NTNU

Norwegian University of
Science and Technology

Human-Computer Collaboration for Faster Document Comparison

Audun Liberg

Master of Science in Computer Science
Submission date: July 2017
Supervisor: Rune Sætre, IDI

Norwegian University of Science and Technology
Department of Computer Science

Audun Liberg

Human–Computer Collaboration for Faster Document Comparison

Master's Thesis in Computer Science, Spring 2017

Data and Artificial Intelligence Group
Department of Computer Science
Faculty of Information Technology and Electrical Engineering
Norwegian University of Science and Technology

Abstract

The amount of data in the world is increasing rapidly. One common operation on large datasets is one-to-all comparison, where the goal is to compare one object to all the other objects in the set. This thesis investigates the possibility of approaching the problem using human–computer collaboration.

To this end, the human-guided filtering model (HGFM) is proposed. The model provides a general framework for one-to-all comparison, where cluster analysis is used to group similar objects together. Through the help of a human domain expert, the contents of irrelevant clusters can be removed from the process.

An implementation of the model is demonstrated, and tested over a series of experiments. During these experiments, it is shown that the model can reduce the size of the dataset with up to 80 % before comparison takes place, creating ample opportunity for saving both time and computational resources.

In light of the model's apparent potential, several directions for future research is proposed at the end of the thesis.

Sammendrag

Mengden data i verden øker hurtig. En vanlig operasjon på store datasett er én-til-alle-sammenligning, hvor målet er å sammenligne ett objekt med alle de andre objektene i settet. Denne oppgaven undersøker mulighetene ved en tilnærming til problemet tuftet på menneske–maskin–samarbeid.

Med dette mål for øyet, foreslås *the human guided filtering model* (HGFM) – den menneske-ledede filtreringsmodellen. Modellen tilbyr et generelt rammeverk for én-til-alle-sammenligning, hvor klyngeanalyse brukes til å gruppere lignende objekter sammen. Ved hjelp av en menneskelig domeneekspert, blir innholdet i irrelevante klynger ekskludert fra prosessen.

En implementasjon av modellen blir demonstrert, og testes gjennom en rekke eksperimenter. Eksperimentene viser at modellen kan redusere størrelsen på datasettet med inntil 80 % før sammenligningen finner sted. Dette gir anledning til å spare både tid og beregningsressurser.

I lys av modellens tilsynelatende potensiale, foreslås flere retninger for videre forskning ved oppgavens slutt.

Preface

The work of this thesis is the culmination of my five-year Computer Science studies at the Norwegian University of Science and Technology (NTNU). The research is conducted at The Department of Computer Science (IDI), and has been supervised by Associate Professor Rune Sætre. I would like to extend a huge thanks to him for his valuable input during the last year.

Oslo, July 12 2017
Audun Liberg

Contents

List of Figures

List of Tables

Acronyms

BoW bag-of-words 7

HCC human–computer collaboration 1

HGFM human-guided filtering model 3

IR Information Retrieval 2, 7

MAP mean average precision 15

NLTK Natural Language Toolkit *(Python library)* 29

NTNU The Norwegian University of Science and Technology 2

OTAC one-to-all comparison 2

tf–idf term frequency–inverse document frequency 8

VSM Vector Space Model 2, 11

1 Introduction

This chapter will introduce the thesis, including its goals and the motivation behind it. It will then present a set of research questions that are to be investigated throughout the report, as well as the methodology used to research them. Lastly, an outline of the remaining report will be given.

1.1 Background and Motivation

The amount of data in the world is increasing rapidly. It is estimated that by 2020, the digital universe will house more than 40,000 exabytes – 40 trillion gigabytes – of data. An emerging problem is our inability to process all of this data fast enough. The heaps of data that remains unprocessed has been referred to as the *Big Data Gap*, and the gap is growing [1].

A large portion of the world's data pool consists of human-produced text. As of 2017, over 31 million messages are sent through the social network Facebook every minute [2]. With this in mind, it is clear that we cannot rely on increased computational power alone if we want to fill the Big Data Gap. To tackle the issue, we will need new techniques designed for faster data processing as well. One such technique that has seen widespread use over the last years, is the MapReduce programming model, which utilizes a parallel, distributed algorithm to manage data clusters [3]. This thesis will explore an alternative approach to the problem, by investigating the effectiveness of human–computer collaboration (HCC).

1.1.1 Why Human–Computer Collaboration?

In 1997, IBM's Deep Blue defeated world chess champion Garry Kasparov. After his loss, Kasparov commented on Deep Blue's playing style, calling it both deeply intelligent and creative, suggesting it could have been aided by a human chess player. William S. Cleveland, Professor of Statistics at Purdue University, has since asked the following question: *Why was Kasparov so obsessed with the notion that IBM was cheating, allowing a Grand Master to alter moves of Deep Blue?* His answer: *He knew that he had no chance to beat a combination of the extraordinary tactical power of the machine learning algorithms and the strategic power of the human* [4].

While it is unlikely that Deep Blue was actually aided by humans, this thought experiment underlines the fact that a human–machine combination can be extremely powerful. While existing algorithms have proven to be adept at solving a number of different problems, they still lack the ability to easily tap into and utilize contextual knowledge

unrelated to what they have been explicitly exposed to. Even with recent advances in machine learning, humans still have the upper hand when it comes to knowledge and reasoning.

By combining the strengths of humans and computers, it is possible to revise the decisions made by the computer programs. This could support algorithms when dealing with unknown data, or save them from committing errors that are obvious to the human-eye, but invisible to the computer. This is not limited to the world of strategic games like chess, but extends to virtually any application where humans have understanding.

There are two possible advantages of human–computer collaboration: As pointed out above, it can lead to improved decision-making. It can also be applied to save time, by having a human expert review the problem and make sub-decisions that are intuitive to the human, but complex and time-consuming to the algorithm. This thesis will mainly investigate the latter option. Whether the data set is too large or the computational resources are simply lacking, introducing a human element could make the problem more manageable.

1.1.2 Textual Similarity

A common task when processing human-produced text, is the act of measuring textual similarity – given two texts, how much alike are they? Similarity on a surface-level is referred to as lexical similarity, while deeper similarity related to meaning is referred to as semantic similarity. Previous applications of textual similarity include automatic essay grading [5], automatic evaluation of machine-translated texts [6], and plagiarism detection [7].

In the context of the ever-growing collections of digital data, plagiarism detection is a particularly interesting problem. The more texts we produce, the more texts we have to search through in order to ensure that plagiarism has not occurred. We formalize this issue as the one-to-all comparison (OTAC) problem. Given one document, we want to find all similar documents in a potentially large collection of documents.

In classical information retrieval (IR), there are several methods for document comparison that can be employed to measure textual similarity. Well-known approaches include simple Boolean models and the vector space model (VSM) [8]. While effective at calculating lexical similarity, utilizing classical methods like these alone would eventually become impractical for OTAC, given a large enough document base. This thesis will research if human–computer collaboration can be used as a more time-efficient approach to the OTAC problem.

1.1.3 Use-Case – The NTNU Course Catalogue

When the Norwegian University of Science and Technology (NTNU) merged with several other educational institutions in early 2016, they became Norway's largest university. This resulted in a heavily extended course catalogue, with more than 4,000 different courses. In several cases, you would want to compare a course to the university's entire course catalogue. For example, a professor might want to find similar courses to his own

at other campuses, or future exchange students could be looking for fitting NTNU courses based on courses from his or her home university. Such scenarios are all OTAC problems, where the goal is to compare a single document to the entire document collection and find the best matches.

1.2 Research Goals and Questions

The overarching goal of the thesis is to explore the potential of saving time on the OTAC problem through the means of human–computer collaboration. To this end, the human-guided filtering model (HGFM) is proposed as a novel method for one-to-all comparison on large datasets. The model provides a stepwise framework for filtering out irrelevant data before the actual comparison takes place. This is done by augmenting standard information retrieval methods with machine learning cluster analysis, and adding a human domain expert to the process.

An implementation of the model will be applied to the NTNU course catalogue, making it possible to find similar courses given an input course. The results should shed light on the usability of the model, as well as its limitations.

1.2.1 Research Questions

The thesis will attempt to answer the following research questions:

RQ1 How much time can be saved by filtering datasets using HGFM?

RQ2 How large portions of the dataset can be filtered out, and how does it affect the output quality?

RQ3 How important is the choice of clustering algorithm to the overall results of the model?

1.3 Research Methodology

In order to answer the aforementioned research questions, a series of experiments will be conducted. Using the NTNU course catalogue as a dataset, the results of the HGFM will be compared to the results of traditional information retrieval algorithms. The various algorithms will be compared with respect to both relevancy of retrieved documents, and time efficiency. Focus will be placed on determining if HGFM can speed up the comparison process, and how the possible speed-up affects the quality and relevancy of the retrieved data.

1.4 Report Structure

The following section will introduce the remaining chapters of the thesis in order.

1.4.1 Background

This chapter will give the reader the necessary background to understand the concepts discussed later in the report. This includes models for document representation, a presentation of relevant preprocessing techniques, an overview of classical textual similarity measures, algorithms for cluster analysis, and metrics for output evaluation in information retrieval.

1.4.2 Related Work

This chapter will provide an overview of related work from relevant fields. This includes previous attempts at human–computer collaboration, as well as conventional ways of doing similarity measurement on big data.

1.4.3 Data

The Data chapter will introduce the dataset used to develop and test the human-guided filtering model. This includes a presentation of document features, how documents can be compared, a list of shortcomings tied to the dataset, and some descriptive statistics.

1.4.4 Model and Implementation

This chapter will introduce the human-guided filtering model. The rationale behind the model will be presented, before each component of the model is described in detail. A concrete implementation of the model is then presented. The selection of algorithms and technology in the system is explained and justified. A brief manual on how to use the system is given last.

1.4.5 Experimental Setup

After presenting the model, this chapter will describe the setup of the experiments which will be conducted on the implementation. It will also cover the methodology used when developing and testing the system.

1.4.6 Experiments and Results

This chapter will document the results of the planned experiments. This includes the HGFM implementation's performance on the NTNU course catalogue, as well as the performance of conventional IR algorithms for later comparison.

1.4.7 Discussion

This chapter will discuss the results of the conducted experiments. It will evaluate the performance of the implementation, by comparing it to the performance of non-human-guided IR algorithms.

1.4.8 Conclusion

Based on the findings of the previous chapters, this last chapter will draw a conclusion. It will review the stipulated research questions and answer them in accordance with the discussed results. The thesis is concluded with a list of proposals for future work based on this study.

2 Background

The human-guided filtering model goes through two separate phases, each with a number of steps. These are summarized below.

Indexing Phase

1. Document Preprocessing
2. Clustering
3. Labeling

Comparison Phase

1. Human-Guided Cluster Selection
2. Similarity Measurement

The model bears the structure of a classical information retrieval IR system, with an additional element of machine learning in the form of cluster analysis. This chapter will provide a theoretical background for the relevant methods and techniques. This includes ways of representing documents and preprocessing them, as well as algorithms for clustering and textual similarity assessment. Finally, standard measures for information retrieval evaluation will be covered.

2.1 Document Representation

In order to process a document, its features must be condensed and represented in some way. This section will present the simple bag-of-words model, as well as the tf–idf extension.

2.1.1 The Bag-of-Words Model

As the name implies, the bag-of-words (BoW) model represents a document as a bag containing its individual words. In mathematics, a bag, also known as a multiset, is defined as a set where multiple instances of an element is allowed. Similar to a regular set, the order of the elements does not matter. As a consequence, BoW is a representational model that disregards the document's order of words, relying only on term frequency.

Given a set of documents, each document will be assigned a vector. The elements of the vector each corresponds to a distinct word in the document collection, and represents its frequency in the given document. Consider the two documents below.

a) John is a baker, not a cook.

b) Mary is married to a baker. Mary has a husband named John.

Using the BoW model, each document would be assigned the following vector representation:

	john	*is*	*a*	*baker*	*not*	*cook*	*mary*	*married*	*to*	*has*	*husband*	*named*
$V_a = [$	1	1	2	1	1	1	0	0	0	0	0	0]
$V_b = [$	1	1	2	1	0	0	2	1	1	1	1	1]

Being fairly simple, the model has some major drawbacks [9]. Disregarding the order of words means that different documents might map to the same vector, as long as the words are the same. An example of this, is "toy dog" and "dog toy". A toy dog is obviously not a dog toy, but their bags of words are the same. Another weakness is the model's obliviousness to semantics. Each word is treated as completely separate from every other word, even when their meaning is close.

2.1.2 tf–idf

Term frequency–inverse document frequency (tf–idf) is a weighting metric for terms in a document collection. Its main purpose is to provide a weighting that reflects not only the frequency of a term in a single document, but also takes the frequency of the word in the overall collection into account [10].

tf–idf weighting can be used to extend the BoW model of document representation [11]. In its unweighted form, BoW representation will treat each word equally, independent of its use in the rest of the document collection. This will create the illusion that common terms, such as stopwords, are important to the document. By weighting each term using tf–idf, terms will be ranked based on how discriminatory they are, placing less importance on stopwords and other common words in the corpus.

Mathematically, tf–idf is defined as

$$\text{tf–idf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D),$$

where $\text{tf}(t, d)$ is the term frequency of term t in document d, and $\text{idf}(t, D)$ is the inverse document frequency of t in document collection D. There are several variations of how to calculate term frequency and inverse document frequency [12]. Common approaches to term frequency include binary counting, raw counting and logarithmic scaling. The inverse document frequency is often logarithmically scaled. No matter the exact calculation, the general idea is that the idf should offset the frequency of words that are common in the corpus.

While tf–idf, like BoW, is blind to semantics and fails to address word order, it has seen wide usage as a term weighting metric [13]. In conjunction with BoW, it ensures that a term's relative importance is taken into account during processing.

2.2 Document Preprocessing

When performing IR-related operations on a document, it is common for the document to go through some kind of preprocessing first. This often involves condensing the full text into a collection of terms that are descriptive of the document, known as index-terms. Preprocessing of documents is usually divided into five primary steps [14]:

1. Lexical analysis
2. Stopword elimination
3. Stemming
4. Index-term selection
5. Thesauri construction

The first four steps are relevant to the thesis, and the following subsections will explain them in detail.

2.2.1 Lexical Analysis

At its rawest form, a document is essentially just a byte stream. The process of lexical analysis is concerned with converting this stream of characters into *tokens* – strings with an associated meaning. This makes it possible to further process the data, for example by removing tokens identified as stopwords. During lexical analysis, one has to consider a number of elements, namely how to deal with punctuation, hyphens and similar characters, and how to treat numbers.

2.2.2 Stopword Elimination

Some words are more common than others. The most common words in a given language are known as stopwords, and the stopword elimination step seeks to remove them. The rationale behind this, is that the final index-terms should be descriptive of their particular document. Common words can by definition be found in many documents, and will therefore have a low discrimination value. One way of performing stopword elimination, is to use a predefined list.

2.2.3 Stemming

Stemming is the act of reducing a word to its root form. For instance, both ”fishing” and ”fished” should be reduced to ”fish”. When searching or comparing documents, stemming is useful because it allows us to bypass the fact that words can be represented on different forms.

2.2.4 Index-Term Selection

After having performed lexical analysis, stopword elimination and stemming, one is left with a term list containing the index candidates. As the name implies, index-term selection is the process of actually selecting the indices. This can be done either manually or automatically, each with its set of strengths and weaknesses. There are a number of things to consider during selection, making the choice hard. One approach is to weight each term by their relative descriptiveness, for example by calculating their tf–idf values.

2.3 Textual Similarity

Textual similarity approaches can be categorized into three different types: String-based similarity, corpus-based similarity and knowledge-based similarity [15]. While the first type measures lexical similarity, the latter types measure semantic similarity. This thesis is mainly concerned with string-based similarity.

String-based similarity can be either character-based or term-based. Character-based similarity compares strings using their individual characters as units, while term-based comparison will use words. The subsequent sections will describe some common string-based similarity measures.

2.3.1 Character-Based Similarity Measures

This section presents two character-based similarity measures.

Jaccard Similarity Coefficient

The Jaccard similarity coefficient, also known as the Jaccard index, is a simple similarity measure coined by Paul Jaccard [16]. Given two sets A and B, their Jaccard index is defined as the size of their intersection divided by the size of their union.

$$J(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

The Jaccard index of two strings can be calculated by treating the characters of each word as sets. Consider the strings "one" and "ten". Because they share two out of four different letters, their Jaccard index equals $\frac{1}{2}$.

Although simple, the Jaccard similarity’s biggest advantage over other string-based similarity measures, is that computations on sets can be performed relatively fast. The

above example illustrates its main weakness; it does not take the character order into account. A consequence of this, among other things, is that anagrams will be treated as equal strings.

Levenshtein Distance

The Levenshtein distance is a string similarity measure, defined as the minimal number of character operations required to transform one string to another [17]. There are three such operations: Insertion, deletion and substitution. Formally, for strings a and b:

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if} \min(i,j) = 0, \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

This edit-distance approach is more computationally taxing than calculating the Jaccard index. However, it mends several weaknesses of Jaccard, by taking both character order and repetition of letters into account.

2.3.2 Term-Based Similarity

This section presents two term-based similarity measures.

The Boolean Model

The Boolean model of information retrieval is a simple method of comparison based on Boolean algebra [18]. By considering the index-terms of a document, the document can be compared to other documents through a query. Each comparison will result in either a match or a mismatch. Consequently, the results of this model cannot be ranked. Additionally, it does not utilize any term-weighting like tf–idf. Still, it has the advantages of being both simple and intuitive.

The Vector Space Model

A widely-applied term-based approach to document comparison is the vector space model (VSM) by Salton et al [8]. Documents are represented as vectors in a bag-of-words fashion, where each dimension corresponds to a term in the global corpus. Each term value says something about the term's frequency in the document, and is usually weighted. One common weight is the tf–idf value. Using these vector representations, it is possible to compare the documents by calculating the cosine of the angle between them. Formally:

$$\cos\theta = \frac{\mathbf{d_1} \cdot \mathbf{d_2}}{|\mathbf{d_1}|\,|\mathbf{d_2}|},$$

where $\mathbf{d_1} \cdot \mathbf{d_2}$ is the dot product and $|\mathbf{d_1}||\mathbf{d_2}|$ is the product of the vector lengths [19]. The relation between the document vectors, terms and angle is illustrated in Figure 2.1.

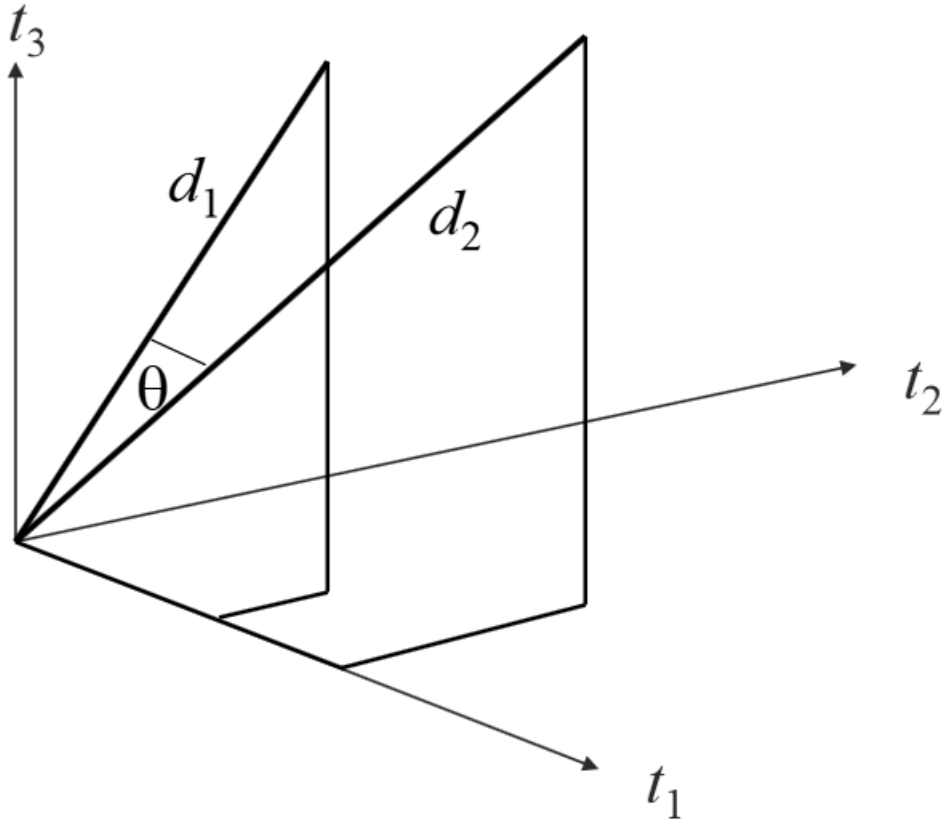

Figure 2.1: Visualization of the vector space model [20].

2.4 Data Clustering

Clustering, also known as cluster analysis, is the act of categorizing objects into groups, where objects in the same group are similar to each other. These groups are called clusters.

There are several algorithms for cluster analysis, with approaches ranging from statistical machine learning to neural networks. This section will present one particular algorithm relevant to the thesis, as well as a method for cluster visualization.

2.4.1 *k*-Means Clustering

The k-means machine learning approach to clustering was first introduced by MacQueen in 1967 [21], while the standard algorithm was presented by LLoyd in 1982 [22]. Clustering is performed by minimizing the objective function

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i,$$

where $\mathbf{x}$ is the set of observations, S is the cluster set, and $\boldsymbol{\mu}$ is the mean of points of its corresponding set. Lloyd's algorithm achieves this by iteratively calculating new means as centroids for each cluster.

2.4.2 Cluster Visualization

One way of visualizing clusters, is through Voronoi diagrams. A Voronoi diagram divides the plane into several regions referred to as Voronoi cells. Each cell is defined as a convex polygon with a generating point, so that every other point in the polygon is closer to that particular generating point, than to any other. Formally, this can be defined as

$$R_k = \{x \in X \mid d(x, P_k) \leq d(x, P_j) \text{ for all } j \neq k\},$$

where $(R_k)_{k \in K}$ is the tuple of cells forming the diagram. K denotes the cell indices. X is the metric space, and d its distance function. P_k is defined as a site with a set of points so that no point is any closer to another site [23].

Intuitively, when visualizing clusters, each cell of the Voronoi diagram represent a cluster. The diagram type is especially fitting for the k-means approach to clustering, as the centroids naturally serve as generating points. An example of a Voronoi diagram can be seen in Figure 2.2.

2.5 Output Evaluation

This section will present some common metrics for evaluating the output quality of an IR system.

2.5.1 Precision and Recall

Precision and recall are performance measures used to calculate the quality of the output of an information retrieval system. Both measures assume binary classification — documents are either relevant or irrelevant to the query. The precision of a result set is defined as the fraction of retrieved documents that are relevant.

$$precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|}$$

The recall value is defined as the fraction of relevant documents that are retrieved.

$$recall = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|}$$

Precision and recall are complementary, meaning they should always be presented together. It is trivial to achieve a high recall value alone, simply by retrieving a high

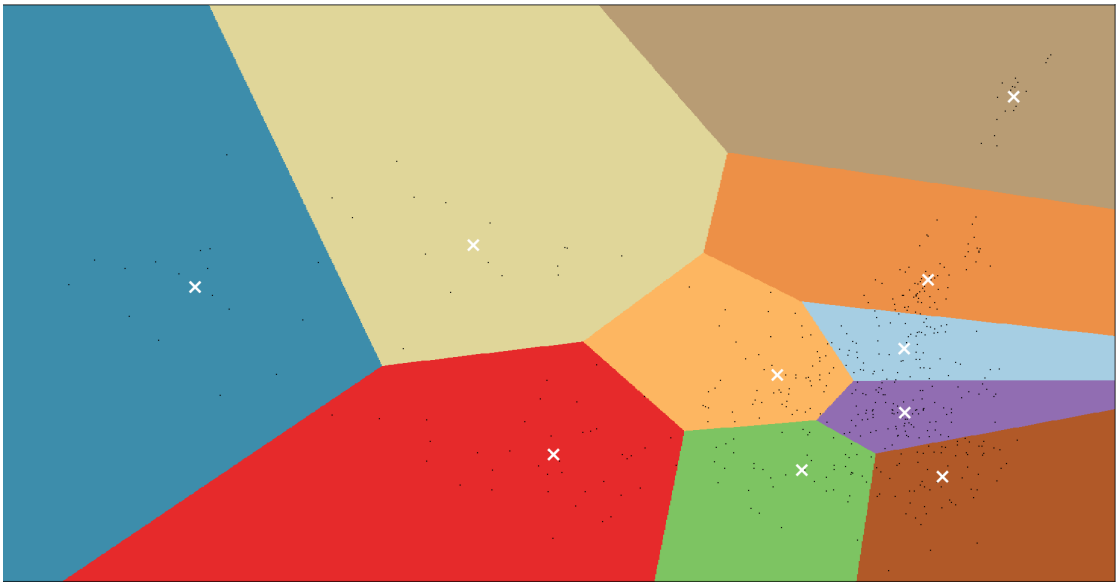

Figure 2.2: An example of a Voronoi diagram. Generating points are marked with a cross.

number of documents. Similarly, one can achieve a high precision value by retrieving very few, but highly probable relevant documents.

The importance of each metric varies with the objective of the system. In cases where it is important to retrieve all relevant documents, one can perhaps afford a lower precision to the benefit of a higher recall value. In other cases, like in web search, the number of relevant documents is so high that the recall value has little to no meaning. One would rather focus on retrieving a subset with the most relevant documents, and optimize the precision to avoid false positives [24].

2.5.2 F-Measure

The F-measure, also known as the harmonic mean, is a way of combining precision and value into a single value.

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

It is 0 when no relevant documents are retrieved, and 1 when all of the retrieved documents are relevant. This makes it a suitable measure of the system's overall performance, as well as an indicator on whether the compromise between precision and recall is good [25].

2.5.3 Mean Average Precision

Precision and recall are calculated based on the whole list of documents retrieved. The average precision takes the order into account as well, by looking at the precision at each recall-level (i.e. the positions where a relevant document is retrieved). The average precision is the mean of these precisions. The mean average precision (MAP) is the average precisions across a set of queries:

$$MAP = \frac{\sum_{q=1}^{Q} \mathrm{AP(q)}}{Q},$$

where $AP(q)$ is the average precision of query q, and Q is the number of queries [26].

3 Related Work

The following chapter will present previous work related to the thesis. This includes a review of human–computer collaboration through the years, as well as conventional ways of doing similarity assessment on big data.

3.1 Human–Computer Collaboration

This section will present various attempts at human–computer collaboration. It will begin with some early research, before moving on to more recent studies.

3.1.1 Early Approaches

In this day and age, it is not obvious that humans are able to increase the performance of well-developed algorithms. Still, studies seem to indicate that a human–machine combination outperforms both humans and machines alone. One early study on the matter was conducted by Krolak et al. in 1971 [27]. Here, a man–machine approach to the NP-hard travelling salesperson problem was proposed. The machine was responsible for organizing and suggesting possible solutions to the problem. The human would then use its eyes and problem solving skills to pick the best option. The results were promising, solving large datasets faster than conventional algorithms at the time.

Since then, man–machine approaches has been attempted at various domains. In 1988, Long et al. proposed a system for evaluating visual clinical research data in medicine [28], and Sargeant et al. proposed a similar system for free-text grading in 2004 [29]. While the benefit of including a human in the process varies from case to case, all the aforementioned studies identifies situations where their solution outperform conventional approaches in some way.

3.1.2 Formalism

Because human–computer collaboration is a broad term, the amount of formalism tied to the field is limited. One attempt at defining the field more formally was done by Terveen in 1994 [30]. Terveen identifies two major approaches to human–machine collaboration: The Human Emulation Approach and the Human Complementary Approach. The former approach draws from the field of artificial intelligence. Both the human and the computer are seen as rational agents with goals to complete. Collaboration occurs when agents communicate in order to reach their goals. Here, the parts can be assumed to have symmetric abilities, both being able to reason and complete their designated

tasks. The Human Complementary Approach, however, assumes asymmetric abilities. The human and the computer have separate strengths and weaknesses, and collaborate by dividing responsibility and exploiting each others strengths.

3.1.3 Human-Guided Machine Learning

Recently, so-called human-guided machine learning has branched out into its own field. This is a type of human–computer collaboration where the human aids the machine learning algorithm during the learning process. Holloway et al. trained a machine learning model using human engineered features instead of regular data [31], while Amershi et al. used human-guided machine learning to achieve fast and accurate network alarm triage [32]. Both parties achieved promising results. The Association for the Advancement of Artificial Intelligence (AAAI) held its first workshop on the topic in February 2017 [33].

3.2 Similarity Assessment on Big Data

As this thesis proposes a novel way of doing similarity measurement on big data, the conventional methods should be introduced first. This section will give an overview of how big data processing is handled in large system, as well as a specific approach to similarity assessment.

Chapter 1 mentioned the MapReduce framework [3]. Today, this is the de facto standard in big data processing [34]. The programming model allows for processing of large amount of data with a parallel, distributed algorithm on a cluster of computers. Apache Hadoop is one of the most popular implementations of the model.

In 2014, Zadeh and Carlsson proposed a method of doing large-scale similarity assessment using MapReduce [35]. Their paper and algorithm was called Dimension Independent Matrix Square using MapReduce (DIMSUM). The DIMSUM algorithm computes all-pairs similarity, meaning it finds the similarity of every document pair in the collection, with the criterion that their similarity have to be above a certain criterion. The algorithm has a probabilistic approach, using sampling to focus the computational effort on the pairs that are above the similarity threshold. The research was done in collaboration with Twitter, and managed to improve the performance of a particular batch job or theirs with 40 %.

4 Data

As established in Chapter 1, the NTNU course catalogue will serve as a dataset for the development and testing of the human-guided filtering model. There are several properties linked to the catalogue and its courses that makes it suited for the purpose of this thesis. First and foremost, it is a sizable collection of human-produced text. Secondly, there are several concepts tied to a course that humans intuitively understand, making them a fitting test-bed for human–computer collaboration.

This chapter will formally introduce the notion of a course, including key features and how they are tied to each other. It will also explain the thesis' definition of similar courses, and how similarity assessments of courses can be objectively judged. A brief statistical overview of the dataset is given at the end.

4.1 Courses

Formally, courses are the building-blocks of a study program. They all have a unique identification code, as well as a non-unique name. Each course has a set curriculum, defining the topics of the course. The curriculum is described by two chunks of text: *Academic content* and *learning outcome*. Each course is administrated by a department, i.e. The Department of Computer Science. A course's corresponding department will naturally say something about its topic, although the lines can get blurred. For example, The Department of Mathematical Sciences has a course on supercomputers. Each department belongs to a faculty. There are 8 faculties at NTNU, and 56 departments. While course code and department are structured values, other features are human-generated text, and as such, highly unstructured.

An overview of course features can be seen in Table 4.1. An example of a course instance can be seen in Table 4.2. Experiments in this thesis has been conducted on course data gathered through IME's Data API for courses[1].

4.2 Comparing Courses

For this thesis, two courses will be classified as similar if they teach some of the same content. In other words, similar courses will have similar curricula. Here, the textual similarity of courses will be assessed using course names, academic content and learning outcome.

[1] http://www.ime.ntnu.no/api/

Feature	Description	Structured
Code	A unique identifier for the course, with an alphabetical prefix related to its field of study.	Yes
Name	The name of the course.	No
Academic content	A description of the course's curriculum.	No
Learning outcome	The skills and knowledge the student will aquire from the course.	No
Department	The university department responsible for the course.	Yes
Credit reduction	A list of courses that will result in a reduction of credits if you have passed them earlier.	Yes

Table 4.1: The features of a course data object.

Feature	Value
Code	TDT4100
Name	Object-Oriented Programming
Academic content	Basic algorithms and data structures, constructs and control flow in object-oriented languages. Modularization and re-use. Standard application programmers interface (API). Unit testing, error detection and tools for this. Object-oriented design. Use of class, sequence and collaboration diagrams in the UML. Use of design patterns. Java is used as implementation language.
Learning outcome	The students will have skills in programming, training in usage of relevant programming methods and tools, Also knowledge and understanding of usage areas, restrictions and underlying theory.
Department	The Department of Computer Science
Credit reduction	IT1104 7.5 credits SIF8005 7.5 credits TDT4102 5.0 credits [...]

Table 4.2: An instance of a course data object.

In order to measure the performance of the system, an objective labeling of similarity is needed as judge. For this purpose, the course credit reduction feature will be used. At NTNU, each course has a list of courses deemed so similar, that students who have passed the course will have their gained course credit reduced if they pass any of the courses on the list. This is done in order to prevent students from obtaining easy credit

using knowledge from courses they have already passed. When analyzing the results of the human-guided filtering model on a given course, the system will be judged based on how many courses in the credit reduction list of the given course are present in the set of retrieved courses.

4.3 Data Quality

This section will present some flaws in the dataset that one should be aware of when working with it.

4.3.1 Lacking Descriptions

While most courses at NTNU have extensive descriptions, some do not. In some cases, descriptions are short or non-existing. This is problematic in cases where two courses have a credit reduction relation, but one or both of the courses have lacking descriptions. Such scenarios will likely result in a false negative, where while the courses are similar, the assessment system do not have the necessary grounds to judge them similar. Table 4.3 shows an example where this is the case. Here, neither courses have any description beside their name.

Code	PPU4241	PPU4941
Name	Education: Teaching English - Full Time	Education: Teaching English - Teaching Practise Excluded
Academic content	—	—
Learning outcome	—	—
Department	Department of Teacher Education	Department of Teacher Education

Table 4.3: Similar courses with lacking descriptions.

4.3.2 Missing Credit Reductions

The credit reduction system, while widespread, does not extend to all courses at NTNU. Some courses that clearly have overlapping curriculum, does not have a credit reduction relation. This phenomena is especially frequent in courses from the other educational institutions NTNU merged with in 2016, which have yet to be completely processed and adapted into the system. A consequence of this, is that an IR system would be likely to retrieve false positives. Table 4.4 shows an example of two courses where their descriptions suggest they are highly similar, but no credit reduction relation exists.

IMT2282	TDT4186
Operating Systems	Operating Systems
– System calls, processes and threads, how they can be synchronized and how they can communicate. – CPU - scheduling algorithms. – Memory management:Virtual memory, swapping, paging and segmentation. – File systems: Implementation, backup, consistens and performance. – IO systems: Polling, interrupt and DMA. interrupt handlers, drivers, device independant layer, disk systems and timers. – Deadlocks: Detection and recovery, prevention and avoidance. – Virtualization. – Security: Access Control and Malware – Programming in C, – Bash, PowerShell	The topic will establish definitions, principles, frameworks and architectures for modern operating systems. The topic will focus on processes, thread systems, synchronization, CPU scheduling, memory management, file systems, input-output units, deadlock management, multi-processor systems and security. Important examples will be WINDOWS, UNIX, ANDROID & MAC OS.
Department of Information Security and Communication Technology	Department of Computer Science

Table 4.4: Excerpt of courses with similar content, but no credit reduction relation.

4.4 Statistical Overview

During the study year of 2016/2017, NTNU offers 4,611 different courses. Out of these, 857 have a credit reduction relation to one or more courses. Table 4.5 displays an overview of the average length of each textual feature. Because academic content and learning outcome both describe the contents of the course, they have been concatinated into a single feature called *curriculum*. Throughout the thesis, curriculum will be used as a replacement for the above features.

Feature	**Average character count**	**Average word count**
Name	33.53	4.36
Curriculum	1 056.71	150.41

Table 4.5: Statistics on length of textual course features.

5 Model and Implementation

This chapter will present the human-guided filtering model (HGFM). It will cover the purpose of the model, its way of working, as well as the motivation behind the concept. The chapter will then go on to describe an implementation of the system, which will be the focus of the experiments of this thesis.

5.1 The Human-Guided Filtering Model

The human-guided filtering model is a novel way of comparing the objects of large datasets. More specifically, it is designed to solve the one-to-all comparison (OTAC) problem. The intuition behind the model, is the idea that domain experts should be able to quickly reduce the size of the dataset by manually filtering out irrelevant data. This should ideally allow for faster comparison.

Because some data is filtered out of the process before the comparison takes place, the results of the model could possibly be worse than the results of one-to-all comparison with no filtering. A good HGFM implementation will minimize the deterioration of the results, while maximizing the time saved.

The model is divided into two different phases, each with a number of steps.

Indexing Phase

1. Document Preprocessing
2. Clustering
3. Labeling

Comparison Phase

1. Human-Guided Cluster Selection
2. Similarity Measurement

The OTAC problem asks the following question: Given a set of data and an additional document (either in the set or outside), which documents in the dataset are most similar to the input document? In its first phase, HGFM will process the data and divide it into clusters. Each cluster will be given a short human-readable label or description. In the second phase, a document is given for comparison. A domain expert will pick the

clusters that are most likely to be relevant to the document in question. Conventional similarity measurement will then be employed on the contents of the selected clusters.

Each phase and step is mostly independent, allowing for several different implementations depending on the nature of the dataset to be compared against. The following sections will present the parts of the system in detail.

5.1.1 The Indexing Phase

The indexing phase is concerned with preprocessing and clustering the data. It is also tasked with giving each cluster a human-readable label. The phase is independent of the actual comparison, and should only be performed once per dataset. By storing the clusters, an arbitrary number of comparisons can be performed later.

Below is a presentation of the phase's three steps: Document preprocessing, clustering and labeling. Figure 5.1 displays a diagram of the phase.

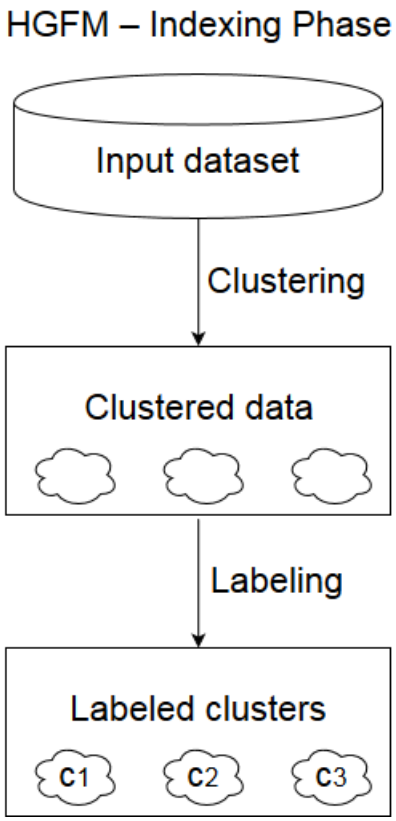

Figure 5.1: The steps of HGFM's indexing phase. Preprocessing is omitted.

Document Preprocessing

First, data should be preprocessed as needed. Depending on the data, this could mean normalization, transformation, feature extraction, etc. Generally, the documents should be preprocessed in such a way that they fit the concrete algorithms used for the subsequent steps.

Clustering

During this step, cluster analysis is performed on the dataset, so that the domain expert later can select the most relevant clusters for further processing. Because the clusters will be inspected by a human, it is essential that the number of clusters are at a manageable level. For the model to work, the human must be able to go through all the clusters. The optimal number of clusters is relative to the resources of whoever is performing the comparison.

Labeling

The last step of the indexing phase involves giving each cluster a human-readable label. Ideally, it should be short and concise, so that a human is quickly able to determine what kind of documents the cluster contains. Erroneously labeled clusters will lead the human to either include irrelevant documents in further steps, or exclude relevant ones. Exclusion of relevant documents would be especially fatal to the overall performance of the system.

Labeling techniques can range from simple to advanced. Intuitively, the features used to cluster the documents in the first place should provide a fitting basis for a label. Some documents will have features which makes summarizing and describing the contents of a cluster trivial. In other cases, it might be necessary to perform actions such as keyword extraction, in order to discover common factors of the documents in the clusters.

5.1.2 The Comparison Phase

In the comparison phase, a document is given as input. A human domain expert is then tasked with selecting the clusters thought to contain data relevant to the input. The input is compared to the contents of the chosen clusters, and the most similar documents are returned.

Below is a presentation of the phase's two steps: Human-guided cluster selection and similarity measurement. Figure 5.2 displays a diagram of the phase.

Human-Guided Cluster Selection

In this step, a human domain expert will have to inspect the input document and available clusters, and decide which clusters will be included in the final step of the process. For this step to be successful, there are two main requirements: First and foremost, the labeling done during the indexing phase must be adequate, so that each

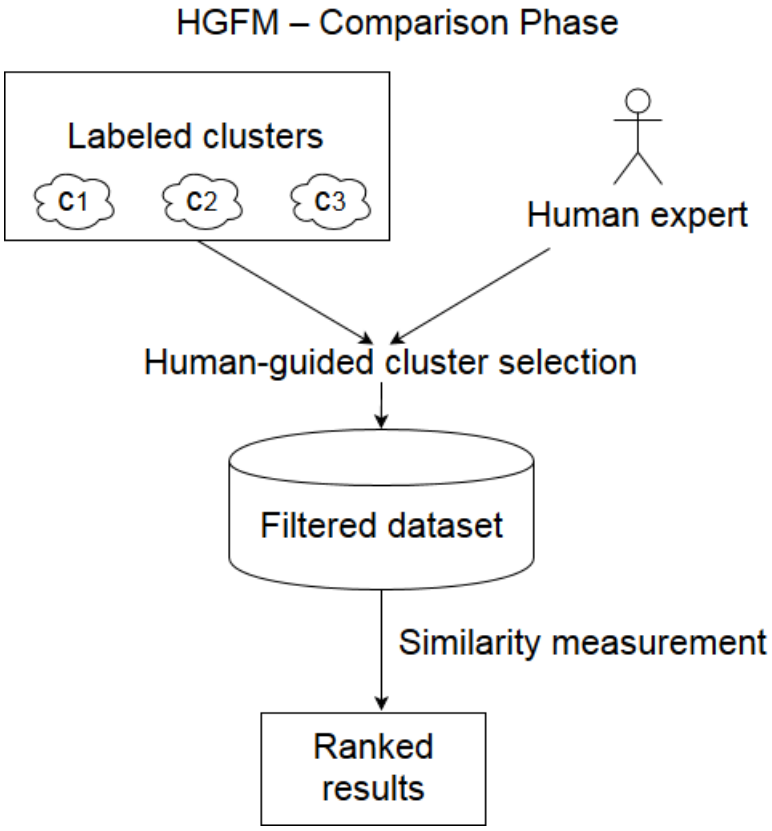

Figure 5.2: The steps of HGFM's comparison phase.

cluster contains what the human expects them to contain. Secondly, the human must be familiar enough with the domain to be able to do the cluster selection swiftly and accurately.

Similarity Measurement

Finally, the documents of the selected clusters will be compared to the input through conventional similarity measures. At this point, the document collection should be significantly smaller than it originally was, allowing for a more time-efficient comparison than if all documents were to be compared against.

The output of this step should be a ranked list of the documents most similar to the input document. This is the final output of the human-guided filtering model.

5.1.3 Remarks

As has already been explained, an implementation of the model is highly dependent on the properties of the data to be compared against. As such, much consideration should be put into choosing fitting algorithms for each step. The general nature of the

model means there are no guarantees on the quality or the efficiency of the comparison. Therefore, experimenting with the workings of each step is crucial to the final outcome.

It should also be noted that for HGFM to perform one-to-all comparison faster than a brute-force approach, the human-guided cluster selection step must be less time-consuming than simply comparing against the documents excluded by that step. If this cannot be achieved, applying the model is meaningless.

5.2 Implementation

In order to test and demonstrate the human-guided filtering model, a system implementing the model has been developed. The project's source code can be found on GitHub[1]. This section will describe the specifics of the implementation, both in regards to the selection of algorithms for each step of the model, and in regards to technological decisions. It also contains a brief manual on how to use the system. Methodology used during development is discussed in Chapter 6.

5.2.1 System Overview

When implementing HGFM, five major choices has to be made, one for each step of the process:

- What kind of preprocessing is needed?
- Which algorithm for cluster analysis should be used?
- How should the clusters be labeled?
- Who should perform the cluster selection?
- Which similarity measurement algorithm(s) should be used to compare the documents?

This section will give an overview of the system by answering these questions.

Preprocessing

Preprocessing is performed on the curriculum feature of each course, in order to extract a list of 30 index-terms. In the implementation, the Python module Natural Language Toolkit (NLTK) is central to this. First, the text is tokenized using NLTK's pre-trained Punkt tokenizer. Stopwords are then removed using a list of stopwords by Porter et al [36]. Terms are stemmed using the Snowball stemmer, and index-terms are finally selected by calculating the tf–idf values for each term and picking the 30 terms with the highest discrimination value. The full preprocessing process is exemplified in Figure 5.3.

[1] https://github.com/AudunLiberg/CourseComparator

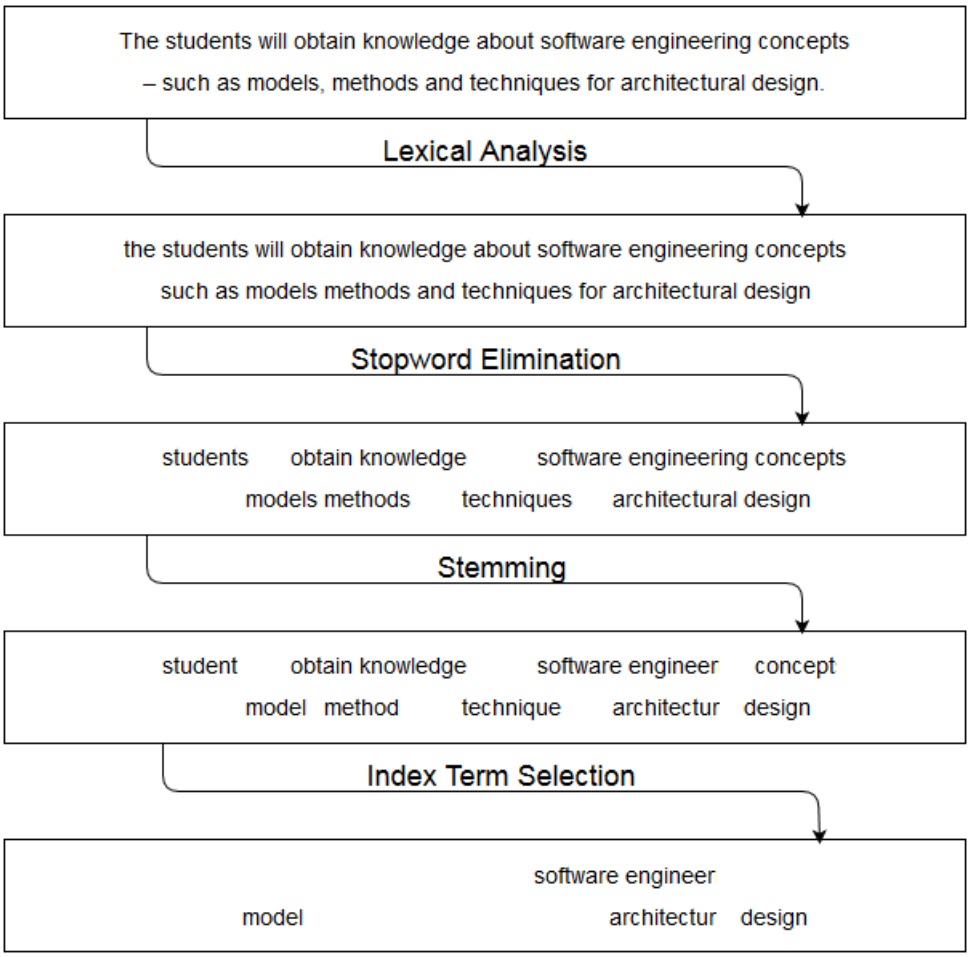

Figure 5.3: Preprocessing of a sentence as done in the HGFM implementation.

Cluster Analysis

In order to examine the importance of the clustering step, two different clustering algorithms have been implemented. The first is a simple baseline clusterer, which does not rely on any machine learning. Instead, it simply clusters courses based on the faculty they belong to.

The second clustering algorithm implemented, is k-means. The implementation utilizes the Python library scikit-learn to achieve this, building on their "Clustering text documents using k-means" approach [37]. Documents are represented in a bag-of-words fashion, and terms are weighted using tf–idf. The algorithm is set to produce 10 clusters, which during testing proved to provide a natural division of courses, without being too much for the human domain expert to handle.

Cluster Labeling

Each of the two clustering algorithms implemented have their own labeling scheme. The baseline clusterer, where each cluster equals a faculty at NTNU, creates labels using the name of the respective faculties. The clusters from the k-means approach, are labeled by identifying the ten most frequent index-terms from the courses inside.

Human Domain Expert

The researcher himself will serve as the domain expert during the experiments. As a longstanding student at NTNU, he is believed to possess adequate knowledge about the various courses at the university. Although sufficient for this preliminary study of the model, more research should be done on the human domain expert's effect on the overall performance of the system. An in-depth study on this is proposed as future work in Section 9.3.2.

Similarity Measures

Similarity will be computed by looking at each course's name and curriculum features. Two sets of similarity measures will be used. One is simple, but computationally fast. The other uses more advanced techniques, which are slower, but should yield better results. These sets will be referred to as the simple and the advanced similarity measure sets, respectively. When retrieving results, courses with a similarity of 0.05 or higher are returned.

The simple set consists of two measures: Jaccard similarity on course names, and the Boolean similarity model on the curriculum feature. The queries of the Boolean model are formed so that two documents are similar if they have at least 15 equal index-terms (out of 30). Otherwise, their similarity is 0. The final similarity is a vote between the measures, with each measure weighting 50 % each.

The advanced set consists of two measures: Levenshtein distance on course names, and VSM-computed similarity on the curriculum feature. The final similarity is a vote between the measures, with each measure weighting 50 % each.

5.2.2 Technology

The system has been implemented using the Python 3 programming language. Table 5.1 lists various libraries the project depends on, as well as their use.

[2]http://www.nltk.org/
[3]http://scikit-learn.org/
[4]https://docs.python.org/3/library/pickle.html
[5]http://docs.python-requests.org/en/master/
[6]https://docs.python.org/3/library/shutil.html

Library	Description
NLTK[2]	An NLP library providing interfaces to various corpora and lexical resources, including stemmers, tokenizers and stopword lists. The system makes use of the *stopwords* and *punkt* packages, which should be downloaded upon first run.
scikit-learn[3]	A machine learning toolkit used for implementing k-means clustering.
pickle[4]	A module for storing ("serializing") objects, allowing the system to cache courses and clusters between runs.
requests[5]	An HTTP library for downloading courses from the web.
shutil[6]	A module for operating on files. Used to manage cache.

Table 5.1: Python libraries used in the implementation of the system.

5.2.3 Usage

The system is started from the command line, by running the file `cc.py`. By providing a course code as argument, a list of the course catalogue's most similar courses will be returned.

```
python cc.py TMA4100.
```

During runtime, the user will be prompted to perform cluster selection. This is depicted in Figure 5.4. For more elaborate instructions on use, including flags which can be used to replicate the thesis' experiments, consult the project's `readme` on GitHub.

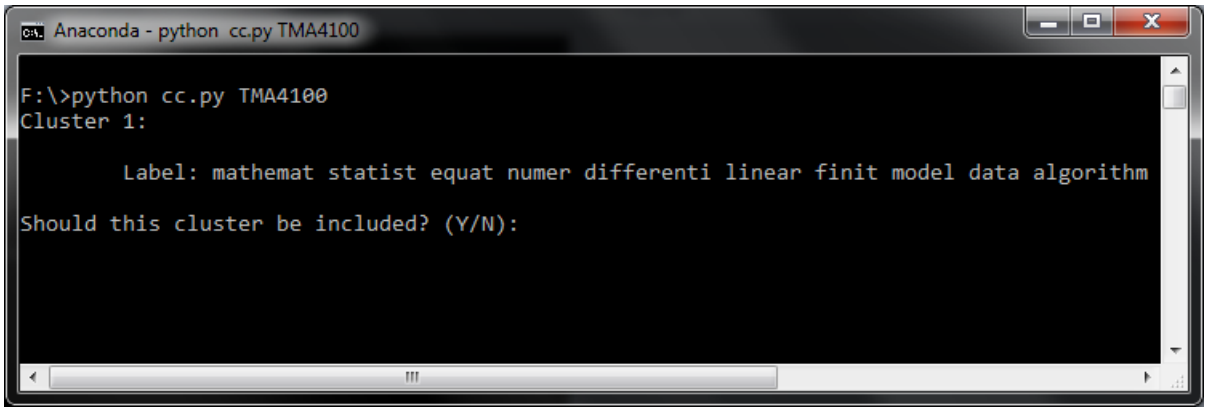

Figure 5.4: Screenshot of cluster selection in practice.

6 Experimental Setup

In order to answer the research questions stipulated in Chapter 1, a series of experiments will be conducted. This chapter will describe the methodology behind the work, as well as the setup for the experiments.

6.1 Preparation

To prevent overfitting during development, courses with credit reduction relations were used to create two sets: A development set with 10 courses, and a validation set with 20 courses. The courses in these sets were chosen at random, and can be found in Appendix A. During the experiments, the system will be tasked with finding the most similar matches to these courses from the full course catalogue of about 4,600 courses.

The validation set was held back until development was done and the experiments began. Results from both sets are provided when describing the experiments.

6.2 Experimental Setup

The experiments are conducted in two rounds: Comparison with and without filtering. This section will describe how these experiments will be carried out, what data will be collected, and how it will be presented.

6.2.1 Experiments With Filtering

In these experiments, the implementation of the human-guided filtering model will be tested. The model consists of two phases, and the experiments will cover results from both. The indexing phase is responsible for clustering the data and giving each cluster a label. During the experiments, the system will be tested using two different clustering algorithms: The baseline clusterer and the k-means clusterer. The distribution of courses withing each cluster will be documented, as will the labels given to each cluster. As mentioned, the researcher will serve as the domain expert for the experiments.

In the comparison phase, the human selects relevant clusters based on the input, and the contents of those clusters are compared to the input. The time taken by the human to pick the clusters, as well as the time taken to do the actual comparison, will be documented.

Finally, similarity will be calculated using algorithms from the *simple* and *advanced* measure sets, as described in the previous chapter. Using the credit reduction list to

assess the relevancy of each retrieved course, precision, recall, F-measure and MAP will be calculated.

6.2.2 Experiments Without Filtering

In order to measure the time saved by employing the human-guided filtering model, comparisons will also be done using the same similarity algorithms as described above, but without human-involvement. This entails using the same dataset with the same preprocessing, but with no filtering. Precision, recall, F-measure and MAP will be calculated as before, in order to observe how the filtering affects the output quality. Time used comparing will also be documented, making it possible to measure if using the model saves time.

6.3 Computer Hardware

Because time-measurement is an important part of the experiments, the specifications of the computer hardware used to run the experiments is documented in Table 6.1. This is done to put the results into context, and to ensure reproducibility.

Hardware	**Model**
Motherboard	GIGABYTE GA-Z77X-D3H Z77 S-1155 ATX IVY
CPU	INTEL CORE I5 3470 3.2GHZ 6MB S-1155 IVY
RAM	CORSAIR 8GB DDR3 XMS3 INTEL I5/I7 PC12800 1600MHZ
SSD	OCZ AGILITY 3 2.5" 120GB SSD SATA/600 MLC

Table 6.1: The computer hardware used to run the experiments.

7 Experiments and Results

This chapter will present the results of the conducted experiments. The first section is concerned with documenting the results of the two implemented approaches to clustering and labeling. The second section is focused on the comparison phase of the process, including human-guided cluster selection and similarity measurement. The third and final section documents the results of similarity measurement without any filtering.

7.1 Clustering and Labeling

Two clustering algorithms were implemented, each with a corresponding labeling scheme. The first is known as the baseline algorithm, the second is k-means clustering.

7.1.1 Baseline Clustering Results

The baseline clusterer simply clusters courses based on their associated faculty, and labels them with the name of that faculty. There are eight faculties responsible for courses at NTNU, in addition to the Education Quality Division, which is responsible for two more courses, summing up to nine clusters. These are summarized in Table 7.1.

#	**Size**	**Label**
1	295	Faculty of Economics and Management
2	758	Faculty of Information Technology and Electrical Engineering
3	248	Faculty of Architecture and Design
4	496	Faculty of Natural Sciences
5	960	Faculty of Engineering
6	501	Faculty of Medicine and Health Sciences
7	672	Faculty of Social and Educational Sciences
8	679	Faculty of Humanities
9	2	Education Quality Division

Table 7.1: The clusters and labels of the baseline clusterer.

7.1.2 *k*-Means Clustering Results

The *k*-means approach divides the dataset into ten different clusters. The resulting Voronoi diagram can be seen in Figure 7.1. Table 7.2 shows the number of courses within each cluster, as well as the associated labels.

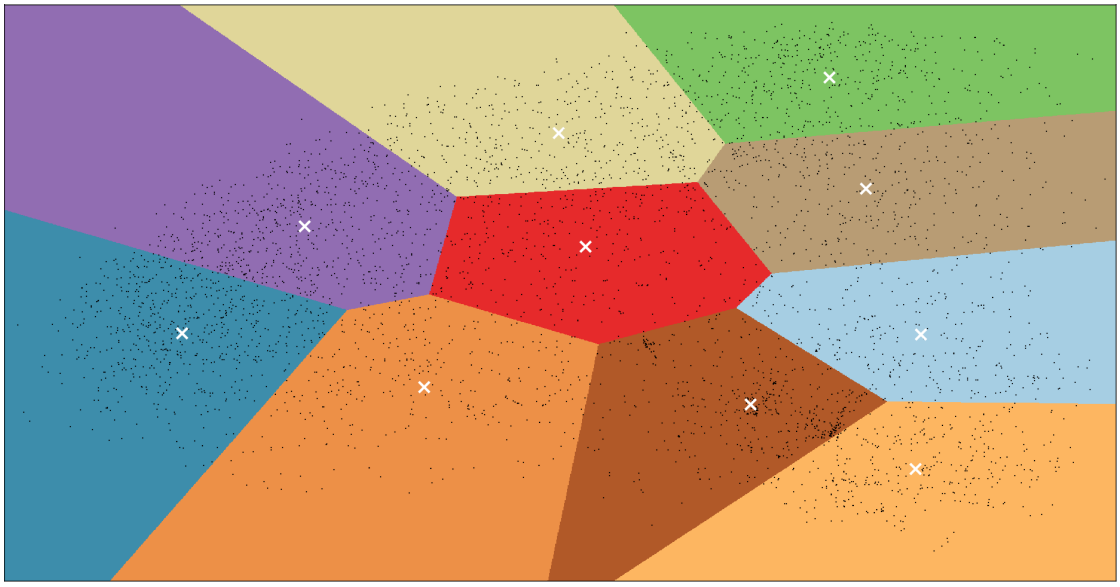

Figure 7.1: Voronoi diagram for the *k*-means clustering algorithm.

#	Size	Label
1	649	design system build architectur læringsutbytt control simul energi product engin
2	886	manag polit social læringsutbytt psycholog busi polici cultur market perspect
3	311	nurs patient care health ill medic diseas clinic profession acut
4	1,014	mathemat statist equat model numer linear method data differenti analysi
5	187	educ teach part teacher time full practis exclud physic theori
6	460	research methodolog scienc qualit scientif candid question thesi disciplin literatur
7	313	project special report mileston work chosen combin manag search literatur
8	273	thesi supervisor must supervis academ page depart scientif report approv
9	106	inform see expert teamwork www.ntnu.no/eit http cours web technolog site
10	412	languag norwegian text grammar european music literari cultur vocabulari histori

Table 7.2: The clusters and labels of the *k*-means clustering algorithm.

7.2 Comparison Phase

This section will present the results of the experiments on the steps in the comparison phase of the model. This includes human-guided cluster selection and similarity measurement. The same experiment has been repeated on both clustering approaches. For statistics on the number of courses filtered for the different clustering approaches, see Table 7.3.

		Development Set		**Validation Set**	
Clusterer	Total Courses	Avg. Filtered	%	Avg. Filtered	%
Baseline	4,611	3,775	0.82	3,577	0.76
k-Means	4,611	3,702	0.8	3,477	0.75

Table 7.3: Average number of courses filtered out before comparing for the different clustering algorithms.

7.2.1 Comparison on Baseline Clusters

The output results of comparison after filtering on the baseline clusters can be seen in Table 7.4. Time spent by the human on cluster selection and time spent on the comparison itself can be seen in Table 7.5.

	Development Set				**Validation Set**			
Comparison Type	Prec.	Recall	F1	MAP	Prec.	Recall	F1	MAP
Simple	0.18	0.35	0.217	0.4	0.07	0.4	0.112	0.358
Advanced	0.08	0.517	0.135	0.429	0.13	0.583	0.194	0.539

Table 7.4: Output results from comparison on courses filtered using the baseline clusters.

	Development Set		**Validation Set**	
Comparison Type	S. Time (s)	C. Time (s)	S. Time (s)	C. Time (s)
Simple	87.3	0.5	179.8	1.3
Advanced	81.4	13.3	162.6	30.6

Table 7.5: Average cluster selection (S.) and comparison (C.) times when using baseline clustering.

7.2.2 Comparison on *k*-Means Clusters

The output results of comparison after filtering on k-means clusters can be seen in Table 7.6. Time spent by the human on cluster selection and time spent on the comparison itself can be seen in Table 7.7.

	Development Set				**Validation Set**			
Comparison Type	Prec.	Recall	F1	MAP	Prec.	Recall	F1	MAP
Simple	0.125	0.557	0.187	0.503	0.167	0.55	0.231	0.6
Advanced	0.14	0.617	0.203	0.583	0.14	0.733	0.214	0.602

Table 7.6: Output results from comparison on courses filtered using the k-means clusters.

	Development Set		**Validation Set**	
Comparison Type	S. Time (s)	C. Time (s)	S. Time (s)	C. Time (s)
Simple	106.2	0.8	187.6	1.4
Advanced	97.8	10.8	183.5	30.6

Table 7.7: Average cluster selection (S.) and comparison (C.) times when using k-means clustering.

7.3 Comparison Phase Without Filtering

This section documents the results of the above experiments without any filtering.

	Development Set				**Validation Set**			
Comparison Type	Prec.	Recall	F1	MAP	Prec.	Recall	F1	MAP
Simple	0.17	0.6	0.237	0.7	0.131	0.667	0.198	0.578
Advanced	0.16	0.717	0.236	0.629	0.15	0.783	0.231	0.714

Table 7.8: Output results from comparison without filtering.

	Development Set	**Validation Set**
Comparison Type	Time (s)	Time (s)
Simple	1.4	3.2
Advanced	57.7	121.0

Table 7.9: Time spent comparing without filtering.

8 Discussion

This chapter will evaluate and discuss the results of the performed experiments, beginning with an analysis of the clustering approaches, labeling and the human-guided cluster selection. Subsequent sections will discuss results related to time and output quality for comparison with and without filtering.

8.1 Clustering

This section will discuss the results of the two approaches to clustering used in the experiments. It will also review the findings of the human-guided cluster selection step.

8.1.1 Clustering Approaches

The baseline clustering approach distributed courses based on their associated faculty. This resulted in eight clusters with sizes ranging from 248 to 960 members. The Education Quality Division represented a ninth cluster, and was a statistical outlier with only two courses. Each cluster was labeled with the name of the faculty it represented.

The k-means clustering approach based on the curriculum feature, was set to produce 10 clusters. These clusters ended up having from 106 to 1,014 members, meaning it had a slightly more uneven distribution than the baseline. Each cluster was labeled using the ten most common index-terms of its members. This revealed some noise in the dataset, namely that some Norwegian words occurs in the course descriptions. For example, "læringsutbytte", meaning *learning outcome*, is commonly used as a heading even in English. Besides this, the index-terms seemed to represent their clusters well. For example, one cluster was labeled `nurs patient care health ill medic diseas clinic profession acu`. As one would expect, this cluster contains courses almost exclusively about nursing and medicine.

Comparing the clusters of the two approaches, we see that their contents are quite varying. The k-means approach has some clusters dedicated to specific types of courses. For example, the cluster labeled `inform see expert teamwork www.ntnu.no/eit http cours web technolog site` consists exclusively of courses from the *Experts in Teamwork* series. These are interdisciplinary courses where students from various programs collaborate on a project. All these courses have similar descriptions, but belong to different faculties. As a consequence, they are placed in different clusters by the baseline approach. A similar example is the cluster labeled `thesi supervisor must supervis academ page depart scientif report approv`, which mainly consists of specialization projects and master theses from various departments. The remaining k-means

clusters were focused on broader topics, such as mathematics (`mathemat statist equat model numer linear method data differenti analys`). This cluster contains mathematical courses not only from The Department of Mathematical Sciences, but also from the fields of economy and engineering, which are governed by different faculties.

8.1.2 Human-Guided Cluster Selection

During the human-guided cluster selection, the human had to select clusters believed to be related to the input course. This resulted in about 75 to 80 % of the courses being filtered out before the actual comparison took place. This goes for both clustering approaches, despite the differences in content pointed out above.

The human found it easier to select clusters labeled with index-terms compared to selecting faculty labels. Index-terms only require general knowledge about which terms relate to which topic. For example, it is intuitive that the course *Mathematical Approximation Methods in Physics* relates most to the cluster labeled `mathemat statist equat model numer linear method data differenti analys`. However, it is not intuitive if the same course belongs to The Faculty of Information Technology and Electrical Engineering (which manages mathematical courses), or The Faculty of Natural Sciences (which manages courses on physics). Detailed domain-knowledge would be needed in this case, something the human (i.e. the researcher) lacked. This is a possible reason for why the results of the baseline clusterer are consistently worse than the results of the k-means clusterer in terms of recall value. This underlines the importance of having a human that is an expert at the domain of the comparison.

The human experienced that cluster selection became easier over time. Tests results were printed between each comparison, meaning the human had the chance to learn from its mistakes. For example, it quickly became clear that courses with the `PPU` suffix in their code belonged in the cluster labeled `educ teach part teacher time full practis exclud physic theor`. In the end, the human was able to include or exclude clusters almost immediately, only using a few seconds per case. The specific clusters from the k-means approach, such as the *Experts in Teamwork* cluster, were especially helpful. The courses in these clusters are seldom related to other courses, meaning they could be excluded in most cases without much thought.

8.2 Performance

This section will discuss the performance of the system with respect to output quality and running time.

8.2.1 Output Evaluation

Without any filtering, the system achieves a recall value of 0.6 with the simple similarity measure set, and 0.717 with the advanced similarity measure set. The precision values are at 0.17 and 0.16 respectively. It is clear that the system generally retrieves too many courses, hence the low precision value. The naive retrieval criterion, where every course

with a similarity of 0.05 or higher are retrieved, is probably the main reason for this. Looking at the MAP values, they are considerably higher than the precision, confirming that relevant documents are ranked high, and that fewer documents should have been retrieved.

When applying filtering, the recall value sinks. This is expected, as removing potentially relevant documents can only affect the recall negatively. With the k-means clustering approach, recall diminishes with 0.05 to 0.1. It sinks even lower with the baseline clustering, being reduced with as much as 0.25 during the experiments (simple comparison, development set). This signifies that incorrect assumptions regarding relevant clusters are being made by the human during cluster selection, resulting in the exclusion of relevant courses. This is especially evident for the baseline clusterer, where the human lacked detailed knowledge about the faculties.

Generally, the results cannot be described as very good, neither with nor without filtering. This is partly due to the similarity measures, which are all fairly simple. Better results could probably have been achieved using more modern, state-of-the-art methods. The dataset itself also has its shortcomings, as documented in Section 4.3. As this has nothing to do with the results of applying the model to the problem, the lackluster results are considered inconsequential.

Finally, it should be noted that the results from the development and validation sets are varying. Given that the number of tests run during the experiments is somewhat low (10 for development, 20 for validation), it is uncertain if the variance is meaningful or coincidental. As the validation set frequently scores a little higher than the development set, there is no reason to suspect overfitting during development.

8.2.2 Time Evaluation

Filter-less comparison with the simple set of similarity measures completes the 10 tests in the development set in 1.4 seconds. The 20 tests in the validation set were completed in 3.2 seconds. Using the advanced set of similarity measures took 57.7 and 121.0 seconds on the same test sets. The implementation of the Levenshtein distance metric was especially time-consuming in practice, explaining the big performance gap between the two measure sets. While slower, the advanced similarity measure set manages to retrieve more relevant courses, as seen above.

With the nine baseline clusters, human-guided cluster selection took around 85 seconds on the tests from the development set, and 165 seconds on the tests from the validation set. On the ten k-means clusters, the selection took around 100 seconds on the tests from the development set, and 185 seconds on the tests from the validation set. As there is one more cluster in the k-means approach, it is expected that selection using this algorithm would take a little longer. As the filtering process excluded up to 80 % of the courses before comparison, comparison times were cut with approximately the same percentage.

Given the above numbers, the human-guided filtering model was not able to save any time. Comparing with the simple similarity measures, comparison was more than 50 times slower. Comparing with the advanced similarity measures, comparison was

about half as fast using the model. While using the model did not save any time in this particular case, the results could have been different under other circumstances. As the time spent on cluster selection is independent of the size of the document collection, the number of documents is crucial to the question if time can be saved. Had there been two or three times as many courses in the course catalogue, using the model with the advanced similarity measures would have meant saving time. This emphasizes the importance of considering the size of your dataset before employing the model.

Furthermore, the cluster selection step relies on the human brain, not the CPU of the computer. As a consequence, time could have been saved if the hardware executing the comparison had been slower. Similarly, it would have been possible to save time if each comparison was more time-consuming. All four similarity measures used during the experiments are fairly simple. More advanced approaches are likely to be slower, creating a greater need for filtering.

9 Conclusion

This thesis has introduced the human-guided filtering model, a novel way of performing one-to-all comparison on large datasets. The model has been implemented and tested on a dataset consisting of course descriptions from The Norwegian University of Science and Technology. Based on the results of the conducted experiments, this chapter will conclude the thesis. Each of the stipulated research questions will be reviewed and attempted answered in accordance with the thesis' findings. Finally, a section is dedicated to suggestions to future work based on this study.

9.1 Review of Research Questions

This section will address the research questions listed in Chapter 1.

RQ1 How much time can be saved by filtering datasets using HGFM?

None of the conducted experiments managed to save any time by employing the model. The human-guided cluster selection proved to be too time-consuming compared to the actual comparison, meaning that even though a large portion of documents were filtered out, time was lost in total. This does, however, not mean that HGFM is without potential. Because the time spent on cluster selection is constant with regards to the size of the document collection, there are multiple cases where time can be saved. Discussion highlighted three such cases, where comparison time will always exceed cluster selection time, given extreme enough situations:

- When the document collection is sufficiently large.
- When the similarity measures are sufficiently complex.
- When the computer hardware is sufficiently slow.

In our time and age, hardware is seldom the bottleneck. Exceedingly large datasets, on the other hand, is a more likely scenario. As discussed in the introduction, the so called Big Data Gap is growing, meaning there are an abundance of larger datasets where the model could prove successful. There are also similarity measures that are more computationally demanding than the simple measures demonstrated in the experiments. Documents with a lot of features are especially resource-demanding to compare. This suggests that HGFM could be well-suited for comparison of high-dimensional objects, such as images or video. The Future Work section at the end of this chapter proposes looking into such use-cases for the model.

In conclusion, there is no theoretical bound to how much time can be saved by using HGFM. With that said, the thesis has shown that saving time is not guaranteed. Therefore, the size of the dataset and the complexity of the similarity measurement has to be taken into consideration before choosing to apply the model in a system.

RQ2 How large portions of the dataset can be filtered out, and how does it affect the output quality?

Experiments showed the dataset being reduced with as much as 80 %. Depending on the amount of clusters selected during human-guided cluster selection, this number will of course vary. This 80 % reduction came with the price of a 0.05 lower recall value at best, and a 0.25 lower value at worst. If such a loss is acceptable, depends on the use of the system being developed.

Important in deciding how much can be filtered out, and how the relevancy of the output is affected, is the human-guided cluster selection. Intuitively, more clusters means more accurate filtering by the human. In the most extreme case, each document is its own cluster, meaning the human would do all of the comparison work manually. This would, of course, be extremely time-consuming. Therefore, when choosing the number of clusters, one should be aware of the trade-off between accurate filtering and time spent on cluster selection.

While it is too early to draw any conclusions based on this study alone, the model does seem to have potential for massive reductions in document collection size. In datasets with millions of documents, being able to filter out anything close to 80 % of the set would lessen the computational burden considerably.

RQ3 How important is the choice of clustering algorithm to the overall results of the model?

The study has tested two different approaches to clustering. The baseline clusterer and the k-means clusterer both managed to filter out about 80 % of the dataset before comparison began. Human-guided cluster selection went fastest when using the baseline clusterer, most likely due to its lower number of clusters.

The baseline approach saw the human struggling with connecting input to the relevant clusters. For this approach, even though approximately the same amount of data was filtered out, fewer relevant courses were retrieved. In other words, relevant courses were removed erroneously. This underlines the importance of not only have a sound clustering algorithm, but also a labeling scheme that communicates the content of each cluster well.

In conclusion, the choice of clustering algorithm is crucial to the overall results of the model, affecting both cluster selection time and output quality in terms of relevance. The number of clusters and how they are labeled are important factors.

9.2 Final Remarks

This study has contributed to big data processing by introducing the human-guided filtering model. The model provides a robust framework for executing human-assisted one-to-all comparison on large datasets. The approach diverges considerably from the conventional ways of processing big data (i.e. with distributed, parallel algorithms, as covered in Chapter 3). With that said, the method could potentially be combined with conventional methods. Employing the DIMSUM algorithm in the final step of the model is one such option.

The high level of filtering seen in this thesis, makes HGFM fall into the ranks of human–computer collaborations where neither the human nor the machine could achieve similar results alone [27] [28] [29]. In line with recent trends in human–computer collaboration, a human-guided machine learning approach to the model could be interesting. This could be achieved by augmenting the clustering step with human assistance.

Limitations of the study includes the dataset used for experiments, which proved to be too small to achieve real time-saving. As such, the full extent of the model's strengths and weaknesses are probably not covered yet. Furthermore, not much emphasis has been placed on the human aspects of the model. This includes the selection, training and impact of the domain experts. Studies where these moments are inspected closer are proposed in the last section of the thesis.

9.3 Future Work

This early research on the human-guided filtering model has been promising. More work should be conducted in order to further explore its potential and limitations. This section will summarize some particular areas of interest.

9.3.1 Experiments on Larger Datasets

The model is most useful on datasets so large that comparison on every object becomes impractical, creating a need for filtering. The NTNU course catalogue has served to create a functional proof of concept, but with approximately 4,600 documents, it cannot be considered big data. In order to further assess the usability of the model, experiments should be conducted on datasets of significantly larger proportions. A suggestion might be the NUS-WIDE image dataset, consisting of 269,648 tagged images [38]. Comparing images should prove more computationally heavy than comparing text. Combined with the large size of the dataset, there should be ample opportunity of proving the model's time-saving capabilities.

9.3.2 Human-Centered Research

By introducing a human into the process, you also introduce several unknown variables. Research should therefore be done to investigate the role of the human in HGFM. What possible biases does the human bring to the table, and how will they influence the

inclusion or exclusion of data? How are the results of the model affected by quick thinking versus slow, methodical thinking? How much domain knowledge is needed in order to do a good job when serving as domain expert? There are many aspects that should be taken into consideration.

9.3.3 Cluster Selection Interfaces

Because an effective cluster selection step is integral to the model, research should be put into discovering good user-interfaces for the process. Not all cluster labels can be represented through a command window. For some datasets, visual labels would be more appropriate than textual ones. In order to make the human's job easier, multiple approaches should be explored.

Bibliography

[1] D. Reinsel J. Gantz. The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East. *IDC IVIEW*, 2012.

[2] D. Ryan. Understanding Digital Marketing. *Kogan Page*, pages 401–402, 2017.

[3] S. Ghemawat J. Dean. MapReduce: Simplified Data Processing on Large Clusters. *Google, Inc.*, 2004.

[4] W. S. Cleveland. Visualization, Statistic Machine Learning, and Massive Data Sets. *Departments of Statistics & Computer Science, Purdue University*, lecture notes.

[5] T. K. Landauer P. W. Foltz, D. Laham. The intelligent essay assessor: Applications to educational technology. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 1(2):939–944, 1999.

[6] K. Papineni, S. Roukos, T. Ward, W. J. Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[7] D. Micol,O. Ferrández, F. Llopis, R. Muñoz. A textual-based similarity approach for efficient and scalable external plagiarism analysis. In *CLEF (Notebook Papers/LABs/Workshops)*, 2010.

[8] C. S. Yang G. Salton, A. Wong. A Vector Space Model for Automatic Indexing. *Information Retrieval and Language Processing*, 1975.

[9] T. Mikolov Q. Le. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[10] H. Schütze C. D. Manning, P. Raghavan. Introduction to Information Retrieval. *Cambridge University Press*, pages 117–119, 2009.

[11] H. B. Low C. L. Tan M. Lan, S. Y. Sung. A comparative study on term weighting schemes for text categorization. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 1, pages 546–551. IEEE, 2005.

[12] C. Buckley G. Salton. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[13] X. Yang C. Shi, C. Xu. Study of tfidf algorithm. *Journal of Computer Applications*, 29(S1):167–170, 2009.

[14] R. B. Yates, B. R. Neto. Modern Information Retrieval: The concepts and technology behind search. Second edition. *Pearson Education Limited*, pages 223–228, 2011.

[15] A. A. Fahmy W. Gomaa. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13), 2013.

[16] P. Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[17] V. Levenshtein I. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[18] R. B. Yates, B. R. Neto. Modern Information Retrieval: The concepts and technology behind search. Second edition. *Pearson Education Limited*, pages 64–66, 2011.

[19] H. Schütze C. D. Manning, P. Raghavan. Introduction to Information Retrieval. *Cambridge University Press*, pages 120–122, 2009.

[20] Document similarity in information retrieval. `https://courses.cs.washington.edu/courses/cse573/12sp/lectures/17-ir.pdf`, 2012. Accessed: 14.12.16.

[21] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[22] S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28:129–137, 1982.

[23] R. Klein F. Aurenhammer. Chp. 5, Voronoi Diagrams. In *Handbook of Computational Geometry*, pages 201–290. Amsterdam, Netherlands: North-Holland, 2000.

[24] H. Schütze C. D. Manning, P. Raghavan. Introduction to Information Retrieval. *Cambridge University Press*, pages 154–155, 2009.

[25] H. Schütze C. D. Manning, P. Raghavan. Introduction to Information Retrieval. *Cambridge University Press*, pages 156–157, 2009.

[26] H. Schütze C. D. Manning, P. Raghavan. Introduction to Information Retrieval. *Cambridge University Press*, pages 159–161, 2009.

[27] P. Krolak, P. Wayne, G. Marble. A man-machine approach toward solving the traveling salesman problem. *Communications of the ACM*, 14(5):327–334, 1971.

[28] J. M. Long, E. A. Irani, D. W. Hunter, J. R. Slagle, J. P. Matts and others. Using a symbiotic man/machine approach to evaluating visual clinical research data. *Journal of medical systems*, 12(5):327–339, 1988.

[29] J. Sargeant, M. W. McGee, S. M. Anderson. A human-computer collaborative approach to the marking of free text answers. *University of Manchester, Department of Computer Science*, 2004.

[30] L. G. Terveen. Overview of human-computer collaboration. *Knowledge-Based Systems*, 8(2-3):67–81, 1995.

[31] C. Holloway, R. Marks II. High dimensional human guided machine learning. *arXiv preprint arXiv:1609.00904*, 2016.

[32] S. Amershi, B. Lee, A. Kapoor, R. Mahajan, B. Christian. Human-guided machine learning for fast and accurate network alarm triage. In *IJCAI*, volume 11, pages 2564–2569, 2011.

[33] Aaai-17 workshop on human-machine collaborative learning. `http://blogs.parc.com/aaai-17/`, 2017.
Accessed: 12.07.17.

[34] F. Bajaber et al. Big data processing systems: state-of-the-art and open challenges. In *Cloud Computing (ICCC), 2015 International Conference on*, pages 1–8. IEEE, 2015.

[35] R. B. Zadeh, G. Carlsson. Dimension independent matrix square using mapreduce. *arXiv preprint arXiv:1304.1467*, 2013.

[36] M. F. Porter. An Algorithm for Suffix Stripping. *Program, 14(3)*, pages 130–137, 1980.

[37] Clustering text documents using k-means. `http://scikit-learn.org/stable/auto_examples/text/document_clustering.html`, scikit-learn.
Accessed: 12.07.17.

[38] Nus-wide. `http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm`, National University of Singapore.
Accessed: 12.07.17.

Appendices

A Courses Used in Experiments

A.1 Development Set

- PPU4729 Teacher Education: Teaching Mathematics, Part II
- IMT3003 Service Architecture Operations
- TFY4305 Nonlinear Dynamics
- SOS8526 Cultural Sociology
- IMT1002 Introduction to engineering - computer science
- KUH2200 Topics in Modern Art: Avant-garde Strategies
- POL3516 The European Union - Rural and Regional Political Economy
- TMA4140 Discrete Mathematics
- PSYPRO4117 Mental Disorders
- NFUT0007 Norwegian for Foreigners, short courses

A.2 Validation Set

- HIST1505 Introduction to Historical Theory and Methods
- TMA4135 Calculus 4D
- FY8410 Light and Force Based Molecular Imaging
- PPU4626 Teacher Education: Teaching Chemistry - Part 1
- PSYPRO4112 Cognitive Psychology I
- PSYPRO4318 Qualitative Research Methodology
- NFUT0005 Norwegian for Foreigners, short courses
- TT3010 Audio Technology and Room Acoustics
- BARN8101 Social Studies of Children and Childhood: Research Perspectives

- BEV8003 Signal Analysis with Matlab in Human Movement Science
- PPU4623 Teacher Education: Teaching Physics - Part 2
- SOS3007 Qualitative Research Methods
- TEP4105 Fluid Mechanics
- TGB4185 Engineering Geology, Basic Course
- KLH3101 Obesity: Epidemiology, Pathophysiology and Consequenses
- NORX1106 Dialectology and History of Language - for Exchange Students
- SOS3513 Welfare State, Family and Integration
- SØK3514 Applied Econometrics
- PSY2016 Personality Psychology II
- KULT3302 Methodes of Qualitative Research Processes II
- EXPH0005 Examen philosophicum for Medical Science and Human Movement Science
- IMT2007 Network Security
- TTT4260 Electronic System Design and Analysis I
- KUH3011 Visual Culture
- POL2022 Petroleum Management, Political Economy and Ethics
- PSY3912 Master Thesis Learning - Brain, Behavior, Environment
- TPK5160 Risk Analysis
- PSYPRO4416 Applied and Clinical Personality Psychology
- IMT4032 Usability and Human Factors in Interaction Design
- FY8908 Quantum Optics