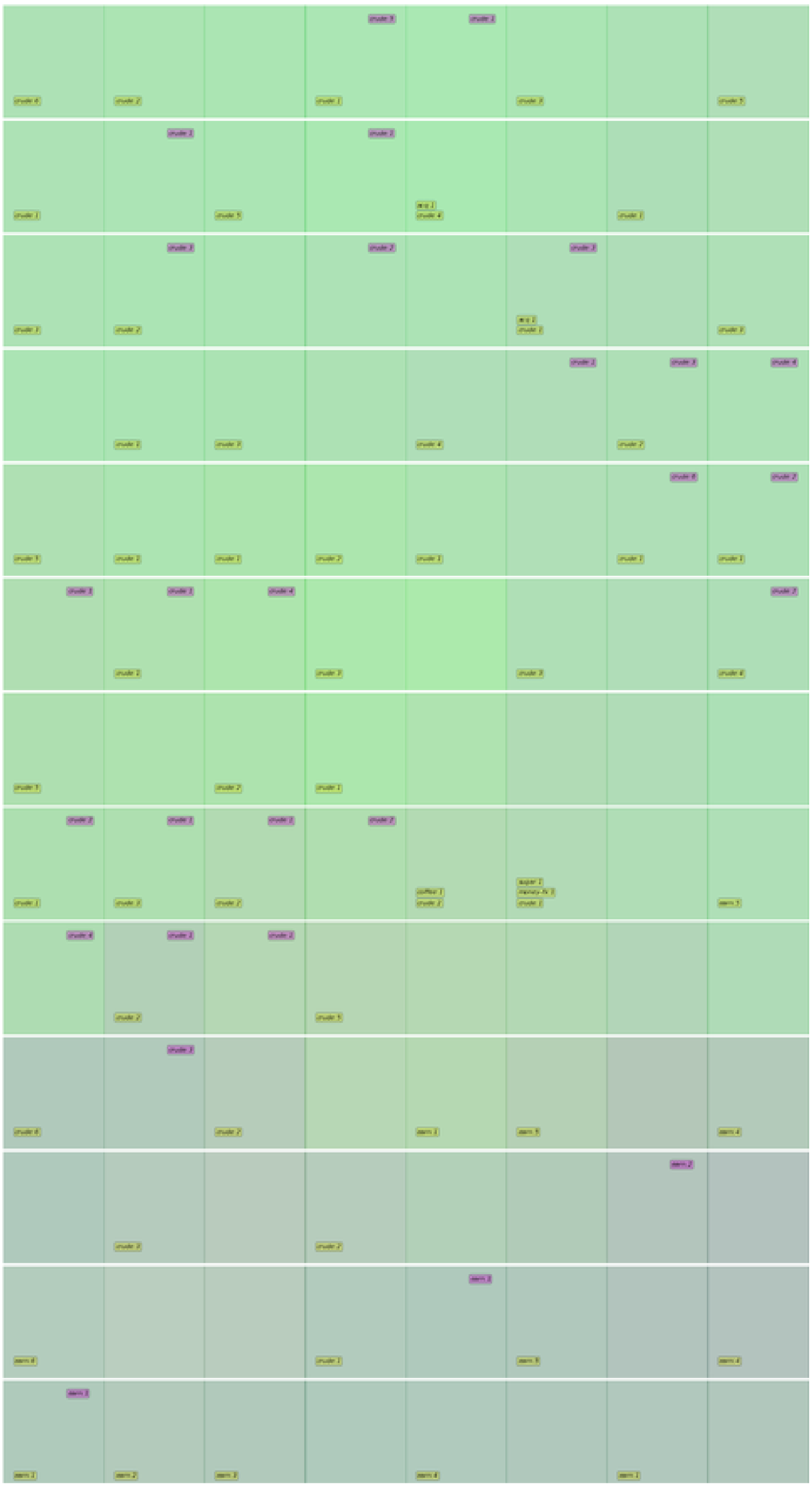


Figure 36: Excerpt of Figure 34a showing the green area dominated by the *crude* label.

Label	Occurrences
acq	326
earn	272
trade	222
interest	172
money-fx	96
sugar	92
money-supply	92
coffee	85
ship	79
crude	12

(a) Cluster 1

Label	Occurrences
earn	2568
acq	1270
crude	241
money-fx	126
money-supply	31
ship	29
trade	28
interest	19
sugar	5
coffee	5

(b) Cluster 2

Table 21: Properties of the clusters of the first partitioning of the Ward-clustered SOM. Corpus: Reuters Corpus (Top 10), dimension: 64x64, features: 33,120.

Figure 36 shows an excerpt of the left side of Figure 34b, and visual inspection shows that this green area contained documents whose labels were dominated by the label *crude*. Tables 21 and 22 list the label occurrences in the clusters from the two partitionings. Furthermore, Table 22 shows that cluster 3 is dominated by the label *crude*, also when the news articles have been dumped into separate directories according to their clusters, turning the clusters into domain-specific text collections and corpora.

8.4.2 *Experiments on the SdeWac Corpus*

This section refers to Figures 37-40 on Pages 172-175. Figure 37 shows the development of a Self-Organizing Map from its randomized beginning (first Epoch) until its completion (last Epoch). Each of the nodes is represented by a vector of the dimension printed in the caption. Figures 37a-37k show that areas with similar vectors (as indicated by reducing to about the same color vector) emerge as training progresses. Until the completion of 30% of the Epochs, the circles visible around the Best-Matching Units indicate their neighborhoods. As the SOM formed, the similarity of the nodes is reflected in these color shadings.

Figure 38 shows a version of the same SOM, annotated with URLs. These URLs do not seem similar (except for two URLs containing

Label	Occurrences
acq	252
trade	189
ship	75
money-fx	46
sugar	44
coffee	21
interest	15
earn	11
crude	8
money-supply	4

(a) Cluster 1

Label	Occurrences
earn	261
interest	157
money-supply	88
acq	74
coffee	64
money-fx	50
sugar	48
trade	33
ship	4
crude	4

(b) Cluster 2

Label	Occurrences
crude	205
acq	169
earn	30
sugar	3
ship	3
trade	1
money-fx	1
coffee	1
interest	0
money-supply	0

(c) Cluster 3

Label	Occurrences
earn	2538
acq	1101
money-fx	125
crude	36
money-supply	31
trade	27
ship	26
interest	19
coffee	4
sugar	2

(d) Cluster 4

Table 22: Properties of the clusters of the second partitioning of the Ward-clustered SOM. Corpus: Reuters Corpus (Top 10), dimension: 64x64, features: 33,120.

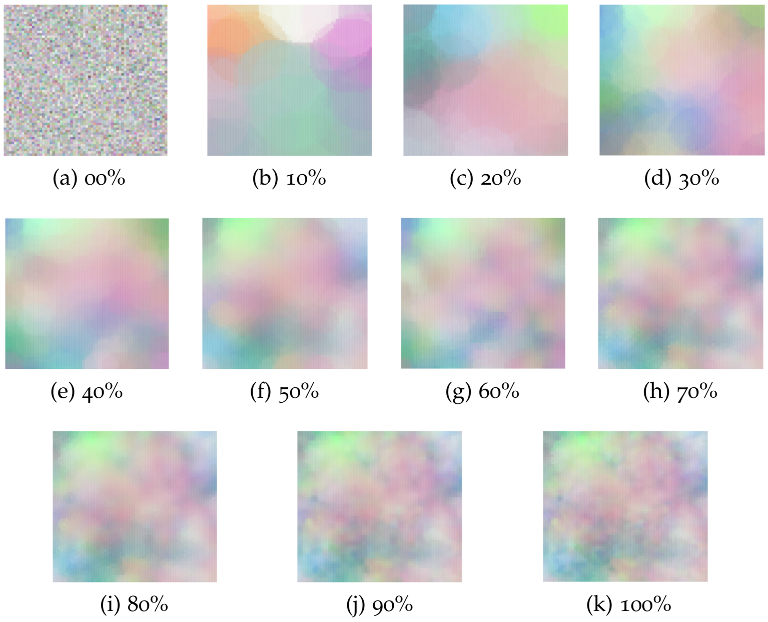


Figure 37: Development of SOM over iterations. Corpus: SdeWac, grid size: 64x64, features: 77,733 dimensions, iterations: 100.

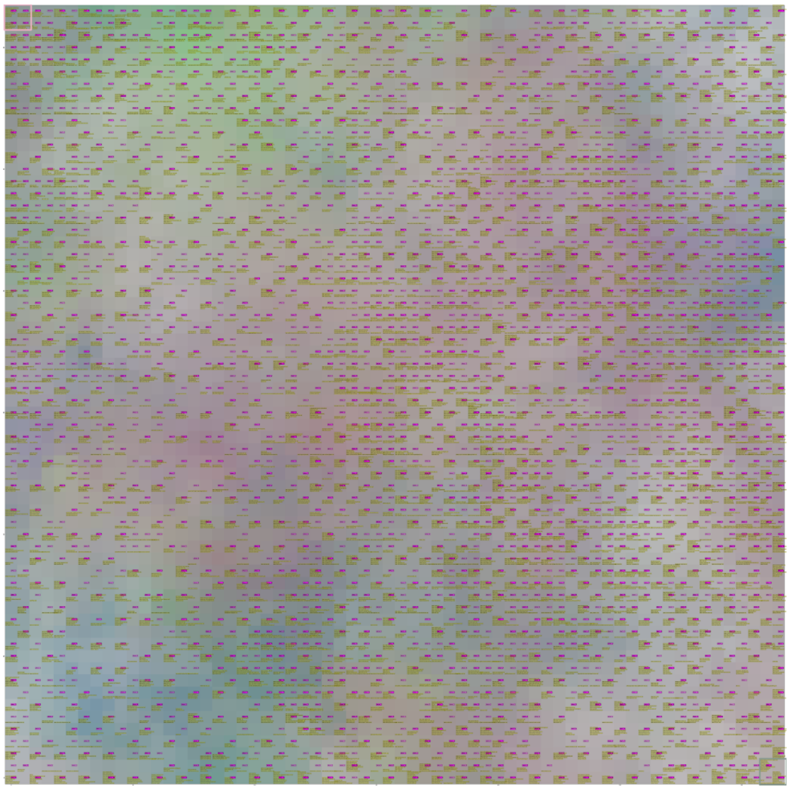
the morpheme *gruen*), and a more thorough analysis of their content would be necessary to determine why they were placed in the same area on the map. Figure 39 shows a detailed analysis of the hierarchical clustering of this Self-Organizing Map. Again, the Ward clustering (row six) provided the most balanced clusters. Figure 40a presents a detailed view of the hierarchical clustering process using Ward's method, with the resulting Self-Organizing Map below in Figure 40b.

Table 23 lists quantitative data on the partitionings resulting from the clustering of this SOM using Ward's method. While the largest cluster is bigger than the next largest by a factor of 3, cluster sizes are more even than for the other methods, which is visible in the figures presented in this section.

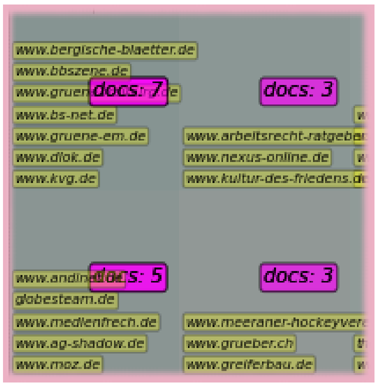
8.4.3 *Experiments on a Subset of the SdeWac Corpus*

Many factors influence the final number of clusters and their sizes, like document vectorization, experimental parameters (e. g., learning rate, initial radius, and number of Epochs), as well as how the numbers of clusters are determined.

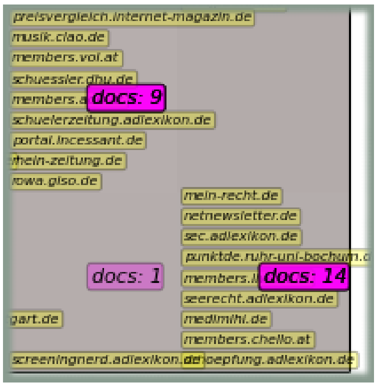
Multiple experiments on different grid sizes and two different vectorizations of the same data were done to explore how the results



(a) Overview



(b) Top-Left



(c) Bottom-Right

Figure 38: SOM annotated with URLs. Top-left and bottom-right corners magnified. Corpus: SdeWac, grid size: 64x64, features 77,733.

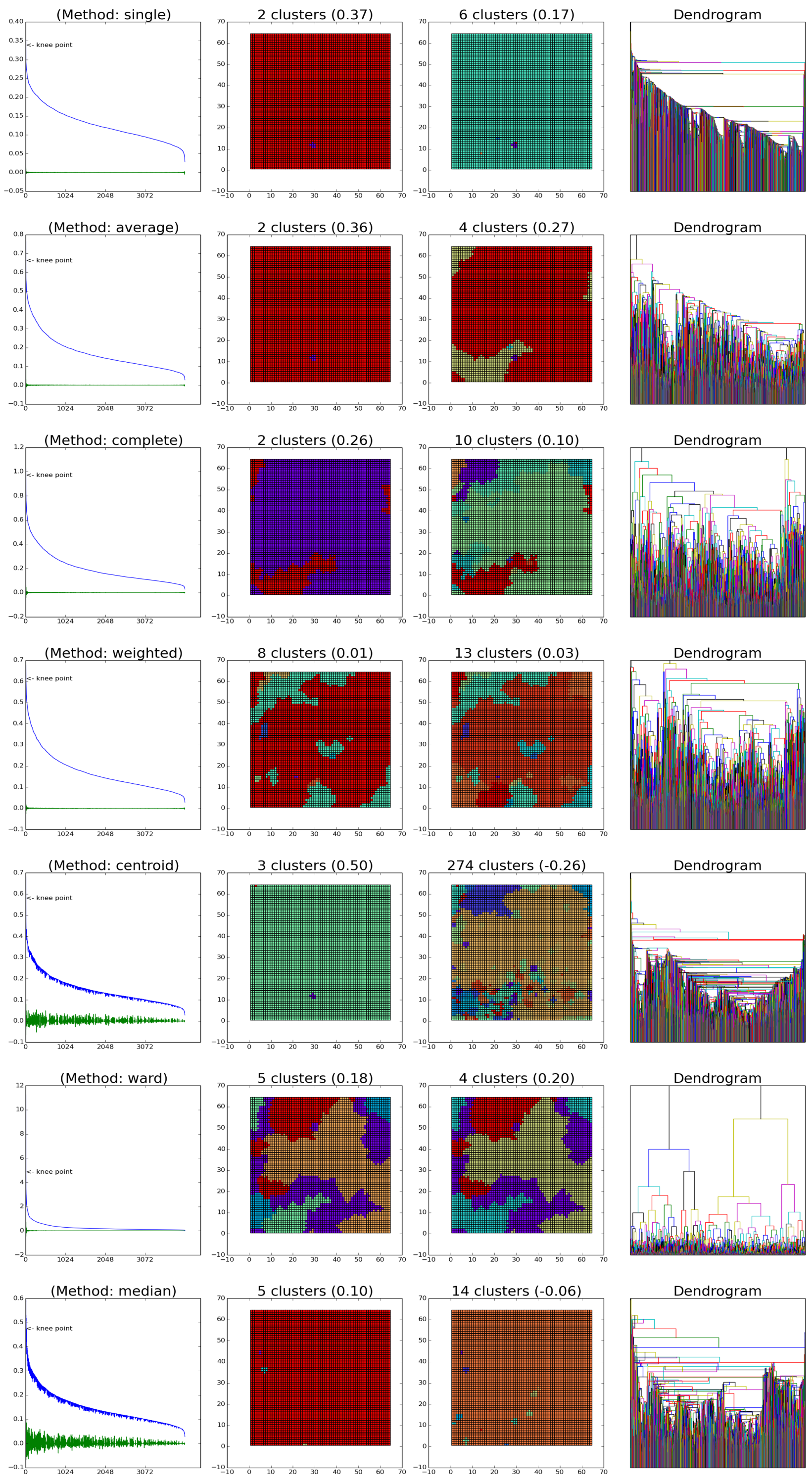
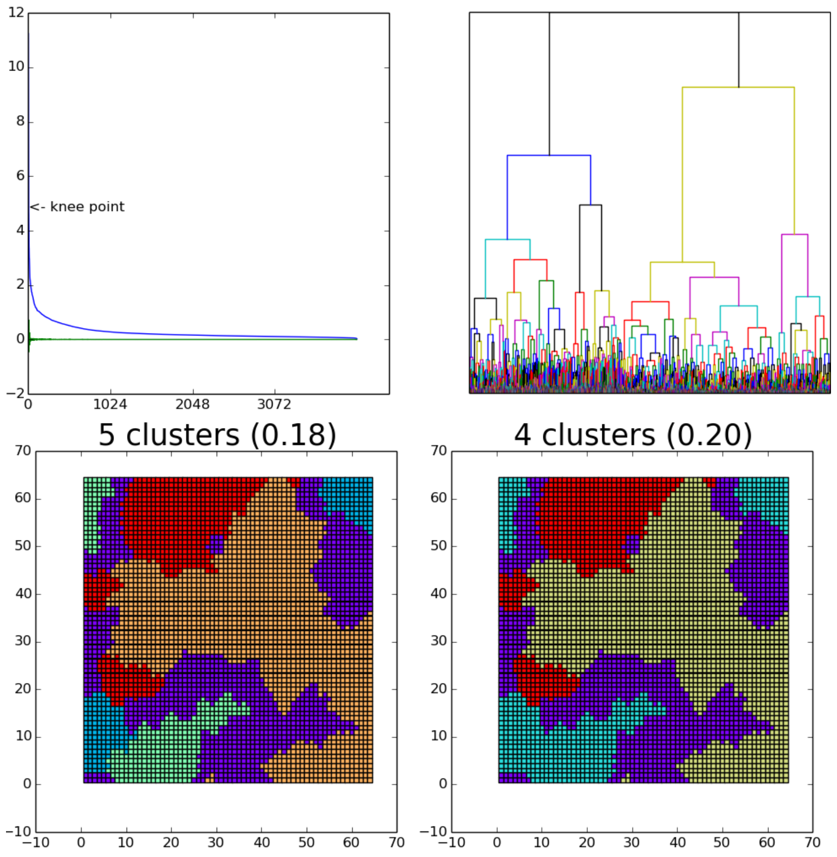


Figure 39: Detailed analysis of hierarchical clustering on SOM. Corpus: SdeWac, grid size: 64x64, features 77,733.



(a) Detailed analysis of hierarchical clustering (Ward method) for SOM.

(b) PCA plot representing nodes as colors.

Figure 40: Juxtaposition of Ward clustering details and PCA-based plotting of SOM. Corpus: SdeWac, grid size: 64x64, features 77,733.

Cluster	Files	Size (MB)
1	2,456	1,209
2	413	23
3	678	500
4	3,601	2,612
5	1,002	272

(a) Partitioning 1

Cluster	Files	Size (MB)
1	2,456	1,357
2	1,091	588
3	3,601	2,939
4	1,002	305

(b) Partitioning 2

Table 23: Properties of two partitionings of Ward-clustered SOM. Corpus: SdeWac, grid size: 64x64, features: 77,733.

changed as each of these variables were changed, keeping the others constant. The experiments were done on 90% of a 1000 document subset of random documents from the SdeWac corpus.

These experiments were separated into Experimental Categories A and B. The parameters of the Self-Organizing Maps in all experiments were: 100 Epochs, a quarter of the grid dimension (i. e., one side) as the initial radius, and an initial learning rate of 0.25. In Experiment Category A, the vectorization process used a cutoff value of 100, and the tf-idf n-grams were computed only from the 1000-document subset of the corpus, resulting in a feature vector dimensionality of 53,160.

In Experiment Category B, the tf-idf transformation of the n-grams up to 7 were computed with a fixed vocabulary from the entire SdeWac corpus (with a frequency threshold of 100), but also applied to a subset of 1,000 random documents, resulting in a feature vector dimensionality of 157,053. Otherwise, the configurations were the same as in Experiment Category A. The number of terms meeting the threshold was larger in the whole corpus than in the subset, and accordingly, the feature space was larger. Appendices A (Figures A2-A8) and B (Figures B2-B8) contain detailed analyses of all these clusterings.

Tables 24 and 25 show the Silhouette Coefficients for partitionings as they vary across the different grid sizes used to create the Self-Organizing Maps and the clustering methods to create clusters. The highest Silhouette Coefficient for each grid size is printed in **bold**, and the highest value per clustering method in *italics*. The 16x16 Self-Organizing Maps had the highest values for most of the experiments in this set. On the clustering method, no clear pattern emerged. (The Silhouette Coefficients for the other partitionings are also included in the figures in the appendices.)

Method	Grid size	8x8	16x16	32x32	64x64
Single	1st Part. (#)	0.18 (3)	**0.41** (7)	*0.36* (3)	0.23 (3)
Average	1st Part. (#)	0.32 (2)	**0.44** (7)	0.35 (4)	0.33 (2)
Complete	1st Part. (#)	0.3 (2)	**0.39** (7)	0.16 (10)	*0.34* (3)
Weighted	1st Part. (#)	**0.32** (2)	0.22 (16)	0.27 (6)	0.18 (25)
Centroid	1st Part. (#)	0.26 (2)	***0.57*** (8)	0.34 (11)	0.01 (128)
Ward	1st Part. (#)	***0.44*** (2)	0.42 (2)	0.16 (4)	0.26 (3)
Median	1st Part. (#)	0.17 (3)	**0.36** (13)	0.18 (17)	0.12 (66)
Single	2nd Part. (#)	*0.68* (46)	0.34 (12)	***1.00*** (504)	0.11 (10)
Average	2nd Part. (#)	0.23 (7)	**0.38** (11)	0.33 (5)	0.32 (4)
Complete	2nd Part. (#)	0.21 (5)	**0.39** (5)	0.17 (12)	0.35 (2)
Weighted	2nd Part. (#)	0.24 (6)	**0.39** (10)	0.35 (4)	0.16 (11)
Centroid	2nd Part. (#)	**0.37** (4)	0.24 (79)	0.26 (36)	0.21 (45)
Ward	2nd Part. (#)	0.28 (3)	0.24 (4)	0.36 (2)	***0.48*** (2)
Median	2nd Part. (#)	0.26 (9)	***0.46*** (10)	0.38 (14)	0.10 (56)

Table 24: Silhouette scores for clusters across grid sizes and clustering methods. Corpus: SdeWac (1000 random documents), features: 53,016.

Method	Grid size	8x8	16x16	32x32	64x64
Single	1st Part. (#)	0.17 (4)	**0.39** (4)	0.32 (2)	0.13 (4)
Average	1st Part. (#)	0.33 (2)	***0.46*** (2)	0.35 (2)	*0.31* (2)
Complete	1st Part. (#)	0.33 (2)	**0.39** (4)	0.09 (4)	0.27 (2)
Weighted	1st Part. (#)	0.14 (2)	**0.30** (18)	0.32 (2)	0.28 (5)
Centroid	1st Part. (#)	*0.46* (2)	**0.59** (2)	0.24 (21)	0.10 (74)
Ward	1st Part. (#)	0.35 (2)	**0.36** (2)	*0.43* (2)	0.27 (3)
Median	1st Part. (#)	0.25 (4)	**0.41** (7)	0.24 (6)	-0.15 (221)
Single	2nd Part. (#)	0.15 (3)	0.36 (7)	***1.00*** (1022)	0.12 (2)
Average	2nd Part. (#)	0.23 (4)	**0.34** (10)	0.31 (4)	*0.29* (4)
Complete	2nd Part. (#)	0.22 (4)	0.21 (8)	0.11 (9)	**0.27** (3)
Weighted	2nd Part. (#)	0.23 (4)	***0.39*** (4)	0.23 (6)	0.28 (2)
Centroid	2nd Part. (#)	0.21 (6)	0.19 (75)	**0.29** (12)	0.00 (152)
Ward	2nd Part. (#)	***0.29*** (4)	0.23 (5)	0.15 (4)	0.2 (5)
Median	2nd Part. (#)	**0.21** (2)	0.01 (40)	-0.05 (50)	-0.39 (46)

Table 25: Silhouette scores for clusters across grid sizes and clustering methods. Corpus: SdeWac (1000 random documents), features: 157,053.

Cluster	File #	Size (MB)
1	1,111	56
2	1,243	61
3	2,146	111

(a) Partitioning 1

Cluster	File #	Size (MB)
1	239	13
2	872	44
3	1,243	61
4	430	22
5	1,716	89

(b) Partitioning 2

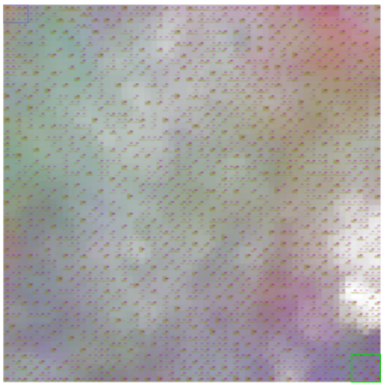
Table 26: Properties of two partitionings of Ward-clustered SOM. Corpus: enTenTen, features: 101,957.

8.4.4 *Experiments on the enTenTen Corpus*

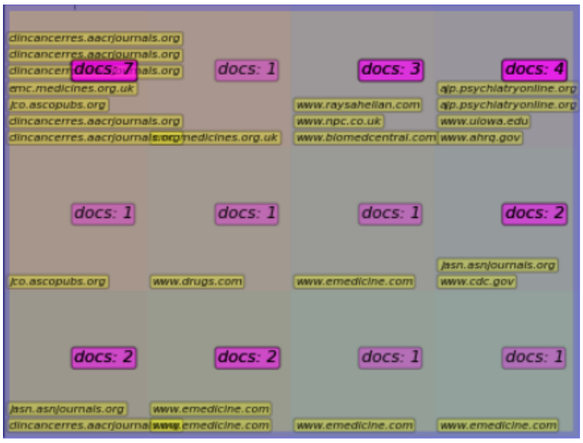
This section refers to Figures 41-44 on Pages 179-182. In this round of experiments, the Self-Organizing Map was trained on a sample of 5,000 documents from the enTenTen corpus, whose document sizes ranged between 30 and 100 Kilobytes. The grid size of the SOM was 64x64, and each document was vectorized with a tf-idf vectorizer using an n-gram range from 1 to 7 and a cut-off of 50, resulting in a vector of 101,957 dimensions representing each document. Figure 41 plots the resulting SOM, with an overview in Figure 41a (annotated with URLs), and two magnified excerpts in Figures 41b-41c. These excerpts indicate that documents with similar URLs cluster together, as medical topics (e. g., `www.emedecines.org` and `www.drugs.com`) in Figure 41b and biblical matters (e. g., `www.bible.org` and `www.bibleteacher.org`) in Figure 41c.

Figure 42 shows a detailed analysis of the clustering process of this SOM and Figure 43 the details of the Ward clustering used in subsequent experiments. Finally, Figure 44 shows a heatmap, Unified Distance Matrix and 3D bar chart of the **win matrix**, i. e., a matrix containing the number of documents domiciled in each node.

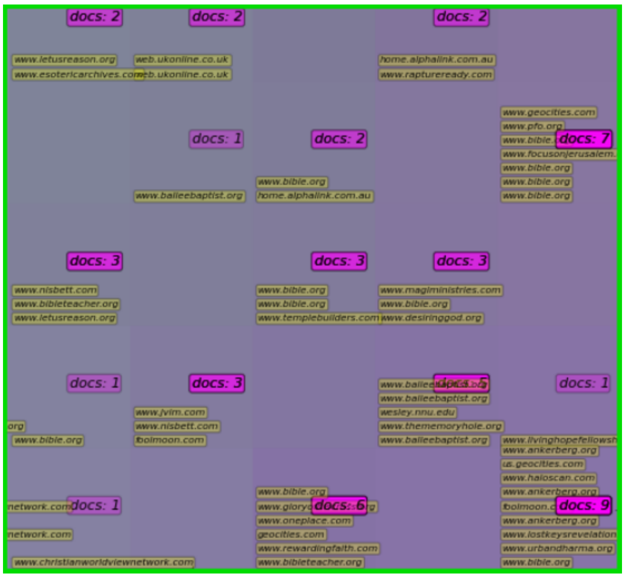
The number of documents in a cluster could be independent of the number of nodes it spans in the Self-Organizing Map, which is also visible in the dendrograms where all nodes are placed next to each other at the bottom. Thus, the above charts illustrate the number of documents situated in each region of the SOM. Table 26 lists quantitative data on the resulting clusters after using Ward's clustering method (shown in the second to last row of Figure 42) and dumping the documents belonging to each cluster.



(a) Overview



(b) Top-Left



(c) Bottom-Right

Figure 41: SOM annotated with URLs. Top-left and bottom-right corners magnified. Corpus: enTenTen (5000 sampled documents), grid size: 64x64, features: 101,957.

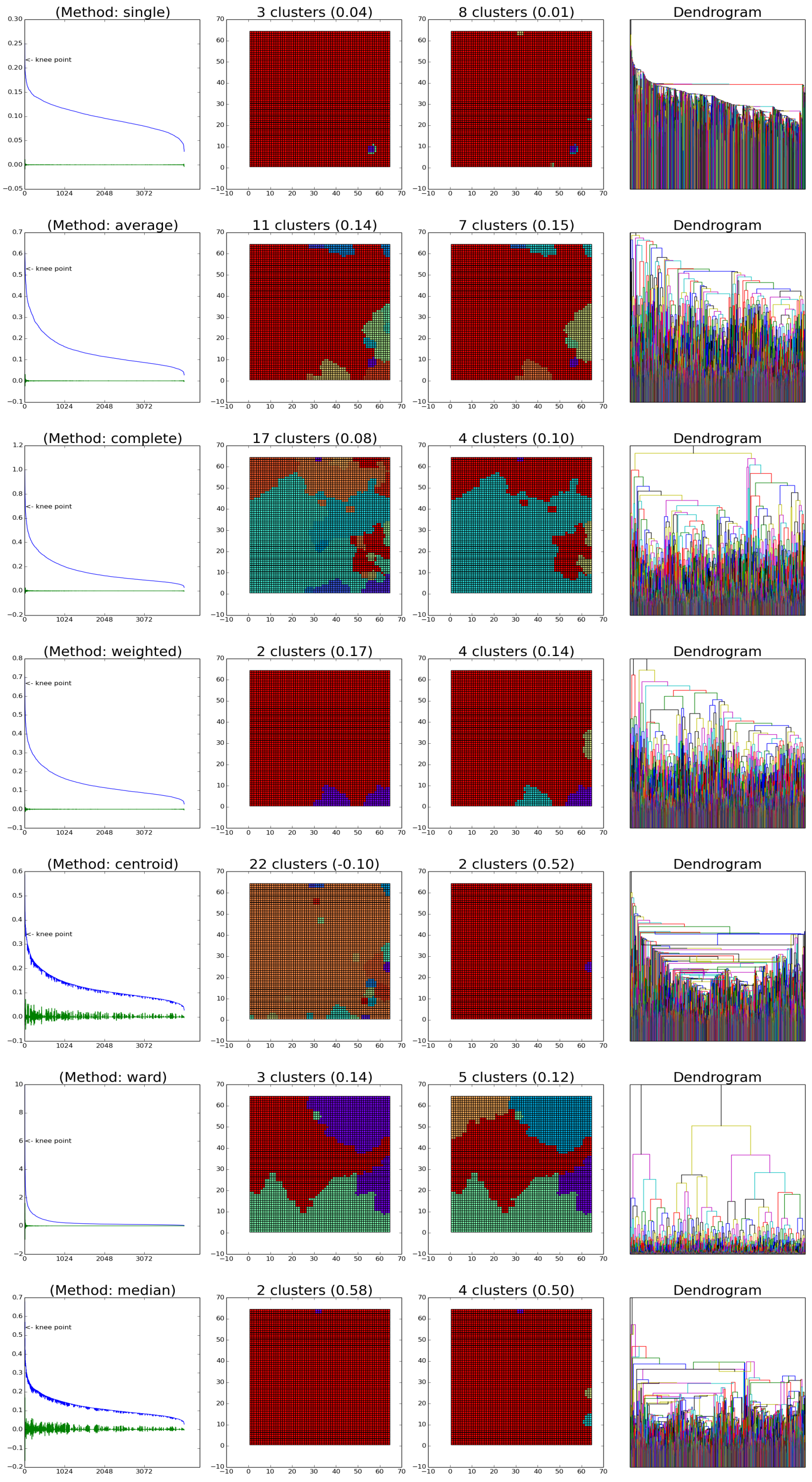
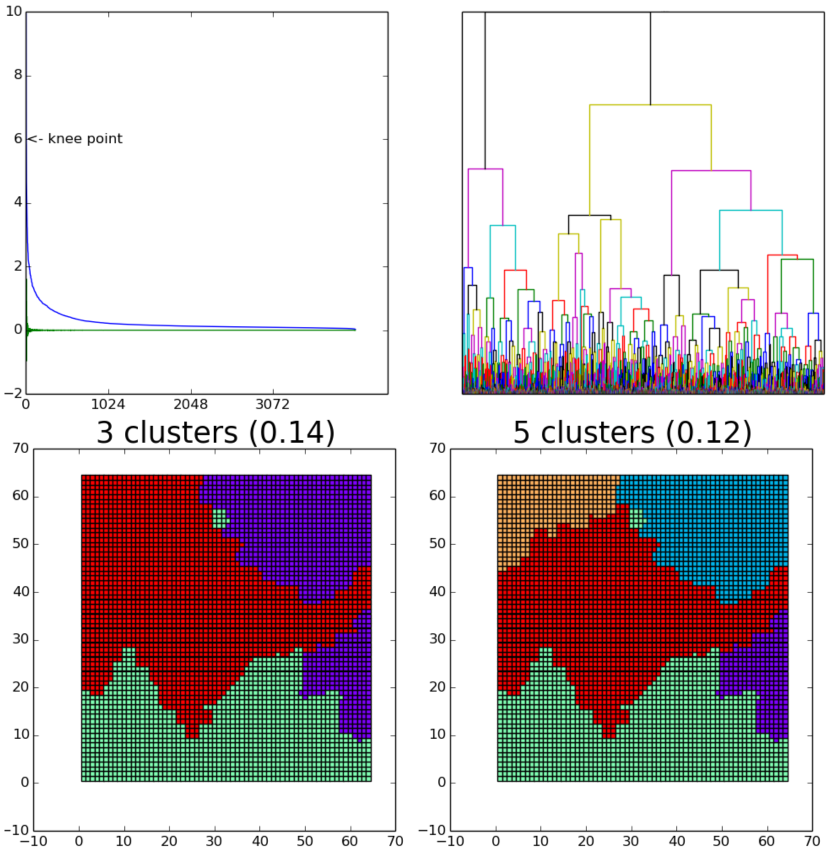
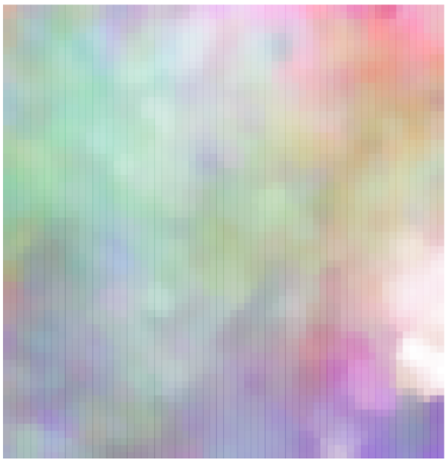


Figure 42: Clustering details for SOM. Corpus: 5000 enTenTen documents, grid size: 64x64, features: 101,957.

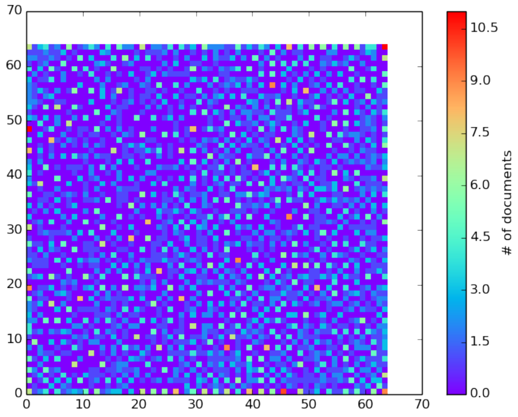


(a) Detailed analysis of hierarchical clustering (Ward method) for SOM.

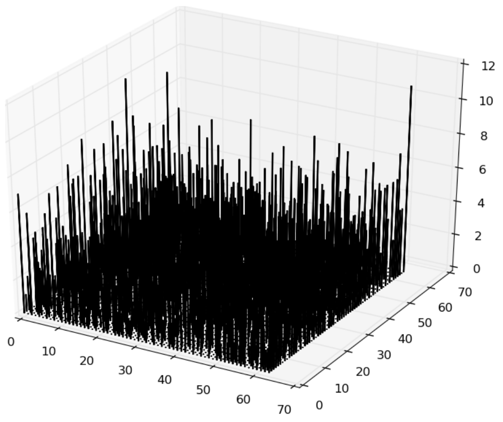


(b) PCA plot representing nodes as colors.

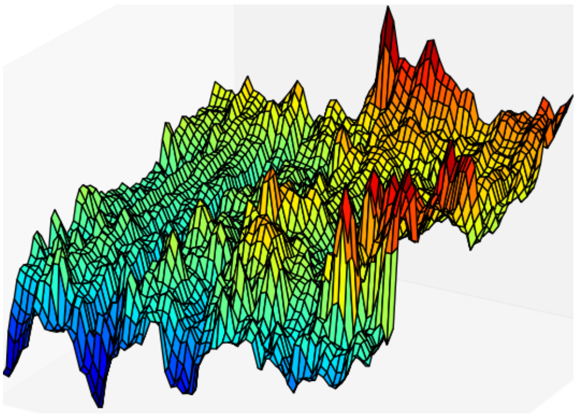
Figure 43: Juxtaposition of Ward clustering details and PCA-based plotting of SOM. Corpus: enTenTen, grid size: 64x64, features 101,957.



(a) Heatmap



(b) 3D bar chart



(c) Unified Distance Matrix

Figure 44: Distribution of documents per node (a and b) and the difference between node vectors (c). Corpus: enTenTen, grid size: 64x64, features: 101,957.

8.5 DISCUSSION

The experiments presented in this chapter were done in two steps; (i) creating a Self-Organizing Map and (ii) clustering its node vectors. The algorithm could also have been applied directly to the observations, i. e., the vectorization of data, and not to only to the resulting Self-Organizing Map. These experiments did not contain such a comparison, but better results from using SOMs as an intermediate step in classification have been observed by, e. g., Vesanto and Alhoniemi [2000].

Hierarchical clustering was done agglomeratively, i. e., successively merging similar clusters. Such clustering requires a distance metric, for which Euclidean distance was used, and a clustering method. Several methods were used, but Ward's method was selected to analyze experiments in further detail. This method was preferred because it provided the most evenly sized clusters, which makes sense for the end task for which they were used (the experiments in Chapter 9 on Domain Adaptation in Machine Translation). However, the hypothesis that Ward's method does better for DA was not tested empirically.

The clusters were evaluated intrinsically, and experiments on different size Self-Organizing Maps had a higher Silhouette Coefficient for the sizes 16x16 and 32x32 than the extremes 8x8 and 64x64. It is also not clear how this would relate to the results on the end-to-end task in Chapter 9.

The only available metadata for the documents in the SdeWac and enTenTen corpora were their URLs, which cannot reliably claim the topic of documents. Manual inspection, however, did show examples of Best-Matching Units being shared by documents of similar-sounding origin, such as documents from the URLs `www.eulenwelt.de` (*owl world*) and `www.eulenmanie.de` (*owl mania*) being adjacent on a Self-Organizing Map of the SdeWac corpus.

9 UNSUPERVISED CLUSTERING AND DOMAIN ADAPTATION

The final experiments of Part II used the corpora that were the output of Self-Organizing Maps-based clustering of unstructured web data (the topic of the previous chapter) for Domain Adaptation in Machine Translation. That way, additional text mined from the web was made available to a generic Machine Translation system for Domain Adaptation to specific domains. These web corpora are generic in the sense that they were not mined or filtered pertinent to these domains. After clustering, portions of the text corpora were used as monolingual pseudo-domain-specific corpora during translation. This additional monolingual text made available for DA is referred to as **auxiliary** text, and the Language Models built from them as auxiliary Language Models.

The experiments with Domain Adaptation had two phases; (i) the segmentation of a large corpus with a Self-Organizing Map and (ii) the utilization of Language Models built from these corpus segments in a Statistical Machine Translation system. The first phase was conducted **off-line**, whereas the leveraging of the LMs was done during decoding — **on-line**. Chapters 7 and 8 described the first phase, and the second is explained below.

This chapter will present these experiments on the integration of the Self-Organizing Map clusters into a Machine Translation pipeline for use in Domain Adaptation. Section 9.1 presents related work, Section 9.2 the datasets, and Section 9.3 explains the proposed method. Section 9.4 shows the experimental results, and finally, Section 9.5 provides a discussion.

9.1 RELATED WORK

Section 4.5 reviewed the main approaches to Domain Adaptation in Machine Translation. The closest architecture to the experiments below was found in Yamamoto and Sumita [2008], who also applied a two-phase approach to multi-domain adaptation. While the authors used a different clustering algorithm, their method similarly discerned between the off-line classification of clusters and their online use as domain-specific Translation- and Language Models.

Additionally, Sennrich et al. [2013] used the K-means algorithm, Hasler et al. [2014] used Latent Dirichlet Allocation, and Yamamoto and Sumita [2008] also clustered sentences using random assignment and entropy reduction, all of which required a fixed number of clusters.

Different selection criteria for choosing text for pseudo-domain corpora include using the cosine difference between tf-idf vectorized documents [Lu et al., 2014], edit distance [Wang et al., 2014] between them, statistical classifiers [Banerjee et al., 2010] or quality estimation (focusing on difficult sentences) [Banerjee et al., 2015].

Multi-domain methods that translate each input unit differently require a mapping between the input and the available domain-specific resources. Any classifier can be applied, as long as it provides an answer to which of the domains is most relevant to the input document. Similarity measures used in data selection methods can also be applied. Xu et al. [2007] found perplexity measures to be more successful than Information Retrieval-based methods. Other approaches include Banerjee et al. [2010] who used Support Vector Machines, Wang et al. [2012] who used perceptrons, and Hasan and Ney [2005] who used regular expressions. Huck et al. [2015] experimented with several versions of Maximum Entropy classifiers and source-side Language Models.

Yamamoto and Sumita [2008] implemented the domain-specific Language Models as extra features in log-linear scoring to apply the domain-specific models on-line. Huck et al. [2015] used interpolated LMs as well as provenance features to indicate domains for sentences.

Cettolo et al. [2016] used hierarchical clustering for clustering commercial domains in a Machine Translation system which uses a vector of domains for the input document to adapt the translation accordingly. By merging domains, and reducing the complexity of these models, only a small degredation of MT output according to BLEU was reported.

9.2 DATA

Two types of datasets were required to conduct these experiments; (i) bilingual corpora were needed to do Machine Translation between the two language pairs English-German and French-English, including annotated datasets to evaluate Domain Adaptation and (ii) auxiliary text resulting from the unsupervised clustering presented in Chapter 8 was used in the DA experiments below.

9.2.1 *Data Collections for Domain Adaptation Experiments*

A dataset from the 2012 Workshop on Statistical Machine Translation (WMT)[1] was used for the English-German experiments. A baseline Statistical Machine Translation model was trained on Europarl version 7 Koehn [2005] and News Commentary [Tiedemann, 2012], containing news text and commentaries from the Project Syndicate. Furthermore, this corpus was used as training data in a series of WMT translation shared tasks on Domain Adaptation. Three test and development sets, distributed for the tasks were used; the European Medicines Agency (EMEA) (medical text), and Subtitles (from translations of TV subtitles) corpora as well as a News corpus not used in training (the Newstest 2011 corpus).

A dataset compiled by Carpuat et al. [2012] was used for the French-English experiments. It is made up of a French-English parallel corpus, with a large Hansard corpus from the Canadian parliament as the out-domain corpus and three domain-specific corpora that were used as test cases; Newstest, EMEA, and Subtitles, available in the Open Parallel Corpus (OPUS) collection [Tiedemann, 2009].

9.2.2 *Data Used for Auxiliary Language Models*

The corpora experimented on in Chapter 8, the SdeWac corpus for German and the enTenTen corpus for English, were used to create auxiliary Language Models in the target languages (see Section 8.2).

See Section 8.4.2 (Figures 38-40) for details on the creation of the German auxiliary text and Section 8.4.4 (Figures 41-43) for the corresponding English text.

9.3 METHOD

Figure 45 outlines the elements of the Domain Adaptation architecture. Leveraging the clustered auxiliary text for Domain Adaptation was done by first building auxiliary Language Models from the clusters, and then using them as auxiliary LMs in Machine Translation. Chapters 7 and 8 described how web corpora were clustered with Self-Organizing Maps. Corpora were compiled from these document clusters, from which 5-gram Language Models models were built. This initial step can be done off-line, independent of the running of the Machine Translation system.

Applying these resources on-line was done by referring each input to the most relevant auxiliary LM during decoding. The experiments

1 `http://www.statmt.org/wmt12/` (Last visited: March 27, 2019.)

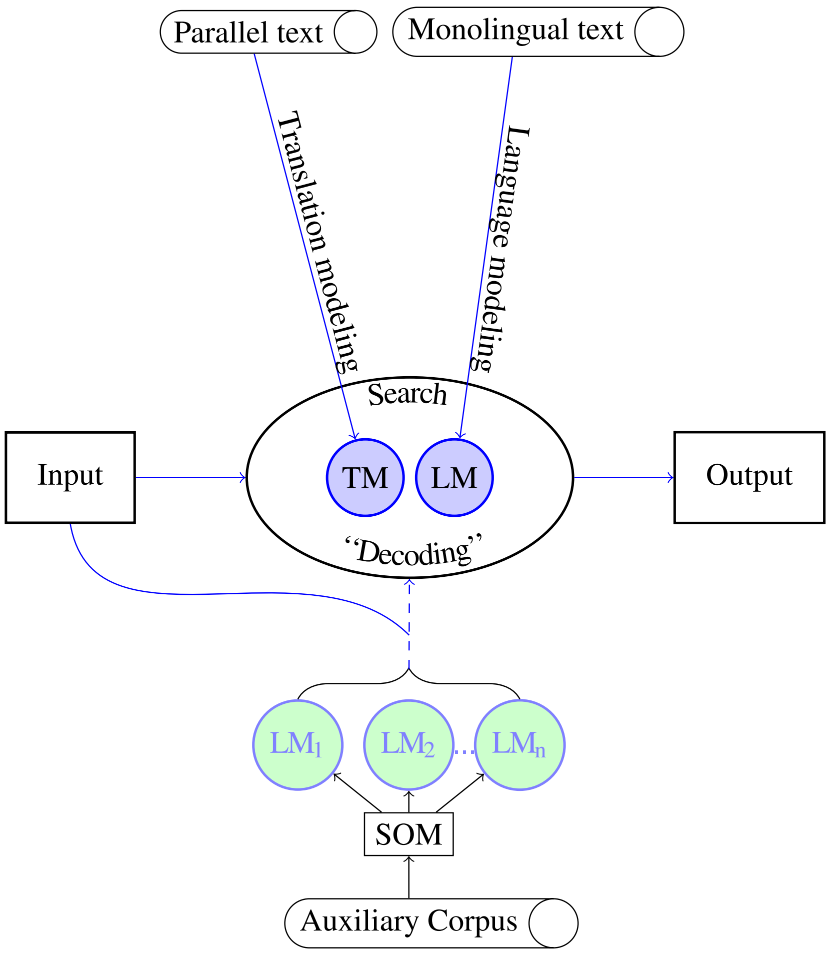


Figure 45: Clustering and Language Model Utilization. The top part is identical to Figure 1 on Page 6.

test a hypothesis that this closest LM would be more helpful for DA compared to the other LMs, or to using no auxiliary LM (referred to as the *No Aux* model) or to using an LM built on all the auxiliary text (referred to as the *Total* LM). After the auxiliary LMs were created, two practical problems remained; (i) determining the closest auxiliary LM for a given input, and (ii) how to apply the auxiliary LM in a Statistical Machine Translation pipeline.

9.3.1 *Selecting Auxiliary Language Model*

A selection method is necessary to choose the auxiliary Language Model for a given input based on some relation between the LMs and the unit of translation (such as document collection, document, or sentence).

If bilingual text is available in the target domain, auxiliary Language Models can be ranked according to perplexity (see Section 3.2.5) of the Target Language version of this data to aid translation of the Source Language version. This makes the unrealistic assumption, however, that a translation of the input to a Machine Translation system is available to aid its very own translation. Providing such translations is precicely what the MT system wants to achieve. Instead, a two-pass solution is proposed, where a document is first translated with the baseline Machine Translation system, and then ranked after the perplexity of the auxiliary LMs on this preliminary translation.

In the experiments presented in Section 9.4, the perplexity of the input documents was used, and the two-pass method was empirically evaluated.

9.3.2 *Clusters and Domain Adaptation*

The auxiliary Language Models were included as separate features in log-linear decoding. An auxiliary LM can be used in a Statistical Machine Translation system also otherwise, e. g., by interpolating LMs directly during language modeling, such that the decoder can query an interpolated model.

These features were added through the configuration file interface of `Moses`), and each test set from each domain was run in separate passes. The same auxiliary LM was preferred for all documents in each domain in the test sets. Furthermore, all auxiliary LMs were applied to all domains.

9.3.3 *Evaluation*

Among the many choices for Machine Translation evaluation (see Section 3.4), four metrics were chosen; BLEU, `Meteor`, Translation Edit Rate, and "Length" — the hypothesis length over reference length as a percentage. Furthermore, statistical testing of the differences between systems was done to assess the significance of results. This testing was done with bootstrap sampling and Approximate Randomization (see Section 3.4.5.1) because of instability in the optimizer determining the parameters in the log-linear model. Henceforth, a parameter set resulting from running the optimizer using the test data is referred to as an **optimizer run**.

`MultEval` by Clark et al. [2011] was used for the calculations using default settings for bootstrap samples and Approximate Randomization re-shuffles ($10,000$). An empirical evaluation of the framework found that the distribution of scores stabilized from 10 optimizer runs. The software applies `jBLEU` $V_{0.1.1}$, `Meteor` 1.5 [Lavie and Denkowski, 2009], and Translation Edit Rate [Snover et al., 2006]. Furthermore, `MultEval` provides the three statistics $\overline{s_{sel}}$, s_{opt}, and p-value. $\overline{s_{sel}}$ is a measure of the variance introduced by the test set selection, measured by the standard deviation of all bootstrap samples, averaged over optimizer runs. This measure is unrelated to optimizer instability. In contrast, s_{opt} is a measure of the variance of the test set using all optimizer configurations, reported as the standard deviation of the metric score on the test set:

$$\overline{s_{sel}} = \sqrt{\sum_{i}^{n} \frac{(m_i - \overline{m}}{n-1})} \tag{63}$$

where m_i is the metric score on the test set for each optimizer run i of n, and $\overline{m}$ their average.

Finally, the p-value is calculated with Approximate Randomization. The reported p-value relates to a baseline (which will be printed in *italics* in the result tables). It is a measure of the statistical probability that a difference of the magnitude presented could be generated again by some random process. Thus, it instills some confidence in the presented results. However, it cannot say anything about, e. g., the magnitude's impact on translation quality as perceived by humans.

9.4 EXPERIMENTS

After a baseline Statistical Machine Translation system was created using a 5-gram Language Model, the perplexity of the three domains' test corpora was measured on all auxiliary LMs provided by the clus-

LM	EMEA	News	Subtitles	EMEA	News	Subtitles
1	**1,532**	893	281	**858**	470	182
2	2,172	1030	374	968	510	242
3	2,162	**744**	**261**	1080	**406**	**167**
4	1,999	1030	556	889	494	331

(a) including OOV words (b) excluding OOV words

Table 27: Perplexity scores using auxiliary Language Models on the German test corpora. (Including and excluding OOV words.)

LM	EMEA	News	Subtitles	EMEA	News	Subtitles
1	5,117	742	**806**	2,533	422	**287**
2	5,619	813	819	2,413	452	309
3	6,085	**661**	866	2,869	**382**	296
4	**4,814**	777	921	**2,175**	430	330

(a) including OOV words (b) excluding OOV words

Table 28: Perplexity scores using the Language Models on the translation of the English test corpora into German with the unadapted system. (Including and excluding OOV words.)

tering experiments. These auxiliary LMs were also 5-gram models with Kneser-Ney smoothing, built with the `KenLM` toolkit [Heafield, 2011]. Perplexity was measured both on the Target Language part of the test corpora as well as the preliminary translation of the Source Language part with the baseline system, and reported values both excluding and including OOV words.

All auxiliary Language Models were used in the translation of all domains, for comparison with the most relevant LM according to the perplexity ranking. Also, two baseline systems were used; (i) the system built without any auxiliary LMs (the *No Aux.* model), and (ii) using all the text that was used for clustering as auxiliary LM (the *Total LM*). Processing all available text did not pose a problem for this limited-size experiment. Feature weights were optimized with Minimum Error Rate Training (MERT) [Bertoldi et al., 2009] (optimized on BLEU score) on a development set, for each Machine Translation configuration and domain. All experiments were repeated 50 times.

9.4.1 *Language Pair: English-German*

Tables 27 and 28 show the perplexity scores on each domain-specific test corpus using the four auxiliary Language Models (second partitioning of the clustering experiments in Chapter 8).

On the German Test corpus (Table 27) perplexity was lowest for LM1 for the EMEA corpus and LM3 for the Subtitles and the News corpora, both when including and excluding Out-of-Vocabulary words on the German test set. LM4 was the smallest in size, making the exclusion of OOV words relevant (see Table 23).

The perplexity score using a translation of the test corpus with a generic system differ markedly from the gold standard version. For the EMEA and Subtitles domains, the top ranking Language Model is different. The translated version of the test corpus prefers the smaller LM4 for the EMEA corpus and LM1 for the Subtitles corpus.

Tables 29-31 on Pages 193-195 show results on the experiments translating domain-specific text from English into German. The preferred model by perplexity ranking (on the test corpus) was better than the other Language Models in all three domains and for all metrics except the Translation Edit Rate on the Subtitles domain. (Note that higher BLEU and `Meteor` values are better, while lower TER values are better as indicated by the arrows in the tables.)

For the EMEA domain (Table 29), LM1 was tied with the Total LM for the BLEU metric and better for `Meteor` and Translation Edit Rate – with significance. As LM3 was relatively large compared to LM1 (2.9G vs. 1.3G), the better scores for LM1 on EMEA domain are noteworthy. The preferred LM when ranking on the translated test corpus, LM4, had the third-best score, of four, for all metrics.

For the News domain (Table 30), results were similarly better for the preferred LM3, but the Total LM scored better for all three metrics – with statistical significance.

On the Subtitles corpus (Table 31), the metrics BLEU and `Meteor` agreed on ranking the preferred LM3 highest with LM1 in second place, whereas the TER was better for LM1. LM1 would have been the preferred LM using the translation of the test corpus to select auxiliary LM. These differences were all statistically significant.

English-German: EMEA (n=50)					
Metric	System	Avg	$\overline{s_{sel}}$	s_{opt}	p-value
BLEU ↑	*LM1*	*19.3*	*0.5*	*0.1*	-
	LM2	18.3	0.5	0.2	0.00
	LM3	18.8	0.5	0.2	0.00
	LM4	18.5	0.6	0.2	0.00
	No Aux	16.8	0.5	0.1	0.00
	Total	**19.3**	0.6	0.1	0.03
Meteor ↑	*LM1*	*35.9*	*0.5*	*0.1*	-
	LM2	35.0	0.5	0.3	0.00
	LM3	35.4	0.5	0.2	0.00
	LM4	35.2	0.5	0.1	0.00
	No Aux	34.3	0.5	0.1	0.00
	Total	35.7	0.5	0.1	0.00
TER ↓	*LM1*	*74.3*	*0.9*	*0.1*	-
	LM2	75.4	0.9	0.5	0.00
	LM3	75.0	0.9	0.2	0.00
	LM4	75.3	0.9	0.2	0.00
	No Aux	76.3	0.9	0.3	0.00
	Total	74.6	0.9	0.2	0.00
Length	*LM1*	*102.7*	*1.1*	*0.2*	-
	LM2	102.6	1.1	0.4	0.00
	LM3	102.7	1.1	0.2	0.00
	LM4	102.7	1.1	0.3	0.08
	No Aux	102.4	1.1	0.3	0.00
	Total	102.7	1.1	0.2	0.00

Table 29: Domain Adaptation results on English-German text across Language Models on the EMEA domain. The closest LM according to the selection criterion is printed in *italics*, the best performing method in **bold**. n is the number of optimizer runs.

English-German: Newstest 2011 (*n*=50)					
Metric	System	Avg	$\overline{s_{sel}}$	s_{opt}	p-value
BLEU ↑	LM1	12.6	0.2	0.1	0.00
	LM2	12.3	0.2	0.1	0.00
	LM3	*13.0*	*0.2*	*0.1*	-
	LM4	12.2	0.2	0.2	0.00
	No Aux	11.5	0.2	0.1	0.00
	Total	**13.2**	0.2	0.2	0.00
Meteor ↑	LM1	34.1	0.2	0.1	0.00
	LM2	33.8	0.2	0.2	0.00
	LM3	*34.3*	*0.2*	*0.1*	-
	LM4	33.7	0.2	0.2	0.00
	No Aux	33.0	0.2	0.2	0.00
	Total	**34.4**	0.2	0.2	0.00
TER ↓	LM1	72.8	0.3	0.3	0.00
	LM2	73.0	0.3	0.4	0.00
	LM3	*72.5*	*0.3*	*0.4*	-
	LM4	73.1	0.3	0.4	0.00
	No Aux	74.0	0.3	0.4	0.00
	Total	**72.4**	0.3	0.4	0.00
Length	LM1	101.9	0.3	0.5	0.00
	LM2	101.3	0.3	0.8	0.00
	LM3	101.7	0.3	0.7	-
	LM4	101.4	0.3	0.8	0.00
	No Aux	101.8	0.3	0.6	0.00
	Total	101.7	0.3	0.7	0.56

Table 30: Domain Adaptation results on English-German text across Language Models on the News domain. The closest LM according to the selection criterion is printed in *italics*, the best performing method in **bold**. *n* is the number of optimizer runs.

English-German: Subtitles (n=50)					
Metric	System	Avg	$\overline{s_{sel}}$	s_{opt}	p-value
BLEU ↑	LM1	10.3	0.4	0.2	0.00
	LM2	9.8	0.3	0.2	0.00
	LM3	*10.4*	*0.2*	*0.2*	-
	LM4	9.0	0.3	0.2	0.00
	No Aux	8.6	0.3	0.1	0.00
	Total	**10.6**	0.4	0.1	0.00
Meteor ↑	LM1	28.1	0.4	0.2	0.00
	LM2	27.8	0.4	0.3	0.00
	LM3	*28.3*	*0.4*	*0.2*	-
	LM4	27.4	0.3	0.3	0.00
	No Aux	27.4	0.3	0.3	0.00
	Total	**28.3**	0.3	0.3	0.07
TER ↓	LM1	**79.7**	0.7	0.4	0.00
	LM2	80.8	0.7	0.5	0.00
	LM3	*79.9*	*0.7*	*0.5*	-
	LM4	81.8	0.7	0.8	0.00
	No Aux	82.6	0.7	0.8	0.00
	Total	79.9	0.7	0.6	0.79
Length	LM1	104.7	0.8	0.8	0.00
	LM2	105.7	0.8	1.1	0.00
	LM3	*105.3*	*0.8*	*0.9*	-
	LM4	106.5	0.8	1.5	0.00
	No Aux	107.4	0.8	1.8	0.00
	Total	105.4	0.8	1.1	0.00

Table 31: Domain Adaptation results on English-German text across Language Models on the Subtitles domain. The closest LM according to the selection criterion is printed in *italics*, the best performing method in **bold**. n is the number of optimizer runs.

LM	EMEA	News	Subtitles	EMEA	News	Subtitles
1	2502	553	**381**	849	324	**200**
2	1857	541	682	703	318	360
3	**1132**	**486**	469	**557**	**287**	250
	(a) including OOV words			(b) excluding OOV words		

Table 32: Perplexity scores using the Language Models on the English test corpora. (Including and excluding OOV words.)

LM	EMEA	News	Subtitles	EMEA	News	Subtitles
1	1885	464	**203**	637	288	**124**
2	1512	443	296	532	274	183
3	**1156**	**404**	233	**453**	**251**	147
	(a) including OOV words			(b) excluding OOV words		

Table 33: Perplexity scores using the Language Models on the translations of the French test corpora into English with the unadapted system. (Including and excluding OOV words.)

9.4.2 *Language Pair: French-English*

Tables 32 and 33 show the perplexity scores for each domain-specific test corpus on the three auxiliary Language Models (second partitioning). The *EMEA* and News corpora selected LM3 and the *Subtitles* selected LM1. The selection was the same on either part of the test corpus.

Tables 34-36 on Pages 197-199 show the results of the experiments comparing the different configurations of the Machine Translation system – with and without auxiliary Language Models on the three domains, EMEA, News, and Subtitles. The LM preferred by the perplexity selection method is printed in *italics*. The p-values represent the probability that the difference between the score of the preferred LM and the other models can be attributed to chance.

For the EMEA domain (Table 34), the preferred model, LM3 performed best of the three auxiliary models but was outperformed by the Total model. However, this was not statistically significant for the BLEU and `Meteor` metrics, but the differences to the other Language Models (1.7 and 2.0 BLEU points respectively) — and the baseline model — were.

French-English: EMEA (n=50)					
Metric	System	Avg	$\overline{s_{sel}}$	s_{opt}	p-value
	LM1	22.2	0.6	0.2	0.00
	LM2	22.5	0.6	0.3	0.00
BLEU ↑	*LM3*	24.2	*0.6*	*0.2*	-
	No Aux	22.0	0.6	0.1	0.00
	Total	**24.2**	0.6	0.2	0.46
	LM1	29.1	0.4	0.1	0.00
	LM2	29.4	0.4	0.2	0.00
`Meteor` ↑	*LM3*	***30.0***	*0.4*	*0.1*	-
	No Aux	29.2	0.4	0.1	0.00
	Total	30.0	0.4	0.1	0.10
	LM1	62.6	0.9	0.3	0.00
	LM2	62.3	0.9	0.6	0.00
Translation Edit Rate ↓	*LM3*	*61.1*	*0.9*	*0.5*	-
	No Aux	62.5	0.9	0.2	0.00
	Total	**61.0**	0.9	0.5	0.00
	LM1	102.3	1.0	0.6	0.00
	LM2	102.4	1.0	0.5	0.00
Length	*LM3*	*102.6*	*1.0*	*0.7*	-
	No Aux	102.3	1.0	0.5	0.00
	Total	102.3	1.0	0.6	0.00

Table 34: Domain Adaptation results on French-English text across Language Models on the EMEA domain. The closest LM according to the selection criterion is printed in *italics*, the best performing method in **bold**. n is the number of optimizer runs.

For the News domain (Table 35), LM3 was preferred as well, but with smaller differences in perplexity. The Total model was better across all three metrics with statistical significance. In these experiments, all auxiliary Language Models had the same score, over the No Aux model, but behind the Total model.

For the Subtitles domain (Table 36), LM1 was preferred. It was the top scoring model for both BLEU and `Meteor`, with statistical significance. The lowest Translation Edit Rate values were tied between

French-English: News (*n*=50)					
Metric	System	Avg	$\overline{s_{sel}}$	s_{opt}	p-value
BLEU ↑	LM1	24.4	0.3	0.1	0.00
	LM2	24.4	0.3	0.2	0.00
	LM3	*24.4*	*0.3*	*0.1*	-
	No Aux	24.0	0.3	0.2	0.00
	Total	**24.7**	0.3	0.2	0.00
`Meteor` ↑	LM1	31.0	0.2	0.1	0.00
	LM2	31.0	0.2	0.1	0.00
	LM3	*31.0*	*0.2*	*0.1*	-
	No Aux	30.9	0.2	0.1	0.00
	Total	**31.1**	0.2	0.2	0.00
Translation Edit Rate ↓	LM1	56.8	0.4	0.2	0.00
	LM2	56.8	0.4	0.2	0.74
	LM3	*56.8*	*0.1*	*0.2*	-
	No Aux	57.1	0.4	0.2	0.00
	Total	**56.6**	0.4	0.2	0.00
Length	LM1	100.9	0.3	0.2	0.02
	LM2	101.0	0.3	0.2	0.00
	LM3	*100.9*	*0.3*	*0.1*	-
	No Aux	101.0	0.3	0.2	0.01
	Total	100.9	0.3	0.2	0.54

Table 35: Domain Adaptation results on French-English text across Language Models on the News domain. The closest LM according to the selection criterion is printed in *italics*, the best performing method in **bold**. *n* is the number of optimizer runs.

Language Models 1 and 3. (The small difference (less than one decimal) in score was not statistically significant.)

For $\overline{s_{sel}}$, the variance due to data selection was smaller for `Meteor` than the other metrics over these three domains, whereas s_{opt}, measuring the standard deviation of the test set, was largely equal between the methods for the three tested domains. Length was included because the brevity penalty in BLEU may skew preferences towards long sentences, while Translation Edit Rate favors short sentences.

French-English: Subtitles (n=50)					
Metric	System	Avg	$\overline{s_{sel}}$	s_{opt}	p-value
	LM1	**13.5**	0.3	0.1	-
	LM2	12.9	0.3	0.1	0.00
BLEU ↑	LM3	13.1	0.3	0.1	0.00
	No Aux	12.7	0.3	0.1	0.00
	Total	13.3	0.3	0.1	0.00
	LM1	**20.3**	0.2	0.1	-
	LM2	19.9	0.2	0.1	0.00
`Meteor` ↑	LM3	20.0	0.2	0.1	0.00
	No Aux	19.8	0.2	0.1	0.00
	Total	20.2	0.2	0.1	0.00
	LM1	**68.9**	0.4	0.4	-
	LM2	69.3	0.4	0.2	0.00
Translation Edit Rate ↓	LM3	68.9	0.4	0.4	0.07
	No Aux	69.8	0.4	0.2	0.00
	Total	69.0	0.4	0.4	0.01
	LM1	86.6	0.5	1.0	-
	LM2	85.9	0.5	0.7	0.00
Length	LM3	85.5	0.5	1.1	0.00
	No Aux	86.4	0.5	0.6	0.00
	Total	86.2	0.5	1.1	0.00

Table 36: Domain Adaptation results on French-English text across Language Models on the Subtitles domain. The closest LM according to the selection criterion is printed in *italics*, the best performing method in **bold**. n is the number of optimizer runs.

These results diverge slightly from those presented by Bungum [2014] and Bungum and Gambäck [2015] on the same datasets. Because the rigorous training with many runs of the optimizer had not been conducted, these papers presented preliminary findings. The conclusions are intact, however, although the figures and some of the rankings have changed. Some variation could also be owing to a slightly different implementation of BLEU and a more recent version of `Meteor`.

9.5 DISCUSSION

As expected, adding more data improved results for all domains in both language pairs. Moreover, the Language Model proposed by the method scored better in almost all cases (higher for BLEU and Meteor, lower for Translation Edit Rate) than the other auxiliary Language Models. These experiments are most similar to Multi-Domain Adaptation work, i. e., methods that adapt a Machine Translation system to multiple domains simultaneously. The improvement in scoring metrics over the baseline was similar to what was reported for studies like Yamamoto and Sumita [2008] and Huck et al. [2015]. However, it was not made explicit how the baseline system was built by Yamamoto and Sumita, i. e., whether it used all the available data concatenated, corresponding to the Total LM in this work.

Since this chapter presented work directed at the use of supplementary data from the web for Domain Adaptation, the method was benchmarked against using no or all supplementary data, as well as the auxiliary Language Models that were not preferred. Nonetheless, it would have been interesting to benchmark the method against using domain-specific resources from supervised sources, such as the development data included in the datasets.

The presented experiments have an exploratory starting point; a corpus was first explored with a Self-Organizing Map to find structure in the high-dimensional data. Next, a hierarchical clustering algorithm was applied to create concrete clusters, with an automatic method for determining how many (the knee method). Hence, the proposed method is unsupervised also regarding the number of clusters.

The discrepancy between the perplexity ranking of the baseline translations of the test corpus and the gold standard version impacted results for the English-German language pair, whereas the conclusions were the same for French-English. The number of Out-of-Vocabulary words when calculating perplexity could explain the differences. Using the gold standard corpus, the numbers of OOV words were 1,885 and 3,042 for LM1 and LM4, respectively. For the translation of the English version of the test corpus, the same figures were 2,624 and 3,413, i. e., a higher number for LM1 and a smaller number for LM4. In other words, the difference had shrunk from 1,157 to 789. Furthermore, the difference in perplexity was relatively small, also on the gold standard version. Additionally, a weak baseline model (as suggested by low BLEU and Meteor scores) may have made estimates based on this translation unreliable. In larger setups with stronger baselines, the difference to the gold standard test corpus should narrow.

The Self-Organizing Maps could also have been used to select auxiliary Language Models for input documents in place of, e. g., perplexity, which was used. Every node in a SOM belongs to a cluster and thereby an LM. By vectorizing an input document with the same features as were used to train the SOM, the Best-Matching Unit of the input document will attribute it to one of the auxiliary LMs.

Repeated runs of MERT optimization were done to ensure that **Monte Carlo error** would not influence the interpretation of results due to randomized initialization. The `Meteor` scores on the Subtitles domain for the language pair French-English ranged from 27.3 to 28.7, with an average score of 28.3. As a consequence, randomly selecting (or worse yet, cherry-picking) an experiment could dramatically alter the interpretation of results, which supports the findings of Clark et al. [2011] on optimizer instability.

Instead of choosing the closest Language Model for a collection of input documents, an auxiliary LM could have been selected for each input on-line, at a chosen granularity, e. g., sentence or document level. A feature function could then look up the preferred auxiliary LM for each line of input, and trigger either the corresponding binary feature or report the real-valued perplexity on all of them.

A feature that distinguishes between the preferred Language Model on a sentence level would require checking the sentence against the available auxiliary LMs at run-time, which could also be simulated by checking each sentence off-line and using a lookup-table during decoding.

Such a feature was developed and tested, but it was not used for these experiments because all documents (and sentences) belonged to the same domain in the available test corpora. A multi-domain corpus could have been created by mixing sentences from the three provided domains in the datasets. The Machine Translation system could then select auxiliary resources on-line, and also optimize weights for all auxiliary Language Models in one pass. The difference would have been that the optimizer would provide weights for all domains features according to the mixture of domains in the development data, as opposed to optimizing only one weight for each LM and domain, which was done in the experiments presented above. Exploring the impact of this alternative method on translation results is left for future research.

Part III

POSTLIMINARIES

The third part of the thesis presents the analysis of the experimental results in Part II on the background of Part I. Chapter 10 summarizes and discusses the results while Chapter 11 concludes on the research questions and proposes ideas for future work.

10 SUMMARY AND DISCUSSION

This chapter will summarize the thesis, and follow up with a discussion of the experiments.

10.1 BRIEF THESIS SUMMARY

Part I presented the research questions and discussed the background for the three main elements of this thesis, Self-Organizing Maps, Machine Translation and Domain Adaptation, in separate chapters.

Part II constituted the experimental part. First, Chapter 5 presented research into the identification of Multiword Expressions applied to a cross-lingual Word Sense Disambiguation task. A focus word (to be translated with a provided context) was translated by looking up the word and its adjacent words in a bilingual dictionary to discover Multiword Expressions as dictionary entries. A list of translation candidates was compiled from the translations. If such a candidate was a member of a provided set of correct translations, it was evaluated as the translation of the focus word in the task.

Chapter 6 followed with experiments on building large-scale Language Models using HPC computing. The experiments into building large-scale n-gram models with the IRSTLM framework used a modified version of scripts for parallel processing for a different PBS scheduler.

The last three experimental chapters presented experiments with Self-Organizing Maps, applied to classification, clustering, and Domain Adaptation. The application to DA of Machine Translation systems had two parts; (i) the use of the SOM algorithm to cluster unstructured data and (ii) the application of these data to the specific problem. The experiments using Self-Organizing Maps will be summarized separately in the next three subsections.

Part III provides analysis and a discussion of results concerning the research questions presented in Chapter 1. Additionally, Part IV contains appendices with additional plots, references, and acronyms.

10.1.1 *Unsupervised Text Clustering on a Structured Document Collection*

Chapter 7 presented an MPI implementation of the on-line Self-Organizing Map algorithm, representing the SOMs as matrices. While it was used for text documents, it could organize any objects that can be represented as real-valued vectors. The SOM algorithm compares the high-dimensional vectors representing the nodes in a low-dimensional topology to each input sample. Subsequently, it updates the closest, termed the Best-Matching Unit — and crucially its neighborhood — towards the input vector, at a given learning rate. The process is repeated for a predefined number of Epochs. A controlling node distributes the SOM matrix to the available MPI processes, such that distances can be calculated in parallel. Each process returns its node with the shortest distance to the input sample to the controlling node, which determines the closest node in the original topology. Example representations of documents are tf-idf conversions (used in the experiments) and raw n-gram counts.

The Self-Organizing Map implementation was used to classify the Reuters Corpus, by making predictions on unseen samples based on an already-trained SOM. Test samples were vectorized and compared to this SOM for a Best-Matching Unit. Majority voting on the labels found in this node after the last epoch of training attributed a label to the test sample. Principal Component Analysis was used to reduce the vectors representing each node to four dimensions, which can be plotted as colors by using them as RGB and alpha channel values. A labeled dataset confirmed that the algorithm clustered similar documents into the same areas. This organization was visualized in the plots where areas were represented by colors and nodes annotated with labels.

10.1.2 *Unsupervised Text Clustering on an Unstructured Document Collection*

Chapter 8 presented experiments on hierarchical clustering of Self-Organizing Maps trained on unstructured text collections (web data). Documents contained some meta-data, such as URLs, which could indicate the document domain. The non-delimited areas in the SOMs were separated into tangible clusters by clustering the SOM nodes. Ward's clustering method produced the most size-balanced clusters, compared to six other methods. Finally, the documents belonging to each cluster were concatenated into text corpora.

The quality of the clustering process was evaluated intrinsically by metrics such as the Silhouette Coefficient, but their relationship to performance in applications remains unclear.

10.1.3 *Incorporating Clusters in an SMT Pipeline*

Language Models were built from the corpora resulting from hierarchical clustering. These LMs were used as auxiliary LMs in a Machine Translation pipeline, and experiments were done on Domain Adaptation datasets for two language pairs and three domains. Input documents were matched to the available auxiliary LMs, ranked after perplexity. Both a Target Language gold standard translation and a back-translation of the input document produced by a baseline system with no auxiliary LMs were used for the ranking.

Results showed that the improvements in Machine Translation performance were largest for the closest Language Model in most cases. The improvement was greater than or equal to using the Total LM for some experiments, i.e., an auxiliary LM built from all supplementary text.

10.2 DISCUSSION

Artificial Neural Network methods have been in use since the beginning of the modern computer era with significant peaks of interest, e. g., around the perceptrons in the 1960s, the backpropagation algorithm [Rumelhart et al., 1986] in the 1980s, and recently with Deep Neural Networks (DNNs). ANNs are now established as the state-of-the-art on several Machine Learning tasks, such as image recognition [Kriegeskorte, 2015] and Machine Translation [Bojar et al., 2016]. While there were continuous improvements in the form of theoretical advances and ideas contributing to this progress, the increased availability of computing power has played a major role in the latest wave of interest.

As a consequence of the new computing capabilities, Collobert et al. [2011] proposed a rethink on Natural Language Processing from the perspective of Deep Neural Networks. The authors used ANNs and multi-task learning for NLP tasks such as Semantic Role Labeling, Part-of-Speech Tagging and Chunking, i. e., finding word categories and syntactic units, respectively.

Similarly, the Self-Organizing Map algorithm dates back to the late 1970s and early 1980s, with a boost of popularity in the 1990s. With its relation to Artificial Neural Networks, it is possible to apply new ideas also to SOMs, such as increasing the number of layers, to use convolutions or recurrent architecture, or simply to scale the number of data points and node vectors up, as was done in the experiments presented in Part II. The unsupervised approach has more biological plausibility than, e. g., methods based on minimizing loss functions. The Self-Organizing Map is useful for mapping complex high-dimensional data to a smaller (mostly 2D) space and visualizing the relationship between the patterns in this space [Floreano and Mattiussi, 2008]. Discovering underlying structures in web data was the idea behind using general web corpora for domain-specific translation.

The supplementary data that was the source of the Language Models used in Chapter 9 could have been clustered with many other algorithms than the Self-Organizing Map. Using an unsupervised method both for clustering unstructured data collections and determining the number of resulting clusters, established such a structure without a predefined number of clusters.

Carter [1994] argued that also in situations where training material cannot be divided into specific domains by extrinsic criteria, clustering would be beneficial to DA, although Carter did point to minor improvements in an end-to-end evaluation. However, the idea of translations customized to each document using a structure established in previously unstructured data collections did offer some promise because using only sections of the supplementary data was sometimes as effective as using all of it. The experimental results in Chapter 9 also saw that effect.

Chapter 7 presented an implementation of the Self-Organizing Map algorithm, where vector comparisons and updates are done in parallel. Since the algorithm requires many such comparisons, it lends itself well to parallel processing because the computation part of the complexity dominates the logarithmic communication part. Thus, computationally expensive high-dimensional vector comparisons can be handled in parallel, but the processing of training samples cannot. It would be necessary to combine the implementation with other ways of reducing the time complexity for the proposed method to scale to even larger datasets, such as variations of the Batch-SOM algorithm.

10.2.1 *Resources*

Web corpora have been extensively for purposes such as n-gram frequencies [Nakov and Hearst, 2005] or translation candidate disambiguation [Grefenstette, 1999]. Volk [2002], and Kilgarriff and Grefenstette [2003] discussed using the web as a corpus for linguistic applications in general. An Association for Computational Linguistics (ACL) Web as Corpus special interest group has been in operation for a decade, with considerable activity[1]. Web-crawled resources have also been applied to Domain Adaptation using directed crawling, i. e., domain-focused web crawling to acquire both domain-specific

1 `https://www.sigwac.org.uk/` (Last visited: March 27, 2019.)

monolingual and bilingual data [Pecina et al., 2015]. Similarly, Lin et al. [2012] used the Google Books[2] n-gram corpus for DA of Part-of-Speech tagging by leveraging **word cluster features** from the Google Books corpus as well as the available in-domain text. Word clusters grouped words that have similar properties (Lin et al. did not present details on the deterministic method).

Recently, large crawls of the web have been made available also to lower-resourced environments, due to increased High-Performance Computing capabilities and initiatives like the Common Crawl, an initiative that regularly publishes full crawls of the Internet under a free license. According to the Common Crawl website[3], their February 2019 crawl consists of over 225TB of data, containing more than 2.9 billion web pages.

10.2.2 *Domains and Domain Adaptation*

Chapter 4 reviewed some definitions of **domains**, none of which provided a clear-cut answer to what the domains are, or how they are delimited [Kittredge, 1983, Carpuat, 2014]. The concept of Semantic Domains [Gliozzo and Strapparava, 2009] entails that domains arise from a given unit of text, and presents a computational way of modeling these. In their experiments, **pseudo-domains** were also induced from unstructured text corpora by algorithms like Latent Semantic Indexing. Offersgaard et al. [2008] also noted that texts not only vary with topic but also, e. g., company guidelines and style. Hence, all input documents exhibit some domain-specificity, and a truly "general" domain does not exist.

Domain-specific Machine Translation has been reported to work well in some cases. Nonetheless, it is hard to translate *any* (general) text with high quality. Lu et al. [2007] pointed out how translation systems are often *blindfolded*, i. e., that the domain of the translated document is unknown at translation time — an argument for **on-line** adaptation, a customized translation for each input. Thus, casting the problem of open-domain Machine Translation as a multi-domain adaptation task is a possible path towards general purpose Machine Translation. This strategy can be seen as applying a divide-and-conquer approach to the problem, where the translation problem is reduced to a multitude of domain-specific translation tasks. However, the number of such tasks, and how they interrelate are open questions.

2 `http://googlebooks.byu.edu/` (Last visited: March 27, 2019.)
3 `http://commoncrawl.org/2019/03/february-2019-crawl-archive-now-available/` (Last visited: March 27, 2019.)

Although the auxiliary data used in the experiments presented in this thesis require some computational resources to process, using all data in one LM model was unproblematic. Given the size of the Internet and its growth, however, it is not possible to add all available text (e.g., from recent crawls) to a Machine Translation system, at least not for all language pairs. With that premise, some segmentation or selection is necessary to leverage the data.

Taking a multi-domain approach using supplementary data that does not require predefined sets of domains or a specific number of them is, therefore, reasonable. In this thesis, a small number of auxiliary Language Models were used (three and four), not enough to give *every* document an adapted translation at the document level. The proposed method must be extended, e. g., by segmenting bilingual data, before a multi-domain system has sufficiently broad coverage to become a general purpose translation system.

Using supplementary data to aid translation also has the advantage that it could account for dynamic changes in languages, i. e., changes in lexical and even grammatical properties over time. While this thesis and related work indicated that multi-domain translation could outperform the mixed-domain (i. e., models trained on many domains simultaneously) and combined data solutions, it was by narrow margins. Thus, a claim that such a multi-domain system can translate any text well would be too strong.

11 CONCLUSIONS AND FUTURE WORK

This final chapter will conclude the thesis by summarizing the findings, particularly addressing the research questions in Chapter 1. The experiments in this thesis indicate answers but also raise questions. Section 11.2 presents some of them as proposals for future research.

11.1 RESEARCH QUESTIONS REVISITED

Chapter 1 posited two research questions, repeated below from Section 1.2:

1. How can unstructured web data be used to facilitate the Domain Adaptation of translation into many different domains simultaneously?
2. How can a Machine Translation system be designed to translate an input document, adapted to the characteristics of that particular document?

Of the three phases of Domain Adaptation, i. e., the identification of structures at some level, the mapping of the structures to distinct domains, and the invocation of this knowledge in the application, Research Question 1 relates to the first phase and Research Question 2 to the third. Phase two has relevance to both questions.

11.1.1 *Facilitating Domain Adaptation With Unstructured Data*

The adaptation of translation systems into multiple domains is known as mixed-domain or multi-domain adaptation, i. e., a combined model covering all domains, or a combination of several models for each target domain [Huck et al., 2015].

Structures in a previously unstructured data collection must be identified before they can facilitate Domain Adaptation. Part II presented experiments on using Self-Organizing Maps and hierarchical clustering for this purpose, with an automated way of choosing the number of clusters.

Experiments on using Self-Organizing Maps for classification of a structured (labeled) corpus showed near the state-of-the-art results. When using this algorithm to cluster unstructured web corpora, the effect of different clustering methods on classification performance could not be evaluated directly. However, an intrinsic evaluation of the clustering indicated that Ward's clustering method produced the most evenly sized clusters. If too much of the data was grouped in one cluster, the difference between the resulting corpus and using all data would be small, and the rest would not be able to impact translation quality. This finding was consistent throughout all experiments, including the experiments presented in the appendices in Part IV. Moreover, URLs indicated that similar documents were grouped in the same clusters.

A mapping between input and a specific domain first requires a decision on the level of granularity (e. g., sentence or document level) and a function from the input instances to the set of domains. The mapping function from the documents to the range of domains does not require that the auxiliary text used for Domain Adaptation is an unstructured text collection.

Chapter 9 presented experiments where auxiliary Language Models were selected based on the lowest perplexity of LMs built on the documents in the clusters. Results showed that using the perplexity measure to rank auxiliary LMs on either gold-standard translations of the input document or back-translations of same most often produced similar selections of auxiliary LMs. Furthermore, the Machine Translation system that applied this method to Domain Adaptation had results that improved over the baseline.

11.1.2 *Adapting Translation to Input Documents*

The second research question addresses how to leverage the resources discussed in the previous section to a Machine Translation system, given a mapping between the identified structures and a set of domains. How the input-specific translation can be done, also depends on the architecture of the Machine Translation system. In the experiments presented in Part II, domain-specific Language Models were used as auxiliary LMs, implemented as log-linear features in the `Moses` framework.

When adapting the translation system to input, two patterns emerge; whether a document collection is considered to represent one *discrete* domain, or if the domain provenance is *continuous*. In the former case, the respective Machine Translation systems can be tuned with a development corpus, e. g., by optimizing feature weights in log-linear decoding. In a multi-domain setting, where a set of domain-specific translation systems or configurations exists, each input is classified as an instance of either of these domains. Whether the system internalizes the classification or translates batches of input documents or sentences with one particular configuration, remains a matter of implementation. As long as the domain provenance is discrete, it is irrelevant whether batch-classification is done off-line or on individual instances at run-time.

In the latter case, with an unknown set of domains (also referred to as dynamic adaptation), an on-line mapping between the input documents and the feature weights is necessary. In such a setting, the feature weights are continuous and idiosyncratic to each input. Examples are Foster and Kuhn [2007] who used the distance between the input document and the components of a mixture model to determine their weighting and Lu et al. [2007] who retrieved similar documents using the input sentence as a query to determine feature weights.

The multi-domain approach was taken in the experiments in Part II, where the batch-level mapping between the input document and the adapted system was done beforehand, and each batch was directed to a particular auxiliary Language Model. Results showed that the translation quality on the batches of discrete documents improved using this method to select the most relevant auxiliary Language Model, in some cases by as much as using all the available supplementary data. However, it was not compared to dynamic adaptation.

Additionally, cache models that continuously update the components of a Machine Translation system as translation progresses could also adapt translations to each input document. This approach was taken, e. g., by Tiedemann [2010] and Louis and Webber [2014].

11.2 FUTURE WORK

More experiments are necessary to refine the proposed method for Domain Adaptation in this thesis. Importantly, it needs to be tested on more datasets and different language pairs and domains, ideally labeled with domains at the document level. The remainder of this section proposes ideas for future research.

11.2.1 *Extension into Parallel Text and Translation Models*

Using the same clustering techniques as presented in this thesis on parallel text could be used to create auxiliary Translation Models for Domain Adaptation. Promising work has been done on mining parallel corpora from the web [Smith et al., 2013].

Using a Self-Organizing Map to cluster such parallel data could provide corpora for auxiliary Translation Models similar to the auxiliary Language Models used in this thesis. These TMs could be used for Domain Adaptation much in the same way as the LMs were used in Chapter 9, i. e., by selecting the most relevant with a similarity measure against the input document. Alternatively, they could be leveraged using other methods such as the mixture modeling of TMs proposed by Sennrich [2012b].

11.2.2 *Other Language Pairs and Domains*

The proposed method should also be tested on other language pairs and corpora to explore whether it generalizes well, e. g., to more distantly separated language pairs. Language domains and sublanguages are not characterized only by thematic variance and genre, but also, e. g., by differences in the properties of the author such as age, style, or emotional state. It is not clear that this method works equally well for all such situations. Some datasets that treat domains purely as differences in topic exist, but more evaluation data is needed to evaluate the method for text that varies along other dimensions.

11.2.3 *Using In-Domain Features to Cluster Text*

Lu et al. [2014] mined more supplementary data from the web based on a limited in-domain corpus. Such ideas could be combined with the proposed method, e. g., by using quantitative data on the in-domain corpus to compute vector representations. This approach would mean that documents are vectorized according to their domain-specific properties before the Self-Organizing Maps are created. Using vectorizations based on a fixed set of domains, would, however, violate the idea of building a general system consisting of an undefined number of domain-specific models.

11.2.4 *The Impact of Distance Metrics and Clustering Methods on Machine Translation*

The proposed method does not dictate the use of a certain distance metric or clustering method when creating the clusters. However, an exhaustive search of this parameter space was not done. Still, the impact of these choices on the end-to-end evaluation is an interesting research question.

Similarly, how the different intrinsic cluster evaluation metrics could potentially correlate differently with final output is worth exploring.

11.2.5 *Using Self-Organizing Maps to Select Auxiliary Language Model*

This thesis discussed several selection criteria for choosing an auxiliary resource in multi-domain settings. It would also be possible to apply the Self-Organizing Maps directly. Input documents could be vectorized with the same vectorizer as used for training of the SOM, and this vector would map to one of the clusters in the SOM, and in turn one of the auxiliary resources.

11.2.6 *Topic Modeling in Multi-Domain Adaptation*

Topic Modeling techniques like Latent Dirichlet Allocation can be used to determine a distribution of topics, where documents have a partial association with multiple topics. Using partial topic memberships in setting configuration parameters in multi-domain adaptation Machine Translation could be a better way to leverage clustered resources.

With partial membership, several auxiliary Language Models could be used in decoding simultaneously, with a weighting according to their degree of membership. Such experiments could also explore how such a weighting would interact with optimization of the log-linear weights.

11.2.7 *Multi-Layered Self-Organizing Maps and Other SOM Configurations*

Multi-layered Self-Organizing Maps algorithms such as MLSOM proposed by Rahman et al. [2007] have been applied to tree-structured image data. Using a configuration with multiple layers, documents can be represented with information at different levels, such as raw corpus counts, tf-idf representations or the output of parsing. A multi-layered Self-Organizing Map would process each information level separately and augment the next layer's feature vectors with new input types.

It is not clear how a Self-Organizing Map including more layers (or recurrency) would cluster a set of documents. Thus, it would be interesting to experiment with architectures inspired by advances in Deep Neural Networks and see if Domain Adaptation results will improve as they have done, e. g., for image recognition tasks.

11.2.8 *Multi-Lingual Domain Models*

Creating multi-lingual domain models to extract mappings between languages can be used to develop domain-specific monolingual and pseudo-parallel corpora from web data. Self-Organizing Maps could be used to cluster text from more languages and to explore the relations between these clusters.

Subsequently, such relations between clusters could be used to create pseudo-parallel corpora, e. g., for use in creating domain-specific Statistical Machine Translation models.

Part IV

APPENDICES

+

A CLUSTERING WITH OF SOMS OF VARIOUS GRID SIZES. CORPUS: SDEWAC, FEATURES: 53,160.

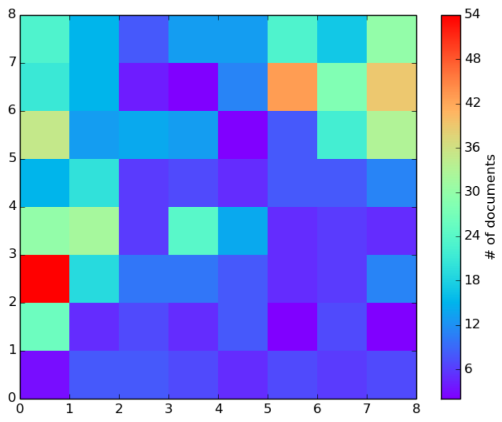
This appendix contains the full set of visualizations of four Self-Organizing Maps created on 90% of a 1000-document subset of the SdeWac corpus, discussed in Subsection 8.4.2. In this appendix, the documents were vectorized with a cut-off value of 100, resulting in a 53,160 features for each document.

Documents per node are visualized with heatmaps and bar charts for each Self-Organizing Map in a combined figure, which also contains a snapshot of a Unified Distance Matrix is plotted showing the separations between areas in the SOM. Additionally, all SOMs are clustered, and a detailed analysis of the clustering processes are plotted in a separate figure.

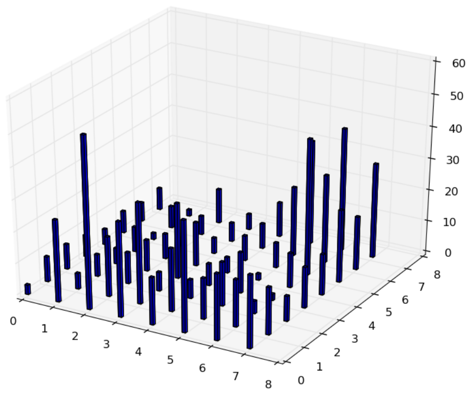
Listing A1: Configuration parameters for Exerpiment Category A.

```
learning rate = 0.25
epochs = 100
initial BMU radius = quarter of grid size

cut-off value for tf-idf-transformation = 100
```



(a) Heatmap


(b) 3D bar chart

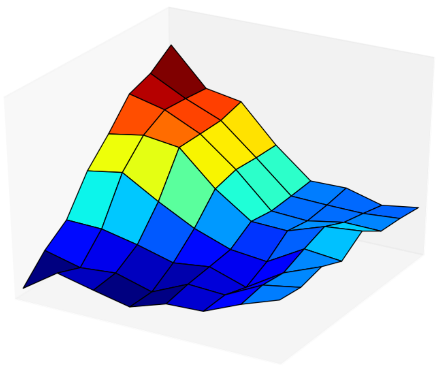

(c) Unified Distance Matrix

Figure A1: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 8x8.

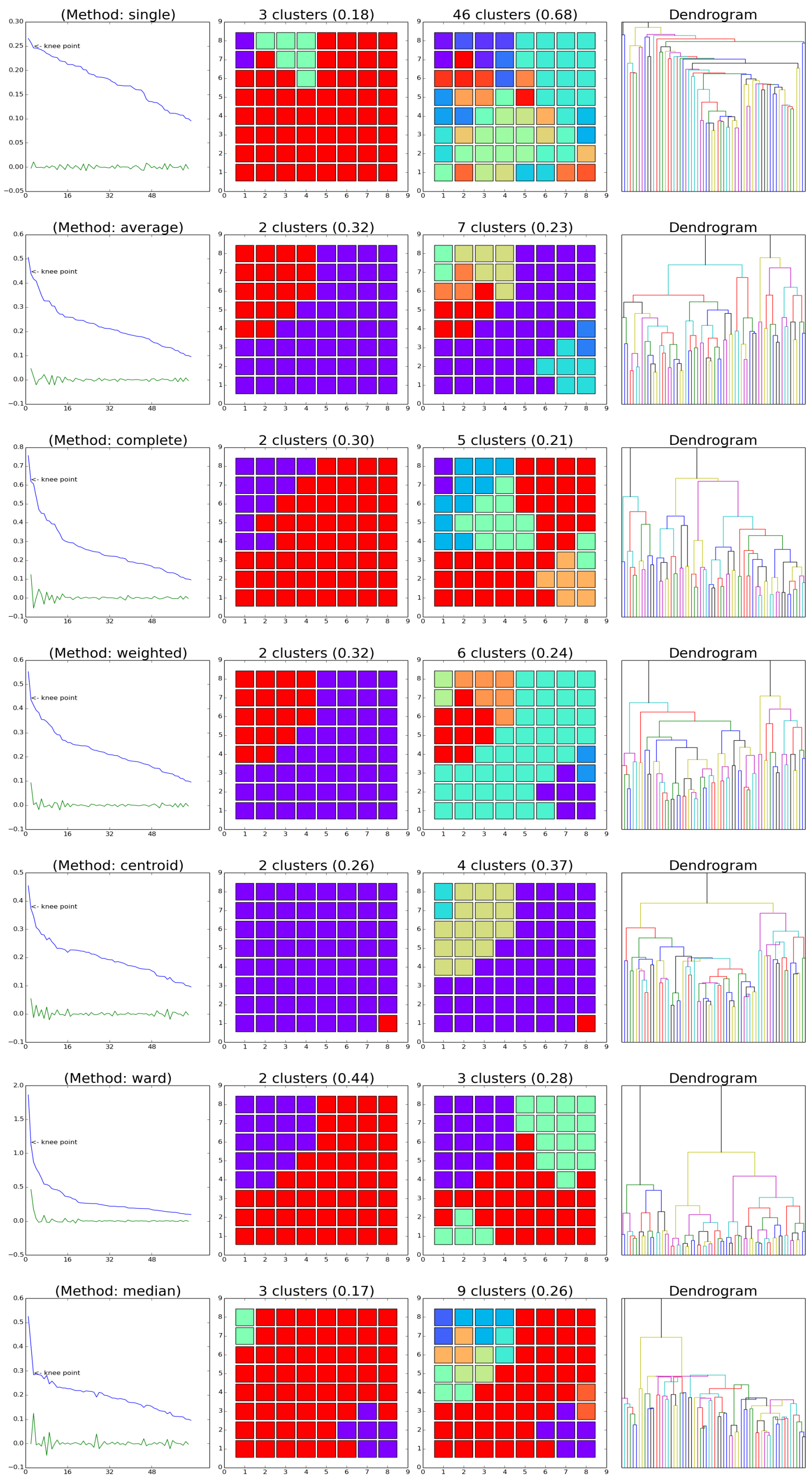
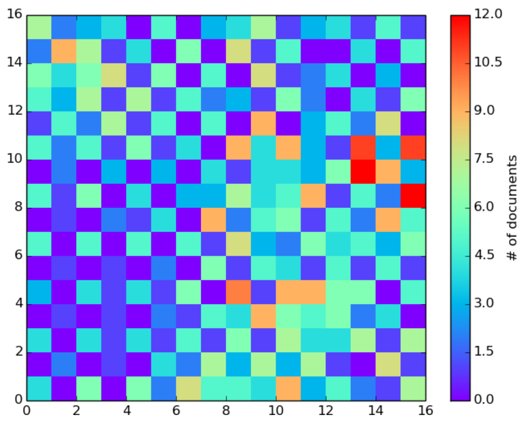
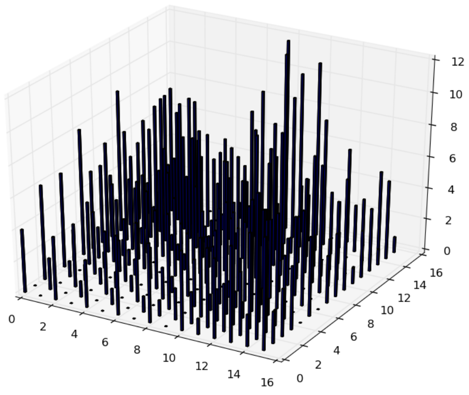


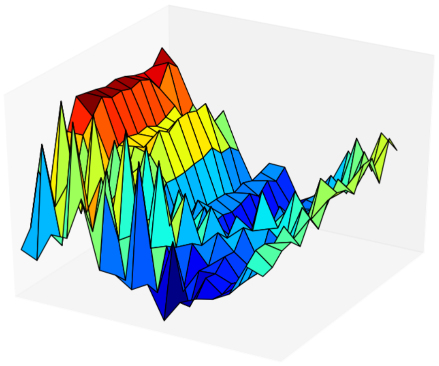
Figure A2: Detailed analysis of hierarchical clustering for 8x8 SOM.



(a) Heatmap



(b) 3D bar chart



(c) U Matrix

Figure A3: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 16x16.

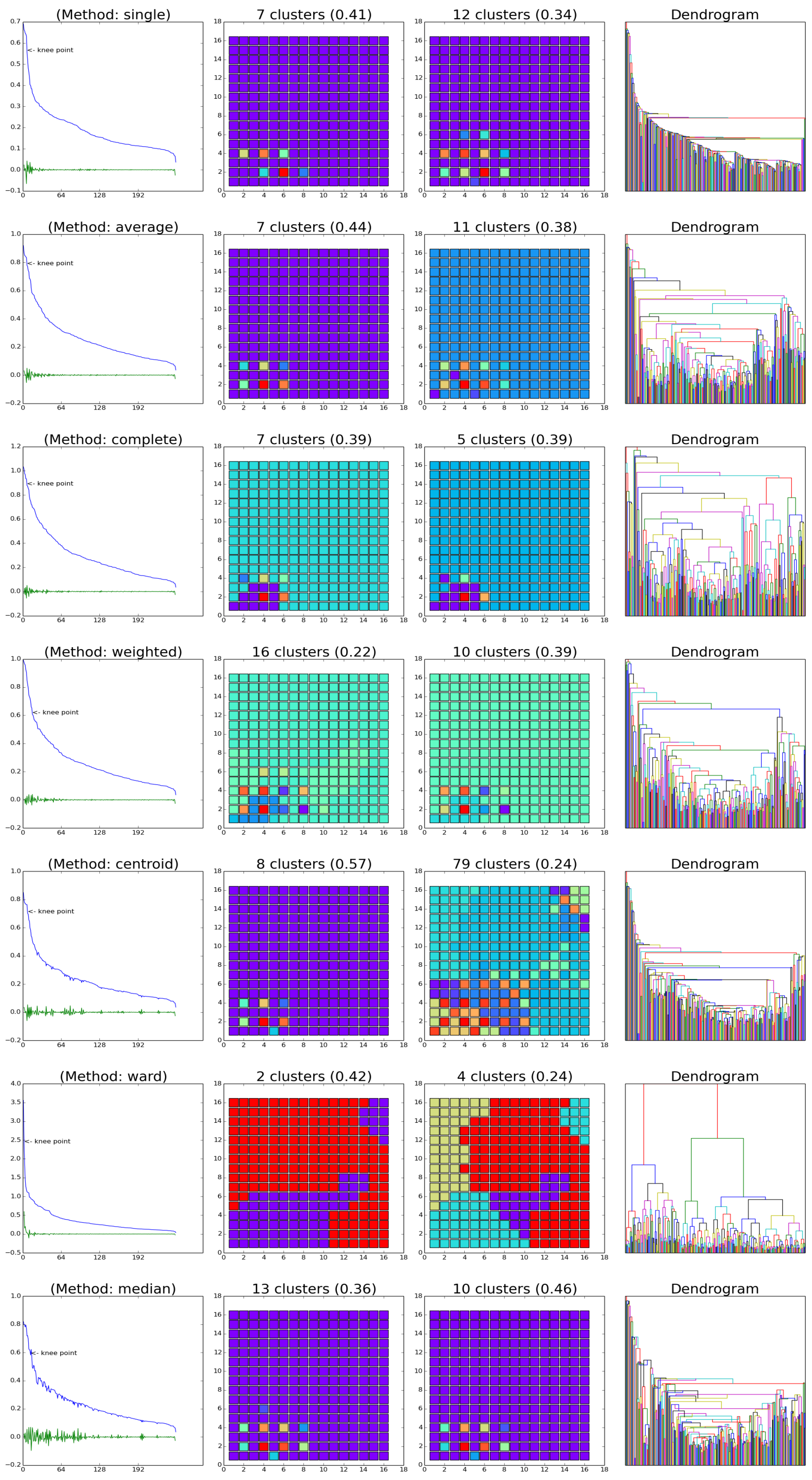
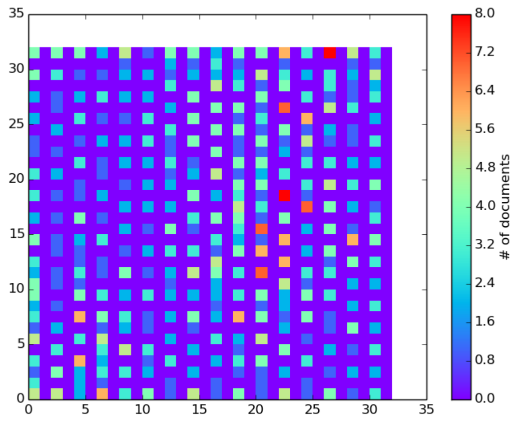
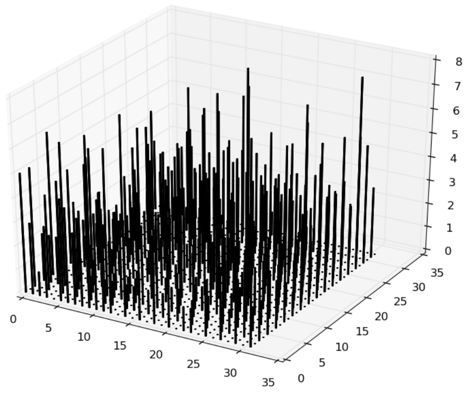


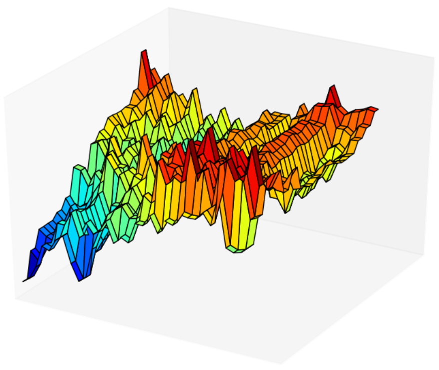
Figure A4: Detailed analysis of hierarchical clustering for 16x16 SOM.



(a) Heatmap



(b) 3D bar chart



(c) U Matrix

Figure A5: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 32x32.

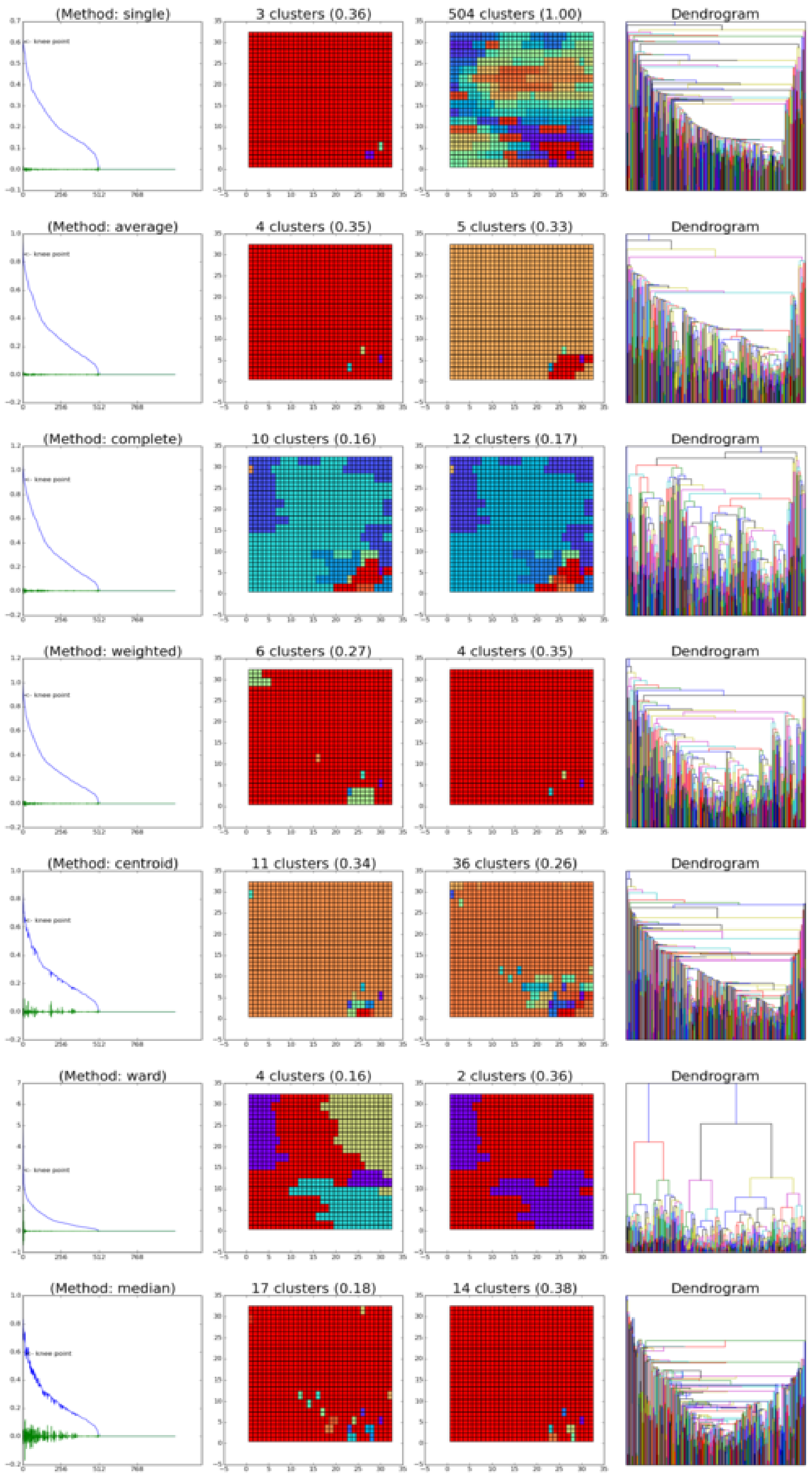
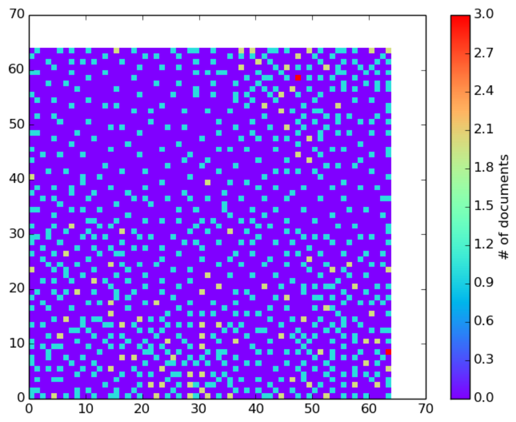
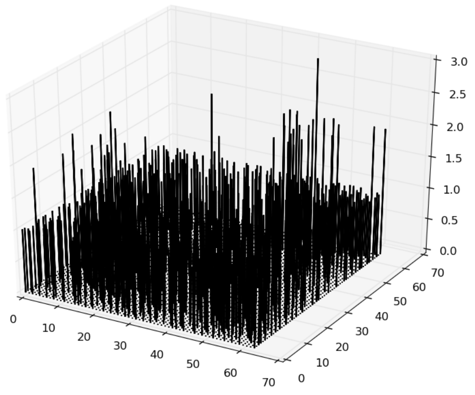


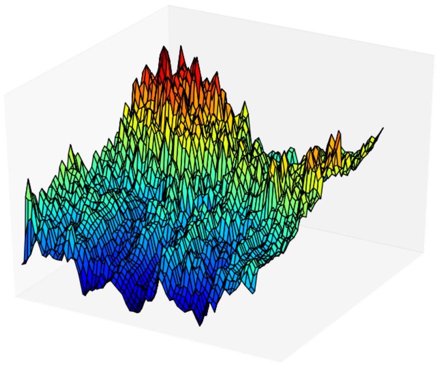
Figure A6: Detailed analysis of hierarchical clustering for 32x32 SOM.



(a) Heatmap



(b) 3D bar chart



(c) Unified Distance Matrix

Figure A7: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 64x64.

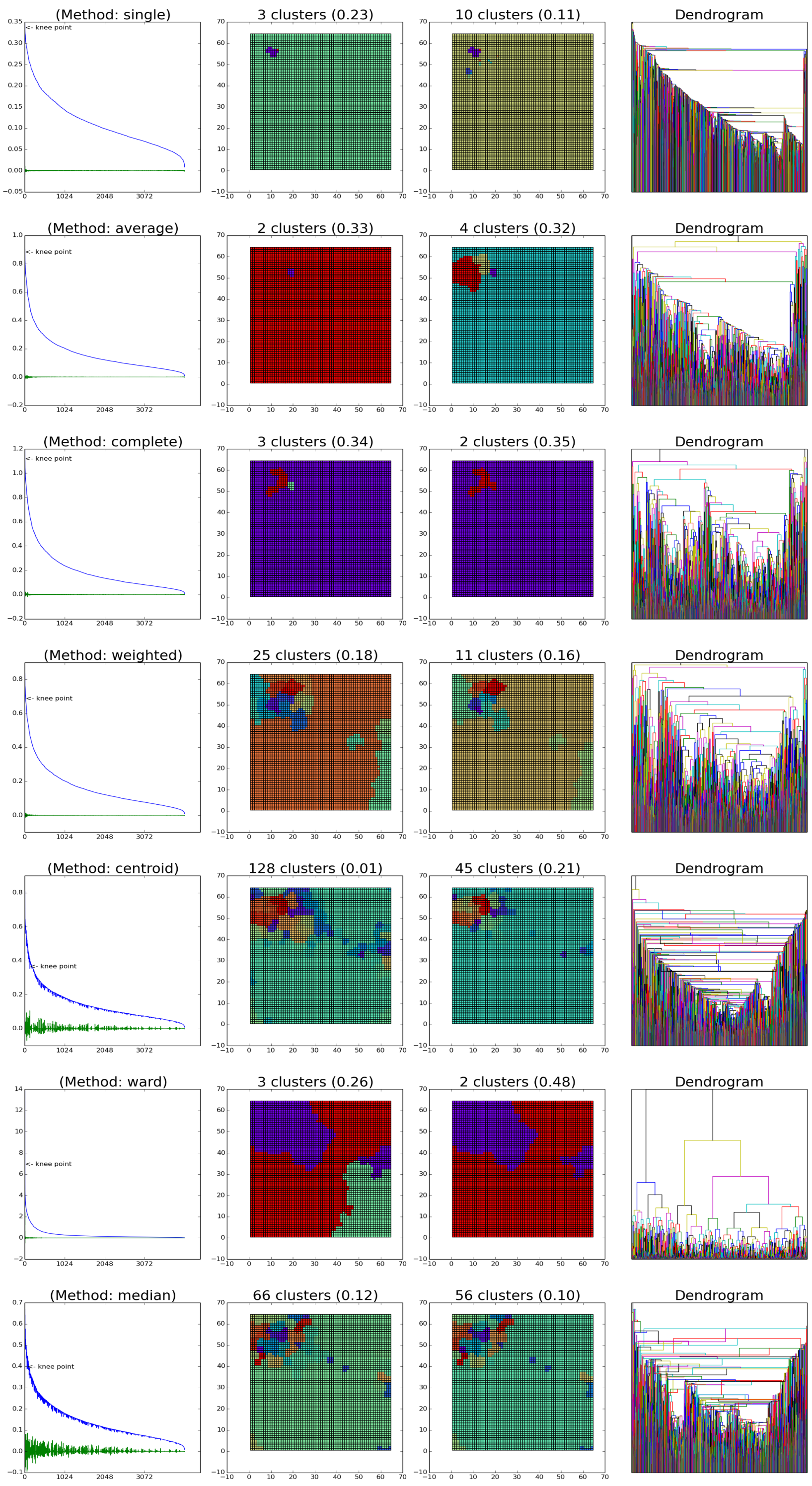


Figure A8: Detailed analysis of hierarchical clustering for 64x64 SOM.

B CLUSTERING WITH SOMS OF VARIOUS GRID SIZES. CORPUS: SDEWAC, FEATURES: 157,053.

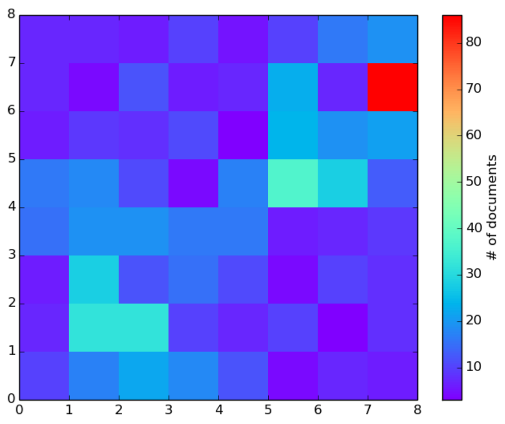
This appendix contains the full set of visualizations of four Self-Organizing Maps created on 90% of a 1000-document subset of the SdeWac corpus, discussed in Subsection 8.4.2. In this appendix, the documents were vectorized based on a fixed vocuabulary derived from the entire SdeWac corpus, resulting in a 157,053 features for each document.

Documents per node are visualized with heatmaps and bar charts for each Self-Organizing Map in a combined figure, which also contains snapshot of a Unified Distance Matrix is plotted showing the separations between areas in the SOM. Additionally, all SOMs are clustered, and a detailed analysis of the clustering processes are plotted in a separate figure.

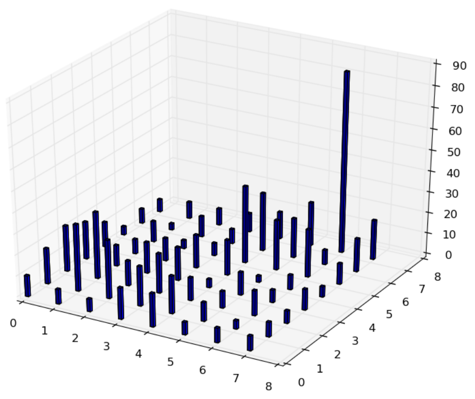
Listing B1: Configuration parameters for Exerpiment Category B.

```
learning rate = 0.25
epochs = 100
initial BMU radius = quarter of grid size

cut-off value for tf-idf-transformation = 100, vectorized
    on the full SdeWac Corpus
```



(a) Heatmap



(b) 3D bar chart

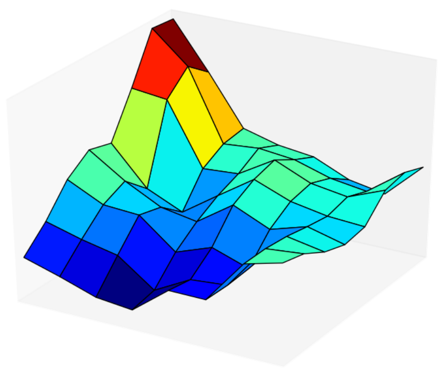

(c) Unified Distance Matrix

Figure B1: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 8x8.

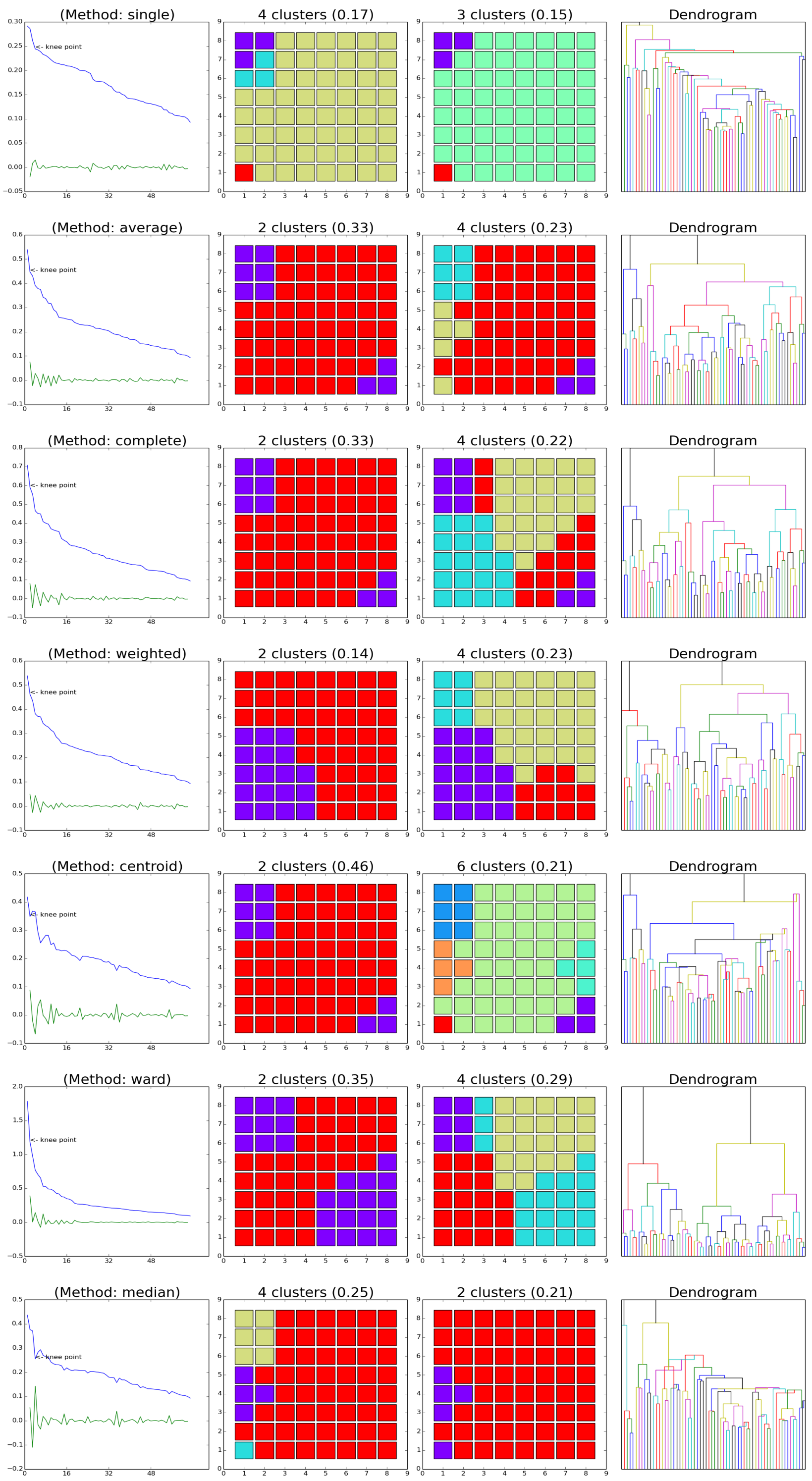
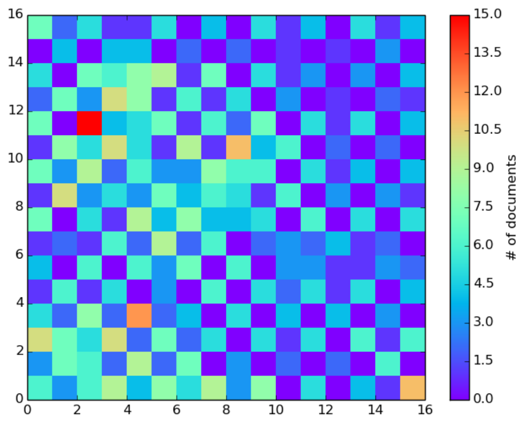
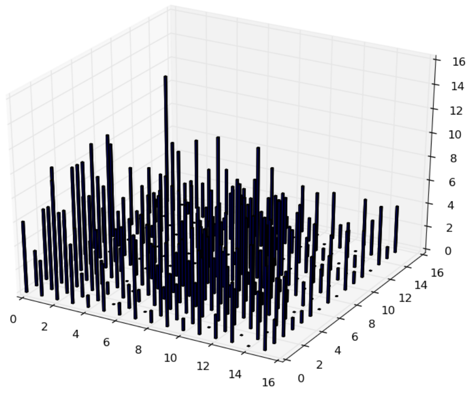


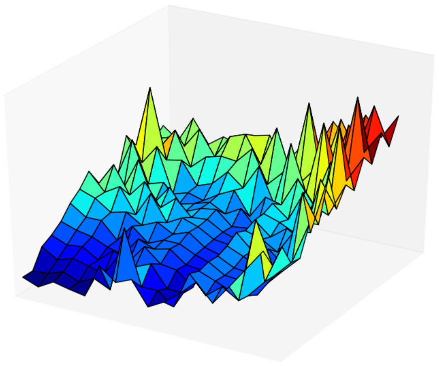
Figure B2: Detailed analysis of hierarchical clustering for 8x8 SOM.



(a) Heatmap



(b) 3D bar chart



(c) Unified Distance Matrix

Figure B3: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 16x16.

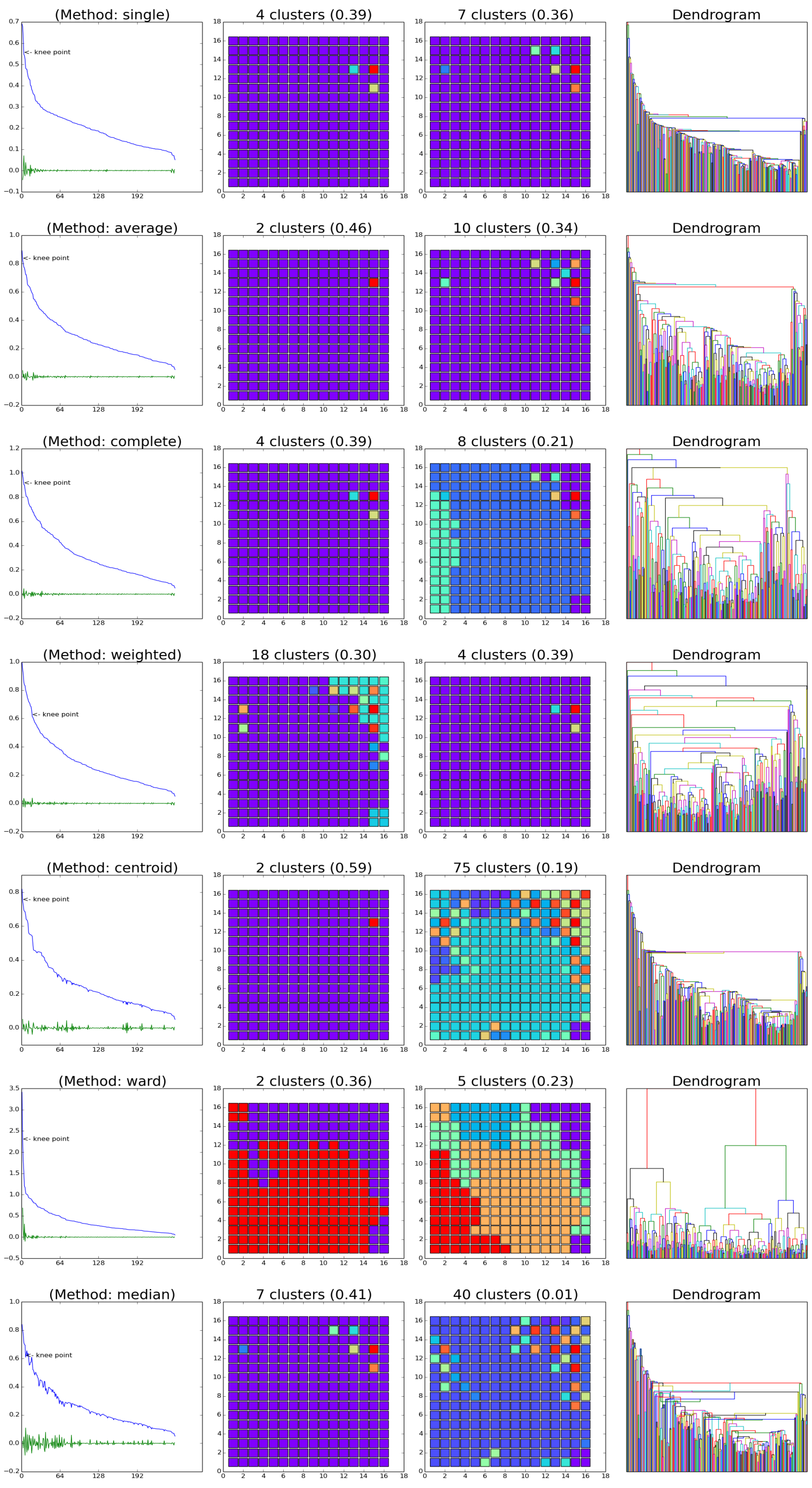
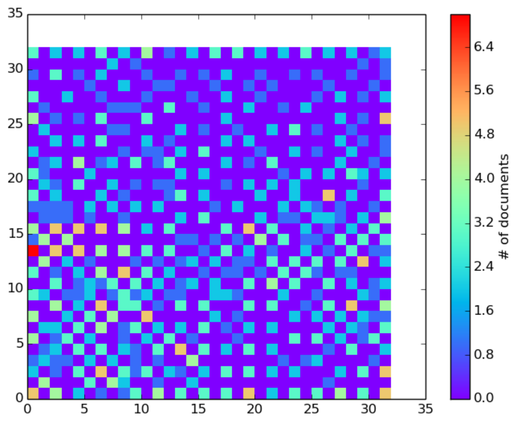
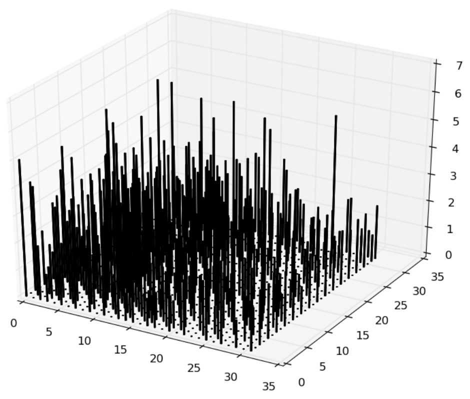


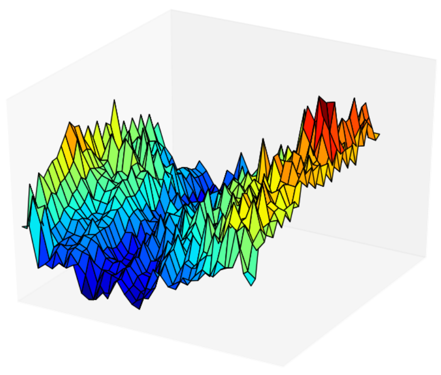
Figure B4: Detailed analysis of hierarchical clustering for 16x16 SOM.



(a) Heatmap



(b) 3D bar chart



(c) Unified Distance Matrix

Figure B5: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 32x32.

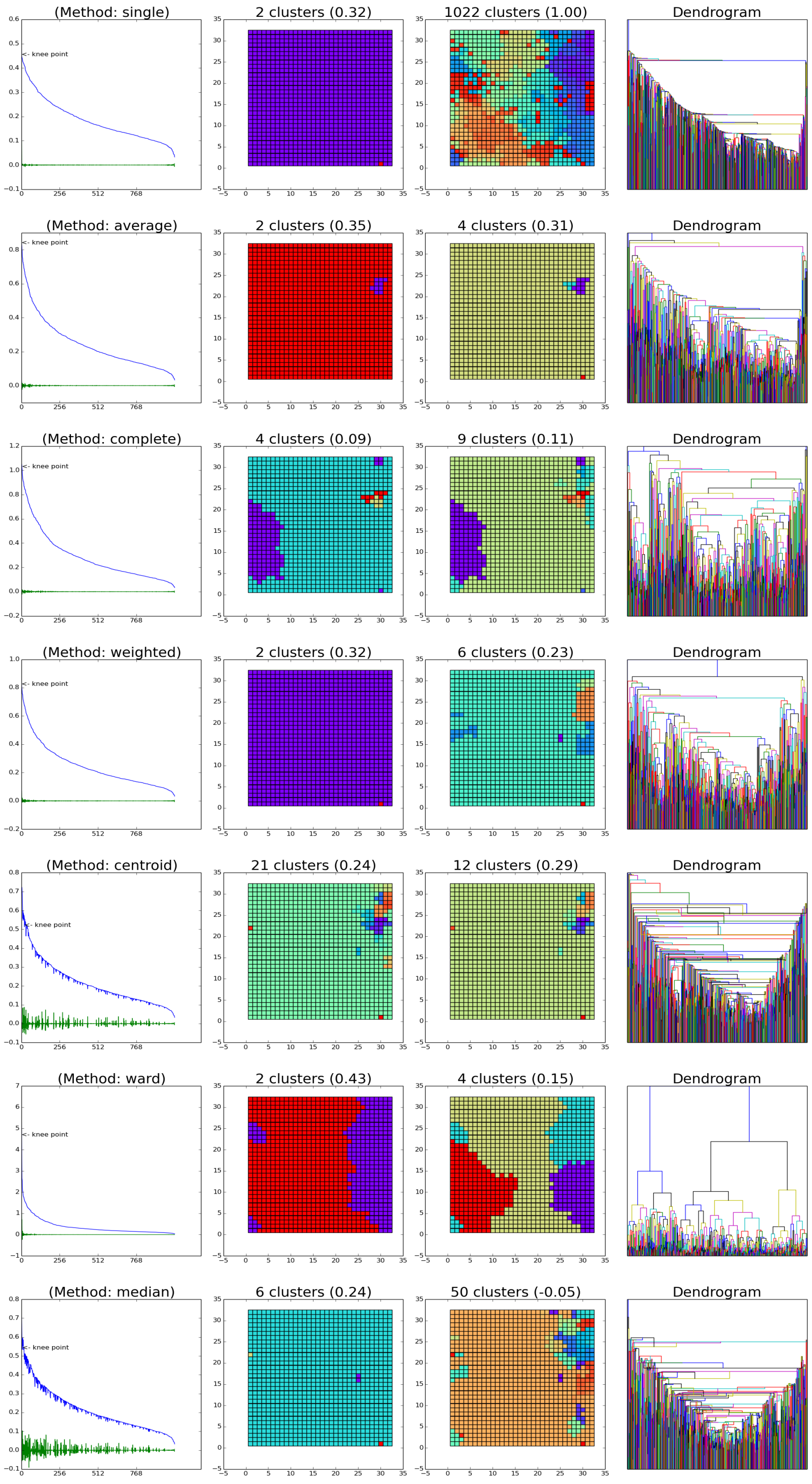
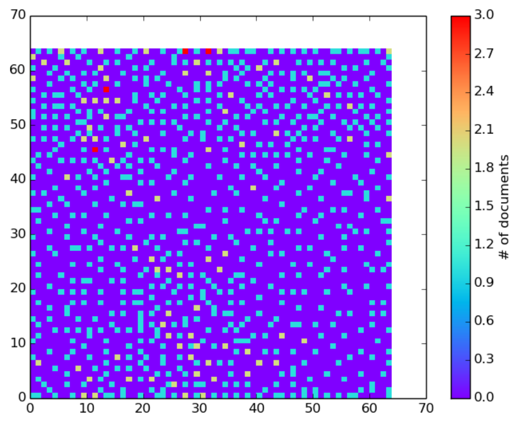
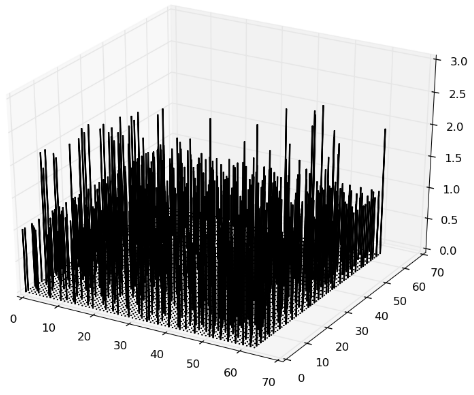


Figure B6: Detailed analysis of hierarchical clustering for 32x32 SOM.



(a) Heatmap



(b) 3D bar chart

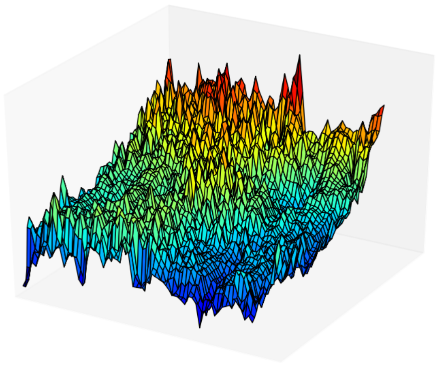

(c) Unified Distance Matrix

Figure B7: Distribution of documents per node (a and b) and the difference between node vectors (c). Dimension: 64x64.

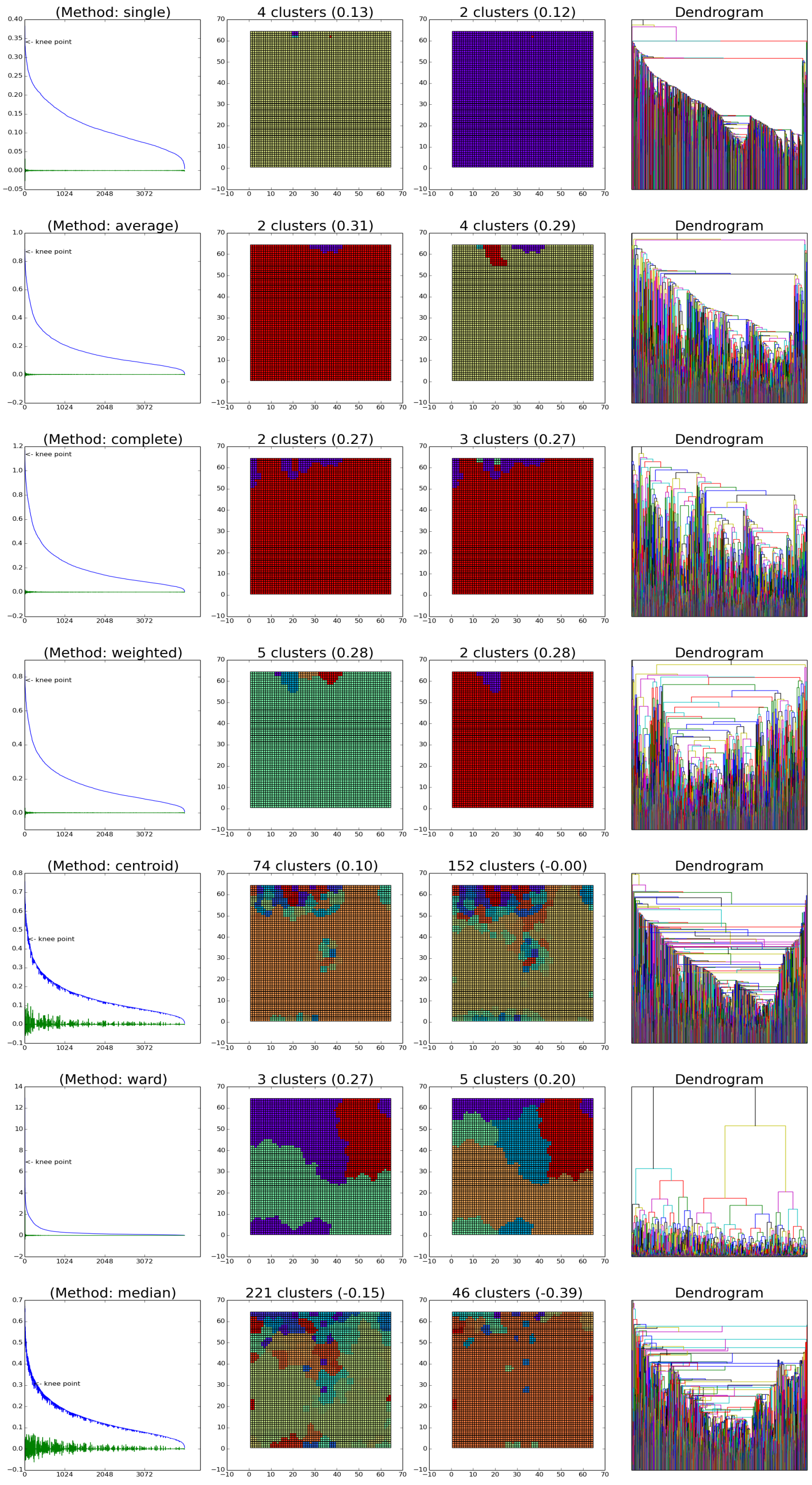


Figure B8: Detailed analysis of hierarchical clustering for 64x64 SOM.

ACRONYMS

ACL Association for Computational Linguistics. 207
ANN Artificial Neural Network. 3, 8, 17, **22**, 23, 24, 43, 44, 67, 68, 96, 205
ASCII American Standard Code for Information Interchange. 66
ASR Automatic Speech Recognition. **17**, 53, 56, 75, 89, 123

BLEU Bilingual Evaluation Understudy. 72, **73**, 76, 91, 93–96, 98, 100, 101, 123, 187, 188, 190–198
BMU Best-Matching Unit. 19, 21, 22, **24**, 25, 26, 37, 38, 140, 142, 143, 148, 152, 156, 165, 168, 177, 198, 204
BNC British National Corpus. 107

CAT Computer-Assisted Translation. 43
CBR Case-Based Reasoning. 47
CDbw Composing Density Between and With Clusters. **160**

DARPA Defense Advanced Research Projects Agency. 72
DGC Dictionary Growth Curve. xxiii, xxiv, **129**, 131, 132

EBMT Example-Based Machine Translation. **47**, 49, 106
EMEA European Medicines Agency. xxiv, 185, 188–191, 194, 195

FAHQT Fully Automated High-Quality Translation. 85
FDA Feature Decay Algorithm. 94
FLOP Floating Point Operation. 20

GFAI Gesellschaft zur Förderung angewandter Informatik. 109
GMM Gaussian Mixture Model. 31
GNU GNU's Not Unix. 16
GPL General Public License. 16, 109
GPU Graphics Processing Unit. 69

HPC High-Performance Computing. 10, 103, 147, 203, 207

IR Information Retrieval. 18, **38**, 39, 56, 90, 94, 95, 138, 184
IRST Istituto per la Ricerca Scientifica e Tecnologica. 126
IRSTLM IRST Language Modeling Toolkit. 13, 15, 116, 125–127, 129, 134, 203

LDA Latent Dirichlet Allocation. **17**, **18**, 95, 96, 183, 213
LDC Linguistic Data Consortium. 72, 93, 97
LSI Latent Semantic Indexing. **17**, 18, 25, 83, 97, 207

MERT Minimum Error Rate Training. 188, 198
MPI Message Passing Interface. 12, 13, 15, 126, **140**, 159, 204
MWE Multiword Expression. xvii, 10, 13, 15, 103, **105**, 106, 109, 110, 112, 114, 119, 120, 203

NLP Natural Language Processing. 7, 22, 25, 53, 56, 61, 67, 76, 79–81, 83, 105, 107, 123, 205
NLTK Natural Language Toolkit. 139, 143
NMT Neural Machine Translation. **44**, 67, 69, 98
NNJM Neural Network Joint Model. 98
NNLM Neural Network Language Model. 64, **67**, 93

OCR Optical Character Recognition. 53, 56, 123
OOF Out-of-Five. 114–116
OOV Out-of-Vocabulary. 129–134, 189, 190, 194, 198

PBS Portable Batch System. 126, 127, 140, 203
PCA Principal Component Analysis. **37**, 148, 164, 173, 180, 204
POS Part-of-Speech. 86, 125, 126, 162, 205, 207
PRESEMT Pattern REcognition-based Statistically Enhanced MT. 56
PSD Phrase-Sense Disambiguation. 99

RBMT Rule-Based Machine Translation. **44**, 45, 47, 56, 71, 79, 106
RNN Recurrent Neural Network. **68**, 69, 93

SCUMBLE Score of ConcUrrence among iMBalanced LabEls. 144, 146
SMT Statistical Machine Translation. vii, 13, 16, **44**, 47, 51, 53–56, 67, 69–71, 79, 81, 87, 88, 90, 91, 93–95, 98, 99, 105, 106, 119, 120, 123, 183–185, 187, 188, 203, 213
SOM Self-Organizing Map. vii, xvii, xxii, xxiii, 1, 3, **7**, 8, 9, 12, 13, 15, 17–26, 31, 32, 35–38, 82, 103, 137–141, 144, 147–150, 152–154, 156, 159, 160, 162–166, 168–180, 183, 185, 198, 203–206, 209, 211–213, 217, 220, 222, 224, 226, 227, 230, 232, 234, 236
SRILM SRI Language Modeling Toolkit. 126, 128
SVD Singular Value Decomposition. **18**, 37, 90, 148
SVM Support Vector Machine. 184

TER Translation Edit Rate. **75**, 98, 187, 190, 192, 195–198

UDM Unified Distance Matrix. **152**, 175, 217, 227
UTF Universal Transformation Format. 125

VSM Vector Space Model. 20, 116, 117, 120

WaCKy The Web-As-Corpus Kool Yinitiative. 160
WER Word Error Rate. 75

WMT Workshop on Statistical Machine Translation. 69, 76, 184, 185
WSD Word Sense Disambiguation. 83, 99, 105–107, **108**, 110, 114, 117, 119, 120, 203
WSK Word Sequence Kernel. 94
WTD Word Translation Disambiguation. 13, 25, **106**, 117

PUBLICATIONS: FROM THESIS

Lars Bungum. Extracting and selecting relevant corpora for domain adaptation in machine translation. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 336–343, Goa, India, December 2014. NLP Association of India/ACL.

Lars Bungum and Björn Gambäck. A survey of domain adaptation in machine translation: Towards a refinement of domain space. In *Proceedings of the India-Norway Workshop on Web Concepts and Technologies*, Trondheim, Norway, 2011. Tapir Academic Press.

Lars Bungum and Björn Gambäck. Efficient n-gram language modeling for billion word web-corpora. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 6–12, Istanbul, Turkey, May 2012. ELRA. Workshop on Challenges in the Management of Large Corpora.

Lars Bungum and Björn Gambäck. Multi-domain adapted machine translation using unsupervised text clustering. In Henning Christiansen, Isidora Stojanovic, and A. George Papadopoulos, editors, *Modeling and Using Context: Proceedings of the Ninth International and Interdisciplinary Conference*, pages 201–213, Larnaca, Cyprus, November 2015a. Springer International Publishing.

Lars Bungum and Björn Gambäck. Self-organizing maps for classification of a multi-labeled corpus. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 39–48, Trivandrum, India, December 2015b. NLP Association of India/ACL.

Lars Bungum, Björn Gambäck, André Lynum, and Erwin Marsi. Improving word translation disambiguation by capturing multiword expressions with dictionaries. In *Ninth Workshop on Multiword Expressions*, Atlanta, US, June 2013. NAACL.

Björn Gambäck and Lars Bungum. Linguistic domains and adaptable companionable agents. In *Proceedings of the First International Workshop on Domain Adaptation for Dialog Agents*, Riva del Garda, Italy, September 2016. ECML.

André Lynum, Erwin Marsi, Lars Bungum, and Björn Gambäck. Disambiguating word translations with target language models. In Petr Sojka, editor, *Proceedings of the 15th International Conference on Text,* *Speech and Dialogue*, pages 378–385, Brno, Czech Republic, September 2012.

Erwin Marsi, André Lynum, Lars Bungum, and Björn Gambäck. Word translation disambiguation without parallel texts. In *Proceedings of the International Workshop on Using Linguistic Information for Hybrid Machine Translation*, pages 66–74, Barcelona, Spain, November 2011.

PUBLICATIONS: OTHER

Lars Bungum and Björn Gambäck. Evolutionary algorithms in natural language processing. In Şule Yildirim and Anders Kofod-Petersen, editors, *Proceedings of the Second Norwegian Artificial Intelligence Symposium*, pages 7–18, Gjøvik, Norway, 2010.

Brage Ekroll Jahren, Valerij Fredriksen, Björn Gambäck, and Lars Bungum. NTNU-SentEval: Combining general classifiers for fast Twitter sentiment analysis. In *Proceedings of Human Language Technologies: The 2016 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 103–108, San Diego, USA, June 2016. ACL. International Workshop on Semantic Evaluation.

Torvald Lekvam, Björn Gambäck, and Lars Bungum. Agent-based modeling of language evolution. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 49–54, Gothenburg, Sweden, April 2014. ACL. Fifth Workshop on Cognitive Aspects of Computational Language Learning.

Erwin Marsi, Hans Moen, Lars Bungum, Gleb Sizov, Björn Gambäck, and André Lynum. NTNU-CORE: Combining strong features for semantic similarity. In *Proceedings of Human Language Technologies: The 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 66–73, Atlanta, USA, June 2013. ACL. Second Joint Conference on Lexical and Computational Semantics.

Johan Reitan, Jørgen Faret, Björn Gambäck, and Lars Bungum. Negation scope detection for Twitter sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 99–108, Lisbon, Portugal, September 2015. ACL. Sixth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis.

Øyvind Selmer, Michael Brevik, Björn Gambäck, and Lars Bungum. NTNU: Domain semi-independent short message sentiment classification. In *Proceedings of Human Language Technologies: The 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 430–437, Atlanta, USA, June 2013. ACL. Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval '13.

REFERENCES

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, March 1994.

Patricia Alexander. Domain knowledge: Evolving themes and emerging concerns. *Educational Psychologist*, 27(1):33–51, 1992.

Paul H. Algoet and Thomas M. Cover. A sandwich proof of the Shannon-McMillan-Breiman theorem. *The Annals of Probability*, 16 (2):899–909, 1988.

Robert B. Allen. Several studies on natural language and backpropagation. In *Proceedings of the IEEE First International Conference on Neural Networks*, pages 335–341, San Diego, USA, 1987. IEEE.

Tarek Aroui, Yassine Koubaa, and Ahmed Toumi. Clustering of the self-organizing map based approach in induction machine rotor faults diagnostics. *Leonardo Journal of Sciences*, 8(15):1–14, April 2008.

Mats Attnäs, Pierre Senellart, and Jean Senellart. Integration of SYSTRAN MT systems in an open workflow. In *Proceedings of the 10th Machine Translation Summit*, pages 211–218, Phuket, Thailand, September 2005. AAMT.

Michael Auli, Adam Lopez, Hieu Hoang, and Philipp Koehn. A systematic analysis of translation model search spaces. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 224–232, Athens, Greece, March 2009. ACL.

Amittai Axelrod. Cynical selection of language model training data. *Computing Research Repository (CoRR)*, abs/1709.02279, 2017.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, United Kingdom, July 2011. ACL.

Jonathan Baarsch and M. Emre Celebi. Investigation of internal validity measures for k-means clustering. *Proceedings of the International Multiconference on Engineers and Computer Scientists*, 1(3):471–476, March 2012.

Ming-Hong Bai, Jia-Ming You, Keh-Jiann Chen, and Jason S. Chang. Acquiring translation equivalences of multiword expressions by normalized correlation frequencies. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 478–486, Singapore, August 2009. ACL.

Timothy Baldwin and Su Nam Kim. *Multiword Expressions*, chapter 4, pages 267–292. Chapman & Hall/CRC, 2nd edition, 2010. ISBN 1420085921, 9781420085921.

R.E. Banchs, L.F. D'Haro, and Haizhou Li. Adequacy–fluency metrics: Evaluating MT in the continuous space model framework. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 23 (3):472–482, March 2015. ISSN 2329-9290.

Pratyush Banerjee, Jinhua Du, Baoli Li, Sudip Naskar, Andy Way, and Josef van Genabith. Combining multi-domain statistical machine translation models using automatic classifiers. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*, pages 141–150, Denver, USA, 2010. AMTA.

Pratyush Banerjee, Raphael Rubino, Johann Roturier, and Josef van Genabith. Quality estimation-guided supplementary data selection for domain adaptation of statistical machine translation. *Machine Translation*, 29(2):77–100, 2015.

Marco Baroni and Adam Kilgarriff. Large linguistically-processed web corpora for multiple languages. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 87–90, Trento, Italy, April 2006. ACL.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009.

Jerome R. Bellegarda. Statistical language model adaptation: review and perspectives. *Speech Communication*, 42(1):93–108, 2004.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. Also published as a book. Now Publishers, 2009.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, March 2003.

Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus phrase-based machine translation quality: a case study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 257–267, Austin, USA, November 2016. ACL.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 995–1005, Stroudsburg, PA, USA, 2012. ACL.

Nicola Bertoldi and Marcello Federico. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 182–189, Athens, Greece, March 2009. ACL.

Nicola Bertoldi, Barry Haddow, and Jean-Baptiste Fouet. Improved minimum error rate training in Moses. *Prague Bulletin of Mathematical Linguistics*, 91:7–16, January 2009.

Ergun Biçici. Domain adaptation for machine translation with instance selection. *The Prague Bulletin of Mathematical Linguistics*, 103: 5–20, April 2015.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

John Blitzer. *Domain Adaptation of Natural Language Processing Systems*. PhD thesis, University of Pennsylvania, Philadelphia, USA, 2008.

Michael Bloodgood and Chris Callison-Burch. Bucking the trend: Large-scale cost-focused active learning for statistical machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 854–864, Uppsala, Sweden, July 2010. ACL.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016. ACL.

R. E. Bonner. On some clustering techniques. *IBM Journal of Research and Development*, 8(1):22–32, January 1964.

Mihaela Elena Braeban. *Clustering: Evolutionary Approaches*. PhD thesis, Alexandru Ioan Cuza University, Timisoara, Romania, 2011.

Thorsten Brants and Alex Franz. Google Web1T 5-gram corpus, version 1. In Linguistic Data Consortium: Catalog Number LDC2006T13, 2006.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 858–867, Prague, Czech Republic, June 2007. ACL.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. Massive exploration of neural machine translation architectures. *Computing Research Repository (CoRR)*, abs/1703.03906, 2017.

Peter F. Brown, John Cocke, Stephen A. Della Piettra, Vincent J. Della Piettra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, June 1990.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, December 1992.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, June 1993.

Lars Bungum. Extracting and selecting relevant corpora for domain adaptation in machine translation. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 336–343, Goa, India, December 2014. NLP Association of India/ACL.

Lars Bungum and Björn Gambäck. Multi-domain adapted machine translation using unsupervised text clustering. In Henning Christiansen, Isidora Stojanovic, and A. George Papadopoulos, editors, *Modeling and Using Context: Proceedings of the Ninth International and Interdisciplinary Conference*, pages 201–213, Larnaca, Cyprus, November 2015. Springer International Publishing.

Lou Burnard, editor. *Reference Guide for the British National Corpus (XML Edition)*. BNC Consortium, Oxford, England, February 2007.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of BLEU in machine translation research. In

Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, pages 249–256, Trento, Italy, April 2006. ACL.

Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. Word sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082, March 2003.

Michael Carl, Silke Gutermuth, and Silvia Hansen-Schirra. Post-editing machine translation. In Aline Ferreira and John W. Schwieter, editors, *Psycholinguistic and Cognitive Inquiries into Translation and Interpreting*, chapter 7. John Benjamins, Amsterdam, The Netherlands, 2015.

Marine Carpuat. Domain adaptation in machine translation. Invited Talk at the Ninth Machine Translation Marathon, Trento, Italy, September 2014. URL `http://www.statmt.org/mtm14/uploads/Main/DomainAdaptationMT_MTM2014.pdf`. Last visited: May 22, 2018.

Marine Carpuat and Dekai Wu. Evaluating the word sense disambiguation performance of statistical machine translation. In *Second International Joint Conference on Natural Language Processing (IJCNLP-2005*, pages 122–127, Jeju, Republic of Korea, 2005.

Marine Carpuat and Dekai Wu. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 61–72, Prague, Czech Republic, June 2007a. ACL.

Marine Carpuat and Dekai Wu. How phrase sense disambiguation outperforms word sense disambiguation for statistical machine translation. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation*, pages 43–52, Skövde, Sweden, September 2007b.

Marine Carpuat, Hal Daumé III, Alexander Fraser, Chris Quirk, Fabienne Braune, Ann Clifton, Ann Irvine, Jagadeesh Jagarlamudi, John Morgan, Majid Razmara, Aleš Tamchyna, Katharine Henry, and Rachel Rudinger. Domain adaptation in statistical machine translation: Final report. In *CLSP Workshop Report*. Johns Hopkins University, 2012. URL `http://www.umiacs.umd.edu/~hal/damt/DAMT_final_report.pdf`. Last visited: March 27, 2019.

David Carter. Improving language models by clustering training sentences. In *Proceedings of the Fourth Conference on Applied Natural Lan-*

guage Processing, ANLC '94, pages 59–64, Stroudsburg, USA, 1994. ACL.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997.

Helena de Medeiros Caseli, Carlos Ramisch, Maria das Graças Volpe Nunes, and Aline Villavicencio. Alignment-based extraction of multiword expressions. *Language Resources and Evaluation*, 44(1-2):59–77, April 2010. Special Issue on Multiword expression: hard going or plain sailing.

Mauro Cettolo, Mara Chinea-Rios, and Roldano Cattoni. Unsupervised clustering of commercial domains for adaptive machine translation. *arXiv e-print arXiv: 1612.04683*, 2016.

Francisco Charte and David Charte. Working with multilabel datasets in R: The mldr package. *The R Journal*, 7(2):149–162, December 2015.

Francisco Charte, Antonio Rivera, María José del Jesus, and Francisco Herrera. Concurrence among imbalanced labels and its influence on multilabel resampling algorithms. In Marios Polycarpou, André C. P. L. F. de Carvalho, Jeng-Shyang Pan, Michał Woźniak, Héctor Quintian, and Emilio Corchado, editors, *Hybrid Artificial Intelligence Systems: 9th International Conference, HAIS 2014*, pages 110–121. Springer International Publishing, June 2014.

Ciprian Chelba and Frederick Jelinek. Structured language modeling. *Computer Speech and Language*, 14(4):283 – 332, 2000.

Boxing Chen, George F. Foster, and Roland Kuhn. Adaptation of reordering models for statistical machine translation. In *Proceedings of the 2013 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 938–946, Atlanta, USA, June 2013. ACL.

Junli Chen, Xuezhong Zhou, and Zhaohui Wu. A multi-label Chinese text categorization system based on boosting algorithm. In *Proceedings of the Fourth International Conference on Computer and Information Technology*, pages 1153–1158, Washington, USA, 2004. IEEE Computer Society.

Stanley F. Chen, Stanley F. Chen, Joshua Goodman, and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical report, Harvard University, Cambridge, USA, 1998.

Colin Cherry and George Foster. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 427–436, Montreal, Canada, June 2012. ACL.

David Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228, June 2007.

Noam Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124, 1956.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. An empirical comparison of simple domain adaptation methods for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 385–391, Vancouver, Canada, August 2017. ACL.

Jorge Civera and Alfons Juan. Domain adaptation in statistical machine translation with mixture modelling. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 177–180, Prague, Czech Republic, 2007. ACL.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, USA, June 2011. ACL.

Stephen Clark. *Vector Space Models of Lexical Meaning*, pages 493–522. John Wiley & Sons, Ltd, Hoboken, USA, 2015.

Trevor Cohn, Phil Blunsom, and Sharon Goldwater. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11: 3053–3096, November 2010.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the International Conference on Machine Learning*, pages 160–167, Helsinki, Finland, 2008.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, November 2011.

Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, USA, 1971. ACM.

Marta R. Costa-Jussà and Mireia Farrús. Statistical machine translation enhancements through linguistic levels: A survey. *ACM Computing Surveys*, 46(3):42:1–42:28, January 2014.

Marta R. Costa-Jussà and José A.R. Fonollosa. Latest trends in hybrid machine translation and its applications. *Computer Speech & Language*, 32(1):3–10, July 2015.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, USA, 1991.

Josep Maria Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, Satoshi Enoue, Chiyo Geiss, Joshua Johanson, Ardas Khalsa, Raoum Khiari, Byeongil Ko, Catherine Kobus, Jean Lorieux, Leidiana Martins, Dang-Chuan Nguyen, Alexandra Priori, Thomas Riccardi, Natalia Segal, Christophe Servan, Cyril Tiquet, Bo Wang, Jin Yang, Dakun Zhang, Jing Zhou, and Peter Zoldan. Systran's pure neural machine translation systems. *CoRR*, abs/1610.05540, 2016.

Gavin E. Crooks. On measures of entropy and information. Online, 2016. URL `http://threeplusone.com/info`. Last visited: March 27, 2019.

Lei Cui, Xilun Chen, Dongdong Zhang, Shujie Liu, Mu Li, and Ming Zhou. Multi-domain adaptation for SMT using multi-task learning. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1055–1065, Seattle, USA, October 2013. ACL.

Hoang Cuong and Khalil Sima'an. A survey of domain adaptation for statistical machine translation. *Machine Translation*, 31(4):187–224, December 2017. ISSN 0922-6567.

Lisandro D. Dalcin, Rodrigo R. Paz, Pablo A. Kler, and Alejandro Cosimo. Parallel distributed computing using Python. *Advances in Water Resources*, 34(9):1124 – 1139, 2011. New Computational Methods and Software Tools.

Arsène Darmesteter. La vie des mots étudiée dans leur significations. Delagrave, Paris, 1887.

Hal Daumé and Jagadeesh Jagarlamudi. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 407–412, Portland, USA, June 2011. ACL.

David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1 (2):224–227, February 1979.

Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. *Commununications of the ACM*, 51(1): 107–113, January 2008.

Franca Debole and Fabrizio Sebastiani. An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56:971–974, 2004.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41 (6):391–407, 1990.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1370–1380, Baltimore, USA, June 2014. ACL.

Lee Raymond Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, July 1945.

George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145, San Francisco, USA, March 2002. Morgan Kaufmann Publishers Inc.

Rebecca Dridan and Stephan Oepen. Tokenization: Returning to a long solved problem a survey, contrastive experiment, recommendations, and toolkit. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 378–382, Jeju, Republic of Korea, July 2012. ACL.

Athanasios S. Drigas and John Vrettaros. Using the self-organizing map (som) algorithm, as a prototype e-content retrieval tool. In *Proceedings of the International Conference on Computational Science and Its Applications – ICCSA*, pages 14–23, Perugia, Italy, June 2008.

David Dubin. The most influential paper Gerard Salton never wrote. *Library Trends*, 52(4):748–764, 2004.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. Adaptation data selection using neural language models: Experiments in machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 678–683, Sofia, Bulgaria, August 2013. ACL.

Susan Dumais, John Platt, Mehran Sahami, and David Heckerman. Inductive learning algorithms and representations for text categorization. In *Proceedings of the Seventh International Conference on Information and Knowledge Management*, pages 148–155, New York, USA, 1998. ACM Press.

C. Dunn, Joseph. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, September 1973.

Keiran J. Dunne. Computer-assisted translation. In Carol A. Chapelle, editor, *The Encyclopedia of Applied Linguistics*. Wiley, Hoboken, USA, 2012.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 7–12, Uppsala, Sweden, July 2010. ACL. System Demonstration.

Christopher Dyer, Aaron Cordova, Alex Mont, and Jimmy Lin. Fast, easy, and cheap: Construction of statistical machine translation models with MapReduce. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 199–207, Columbus, USA, 2008. ACL.

Matthias Eck, Stephan Vogel, and Alex Waibel. Language model adaptation for statistical machine translation based on information retrieval. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, Lisbon, Portugal, May 2004. ELRA.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, New York, NY, 1993.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2): 179–211, 1990.

Euromatrix. Survey of machine translation evaluation. Deliverable 1.3,. Online, December 2007. URL `http://www.euromatrix.net/deliverables/Euromatrix_D1.3_Revised.pdf`. Last visited: March 27, 2019.

Brian S. Everitt, Sabine Landau, Morven Leese, and Daniel Stahl. *Cluster Analysis*. John Wiley & Sons, Ltd, Chichester, United Kingdom, fifth edition, 2011.

Samuel Eyassu and Björn Gambäck. Classifying Amharic news text using self-organizing maps. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 71–78, Ann Arbor, USA, June 2005. ACL. Workshop on Computational Applications to Semitic Languages.

Gertrud Faaß and Kerstin Eckart. SdeWac – a corpus of parsable sentences from the web. In Iryna Gurevych, Chris Biemann, and Torsten Zesch, editors, *Language Processing and Knowledge in the Web*, volume 8105 of *Lecture Notes in Computer Science*, pages 61–68. Springer Berlin Heidelberg, 2013.

M. Amin Farajian, Marco Turchi, Matteo Negri, Nicola Bertoldi, and Marcello Federico. Neural vs. phrase-based machine translation in a multi-domain scenario. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 280–284, Valencia, Spain, April 2017. ACL.

Marcello Federico and Mauro Cettolo. Efficient handling of n-gram language models for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 88–95, Prague, Czech Republic, 2007. ACL.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. IRSTLM: an open source toolkit for handling large scale language models. In *Proceedings of the Ninth Annual INTERSPEECH Conference*, pages 1618–1621, Brisbane, Australia, September 2008. ISCA.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. *IRST Language Modeling Toolkit, Version 5.50.02: User Manual*. FBK-irst, Trento, Italy, November 2010.

Dario Floreano and Claudio Mattiussi. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. The MIT Press, Cambridge, USA, 2008.

Mikel L. Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, July 2011.

Jean-Claude Fort, Patrick Letrémy, and Marie Cottrell. Advantages and drawbacks of the Batch Kohonen algorithm. In Michel Verleysen, editor, *Proceedings of the 10th European Symposium on Artificial Neural Networks*, pages 223–230, Bruges, Belgium, 2002.

George Foster and Roland Kuhn. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 128–135, Prague, Czech Republic, 2007. ACL.

Björn Gambäck and Lars Bungum. Linguistic domains and adaptable companionable agents. In *Proceedings of the First International Workshop on Domain Adaptation for Dialog Agents*, Riva del Garda, Italy, September 2016. ECML.

Jianfeng Gao, Joshua Goodman, Mingjing Li, and Kai-Fu Lee. Toward a unified approach to statistical language modeling for chinese. *ACM Transactions on Asian Language Information Processing*, 1 (1):3–33, March 2002.

Jianfeng Gao, Hisami Suzuki, and Wei Yuan. An empirical study on language model adaptation. *ACM Transactions on Asian Language Information Processing*, 5(3):209–227, September 2005.

Paul L. Garvin. Some linguistic problems in machine translation. In *For Roman Jakobson; Essays on the Occasion of His Sixtieth Birthday*, pages 180–186. Mouton, The Hague, The Netherlands, 1956.

Dirk Geeraerts. *Theories of Lexical Semantics (Oxford Linguistics)*. Oxford University Press, 2010.

Ulrich Germann. Dynamic phrase tables for machine translation in an interactive post-editing scenario. In *Proceedings of the Third Conference of the Association for Machine Translation in the Americas*, pages 20–31, Vancouver, Canada, 2014. AMTA.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting and the Ninth Confer-*

ence of the European Chapter of the Association for Computational Linguistics, pages 228–235, Toulouse, France, July 2001. ACL.

Alfio Massimiliano Gliozzo and Carlo Strapparava. *Semantic Domains in Computational Linguistics.* Springer, New York, USA, 2009.

Yoav Goldberg. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726, 2015.

Yoav Goldberg and Omer Levy. word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, USA, 2016. `http://www.deeplearningbook.org`.

Joshua T. Goodman. A bit of progress in language modeling. *Computer Speech & Language*, 15(4):403–434, October 2001.

Yvette Graham, Timothy Baldwin, Aaron Harwood, Alistair Moffat, and Justin Zobel. Measurement of progress in machine translation. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, pages 70–78, 2012.

Yvette Graham, Nitika Mathur, and Timothy Baldwin. Randomized significance tests in machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 266–274, Baltimore, USA, June 2014. ACL.

João Graça. Crowdsourcing for MT. Online, September 2014. URL `http://statmt.org/mtm14/uploads/Main/Crowdsourcing_MTM2014.pdf`. Keynote at Ninth Machine Translation Marathon, Trento, Italy. Last visited: March 27, 2019.

Gregory Grefenstette. The World Wide Web as a resource for example-based machine translation tasks. In *Translating and the Computer 21: Proceedings of the 21st International Conference on Translating and the Computer*, London, United Kingdom, 1999.

Gregory Grefenstette and Pasi Tapainen. What is a word, what is a sentence? Problems of tokenization. In *Proceedings of the 3rd International Conference on Computational Lexicography (COMPLEX*, Budapest, Hungary, 1994.

Antonio Grieco, Massimo Pacella, and Marzia Blaco. On the application of text clustering in engineering change process. *Procedia {CIRP}*, 62:187 – 192, 2017. 10th {CIRP} Conference on Intelligent

Computation in Manufacturing Engineering - {CIRP} {ICME} '16. [Edited by: Roberto Teti, Manager Editor: Doriana M. D'Addona].

Georges Grinstein, Marjan Trutschl, and Urŝka Cvek. High-dimensional visualizations. In *Proceedings of the Data Mining Conference (KDD)*, August 2001.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. Active learning for statistical phrase-based machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 415–423, Boulder, USA, June 2009. ACL.

Jan Hajič and Eva Hajičová. Some of our best friends are statisticians. In Václav Matousek and Pavel Mautner, editors, *Proceedings of the 10th International Conference on Text, Speech and Dialogue*, pages 2–10, Pilsen, Czech Republic, September 2007.

Maria Halkidi and Michalis Vazirgiannis. A density-based cluster validity approach using multi-representatives. *Pattern Recognition Letters*, 29(6):773–786, April 2008.

Timo D. Hämäläinen. Parallel implementation of self-organizing maps. In Udo Seiffert and Lakhmi C. Jain, editors, *Self-Organizing Neural Networks*, pages 245–278. Springer, New York, USA, 2002.

Jiawei Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, USA, second edition, 2005.

Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, USA, third edition, 2011.

Saša Hasan and Hermann Ney. Clustered language models based on regular expressions for SMT. In *Proceedings of the 10th Annual Conference of the European Association for Machine Translation*, pages 119–125, Budapest, Norway, May 2005. EAMT.

Eva Hasler. *Dynamic Topic Adaptation for Improved Contextual Modelling in Statistical Machine Translation*. PhD thesis, University of Edinburgh, Edinburgh, United Kingdom, 2014.

Eva Hasler, Barry Haddow, and Philipp Koehn. Combining domain and topic adaptation for SMT. In *Proceedings of the Third Conference of the Association for Machine Translation in the Americas*, pages 445–456, Vancouver, Canada, 2014. AMTA.

Trevor J. Hastie, Robert John Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, New York, USA, 2009.

Kenneth Heafield. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, United Kingdom, July 2011. ACL.

Kenneth Heafield, Michael Kayser, and Christopher D. Manning. Faster Phrase-Based decoding by refining feature state. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 130–135, Baltimore, USA, June 2014. ACL.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the 14th Annual Conference of the European Association for Machine Translation*, pages 133–142, Budapest, Hungary, May 2005. EAMT.

Hieu Hoang and Philipp Koehn. Design of the Moses decoder for statistical machine translation. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP, pages 58–65. ACL, 2008.

Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Richard Zens, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Brooke Cowan, Wade Shen, Christine Moran, and Ondřej Bojar. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic, June 2007. ACL.

Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57, New York, USA, 1999. ACM.

Timo Honkela. *Self-Organizing Maps in Natural Language Processing*. PhD thesis, Helsinki University of Technology, Espoo, Finland, 1997.

Matthias Huck, Alexandra Birch, and Barry Haddow. Mixed-domain vs. multi-domain statistical machine translation. In *Proceedings of the 15th Machine Translation Summit*, pages 240–255, Miami, USA, September 2015. AMTA.

John D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

W. John Hutchins. Milestones no.6: Bar-Hillel and the non-feasibility of FAHQT. *International Journal of Language and Documentation*, pages 20–21, September 1999.

W. John Hutchins. The first public demonstration of machine translation: the Georgetown-IBM system, january 7th, 1954. Online, 2005. URL `http://www.hutchinsweb.me.uk/GU-IBM-2005.pdf`.

W. John Hutchins. Machine translation: A concise history. *Journal of Translation Studies*, 13(1-2):20–70, 2010.

Heikki Hyötyniemi. Text document classification with self-organizing maps. In J. Alander, T. Honkela, and M. Jakobsson, editors, *Proceedings of Seventh Finnish Artificial Intelligence Conference*, pages 64–72, Vaasa, Finland, 1996.

Ann Irvine, John Morgan, Marine Carpuat, Hal Daumé III, and Dragos Stefan Munteanu. Measuring machine translation errors in new domains. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 429–440, Seattle, USA, October 2013. ACL.

Dino Isa, Vish P. Kallimani, and Lam Hong Lee. Using the self organizing map for clustering of text documents. *Expert Systems with Applications*, 36(5):9584 – 9591, July 2009.

Rukmini Iyer, Mari Ostendorf, and Herb Gish. Using out-of-domain data to improve in-domain language models. In *Signal Processing Letters*, number 8, pages 221–223. IEEE, August 1997.

Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, USA, 1988.

David Janiszek, Renato de Mori, and Frederic Béchet. Data augmentation and language model adaptation using singular value decomposition. *Pattern Recognition Letters*, 25(1):15–19, January 2004.

Frederick Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, USA, 1997.

Frederick Jelinek. Some of my best friends are linguists. *Language Resources and Evaluation*, 39(1):25–34, February 2005.

Jing Jiang. *Domain Adaptation in Natural Language Processing*. PhD thesis, University of Illinois at Urbana-Champaign, Champaign, USA, 2008.

Ian T. Jolliffe. *Principal Component Analysis*. Springer Verlag, New York, USA, 1986.

Shafiq Joty, Hassan Sajjad, Nadir Durrani, Kamla Al-Mannai, Ahmed Abdelali, and Stephan Vogel. How to avoid unwanted pregnancies: Domain adaptation using neural network models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1259–1270, Lisbon, Portugal, September 2015. ACL.

Daniel Jurafsky and James H. Martin. Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition, 2017. Draft of January 2017.

Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, USA, October 2013. ACL.

Jussi Karlgren. Sublanguages and registers - a note on terminology. *Interacting with Computers*, 5:348–350, 1993.

Samuel Kaski. *Data Exploration Using Self-Organizing Maps*. PhD thesis, Helsinki University of Technology, Espoo, Finland, 1997.

Samuel Kaski, Jari Kangas, and Teuvo Kohonen. Bibliography of Self-Organizing Map (SOM) papers: 1981–1997. *Neural Computing Surveys*, 1:102–350, 1998.

Slava Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, March 1987. ISSN 0096-3518.

Martin Kay. The proper place of men and machines in language translation. *Machine Translation*, 12:3–23, 1997. First appeared as a Xerox PARC Working paper in 1980.

Adam Kilgarriff and Gregory Grefenstette. Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29(3):333–348, 2003.

Adam Kilgarriff, Siva Reddy, Jan Pomikálek, and Avinesh PVS. A corpus factory for many languages. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 904–910, Valetta, Malta, May 2010. ELRA.

Adam Kilgarriff, Avinesh PVS, and Jan Pomikálek. Comparable corpora BootCaT. In Iztok Kosem and Karmen Kosem, editors, *Proceedings of eLex 2011, Electronic Lexicography in the 21st Century: New Applications for New Users*, pages 122–128, Bled, Slovenia, November 2011. Trojina, Institute for Applied Slovene Studies.

Richard Kittredge. Semantic processing of texts in restricted sublanguages. In Nick Cercone, editor, *Computational Linguistics*, International series in modern applied mathematics and computer science, pages 45–59, Amsterdam, The Netherlands, 1983. Elsevier Science Inc.

Richard Kittredge and John Lehrberger, editors. *Sublanguage: studies of language in restricted semantic domains*, Berlin, Germany, 1982. De Gruyter.

Richard I. Kittredge. Sublanguages and controlled languages. In Ruslan Mitkov, editor, *The Oxford Handbook of Computational Linguistics*, chapter 23. Oxford University Press, Oxford, United Kingdom, 1 edition, 2003.

Reinhard Kneser and Hermann Ney. Improved backing-off for n-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, pages 181–184, Michigan, USA, May 1995.

Kevin Knight. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615, 1999.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain, July 2004. ACL, ACL.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86, Phuket, Thailand, September 2005. AAMT.

Philipp Koehn. *Statistical Machine Translation*. Cambridge University Press, New York, USA, first edition, 2010.

Philipp Koehn and Christof Monz. Manual and automatic evaluation of machine translation between european languages. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 102–121, New York, USA, June 2006. ACL.

Philipp Koehn, Franz J. Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133, Edmonton, Canada, May 2003. ACL.

Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69, 1982.

Teuvo Kohonen and Timo Honkela. Kohonen network. *Scholarpedia*, 2(1):1568, 2007. revision #122029.

Teuvo Kohonen, Samuel Kaski, Krista Lagus, and Timo Honkela. Very large two-level SOM for the browsing of newsgroups. In C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, editors, *Proceedings of ICANN, International Conference on Artificial Neural Networks*, pages 269–274. Springer, New York, USA, July 1996.

Teuvo Kohonen, Samuel Kaski, Krista Lagus, Jarkko Salojärvi, Vesa Paatero, and Antti Saarela. Self-organization of a massive document collection. *IEEE Transactions on Neural Networks, Special Issue on Neural Networks for Data Mining and Knowledge Discovery*, 11(3): 574–585, May 2000.

Mikko T. Kolehmainen. *Data exploration with self-organizing maps in environmental informatics and bioinformatics*. PhD thesis, Helsinki University of Technology, Espoo, Finland, 2004.

Nikolaus Kriegeskorte. Deep neural networks: A new framework for modeling biological vision and brain information processing. *Annual Review of Vision Science*, 1(1):417–446, 2015.

A.-L. Lagarda, V. Alabau, F. Casacuberta, R. Silva, and E. Díaz-de Liaño. Statistical post-editing of a rule-based machine translation system. In *Proceedings of the 2009 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 217–220, Boulder, USA, June 2009. ACL.

Krista Lagus, Samuel Kaski, and Teuvo Kohonen. Mining massive document collections by the WEBSOM method. *Information Sciences*, 163(1-3):135–156, 2004.

Patrik Lambert, Simon Petitrenaud, Yanjun Ma, and Andy Way. What types of word alignment improve statistical machine translation? *Machine Translation*, 26(4):289–323, December 2012.

Andrea Lancichinetti and Santo Fortunato. Community detection algorithms: A comparative analysis. *Physical Review E*, 80:056117, Nov 2009.

John Langford, Lihong Li, and Alex Strehl. Vowpal Wabbit open source project. Online, 2007. URL `https://github.com/JohnLangford/vowpal_wabbit/wiki`. Last visited: March 27, 2019.

Alon Lavie and Michael J. Denkowski. The meteor metric for automatic evaluation of machine translation. *Machine Translation*, 23 (2-3):105–115, September 2009.

Richard D. Lawrence, George S. Almasi, and Holly E. Rushmeier. A scalable parallel algorithm for self-organizing maps with applications to sparse data mining problems. *Data Mining and Knowledge Discovery*, 3(2):171–195, 1999.

David Y.W. Lee. Genres, registers, text types, domains and styles: clarifying the concepts and navigating a path through the BNC jungle. *Language Learning and Technology*, 5(3):37–72, September 2001.

Els Lefever and Véronique Hoste. SemEval-2010 Task 3: Cross-lingual word sense disambiguation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 15–20, Uppsala, Sweden, July 2010a. ACL. Fifth International Workshop on Semantic Evaluation.

Els Lefever and Véronique Hoste. Construction of a benchmark data set for cross-lingual word sense disambiguation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 1584–1590, Valetta, Malta, May 2010b. ELRA.

Els Lefever and Véronique Hoste. SemEval-2013 task 10: Cross-lingual word sense disambiguation. In *Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 158–166, Atlanta, USA, June 2013. ACL.

Csaba Legány, Sándor Juhász, and Attila Babos. Cluster validity measurement techniques. In *Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, pages 388–393, Stevens Point, USA, 2006. World Scientific and Engineering Academy and Society (WSEAS).

John Lehrberger. Automatic translation and the concept of sublanguage. In Kittredge and Lehrberger [1982], chapter 7.

Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–10, 1966.

David D. Lewis. Reuters-21578 text categorization test collection. Online, 1997. URL `http://www.daviddlewis.com/resources/testcollections/reuters21578/`. Last visited: March 27, 2019.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, December 2004.

Sung-Chien Lin, Chi-Lung Tsai, Lee-Feng Chien, Keh-Jiann Chen, and Lin-Shan Lee. Chinese language model adaptation based on document classification and multiple domain-specific language models. In G. Kokkinakis, N. Fakotakis, and E. Dermatas, editors, *Proceedings of the Fifth European Conference on Speech Communication and Technology*, pages 1463–1466, Rhodes, Greece, September 1997. ESCA.

Yuri Lin, Jean-Baptiste Michel, Erez Aiden Lieberman, Jon Orwant, Will Brockman, and Slav Petrov. Syntactic annotations for the Google Books n-gram corpus. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 169–174, Jeju, Republic of Korea, July 2012. ACL. System Demonstrations.

Thomas Lippincott, Diarmuid Ó Séaghdha, and Anna Korhonen. Exploring subdomain variation in biomedical language. *BMC Bioinformatics*, 12:212, 2011.

Zhen-Ping Lo, Masahiro Fujita, and Behnam Bavarian. Analysis of neighborhood interaction in Kohonen neural networks. In *Proceedings of the Fifth International Parallel Processing Symposium*, pages 246–249. IEEE, May 1991.

Edward Loper and Steven Bird. NLTK: The natural language toolkit. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 63–70, Philadelphia, USA, July 2002. ACL. Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics.

Adam Lopez. Statistical machine translation. *ACM Computing Surveys*, 40(3):8:1–8:49, August 2008.

Adam Lopez and Matt Post. Beyond bitext: Five open problems in machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, USA, October 2013. ACL. Workshop: Twenty Years of Bitext.

Annie Louis and Bonnie Webber. Structured and unstructured cache models for SMT domain adaptation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 155–163, Gothenburg, Sweden, April 2014. ACL.

Yajuan Lu, Jin Huang, and Qun Liu. Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 343–350, Prague, Czech Republic, June 2007. ACL.

Yi Lu, Longyue Wang, Derek F. Wong, Lidia S. Chao, and Yiming Wang. Domain adaptation for medical text translation using web resources. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 233–238, Baltimore, USA, June 2014. ACL.

Heinz-Dirk Luckhardt. Sublanguages in machine translation. In *Proceedings of the Fifth Conference of the European Chapter of the Association for Computational Linguistics*, pages 306–308, Berlin, Germany, April 1991. ACL.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *CoRR*, abs/511.06114, 2016. Published at the 5th International Conference on Learning Representations, San Juan, Puerto Rico.

André Lynum, Erwin Marsi, Lars Bungum, and Björn Gambäck. Disambiguating word translations with target language models. In Petr Sojka, editor, *Proceedings of the 15th International Conference on Text, Speech and Dialogue*, pages 378–385, Brno, Czech Republic, September 2012.

Xiaoyi Ma and Christopher Cieri. Corpus support for machine translation at LDC. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, Genova, Italy, May 2006. ELRA.

Bente Maegaard. *MLIM—Multilingual Information Management: Current levels and Future Abilities Report*, chapter 4. Insituti Editoriali e Poligrafici Internazionali, Pisa, Italy, April 1999.

Bernardo Magnini and Gabriela Cavaglià. Integrating subject field codes into wordnet. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 1413–1418, Athens, Greece, May 2000. ELRA.

Milind Mahajan, Doug Beeferman, and X. D. Huang. Improved topic-dependent language modeling using information retrieval techniques. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 541–544, 1999.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, USA, 1999.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, USA, 2008.

Gary Marcus and Jeremy Freeman, editors. *Consciousness, Big Science and Conceptual Clarity*. Princeton University Press, Princeton, USA, 2014.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn Treebank: Annotating predicate argument structure. In *HLT '94: Proceedings of the Workshop on Speech and Natural Language*, pages 114–119, Stroudsburg, PA, USA, 1994. ACL.

Erwin Marsi, André Lynum, Lars Bungum, and Björn Gambäck. Word translation disambiguation without parallel texts. In *Proceedings of the International Workshop on Using Linguistic Information for Hybrid Machine Translation*, pages 66–74, Barcelona, Spain, November 2011.

Juan Alberto Alonso Martín and Anna Civil Serra. Integration of a machine translation system into the editorial process flow of a daily newspaper. *Procesamiento del Lenguaje Natural*, 53(53):193–196, September 2014.

Héctor Martínez Alonso and Barbara Plank. When is multitask learning effective? Semantic sequence prediction under varying data conditions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 44–53, Valencia, Spain, April 2017. ACL.

Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, 5(4): 115–133, December 1943.

Alan Melby. Some notes on the proper place of men and machines in language translation. *Machine Translation*, 12(1/2):29–34, 1997.

Tomáš Mikolov. *Statistical Language Models Based on Neural Networks*. PhD thesis, Brno University of Technology, Brno, Czech Republic, 2012.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of the 11th Annual INTERSPEECH Conference*, pages 1045–1048, Makuhari, Japan, September 2010. ISCA.

Tomáš Mikolov, Q. V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, September 2013a.

Tomáš Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Proceedings of the 27th Annual Conference on Advances in Neural Information Processing Systems*, pages 3111–3119. Curran Associates, Inc., 2013b.

Tomáš Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 746–751, Atlanta, USA, June 2013c. ACL.

George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November 1995.

Robert C. Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 220–224, Uppsala, Sweden, July 2010. ACL.

Roland G. Morris. Donald O. Hebb: The organization of behavior, Wiley: New York; 1949. *Brain research bulletin*, 50(5-6), 1999.

István Nagy T., Veronika Vincze, and Gábor Berend. Domain-dependent identification of multiword expressions. In Galia Angelova, Kalina Bontcheva, Ruslan Mitkov, and Nicolas Nicolov, editors, *Proceedings of the Eighth International Conference on Recent Advances in Natural Language Processing*, pages 622–627, Hissar, Bulgaria, September 2011.

Preslav Nakov and Marti Hearst. A study of using search engine page hits as a proxy for n-gram frequencies. In *Proceedings of the Fifth International Conference on Recent Advances in Natural Language Processing*, pages 347–353, Borovets, Bulgaria, September 2005.

Cyrine Nasri, Kamel Smaili, and Chiraz Latiri. Training phrase-based SMT without explicit word alignment. In Alexander Gelbukh, editor, *Proceedings of the 14th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 233–241, Samos, Greece, March 2013.

Graham Neubig and Kevin Duh. On the elements of an accurate tree-to-string machine translation system. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 143–149, Baltimore, USA, June 2014. ACL.

Graham Neubig and Taro Watanabe. Optimization for statistical machine translation: A survey. *Computational Linguistics*, 42(1):1–54, March 2016. ISSN 0891-2017.

Noreen. *Introduction to Testing Hypotheses Using Computer Intensive Methods*. John Wiley & Sons, Inc., New York, USA, 1988.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, College Park, USA, 1999.

Lene Offersgaard, Claus Povlsen, Lisbeth Kjeldgaard Almsten, and Bente Maegaard. Domain specific MT in use. In *Proceedings of the 12th Annual Conference of the European Association for Machine Translation*, pages 150–159, Hamburg, Germany, September 2008. EAMT.

Santanu Pal, Tanmoy Chakraborty, and Sivaji Bandyopadhyay. Handling multiword expressions in phrase-based statistical machine translation. In *Proceedings of the 13th Machine Translation Summit*, pages 215–224, Xiamen, China, September 2013. AAMT.

Martha Stone Palmer, Owen Rambow, and Alexis Nasr. Rapid prototyping of domain-specific machine translation systems. In *Proceedings of the Fourth Conference of the Association for Machine Translation in the Americas*, pages 95–102, Langhorne, USA, 1998. AMTA.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, USA, July 2002. ACL.

Bhavik Patel, Anurag Jajoo, Yash Tibrewal, and Amit Joshi. An efficient parallel algorithm for self-organizing maps using MPI-OpenMP based cluster. In *Proceedings of the International Conference on Advanced Computing and Communication Techniques for High Performance Applications*, pages 5–9, Panipat, India, February 2015.

Adam Pauls and Dan Klein. Faster and smaller n-gram language models. In *Proceedings of the 49th Annual Meeting of the Association for*

Computational Linguistics: Human Language Technologies, pages 258–267, Portland, USA, June 2011. ACL.

Pavel Pecina, Antonio Toral, Vassilis Papavassiliou, Prokopis Prokopidis, Aleš Tamchyna, Andy Way, and Josef Genabith. Domain adaptation of statistical machine translation with domain-focused web crawling. *Language Resources and Evaluation*, 49(1):147–193, 2015.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(1):2825–2830, 2011.

Slobodan Petrović. A comparison between the silhouette index and the Davies-Bouldin Index in labelling IDS clusters. In *Proceedings of The 11th Nordic Workshop on Secure IT-systems*, pages 53–64, Linköping, Sweden, October 2006.

Barbara Plank. *Domain Adaptation for Parsing*. PhD thesis, University of Groningen, Groningen, The Netherlands, 2011.

Sushil Kumar Podder. Unsupervised clustering and automatic language model generation for ASR. Master's thesis, University of Waterloo, Canada, Waterloo, Canada, 2004.

Matt Post and Daniel Gildea. Parsers as language models for statistical machine translation. In *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, pages 172–181, Waikiki Beach, USA, 2008. AMTA.

Matt Post, Yuan Cao, and Kumar Gaurav. Joshua 6: A phrase-based and hierarchical statistical machine translation system. *Prague Bulletin of Mathematical Linguistics*, 104:5–16, September 2015.

Masukur Kaled M. Rahman, Wang Pi Yang, Tommy W. S. Chow, and Sitao Wu. A flexible multi-layer self-organizing map for generic processing of tree-structured data. *Pattern Recognition Letters*, 40(5): 1406–1424, May 2007.

Tapani Raiko, Alexander Ilin, and Juha Karhunen. Principal component analysis for sparse high-dimensional data. In Masumi Ishikawa, Kenji Doya, Hiroyuki Miyamoto, and Takeshi Yamakawa, editors, *Neural Information Processing: 14th International Conference,* *ICONIP 2007, Kitakyushu, Japan, November 13-16, 2007, Revised Selected Papers, Part I*, pages 566–575, Berlin, Heidelberg, Germany, 2008. Springer.

Bhiksha Raj and Ed Whittaker. Lossless compression of language model structure and word identifiers. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 388–391, Hong Kong, China, 2003. IEEE.

Carlos Ramisch. *Multiword Expressions Acquisition: A Generic and Open Framework*, chapter State of the Art in MWE Processing, pages 53–102. Springer International Publishing, Cham, Switzerland, 2015.

Uwe Reinke. State of the art in translation memory technology. *Translation: Computation, Corpora, Cognition*, 3(1), 2013.

Zhixiang Ren, Yajuan Lü, Jie Cao, Qun Liu, and Yun Huang. Improving statistical machine translation using domain bilingual multiword expressions. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*, pages 47–54, Singapore, August 2009. ACL. Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications.

Eréndira Rendón, Itzel M. Abundez, Citlalih Gutierrez, Sergio Díaz Zagal, Alejandra Arizmendi, Elvia M. Quiroz, and H. Elsa Arzate. A comparison of internal and external cluster validation indexes. In *Proceedings of the 2011 American Conference on Applied Mathematics and the 5th WSEAS International Conference on Computer Engineering and Applications*, pages 158–163, Stevens Point, USA, 2011. World Scientific and Engineering Academy and Society.

Stefan Riezler and John T. Maxwell. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, Ann Arbor, USA, June 2005. ACL.

Christian Rishøj and Anders Søgaard. Factored translation with unsupervised word clusters. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 447–451, Edinburgh, United Kingdom, July 2011. ACL.

Thomas Roelleke and Jun Wang. TF-IDF uncovered: A study of theories and probabilities. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’08, pages 435–442, New York, USA, 2008. ACM.

Frank Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, Washington, USA, 1962.

Ronald Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. *Computer Speech & Language*, 10:187–228, July 1996.

Ronald Rosenfeld. Two decades of statistical language modeling: Where do we go from here. *Proceedings of the IEEE*, 88:1270–1278, September 2000.

Ronald Rosenfeld, Stanley F. Chen, and Xiaojin Zhu. Whole-sentence exponential language models: a vehicle for linguistic-statistical integration. *Computer Speech & Language*, 15(1):55–73, August 2001.

Dmitri G. Roussinov and Hsinchun Chen. A scalable self-organizing map algorithm for textual classification: A neural network approach to thesaurus generation. *Communication Cognition and Artificial Intelligence*, 15(1-2):81–112, 1998.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323 (6088):533–536, October 1986.

Jyri Saarikoski. A study on the use of self-organised maps in information retrieval. *Journal of Documentation*, 65(2):304–322, 2009.

Jyri Saarikoski, Jorma Laurikkala, Kalervo Järvelin, and Martti Juhola. Self-organising maps in document classification: A comparison with six machine learning methods. In Andrej Dobnikar, Uroš Lotric, and Branko Šter, editors, *Proceedings of the 10th International Conference on Adaptive and Natural Computing Algorithms*, pages 260–269, Ljubljana, Slovenia, April 2011.

Ivan Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. Multiword expressions: A pain in the neck for NLP. In Alexander Gelbukh, editor, *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 189–206, Mexico City, Mexico, February 2002.

Stan Salvador and Philip Chan. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *16th IEEE International Conference on Tools with Artificial Intelligence*, pages 576–584, Boca Raton, USA, November 2004. IEEE Computer Society.

John W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18(5):401–409, May 1969.

Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, pages 44–49, Manchester, United Kingdom, 1994.

Helmut Schmid. Improvements in part-of-speech tagging with an application to german. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 47–50, Ann Arbor, USA, June 1995. ACL. SIGDAT Workshop.

Holger Schwenk. Continuous space language models. *Compututer Speech Language*, 21(3):492–518, July 2007.

Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March 2002.

Yoav Seginer. *Learning Syntactic Structure*. PhD thesis, University of Amsterdam, Amsterdam, The Netherlands, 2007.

Nasredine Semmar and Meriama Laib. Building multiword expressions bilingual lexicons for domain adaptation of an example-based machine translation system. In *Proceedings of the Ninth International Conference on Recent Advances in Natural Language Processing*, pages 661–670, Varna, Bulgaria, September 2017.

Rico Sennrich. Combining multi-engine machine translation and online learning through dynamic phrase tables. In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation*, pages 89–96, Leuven, Belgium, May 2011. EAMT.

Rico Sennrich. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 539–549, Avignon, France, April 2012a. ACL.

Rico Sennrich. Mixture-modeling with unsupervised clusters for domain adaptation in statistical machine translation. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pages 185–192, Trento, Italy, May 2012b. EAMT.

Rico Sennrich. Promoting flexible translations in statistical machine translation. In *Proceedings of the 14th Machine Translation Summit*, pages 207–214, Nice, France, September 2013a. EAMT.

Rico Sennrich. *Domain adaptation for translation models in statistical machine translation*. PhD thesis, University of Zurich, Zurich, Switzerland, 2013b.

Rico Sennrich. A CYK+ variant for SCFG decoding without a dot chart. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 94–102, Doha, Qatar, October 2014. ACL. SSST-8 Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation.

Rico Sennrich, Holger Schwenk, and Walid Aransa. A multi-domain translation model framework for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 832–840, Sofia, Bulgaria, August 2013. ACL.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 86–96, Berlin, Germany, August 2016. ACL.

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(4):623–656, October 1948.

Dana Shapira and James A. Storer. Edit distance with move operations. *Journal of Discrete Algorithms*, 5(2):380–392, 2007.

Stuart C. Shapiro. *Encyclopedia of Artificial Intelligence (Article: Artificial Intelligence)*. John Wiley & Sons, Inc., New York, USA, second edition, 1992.

Serge Sharoff, Bogdan Babych, and Anthony Hartley. Using collocations from comparable corpora to find translation equivalents. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 465–470, Genova, Italy, May 2006. ELRA.

Stephen Shum, William M. Campbell, and Douglas A. Reynolds. Large-scale community detection on speaker content graphs. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7716–7720, Vancouver, Canada, May 2013. IEEE.

Stephen H. Shum, Douglas A. Reynolds, Daniel Garcia-Romero, and Alan McCree. Unsupervised clustering approaches for domain adaptation in speaker recognition systems. In *Proceedings of the Speaker and Language Recognition Workshop*, pages 265–272, Joensuu, Finland, June 2014.

Daniel D. Sleator and Davy Temperley. Parsing English with a link grammar. In *Proceedings of Third International Workshop on Parsing Technologies*, pages 277–292, Tilburg, The Netherlands, 1993.

Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. Dirt cheap web-scale parallel text from the Common Crawl. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1374–1383, Sofia, Bulgaria, August 2013. ACL.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the Seventh Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, USA, 2006. AMTA.

Matthew Snover, Nitin Madnani, Bonnie J Dorr, and Richard Schwartz. TERp system description. In *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, Waikiki Beach, USA, 2008. AMTA. Published in the proceedings of Metrics MATR Workshop.

Harold Somers. *Computers and translation: Translation memory systems*, chapter 3, pages 31–47. John Benjamins, 2003.

Xingyi Song, Trevor Cohn, and Lucia Specia. BLEU deconstructed: Designing a better MT evaluation metric. *Computational Linguistics and Applications*, 4(2):29–44, 2013. ISSN 0976-0962.

Ankit Srivastava, Sergio Penkale, Declan Groves, and John Tinsley. Evaluating syntax-driven approaches to phrase extraction for MT. In *Proceedings of the Third Workshop on Example-Based Machine Translation*, pages 19–28, Dublin, Ireland, November 2009.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufiş, and Dániel Varga. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 2142–2147, Genova, Italy, May 2006. ELRA.

Andreas Stolcke. SRILM – an extensible language modeling toolkit. In *Proceedings of the Seventh International Conference on Spoken Language Processing*, pages 901–904, Denver, USA, September 2002. ISCA.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. SRILM at sixteen: Update and outlook. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*, Waikoloa, USA, December 2011. IEEE.

David Talbot and Miles Osborne. Randomised language modelling for statistical machine translation. In *Proceedings of the 45th Annual*

Meeting of the Association for Computational Linguistics, pages 512–519, Prague, Czech Republic, June 2007. ACL.

George Tambouratzis, Georgios Tsatsanifos, Ioannis Dologlou, and Nikos Tsimboukakis. SOM-based corpus modeling for disambiguation purposes in MT. In Petr Sojka, editor, *Proceedings of the 15th International Conference on Text, Speech and Dialogue*, pages 29–36, Brno, Czech Republic, September 2012a. Workshop on Hybrid Machine Translation.

George Tambouratzis, Marina Vassiliou, and Sokratis Sofianopoulos. Agent-based modeling of language evolution. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–68, Avignon, France, April 2012b. ACL. Joint Workshop on Exploiting Synergies Between Information Retrieval and Machine Translation (ESIRMT) and Hybrid Approaches to Machine Translation (HyTra).

George Tambouratzis, Sokratis Sofianopoulos, and Marina Vassiliou. Language-independent hybrid MT with PRESEMT. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 123–130, Sofia, Bulgaria, August 2013. ACL. HYTRA-2013 Workshop.

Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualization large-scale and high-dimensional data. *CoRR*, abs/1602.00370, 2016.

Irina Temnikova, William A. Baumgartner Jr., Negacy D. Hailu, Ivelina Nikolova, Tony McEnery, Adam Kilgarriff, Galia Angelova, and K. Bretonnel Cohen. Sublanguage corpus analysis toolkit: A tool for assessing the representativeness and sublanguage characteristics of corpora. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 1714–1718, Reykjavik, Iceland, May 2014. ELRA.

Benoît Thouin. The Meteo system. In Veronica Lawson, editor, *Practical Experience of Machine Translation*, pages 39–44. North-Holland, Amsterdam, The Netherlands, 1982.

Jörg Tiedemann. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. *Recent Advances in Natural Language Processing: Selected Papers from RANLP 2007. Borovets, Bulgaria.*, 5:237–248, 2009.

Jörg Tiedemann. Context adaptation in statistical machine translation using models with exponentially decaying cache. In *Proceedings of* *the 2010 Workshop on Domain Adaptation for Natural Language Processing*, DANLP 2010, pages 8–15, Stroudsburg, USA, July 2010. ACL, ACL.

Jörg Tiedemann. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 2214–2218, Istanbul, Turkey, May 2012. ELRA.

Antonio Toral and Víctor M. Sánchez-Cartagena. A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1063–1073, Valencia, Spain, April 2017. ACL.

Jost Trier. Das sprachliche Felt: eine Auseinandersetzung. In *Neue Jahrbücher für Wissenschaft und Jugendbildung*, volume 10, pages 428–49. Mouton & Co, The Hauge, The Netherlands, 1973.

Nikolaos Tsimboukakis and George Tambouratzis. Self-organising word map for context-based document classification. In *Proceedings of the WSOM-2007 Conference*, Bielefeld, Germany, September 2007.

Hélène Valbret, Eric Moulines, and Jean-Pierre Tubach. Voice transformation using PSOLA technique. *Speech Communication*, 11(2-3): 175–187, June 1992.

Marlies van der Wees, Arianna Bisazza, Wouter Weerkamp, and Christof Monz. What's in a domain? analyzing genre and topic differences in statistical machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 560–566, Beijing, China, 01 2015. ACL.

Stijn van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, Utrecht, The Netherlands, 2000.

Bernard Vauquois. Automatic translation — a survey of different approaches. In Hans Karlgren, editor, *Proceedings of the Sixth International Conference on Computational Linguistics*, pages 127–135, Ottawa, Canada, 1976. ACL.

Juha Vesanto and Esa Alhoniemi. Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11:586–600, May 2000.

Martin Volk. Using the web as corpus for linguistic research. In Renate Pajusalu and Tiit Hennoste, editors, *Tähendusepüüdja. Catcher of the Meaning. A Festschrift for Professor Haldur Õim*. University of Tartu, 2002.

Martin Volk, Noah Bubenhofer, Adrian Althaus, Maya Bangerter, Lenz Furrer, and Beni Ruef. Challenges in building a multilingual alpine heritage corpus. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, Valetta, Malta, May 2010. ELRA.

Longyue Wang, Derek F Wong, Lidia S Chao, Yi Lu, and Junwen Xing. A systematic comparison of data selection criteria for SMT domain adaptation. *The Scientific World Journal*, 2014.

Rui Wang, Hai Zhao, Lu Bao-Liang, Masao Utiyama, and Eiichiro Sumita. Connecting phrase based statistical machine translation adapation. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 3135–3145, Osaka, Japan, December 2016. ACL.

Wei Wang, Klaus Macherey, Wolfgang Macherey, Franz Och, and Peng Xu. Improved domain adaptation for statistical machine translation. In *Proceedings of the 10th Conference of the Association for Machine Translation in the Americas*, San Diego, USA, 2012. AMTA.

Joe H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

Warren Weaver. Correspondence with Norbert Wiener. Online. With permission from the Rockefeller Foundation Archives., 1947. URL `http://www.mt-archive.info/50/Weaver-1947-original.pdf`. Last visited: March 27, 2019.

Stefan Wermter and Chihli Hung. Self-organizing classification on the Reuters news corpus. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan, August 2002. ACL.

Philip Williams, Rico Sennrich, Maria Nadejde, Matthias Huck, and Philipp Koehn. Edinburgh's syntax-based systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 199–209, Lisbon, Portugal, September 2015. ACL. 10th Workshop on Statistical Machine Translation.

Ludwig Wittgenstein. *Philosophical Investigations*. Basil Blackwell, Oxford, United Kingdom, 1953.

Petra Wolf and Ulrike Bernardi. Hybrid domain adaptation for a rule based MT system. In *Proceedings of the 14th Machine Translation Summit*, pages 321–328, Nice, France, September 2013. EAMT.

Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting TF-IDF term weights as making relevance decisions. *ACM Trans. Inf. Syst.*, 26(3):13:1–13:37, June 2008a.

Hua Wu, Haifeng Wang, and Chengqing Zong. Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 993–1000, Manchester, United Kingdom, August 2008b. ACL.

Sitao Wu and Tommy W. S. Chow. Clustering of the self-organizing map using a clustering validity index based on inter-cluster and intra-cluster density. *Pattern Recognition*, 37(2):175–188, 2004.

Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, 2015.

Jia Xu, Yonggang Deng, Yuqing Gao, and Hermann Ney. Domain dependent statistical machine translation. In *Proceedings of the 11th Machine Translation Summit*, pages 515–520, Copenhagen, Denmark, September 2007. EAMT.

Hirofumi Yamamoto and Eiichiro Sumita. Bilingual cluster based models for statistical machine translation. *IEICE Transactions on Information and Systems*, E91-D(3):588–597, March 2008.

Roman V. Yampolskiy. *Artificial Superintelligence: A Futuristic Approach*. Chapman & Hall/CRC, 2015.

Shuang-Hong Yang, Hongyuan Zha, and Bao-Gang Hu. Dirichlet-Bernoulli alignment: A generative model for multi-class multi-label multi-instance corpora. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2143–2150. Curran Associates, Inc., Red Hook, USA, 2009.

Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49, New York, USA, 1999. ACM.

Majid Yazdani, Meghdad Farahmand, and James Henderson. Learning semantic composition to detect non-compositionalityof multiword expressions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1733–1742, Lisbon, Portugal, September 2015. ACL. Sixth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis.

Omar F. Zaidan and Chris Callison-Burch. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1220–1229, Portland, USA, June 2011. ACL.

M. Zhang and Z. Zhou. A review on multi-label learning algorithms. *Knowledge and Data Engineering, IEEE Transactions on*, PP(99):1, 2013.

Bing Zhao, Matthias Eck, and Stephan Vogel. Language model adaptation for statistical machine translation with structured query models. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 327–330, Geneva, Switzerland, August 2004. ACL.

George Kingsley Zipf. *The Psychobiology of Language: An Introduction to Dynamic Philology*. MIT Press, Cambridge, USA, 1935.

Andreas Zollmann, Ashish Venugopal, Franz Och, and Jay Ponte. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1145–1152, Manchester, United Kingdom, August 2008. ACL.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, USA, November 2016. ACL.

Arnold M. Zwicky and Ann D. Zwicky. Register as a domain of linguistic variance. In Kittredge and Lehrberger [1982], chapter 9.

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

`http://code.google.com/p/classicthesis/`

Final Version as of April 1, 2019 (`classicthesis` version 1.5 RC VIII).