Master's thesis

Tiril Sætersdal Andreassen

Competence mining for better cross-organization communication and cooperation at Bouvet ASA

A feasibility study

Master's thesis in Computer Science
Supervisor: Pieter Jelle Toussaint, IDI
Co-supervisor: Anja Bergby, Bouvet ASA
March 2021

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

Tiril Sætersdal Andreassen

Competence mining for better cross-organization communication and cooperation at Bouvet ASA

A feasibility study

Master’s thesis in Computer Science
Supervisor: Pieter Jelle Toussaint, IDI
Co-supervisor: Anja Bergby, Bouvet ASA
March 2021

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

Abstract

This thesis is a feasibility study that attempts to map the opportunities and impossibilities of a competence mining functionality. The project is motivated by an idea at Bouvet ASA - a Norwegian IT consultancy firm with a distributed, diverse and knowledgeable workforce. The overall idea is a functionality for internally searching for Bouvet's employees' competences based on what they have done and produced, instead of what they themselves say they know through project CVs. The objective of this study is to do a qualitative assessment of the possibilities of mining the employees' competences from natural language documents written by the employees at Bouvet, utilizing natural language processing and machine learning. The study first describes an investigative analysis of different technologies considered as the text mining system, before landing on Google Cloud Platform AutoML Natural Language. The remainder of the study researches, assesses and tests the feasibility of mining competences based on the data available at Bouvet and the technology chosen. This is done by first annotating documents and training a ML model for filtering out relevant texts, and then evaluating the quality of the model. The results show that the quality of the models increases with the number of annotated documents, although there are a relatively limited amount of relevant texts available. The thesis concludes that competence mining is possible, however there are substantial challenges, especially concerning the quality of the data available.

Sammendrag

Denne avhandlingen er en mulighetsstudie som prøver å kartlegge mulighetene og umulighetene for en funksjonalitet for kompetanseutvinning. Prosjektet er motivert av en idé fra Bouvet ASA - et norsk IT-konsulentselskap med distribuert, mangfoldig og kunnskapsrik arbeidsstyrke. Den overordnede ideen er en funksjonalitet for intern søking på Bouvet sine ansattes kompetanser basert på hva de har gjort og produsert, i stedet for hva de selv sier de kan gjennom prosjekt-CVer. Målet med denne studien er å gjøre en kvalitativ vurdering av mulighetene for å utvinne de ansattes kompetanser fra naturlige språkdokumenter skrevet av de ansatte ved Bouvet, ved å bruke språkbehandling (NLP) og maskinlæring. Rapporten beskriver først en undersøkende analyse av forskjellige teknologier vurdert som tekstanalysesystem, som ender med Google Cloud Platform AutoML Natural Language. Resten av studien undersøker, vurderer og tester muligheten for å hente ut kompetanser basert på den dataen som er tilgjengelig hos Bouvet og den valgte teknologien. Dette gjøres ved å først klassifisere dokumenter og trene en ML-modell for filtrering av relevante tekster, og deretter evaluere kvaliteten på modellen. Resultatene viser at kvaliteten på modellene øker med antall klassifiserte dokumenter, selv om det er en relativt begrenset mengde relevante tekster tilgjengelig. Oppgaven konkluderer med at kompetanseutvinning er mulig, men det er store utfordringer, spesielt når det gjelder kvaliteten på dataen som er tilgjengelig.

Preface

This thesis is a master's thesis written as part of the Computer Science program at the Norwegian University of Science and Technology (NTNU) in 2020 and 2021.

I would like to thank NTNU, Bouvet ASA and all the people that have helped me during the process of working with this thesis. Without them, the result would not have been the same.

Acknowledgements:
Pieter Jelle Toussaint - My supervisor at NTNU

Bouvet ASA, represented by:
Anja Bergby - My co-supervisor
Henriette Høyer - Manager of The Communications Department
Andreas Kjerstad - Consultant responsible for back-end at bouvet.no and "Min Side"
Simen Sommerfeldt - Chief Technology Officer (CTO)
Niels Henrik Sandaa Hagen - Consultant in department PIA (platform, insight and analysis)

Sesam, represented by:
Pål Andreassen
Erik Leven

Table of Contents

List of Figures

List of Tables

Abbreviations

AutoML	=	Google Cloud Auto Machine Learning
GCP	=	Google Cloud Platform
IQ	=	Information Quality
KM	=	Knowledge Management
ML	=	Machine Learning
NL-API	=	Google Cloud Natural Language API
NLP	=	Natural Language Processing
POC	=	Proof of Concept
SLR	=	Structured Literature Review
URI	=	Uniform Resource Identifier (Here, the file path to a resource in Cloud Storage)

Chapter 1

Introduction

How can we use competence mining to improve access to competence, meaning people, across an organization? Many organizations have a wide structure, with several regions and employees often not working from the same office. Employees can often ask: "Who (in this company) knows or have worked on the same subject that I am working with now?" In this project, the aim is to see if data already available to the company, such as internal blog post or articles, can be used to answer that question.

This project is done in collaboration with and on behalf of the Norwegian IT consulting company Bouvet ASA. The Communications department at Bouvet is responsible for the development and maintenance of the internal pages "Min Side" (My page) at Bouvet. Here, employees have access to different kinds of internal documents as well as information about other employees in the form of project-CVs. In collaboration with them and the Data Science department at Bouvet, the idea for this project was formulated.

A member of the Data Science department suggested that I could look into a functionality for searching on employees' competences based on what they have done instead of (or in addition to) what they themselves say they know through CVs. What they have done can be represented by projects and technologies they have worked on, and professional and technical articles, blog posts and presentations they have written, and so on. To limit the scope of the project we decided that I am going to assess the possibilities with regards to utilizing natural language processing (NLP) and machine learning (ML) in order to extract competences through subject areas and themes found in natural language documents written by the employees at Bouvet. A vital assumption was then that the author of the document inhabits the competences described in the text.

For this project the data sources are limited to natural language documents, specifically internal articles and news on Bouvet's internal pages in addition to blog-posts on Bouvet Deler. This is done to narrow the scope of the project and because other sources proved difficult or impossible to obtain. Bouvet has no project database and the source code for many of the projects belong to and are stored by the customers. Internal chats and discussion platforms were also considered, but this raised potential privacy and GDPR issues that would have been out of scope for this project. Other potential data sources for future work are discussed in chapter 5.

The project is conducted as a feasibility study, attempting to map the opportunities and impossibilities of the suggested competence mining functionality. The Oxford dictionary defines feasibility study as a noun meaning "An assessment of the practicality of a proposed plan or method" (*Oxford Dictionary on Lexico.com*, n.d.). The aim of the project is therefor not to create or implement a proof of concept (POC), but rather to do a qualitative assessment of whether or not it is possible to extract employees' competences from written human language text based on the data provided by Bouvet and the technology chosen as the text mining system. During the process, I will document and discuss different aspects and challenges that arise and how they are or could be potentially resolved.

Based on the project goal the following research questions were formulated:

RQ1: Is it possible to mine competences of employees from the natural language documents given by Bouvet (articles, news and blog posts) using Google Cloud Platform AutoML?

RQ2 Is the assumption that the author of a text in the collection has the competences described in that text always true?

RQ3 If RQ1 and/or RQ2 fails, What potential changes needs to be implemented for it to work and are those changes possible to do?

Technical theory and background for the project is described in chapter 2, while chapter 3 outline the method and work done to first chose a text mining system and then access, annotate and train testsets of the documents given by Bouvet. In chapter 4, I will view and discuss the results of the analysis of text mining systems, and of the training of the ML models. Finally, in chapter 5, I will discuss the results and touch on the topics of data quality, technical restrictions and more. The final conclusion of the study is given in chapter 6.

Chapter 2

Background and Theory

2.1 Background

2.1.1 Bouvet ASA

Bouvet ASA (*Bouvet Norge*, n.d.) is a Norwegian consulting company within information technology. They consist of approximately 1600 employees divided on 10 offices in Norway and three in Sweden. This means they are a relatively large and quite distributed company. They consist of departments within data science, web development, graphic design and more. "Min Side", meaning "my site/page" is the internal resource pages for employees. Here, employees can find everything from personal employment information and project CVs to the employee handbook, graphic profiles and information about company development and sales. This is also one of the company's main channels for sharing knowledge and experiences internally both through articles and contact information to fellow employees. Every employee is required to have and keep an updated CV in the internal pages. This is used both to find employees with certain competences and by the sales department and project leaders when creating teams for costumer projects. This project aims to test if it is possible to extract and possibly confirm employee competences using what the employees have done and produced instead of what they say they know through CVs. What they have done can mean articles they have written, roles and responsibilities they have had, technical code and projects they have worked on etc.

2.1.2 Sesam

Sesam (*Sesam - Democratising Data*, n.d.) is a data integration platform and subsidiary of Bouvet. It was originally a part of Bouvet and was separated and established as a corporate spin-off in 2014. It functions as a hub connected to several of Bouvet's systems, sending data between them.

2.2 Theory

2.2.1 Knowledge Management and Competence Mining

Knowledge management (KM) refers to the management of the knowledge existing within an organization. Macintosh (1999) defined that "Knowledge management involves the identification and analysis of available and required knowledge assets and knowledge asset related processes, and the subsequent planning and control of actions to develop both the assets and the processes so as to fulfil organizational objectives." Through a study in precisely the definition of knowledge management, Girard and Girard (2015) used this definition along with over 100 other existing definitions of KM to formulate a more general definition based on the most common verbs and nouns: "Knowledge Management is the process of creating, sharing, using and managing the knowledge and information of an organization."

Based on these definitions, we see that competence mining is a part of knowledge management. Rodrigues, Oliveira, and Souza (2004), found during my search for relevant work, loosely defines competence mining as an organisations "[...] need to know what they know (internal competences), and who the owners of this knowledge are." Gartner glossary defines competence mining, or skill mining more detailed as "A knowledge management (KM) functionality that automatically identifies the skills of knowledge workers by analyzing past behavior. This behavior may be implicit (e.g., looking for recurring concepts in documents that the worker has produced), or explicit (e.g., a worker's willingness and ability to answer a question in the past). Skill mining enables users to identify who in their enterprise has the expertise to address specific questions or problems" (*Definition of Skill Mining - Gartner Information Technology Glossary*, n.d.). Based on this definition, this project aims to assess the possibility of automatically identifying the skills or competences of employees by analysing their implicit behavior.

2.2.2 Natural Language Processing and Machine Learning

Text analysis, or text mining, refers to the process of deriving or extracting new information out of unstructured data. The unstructured data are often collections of different types of documents, files and text as opposed to the structured, homogeneous data in a relational database (Feldman & Sanger, 2006). The new information derived from the texts is structured and machine-readable, derby "create[ing] structured data out of free text content" (*What is Text Analysis*, n.d.). This is done by utilizing different text analysis tools. Text analysis is important and highly relevant in today's society where data and information are some of the biggest resources available. If a company or organization is not able to organize and manage their unstructured data, then that data is virtually unusable (*Text Mining: The Beginner's Guide*, n.d.).

In order for a text analysis system to be useful in this project, it needs to be adaptable and changeable. It needs to provide the understanding and processing of natural language text, and convey the meaning/semantics of text. Specifically, the system needs to provide multi-label classification of natural language text. The system also potentially needs to work successfully within a specific domain and/or language.

The field of natural language processing (NLP) revolves around the communication between computers and human language. NLP is considered a subfield of computer science, information engineering, linguistics and artificial intelligence (Razno, 2019). The idea is to program computers to be able to perform NLP tasks that process, analyse and "understand" unstructured human natural language. There are several different types of NLP tasks. Cambria and White (2014) states that "Since its inception in 1950s, NLP research has been focusing on tasks such as machine translation, information retrieval, text summarization, question answering, information extraction, topic modeling, and more recently, opinion mining." In this project, I am most interested in a variation of information extraction and topic modeling called text classification. According to Razno (2019), "Text classification is one of the most important and typical task in supervised machine learning". The process revolves around assigning predefined labels to natural language text, thereby classifying or categorising the texts into different categorisations based on the content of the texts. How text classification is done has changed over time, from the simpler method of keyword spotting, to the probability-based method of lexical affinity, to the more modern method of statistical NLP. Cambria and White (2014) describes how statistical classification NLP utilizes ML like this: "By feeding a large training corpus of annotated texts to a machine-learning algorithm, it is possible for the system to not only learn the valence of keywords (as in the keyword spotting approach), but also to take into account the valence of other arbitrary keywords (like lexical affinity), punctuation, and word co-occurrence frequencies." In this project I will follow this method by assessing the possibility of custom-annotating the data given by Bouvet (my corpus) and training it using the machine-learning algorithm provided by Google Cloud AutoML.

Jordan and Mitchell (2015) stated that "Machine learning addresses the question of how to build computers that improve automatically through experience. It is one of today's most rapidly growing technical fields, lying at the intersection of computer science and statistics, and at the core of artificial intelligence and data science". Alpaydin (2020) echoes in the statement that "The goal of machine learning is to program computers to use example data or past experience to solve a given problem. Machine learning underlies such exciting new technologies as self-driving cars, speech recognition, and translation applications".

2.2.3 Data Quality

In this project, I will be working with data given by Bouvet in order to assess the opportunities and impossibilities of the competence mining feature described in chapter 1. The results of the feasibility study largely depends on the quality of the data I receive. Many researchers have proposed different definitions for information quality (IQ), and have come to the conclusion that information quality is a multi-dimensional concept (Alter, 2006; Fehrenbacher & Helfert, 2012; Ge & Helfert, 2007; Wang & Strong, 1996). Several of these researchers have presented suggestions for these different dimensions and measurements or criteria that belong in the different dimensions. Alter (2006) and Wang and Strong (1996) proposes many of the same categories for the dimensions: Intrinsic Quality of Information, Accessibility of Information, Contextual Quality of Information and Representational Quality of Information. Intrinsic IQ refers to the quality of the data outside of the context, contextual IQ refers to the IQ requirements within the context, representational IQ refers to the utilization of the data and accessibility refers to the accessibility of the data (Ge & Helfert, 2007; Wang & Strong, 1996). Table 2.1 list the dimensions that may be considered in this project.

Table 2.1: Information Quality Dimensions

Dimension	**Description**
Intrinsic Quality of Information	
Accuracy	The extent to which the information is correct and error free
Precision	The fineness of detail in expressing the information
Age	The amount of time between when the source information was created and when the final information is used
Contextual Quality of Information	
Completeness	The information contains all the relevant facts that are needed to take a decision or take action
Timeliness	The extent to which the information's age and availability are appropriate for the task and the user
Relevance	The extent to which the information makes a difference in the context of use
Appropriate amount of Data	The extent to which the amount of data are appropriate for the task and the user
Representational Quality of Information	
Consistency of Representation	The same information is always represented in the same way

Table 2.1 – continued

Dimension	**Description**
Conciseness	Information is to the point and compactly presented
Ease of understanding	The extent to which the information is clear, and unambiguous
Accessibility of Information	
Access control	The extent to which only authorized users have access to the information
Access time	The average time to retrieve information

2.3 Relevant Work

While searching for other relevant work within competence mining, I utilized some of the aspects of structured literature review (SLR), specifically steps from the second phase (Kofod-Petersen, 2014). The second phase concerns the actual search and review of the literature. In this step, I determined the sources where I would search and how to search them. I chose to use **ACM digital library** (*ACM Digital Library*, n.d.) and **IEEE Xplore** (*IEEE Xplore*, n.d.) as my sources because they are two of the largest sources of technical journals, conference proceedings, books and other articles in the world. In order to get as many results as possible about competence mining specifically, I eluded searches only containing the terms "competence" and "mining" separately, and included searches on synonyms of the complete phrase "competence mining". Table 2.2 depicts the different search terms (or phrases) I search for on the selected sources.

Table 2.2: Relevant work: Search terms

Search Term 1	*Competence mining*

Search Term 2	*Competence matching*
Search Term 3	*Skill mining*
Search Term 4	*Skill matching*

In order to initially filter down the search-results, I utilized some of the general removal criteria points given by SLR (Kofod-Petersen, 2014):

1. Duplicates (keep the highest ranking source)
2. The same study published in different sources (keep the highest ranking source)
3. Studies published before a certain date (or even after): (keep sources published between 1.1.2000 - d.d.)

The results of the searches generally suggested that there have not been carried out many studies on this theme. Out of a substantial amount of immediate search results, only a few could be considered to have researched and/or attempted to test or implement some of the same functionality that I am assessing in this project. One of them is Rodrigues, Oliveira, and de Souza (2006); Rodrigues et al. (2004). This was a study conducted over several years at the Federal University of Rio de Janeiro, Brazil. Rodrigues et al. (2004) explained the goal of the study by stating that "This work aims at mapping researcher's competence in his/her publications, using methods and techniques applied to the area of knowledge discovery from texts or, as commonly entitled, text mining. This discovery is essential for a scientific organisation to be able to discover which areas of knowledge have active professionals, as well as how internal knowledge is divided". I have unfortunately not been able to find an article that declares the final results of the prolonged study and can therefor not say whether or not their theory worked or was ever deployed for use. Another article describing relevant work is Jin, Li, Zhang, Xu, and Chen (2020). This study aimed at researching the use of competence mining aspects in order to create a decision support model for team formation. Curran and Gualtieri (2016) also briefly mentions a similar functionality as "use search to discover who the internal experts are by leveraging NLP to analyze their “footprint” (authorship, contributions, etc.) in documents" as a part of a brief on cognitive search in enterprises.

Chapter 3

Method

As mentioned in the Introduction chapter (chapter 1), the main objective of this project is to research whether or not it is possible to mine competences of employees based on the data available at Bouvet and the technology used. That is, to make a qualitative assessment and not a Proof of concept (POC). Because of this, not all processes or test are fully completed if earlier results or challenges during the process proves them unnecessary or impossible. In this chapter, I first describe the process of analysing different text mining systems to be potentially used. I then describe how and what data is accessed from Bouvet and finally, I train and evaluate machine learning models in order to mine competences from the data.

3.1 Analysis of Text Mining Systems

In order to analyse and compare relevant text analysis systems, I worked in three steps. First, I collected information about several potential candidate systems, I then filtered them down to the three most relevant, and finally I analysed them based on a list of questions I formulated.

3.1.1 Gather Information on Systems

To find relevant candidate text mining systems to explore, I first gathered information about several potential systems by searching for "Free open source text analysis systems" on Google search. As a result of the search, I found several websites and reviews that listed and described multiple such systems. I took basis in these lists: Maffeo (2019); PATresearch (2019), and further explored the text analysis systems described. Then, I collected and wrote down the systems that were most relevant to the project with the analysis they provided and how easily available information about the system were. See Table 4.1 in chapter 4 for the full list of the systems. Finally, I filtered the list down to three text mining systems for further analysis by using the criteria listed in Figure 3.1 below. The criteria are meant to filter out the systems that are potentially most relevant for use in an intelligent search system. Table 4.1 also describes the reasons for discarding all the systems that were not chosen, based on the criteria.

3.1.2 Filter Systems

- The system should not be too old and/or outdated.
- The system should not be a "hobby project".
- The system should be widely used and tested.
- The text analysis should provide some form of Natural Language Processing (NLP) like semantic analysis, information extraction and/or content classification that will be able to convey the meaning of text.
- The system should be open source or otherwise possible to modify and use.
- The system should be free or pricewise manageable for a small to large company.
- The system should have plenty of information and documentation openly available.
- The system should be possible to test in practice.

Figure 3.1: Criteria used for filtering down the systems.

The three final systems I ended up with were Apache OpenNLP, Natural Language Toolkit and Google Cloud Natural Language. These were also chosen because they represent different technologies and use cases within NLP. When I had filtered the list down to the three systems, the main analysis process started.

3.1.3 Analyse Systems

I formulated a list of questions and points to ask each system as the basis of the analysis and comparison of the systems. The questions are meant to provide insight into the use of each system and to show if and how they could be used for competence mining of natural language texts. They should especially reveal the adaptability of each system since this is crucial. In addition, the questions function as a double check that the text analysis system fits with the criteria previously listed in Figure 3.1. The questions are listed below in Figure 3.2. The questions were then answered for each system separately by reading documentation about each one. This included everything from product webpages, source codes, READMEs and other developer documentations to published books, technical blogs and news posts. Some of the systems also had light version demos, lectures and other more hands on resources available. The answers to the questions for each system are described in Table 4.2 in chapter 4. Finally, I used the summed up information in Table 4.2 as the basis for the discussion, analysis and comparison of the three systems with regards to being used in this project.

1. Meta information (When was it started?, How was it developed? etc.)
2. What is the system used for? What type of analysis does it provide?
3. Is the system meant for any specific technologies? If so, which?
4. Is the system meant for any specific domains? If so, which?
5. What/which natural language(s) is the system built for? Is it possible to change language?
6. What is the input and output of the system?
7. How easy is it to make changes and modifications to the system?
8. What/which part(s) of the system is/are potentially most relevant concerning competence mining?

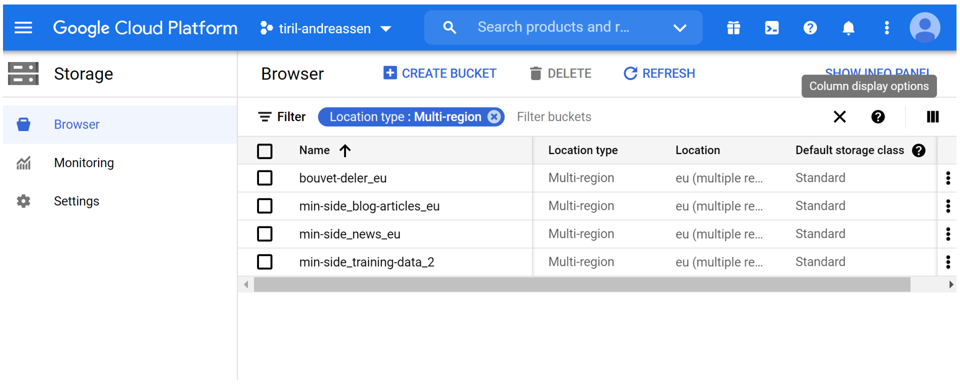
Figure 3.2: Questions for each of the three final text mining systems.

Based on the analysis shown in chapter 4, Google Cloud Natural Language, specifically Google Cloud AutoML, was selected to be used together with Google Cloud Storage.

3.1.4 Google Cloud Platform

Google Cloud Platform and Storage

Google Cloud Platform is Google's cloud service and thereby one of the largest and most utilized cloud services in the world. It was first launched in April of 2008, and advertised that it would be "a developer tool that enables you to run your web applications on Google's infrastructure" (McDonald, 2008). Since then, Google has continued developing the service and added a multitude of different cloud products like Cloud SQL, BigQuery and API Analytics (*Products and Services*, n.d.). The first of these products to be deployed was Google Cloud Storage, a cloud service for storing large amounts of data (Jiang, 2010). Figure 3.3 shows the view of Google Cloud Storage with the storage-buckets used in this project.



Name	Location type	Location	Default storage class
bouvet-deler_eu	Multi-region	eu (multiple re...	Standard
min-side_blog-articles_eu	Multi-region	eu (multiple re...	Standard
min-side_news_eu	Multi-region	eu (multiple re...	Standard
min-side_training-data_2	Multi-region	eu (multiple re...	Standard

Figure 3.3: The view of Google Cloud Storage with the storage-buckets used in this project.

Google Cloud Natural Language

The Google Cloud Natural Language systems are made and distributed by Google LLC and are part of the Google Cloud platform. The systems can therefor be combined with Cloud Storage and other Google Cloud services (*Google Cloud Natural Language*, n.d.). Cloud offers two different natural language systems; Cloud Natural Language API (hereafter NL-API) (*Cloud Natural Language API documentation*, n.d.) and Cloud AutoML Natural Language (hereafter AutoML) (*AutoML Natural Language documentation*, n.d.). The systems both provide natural language understanding technologies reveling the structure and meaning of text. The difference is that the systems use different machine learning models to do this. NL-API uses pre-trained ML models that are backed by Googles huge amount of data, whereas AutoML gives users the chance to build, train and deploy custom ML models. AutoML can therefor be particularly interesting if one needs very domain specific analysis. The systems provide sentiment analysis, entity analysis, syntax analysis and entity sentiment analysis as well as content classification. For this project, classification is most relevant seeing as I want to classify documents in regards to what competences they reference. The pre-trained ML models used by NL-API utilizes the same deep ML technology that is used by both Google Search and Google Assistant (*Google Cloud Natural Language*, n.d.). Google also provides both API client libraries and cloud client libraries for easier programmatic access and integration with their cloud services (*API Client Libraries*, n.d.; **?**). AutoML classification provides built in support for several languages, including English and Swedish, whereas NL-API classification only supports English initially. Cloud Natural Language API was made generally available (GA) in 2016 (Craft, 2016) with the latest stable release in March 2020 (*Release notes | Cloud Natural Language API*, n.d.). AutoML Natural Language was made GA in 2019 (Liu, 2019) with the latest stable release in April 2020 (*Release notes | AutoML Natural Language*, n.d.).

3.2 Access and Modify Data

Bouvet uses Sesam as their data integration platform. Sesam originally started as a department within Bouvet, but split and became a stand-alone subsidiary. In order to get access to data from Bouvet both Sesam and Bouvet's Communications department had to be involved. Before I could get access to documents from Bouvet, they had to be somewhat modified and sent from the Communications department, through Sesam and finally to my project on Google Cloud Platform.

3.2.1 Access Data

1. First, a Google Cloud Platform license was established and a project was created. This was done by the IT-department at Bouvet.

2. I then created storage buckets for each datatype in Google Cloud Storage. A list of the buckets and the data stored in them can be found in Table 3.1. In order to be accessed by AutoML Natural Language the buckets must be in either the "us-central1" region or "eu"

multi-region location. For this project the "eu" region was chosen due to the project owner being a Norwegian company. The buckets requirements for both the buckets containing the documents and the annotated training-data were then:

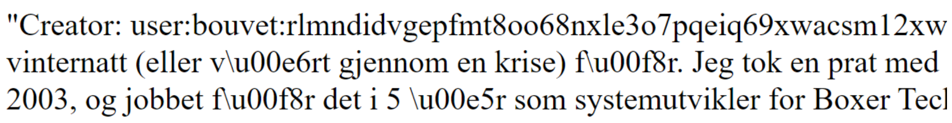
- Location type: Multi-region
- Location: eu
- Storage class: Standard

Table 3.1: Storage Buckets

Bucket	**Content**
bouvet-deler_eu	Public articles from the blog Bouvet Deler
min-side_blog-articles_eu	Internal blog articles
min-side_news_eu	Internal news articles
min-side_training-data_2	csv files containing the custom annotated training data

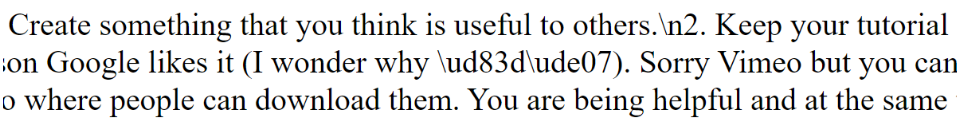
3. In order for Sesam to be able to upload data to GCP, I created a service account for Sesam with writing-rights. This was done following the steps for creating a service account: *Getting started with authentication | Authentication* (n.d.).

4. The documents were slightly modified and sent from the communications department through Sesam and directly into the corresponding buckets in Google Storage. The data was originally stored as Json files and had to be changed to content_type "text/html" in order to make them more readable and remove Json specifics such as "{}". The files were also UTF-8 encoded however, the microservice used by Sesam to upload data to GCP only have a parameter for setting the metadata field "content_type". This means that most of the documents do not handle Norwegian letters *ÆØÅ* and some other special characters. In the .txt files, some of the metadata fields within the texts were removed, leaving only creator, title and the text for readability and relevance. The creator is usually the Bouvet email address of the creator of the document. The buckets consist of mostly Norwegian documents, but also some English and Swedish documents. See Figure 3.4 for examples from the document texts. The documents I received were all the document available within each datatype up to December 2020.



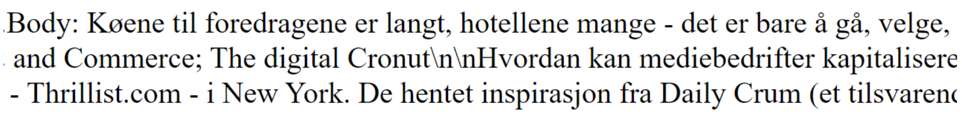
"Creator: user:bouvet:rlmndidvgepfmt8oo68nxle3o7pqeiq69xwacsm12xw
vinternatt (eller v\u00e6rt gjennom en krise) f\u00f8r. Jeg tok en prat med
2003, og jobbet f\u00f8r det i 5 \u00e5r som systemutvikler for Boxer Tecl

(a) Example of parts of a Norwegian document. Here, the creator field is also useless.



Create something that you think is useful to others.\n2. Keep your tutorial
on Google likes it (I wonder why \ud83d\ude07). Sorry Vimeo but you can
o where people can download them. You are being helpful and at the same

(b) Example of parts of an English document.

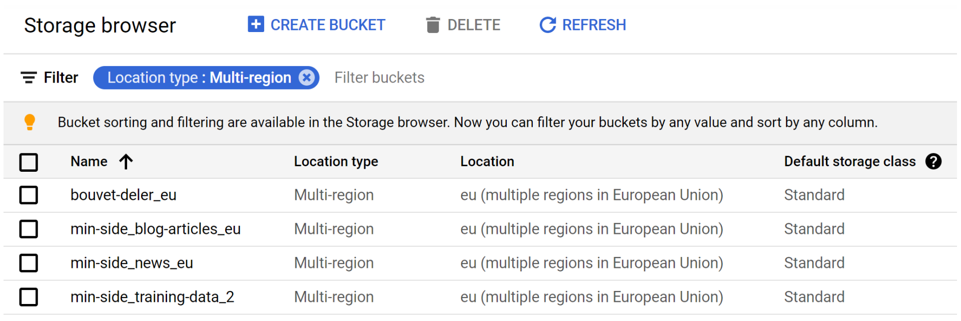


Body: Køene til foredragene er langt, hotellene mange - det er bare å gå, velge,
and Commerce; The digital Cronut\n\nHvordan kan mediebedrifter kapitalisere
- Thrillist.com - i New York. De hentet inspirasjon fra Daily Crum (et tilsvarenc

(c) Example of parts of a Norwegian document with æøå.

Figure 3.4: Examples of extracts of the texts from different documents from the "Bouvet Deler" bucket.

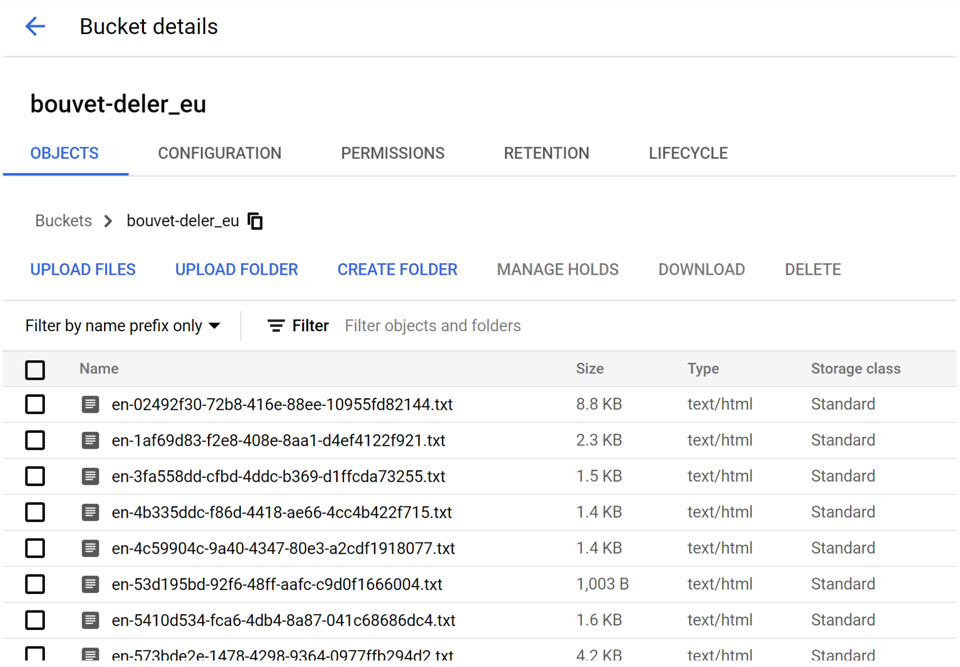
Figure 3.5 and Figure 3.6 shows what two views of the buckets and documents in Google Cloud Storage looks like with examples of filenames for the documents.



Bucket sorting and filtering are available in the Storage browser. Now you can filter your buckets by any value and sort by any column.

Name	Location type	Location	Default storage class
bouvet-deler_eu	Multi-region	eu (multiple regions in European Union)	Standard
min-side_blog-articles_eu	Multi-region	eu (multiple regions in European Union)	Standard
min-side_news_eu	Multi-region	eu (multiple regions in European Union)	Standard
min-side_training-data_2	Multi-region	eu (multiple regions in European Union)	Standard

Figure 3.5: The view of the buckets in Google Cloud Storage.


Bucket details

bouvet-deler_eu

OBJECTS CONFIGURATION PERMISSIONS RETENTION LIFECYCLE

Buckets > bouvet-deler_eu

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER MANAGE HOLDS DOWNLOAD DELETE

Filter by name prefix only Filter Filter objects and folders

Name	Size	Type	Storage class
en-02492f30-72b8-416e-88ee-10955fd82144.txt	8.8 KB	text/html	Standard
en-1af69d83-f2e8-408e-8aa1-d4ef4122f921.txt	2.3 KB	text/html	Standard
en-3fa558dd-cfbd-4ddc-b369-d1ffcda73255.txt	1.5 KB	text/html	Standard
en-4b335ddc-f86d-4418-ae66-4cc4b422f715.txt	1.4 KB	text/html	Standard
en-4c59904c-9a40-4347-80e3-a2cdf1918077.txt	1.4 KB	text/html	Standard
en-53d195bd-92f6-48ff-aafc-c9d0f1666004.txt	1,003 B	text/html	Standard
en-5410d534-fca6-4db4-8a87-041c68686dc4.txt	1.6 KB	text/html	Standard
en-573bde2e-1478-4298-9364-0977ffb294d2.txt	4.2 KB	text/html	Standard

Figure 3.6: The view of the bucket for "Bouvet Deler" in Google Cloud Storage.

3.3 Annotate, Train and Validate ML Models

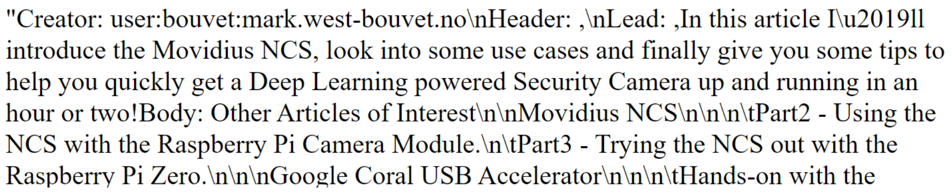
For training and creating ML models, AutoML was chosen over NL-API. This is because AutoML gives the opportunity to create custom annotated training data and because it has a built in support for both English and Swedish when training classification. AutoML gives the opportunity to train models for single-label classification or multi-label classification, both of which are relevant to this project. I this project I only use the Web UI provided by Google Cloud platform.

3.3.1 Annotate and Train AutoML Models

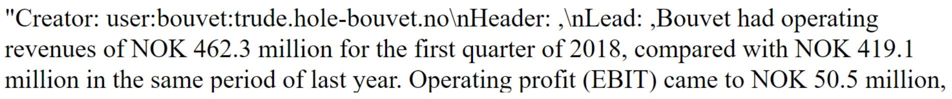
I started the training process by reading and annotating a number of the different documents received in Google Storage. Quickly, many of the documents turned out to not be relevant for this project, so the annotation was done in two steps in order to train two models. One to filter out relevant documents and one to classify competences. I first read and labeled documents based on relevancy. Secondly, I annotated the documents I had deemed relevant with different competences, and finally, I trained and validated the filtering model.

1. First, a filtering dataset was made for the filtering model. This is a .csv file called "*Filtering_competences.csv*", containing a set of the documents to train the model. In order to make this file, I read some of all the documents available and annotated them to either "In_domain" or "Out_of_domain". Any document that were a professional article or in some other way talked and informed about a technology, IT-process or method, I labeled "In_domain". Any document that did not reference a technological or IT-relevant competence I labeled "Out_of_domain". This was done based only on the pure text, regardless of whether the creator field was filled in the correct format. See Figure 3.7 for examples of documents and how I labeled them. For readability, all the examples are English documents. See also Figure 3.8 for an extract of the annotated filter data as shown in "*Filtering_competences.csv*". The first column is the URI for the document (file path to the resource in Cloud Storage) and the second is the label given to that document. This file was then used to train a single-label classification model meant as a filtering process before the actual classification of competences was to be done on the documents deemed "In_domain".

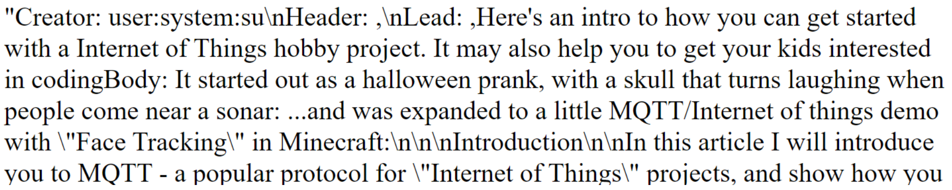
2. During the process of annotating documents "In_domain" or "Out_of_domain", I also wrote down all documents deemed relevant for competence mining (labeled "In_domain") in a second .csv file named "*Classification_competences.csv*". Here, I filled column two, three and so on with competences mentioned or written about in the documents. Each document may have more than one label this time, and the model would be trained as a multi-label classification model. At first, the different competence-labels were the labels I first associated with that text; often words or phrases taken directly from the text. The idea was to then go over all the labels written down for all the documents and gather them in groups based on similarities and correlations, limiting the number of different labels in total. Because this projects' focus is on testing the possibilities for this to work and not make a complete POC, the number of labels were limited even further by only focusing on competences relevant for one department within Bouvet. This was done to minimize the scope of the test. The department *Øst Tech 2* consists of subdivisions within digitization, cloud development, interaction and security. On Bouvet's internal pages, each subdivision has a list of labels named "Ting vi kan" literally meaning "Stuff we know". This, along with a list of more technical competences provided by the head of the department, was meant to act as a guide when formulating the final labels. See Figure 3.9 for the list given


"Creator: user:bouvet:mark.west-bouvet.no\nHeader: ,\nLead: ,In this article I\u2019ll introduce the Movidius NCS, look into some use cases and finally give you some tips to help you quickly get a Deep Learning powered Security Camera up and running in an hour or two!Body: Other Articles of Interest\n\nMovidius NCS\n\n\n\tPart2 - Using the NCS with the Raspberry Pi Camera Module.\n\tPart3 - Trying the NCS out with the Raspberry Pi Zero.\n\n\nGoogle Coral USB Accelerator\n\n\n\tHands-on with the

(a) Example of an English document labeled "In_domain".


"Creator: user:bouvet:trude.hole-bouvet.no\nHeader: ,\nLead: ,Bouvet had operating revenues of NOK 462.3 million for the first quarter of 2018, compared with NOK 419.1 million in the same period of last year. Operating profit (EBIT) came to NOK 50.5 million,

(b) Example of an English document labeled "Out_of_domain".

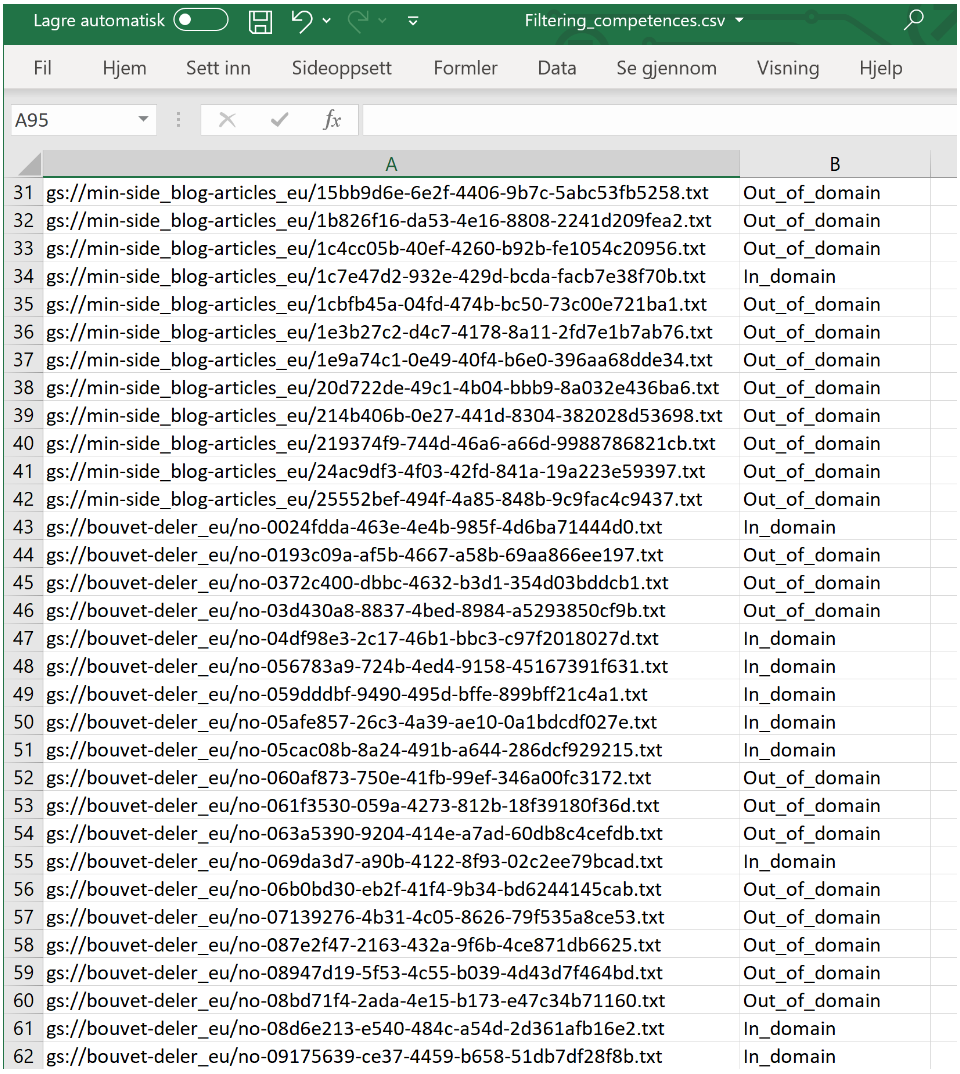

"Creator: user:system:su\nHeader: ,\nLead: ,Here's an intro to how you can get started with a Internet of Things hobby project. It may also help you to get your kids interested in codingBody: It started out as a halloween prank, with a skull that turns laughing when people come near a sonar: ...and was expanded to a little MQTT/Internet of things demo with \"Face Tracking\" in Minecraft:\n\n\nIntroduction\n\nIn this article I will introduce you to MQTT - a popular protocol for \"Internet of Things\" projects, and show how you

(c) Example of an English document labeled "In_domain".

Figure 3.7: Examples of extracts of the texts from different documents and how they are labeled.

by the head of the department.

3. In order to train the model to filter out relevant documents, I uploaded the "*Filtering_competences.csv*" file to the bucket called "*min-side_training-data_2*" and then created a dataset in AutoML where I imported the annotated documents using that file. AutoML then automatically uses 80% of the labeled documents for training the model, 10% for validating and then 10% for testing the model. This process was done twice, first with 69 labeled documents, then with 119 labeled documents. When a AutoML model is trained, Cloud AutoML provides some evaluation statistics of the model based on the 10% of the documents AutoML uses for testing the model. The results of the automatic evaluation of both the two filtering models are shown in chapter 4 and are discussed further in chapter 5. Figure 3.10 shows the import-view in Google Cloud AutoML where the .csv file is imported. Here it is imported to the dataset named "*bouvet_test2_1610540190934*" which was the first filtering model with 69 labeled documents.

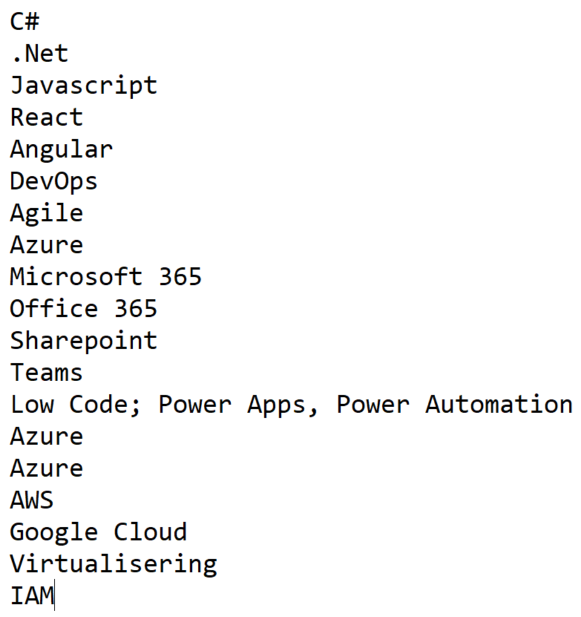


	A	B
31	gs://min-side_blog-articles_eu/15bb9d6e-6e2f-4406-9b7c-5abc53fb5258.txt	Out_of_domain
32	gs://min-side_blog-articles_eu/1b826f16-da53-4e16-8808-2241d209fea2.txt	Out_of_domain
33	gs://min-side_blog-articles_eu/1c4cc05b-40ef-4260-b92b-fe1054c20956.txt	Out_of_domain
34	gs://min-side_blog-articles_eu/1c7e47d2-932e-429d-bcda-facb7e38f70b.txt	In_domain
35	gs://min-side_blog-articles_eu/1cbfb45a-04fd-474b-bc50-73c00e721ba1.txt	Out_of_domain
36	gs://min-side_blog-articles_eu/1e3b27c2-d4c7-4178-8a11-2fd7e1b7ab76.txt	Out_of_domain
37	gs://min-side_blog-articles_eu/1e9a74c1-0e49-40f4-b6e0-396aa68dde34.txt	Out_of_domain
38	gs://min-side_blog-articles_eu/20d722de-49c1-4b04-bbb9-8a032e436ba6.txt	Out_of_domain
39	gs://min-side_blog-articles_eu/214b406b-0e27-441d-8304-382028d53698.txt	Out_of_domain
40	gs://min-side_blog-articles_eu/219374f9-744d-46a6-a66d-9988786821cb.txt	Out_of_domain
41	gs://min-side_blog-articles_eu/24ac9df3-4f03-42fd-841a-19a223e59397.txt	Out_of_domain
42	gs://min-side_blog-articles_eu/25552bef-494f-4a85-848b-9c9fac4c9437.txt	Out_of_domain
43	gs://bouvet-deler_eu/no-0024fdda-463e-4e4b-985f-4d6ba71444d0.txt	In_domain
44	gs://bouvet-deler_eu/no-0193c09a-af5b-4667-a58b-69aa866ee197.txt	Out_of_domain
45	gs://bouvet-deler_eu/no-0372c400-dbbc-4632-b3d1-354d03bddcb1.txt	Out_of_domain
46	gs://bouvet-deler_eu/no-03d430a8-8837-4bed-8984-a5293850cf9b.txt	Out_of_domain
47	gs://bouvet-deler_eu/no-04df98e3-2c17-46b1-bbc3-c97f2018027d.txt	In_domain
48	gs://bouvet-deler_eu/no-056783a9-724b-4ed4-9158-45167391f631.txt	In_domain
49	gs://bouvet-deler_eu/no-059dddbf-9490-495d-bffe-899bff21c4a1.txt	In_domain
50	gs://bouvet-deler_eu/no-05afe857-26c3-4a39-ae10-0a1bdcdf027e.txt	In_domain
51	gs://bouvet-deler_eu/no-05cac08b-8a24-491b-a644-286dcf929215.txt	In_domain
52	gs://bouvet-deler_eu/no-060af873-750e-41fb-99ef-346a00fc3172.txt	Out_of_domain
53	gs://bouvet-deler_eu/no-061f3530-059a-4273-812b-18f39180f36d.txt	Out_of_domain
54	gs://bouvet-deler_eu/no-063a5390-9204-414e-a7ad-60db8c4cefdb.txt	Out_of_domain
55	gs://bouvet-deler_eu/no-069da3d7-a90b-4122-8f93-02c2ee79bcad.txt	In_domain
56	gs://bouvet-deler_eu/no-06b0bd30-eb2f-41f4-9b34-bd6244145cab.txt	Out_of_domain
57	gs://bouvet-deler_eu/no-07139276-4b31-4c05-8626-79f535a8ce53.txt	Out_of_domain
58	gs://bouvet-deler_eu/no-087e2f47-2163-432a-9f6b-4ce871db6625.txt	Out_of_domain
59	gs://bouvet-deler_eu/no-08947d19-5f53-4c55-b039-4d43d7f464bd.txt	Out_of_domain
60	gs://bouvet-deler_eu/no-08bd71f4-2ada-4e15-b173-e47c34b71160.txt	Out_of_domain
61	gs://bouvet-deler_eu/no-08d6e213-e540-484c-a54d-2d361afb16e2.txt	In_domain
62	gs://bouvet-deler_eu/no-09175639-ce37-4459-b658-51db7df28f8b.txt	In_domain

Figure 3.8: Extract of the custom annotated trainingdata from the filtering model. The first column are URIs for the documents and the second is the custom label appointed to that file.

3.3.2 Competence Model

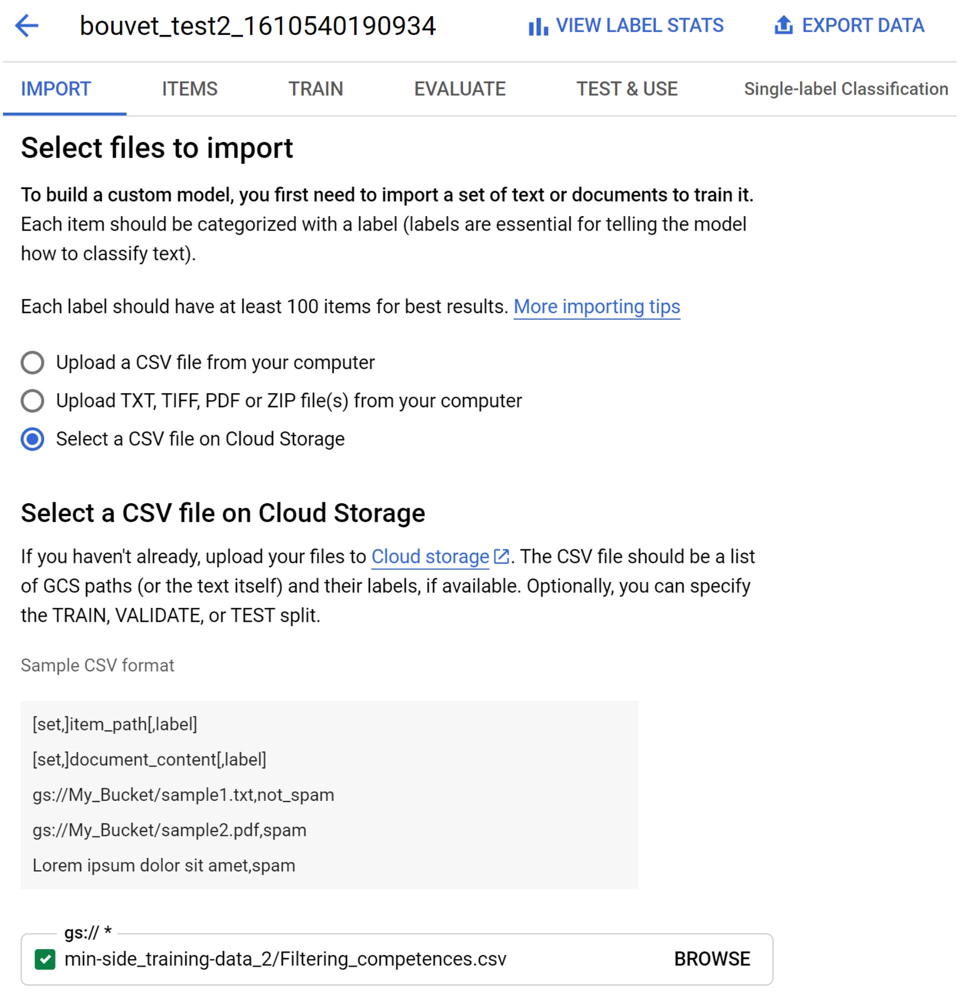
As can be somewhat seen in Figure 3.8, well over half the documents I annotated for the filtering model are labeled "Out_of_domain". This was especially the case concerning the internal documents from the buckets *min-side_blog-articles_eu* and *min-side_news_eu*. Of the annotated documents within these buckets less then 10% and less then 20% respec-



```
C#
.Net
Javascript
React
Angular
DevOps
Agile
Azure
Microsoft 365
Office 365
Sharepoint
Teams
Low Code; Power Apps, Power Automation
Azure
Azure
AWS
Google Cloud
Virtualisering
IAM
```

Figure 3.9: The technical competence labels provided by the head of the department *Øst Tech 2*.

tively were labeled "In_domain". In the final filtering, with 119 annotated documents, only 44 were "In_domain". Based on the proportion of relevant documents in the two filtering tests and the amount of documents available, I can estimate that between 600 and 800 documents can be assumed to be relevant in total. GCP AutoML recommends to have at least 100 documents per label for optimal training results. This could potentially indicate that there could only be a maximum of 8 different competence labels depending on the amount of overlap of labeling in the final multi label classification. In this kind of company, with departments covering everything within IT-consulting from web development to change management, the distinct labels would have to be very general and broad to cover all competences with so few labels. Given the intended use of the competence mining proposed in this project, a search on either a competence or directly on an employee would not say enough about the actual specific competences of the employees to be useful. For instance, all system developers may be labeled the same even though they may have different competences regarding technologies and types of development (e.g. front end vs. back end). The fact that the amount of relevant data is so limited would also mean that I would have to annotate and train all of the available data, which would leave no data to use the model on, and therefor very limited use for the classification model after it was trained.



Select files to import

To build a custom model, you first need to import a set of text or documents to train it. Each item should be categorized with a label (labels are essential for telling the model how to classify text).

Each label should have at least 100 items for best results. More importing tips

Upload a CSV file from your computer
Upload TXT, TIFF, PDF or ZIP file(s) from your computer
[✓ Select a CSV file on Cloud Storage]

Select a CSV file on Cloud Storage

If you haven't already, upload your files to Cloud storage. The CSV file should be a list of GCS paths (or the text itself) and their labels, if available. Optionally, you can specify the TRAIN, VALIDATE, or TEST split.

Sample CSV format

```
[set,]item_path[,label]
[set,]document_content[,label]
gs://My_Bucket/sample1.txt,not_spam
gs://My_Bucket/sample2.pdf,spam
Lorem ipsum dolor sit amet,spam
```

gs:// *
✓ min-side_training-data_2/Filtering_competences.csv BROWSE

Figure 3.10: The import-view of the dataset "*bouvet_test2_1610540190934*" showing the format of the .csv file as well as some other import options.

While I annotated the data, I also discovered that the "creator"-field, supposed to contain the Bouvet email address of the creator of the text, was often filled wrong or not filled at all. An example of this is shown in Figure 3.4a. Hence, I would not always be able to connect the classified competences back to an employee even if it is possible to train a classification model based on the data available. This is potentially an essential obstacle, since this would be the main purpose of the competence mining feature assessed in this project. This is discussed further in chapter 5.

Based on the evaluation of these aspects, I chose not to annotate and train a multi label classification model for competences as originally planned. This is because I saw that a limited training test would not give any more information about the opportunities or impossibilities for the competence mining functionality to work. That is, this test would not contribute any new information to the feasibility study conducted in this project. This is also the reason why the list of the final competence-labels for the *Øst Tech 2* department was not completed, as it would no longer used in this project.

Chapter 4

Results

In this chapter, the results of the different analyses and tests throughout the project are described. First, I go over the gathering and filtering of the different text analysis systems that were considered as the mining technology. Then, I show the further analysis and discussion around the three final candidates. Finally, I depict the results given by the automatic evaluation of the AutoML models that were trained for filtering the relevant documents.

4.1 Text Mining System

Table 4.1 shows the initial candidate text mining systems considered for further analysis and the results of the filtering based on the criteria listed in Figure 3.1 in chapter 3. After that, the further analysis of the final systems based on the questions in Figure 3.2 is shown and discussed.

4.1.1 Filter Initial Systems

Table 4.1: Initial text mining systems, with reasons for discarding. Chosen systems are marked in green.

System	Description	Notes
Aika	An artificial neural network for NLP. Provides aid in text analysis tasks such as text classification by generating multiple interpretations of a word and selecting the most likely one. Stable release 2017. (*Aika: An Artificial Intelligence for Knowledge Acquisition*, n.d.)	Too heavily focused on AI and not widely used.
Apache OpenNLP	A Java ML toolkit for NLP. Provides basic NLP tasks such as tokenization, sentence segmentation and named entity extraction. Initially released in 2004. (*Apache Solr*, n.d.-a)	
Apache Solr	An enterprise search platform that is part of Apache Lucene. Provides tasks such as full-text search and indexing. Initially published in 2006. (*Apache Solr*, n.d.-b)	This is a full search plattform and not a text analysis system.
Carrot 2	A clustering engine meant for search results. Categorizes documents into thematic clusters. Initially released in 2006. (Weiss & Osinski, n.d.)	Does not provide general NLP.
GATE ANNIE	A Java information extraction system. Provides basic NLP such as tokenization, lemmatization and semantic tagging. Initially released late 90's/early 2000's. (Cunningham et al., 2014)	Could not find evidence it is widely used and tested.

Table 4.1 – continued

System	**Description**	**Notes**
GATE TwitIE	A Java information extraction system for microblog text, specifically Twitter. Initially released in 2013. (Bontcheva et al., 2013)	Only focused on social media texts.
Gensim	A Python library for topic modelling. Mainly provides analysis of semantic structure and similarity retrieval. Initially released in 2009. (Řehůřek & Sojka, 2009)	Similar to NLTK, but appears less widely used and more focused on similarity generation.
Google cloud natural language	A text analysis system based on ML with two versions for NLP. Provides NLP tasks such as entity analysis and syntax analysis, as well as custom ML models. Initially released 2016. (*Google Cloud Natural Language*, n.d.)	
KH coder	A software for quantitative content analysis. Provides analysis such as word frequency and co-occurrence. Initially released early 2000's. (Higuchi, n.d.)	Could not find evidence it is widely used and tested. Initially made for Japanese.
Knime	A software analytic platform for creating data science. Provides tool for building data science workflows. Initially released in 2006. (*KNIME Analytics Platform*, n.d.)	More focused on data science and organization than natural language analysis.

Table 4.1 – continued

System	**Description**	**Notes**
Natural language toolkit (NLTK)	A Python library for NLP. Provides libraries for tasks such as classification, parsing and semantic reasoning. Initially released in 2001 as an educational tool. (Bird et al., 2009, 2001)	
QCAmap	An online qualitative content analysis tool. Mainly used for research analysis in social science research. Initially released early 2010's. (Mayring et al., n.d.)	Not open source and does not provide NLP.
Pattern	A Python module for web mining. Provides web mining, NLP, ML and network analysis. Initially released in 2011. (De Smedt & Daelemans, n.d.)	Gives impression of being a bit out of date. Similar to NLTK.
Sannsyn	A consulting company specializing in data science and analysis. Included because it was the only Norwegian system I found. Initially started in 2012. (*Sannsyn*, n.d.)	Not an available system ready for use, but a consulting service.
Textable	A software for visual text analysis. Enables building data tables from text data. Initially released in 2013 as an add-on to Orange Canvas. (Sarl, n.d.)	More focused on data science and organization than natural language analysis.

4.1.2 Analysis of Text Mining Systems

This section shows the results of the further analysis of the three remaining text mining systems based on the questions presented in chapter 3. The questions are repeated in Figure 4.1 for readability. Table 4.2 presents the answers to all the questions for each system. The systems are then compared, and the similarities and differences between the systems in how they are used and what they provide are discussed.

1. Meta information (When was it started?, How was it developed? etc.)
2. What is the system used for? What type of analysis does it provide?
3. Is the system meant for any specific technologies? If so, which?
4. Is the system meant for any specific domains? If so, which?
5. What/which natural language(s) is the system built for? Is it possible to change language?
6. What is the input and output of the system?
7. How easy is it to make changes and modifications to the system?
8. What/which part(s) of the system is/are potentially most relevant concerning competence mining?

Figure 4.1: Questions for each of the three final systems.

Table 4.2: Analysis of the three systems. Answers to the questions in Figure 4.1

Question	**Apache OpenNLP**	**Google Cloud Natural Language**	**Natural Language Toolkit**
1	Made and distributed by: Apache Software Foundation. Launched: 2004. Stable release: v1.9.2 December 11th 2019. License: Apache 2.0. Prices: Free.	Made and distributed by: Google LLC. Launched: NL-API: 2016. AutoML: 2019. Stable release: NL-API: March 20th 2020. AutoML: April 3rd 2020. License: Apache 2.0. Prices: See NL-API and AutoML.	Made and distributed by: NLTK Project. Launched: 2001. Stable release: v3.5 April 13th 2020. License: Apache 2.0. Prices: Free.
2	Supports NLP tasks such as tokenization, sentence segmentation, POS tagging and named entity extraction. For a full list of OpenNLP components, see OpenNLP Manual.	Reveals structure and meaning of text and classifies content: Entity analysis Sentiment analysis Entity sentiment analysis Syntax analysis Content classification Pre-trained ML models (NL-API) or custom models (AutoML)	Supports a wide variety of NLP tasks like string processing, classification, parsing and semantic interpretation. For a full list of NLTK modules, see NLTK Modules.

Table 4.2 – continued

Question	Apache OpenNLP	Google Cloud Natural Language	Natural Language Toolkit
3	Written in Java and used as a Java library. The CLI is available in Windows, and Linux or compatible systems.	Provides API client libraries in Java, Python, Node.js, C#, GO, PHP and Ruby, as well as REST API.	Written in Python and used as a Python library. Available for Windows, Mac OS X, and Linux.
4	Domains represented by the pre-trained models or as domain specific you want (custom trained models on existing or custom-annotated corpus).	General / not specific (NL-API) or as domain specific you want (AutoML)	Originally made for educational purposes to learn NLP. Support for some different domains have been added through corpora over time. It is also possible to add custom corpora.
5	Languages represented by the pre-trained models or in a specific language (custom trained models on existing or custom-annotated corpus). The toolkit also offers support for a Language Detector.	Both systems supports a multitude of languages on their own: Language Support NL-API, Language Support AutoML.	Originally made for American English. Support for some other languages have been added through corpora over time. It is also possible to add custom corpora.

Table 4.2 – continued

Question	Apache OpenNLP	Google Cloud Natural Language	Natural Language Toolkit
6	Input: A model (usually loaded via a FileInputStream) and an input-text to analyse (usually a String or an array of String). Output: Usually an array of String. (*Apache OpenNLP | Home*, n.d., Chap. 1)	Input: A String (the text to be analysed) and sometimes encoding type, language and/or model depending on which task is performed and if custom models are used. Output: Special classes representing the requested output. (e.g. an Entity class with methods such as getName and getSalience) (*AutoML Natural Language documentation*, n.d.; *NL-API | How-to Guides*, n.d.)	Differs for all modules, see NLTK Modules for details.

Table 4.2 – continued

Question	Apache OpenNLP	Google Cloud Natural Language	Natural Language Toolkit
7	NLP components may be used and combined freely to build a NLP program, and all models may be custom-trained on chosen corpora. Also provides support for Brat format (custom-annotated corpus).	NLP tasks may be executed and combined with an API or through an API client library and models may be custom-trained (AutoML).	NLP modules may be used and combined freely to build a NLP program. New corpora may be added easily only if it's format is already supported by one of the existing corpus readers. If a new corpus reader is to be added, it needs to be added to the source code directly. This makes modifications possible, but somewhat difficult.
8	Provides NLP. Programmatically adaptable and usable in Java. Supports different domains and languages through custom-trained models.	Provides NLP and classification. Programmatically usable through an API or API Client libraries. Supports different domains and languages through custom-trained models (AutoML).	Provides NLP. Programmatically adaptable and usable in Python. Supports some different domains and languages (limited to corpora formats supported by existing corpus readers).

Table 4.2 shows that all the systems provide some form of natural language processing. All systems provide classification or categorization and some form of named-entity recognition. Apache OpenNLP and NLTK both provide tokenization, tagging, lemma/stemming and parsing as separate tasks, whereas Cloud Natural Language combines these in a syntactic analysis. This potentially makes Cloud Natural Language less modular. NLTK and Cloud Natural Language also provides sentiment analysis.

All three of the systems offer the possibility to custom train ML models, but NLTK have some restrictions. When adding a new corpus in NLTK, the format must either be supported by one of the existing corpus readers or a new corpus reader has to be developed first. This then has to be committed to the public source code. If the corpus is already supported by a reader however, the process is simple and can be performed without interfering with the public source code. Custom-trained models in Cloud AutoML also has a limitation, in that they can last no longer than 6 months before they have to be renewed (*AutoML Natural Language documentation*, n.d.). Apache OpenNLP has built-in support for Brat format custom-annotated corpora, making it even more flexible in regards to custom models (*Apache OpenNLP | Home*, n.d., Chap. 13).

The text mining systems all have the opportunity to work with different domains and/or languages, both through existing built-in support for specific domains or languages and through adding new corpora and custom-training models. Non of the systems have pre built-in support for Norwegian language, but some have support for Swedish and other closely related languages.

NLTK and Cloud Natural Language both offers simple demos of some of their analysis tools (*Demo | Cloud Natural Language*, n.d.; *NLTK Demos for NLP*, n.d.), and OpenNLP offers a CLI for experimenting with their tools. This means that all three systems offers ways to test smaller projects or texts before committing to that system with a large scale project.

All systems provide programmatic access to their tools, but with support for different programmatic languages. Cloud Natural Language offers API Client Libraries in a variety of programming languages, OpenNLP is a Java toolkit and NLTK is a Python toolkit.

Google Cloud Natural Language is the only one of the systems that charges for extended use of the tool. However, Cloud Natural Language is also a part of the Google Cloud platform, and therefor offers additional support with Cloud Storage and other Google Cloud services. Neither OpenNLP nor NLTK offers similar support for direct connection to a cloud storage system or other related services and APIs.

Finally, all three text analysis systems are widely used and tested. They are all released and maintained by professional and respected sources, either through academia or as parts of larger computer science platforms (Apache and Google). All systems are also regularly updated and have had a new stable release within the last 7 months (as of June 2020).

The analysis shows that the three text mining systems offer a lot of the same tasks and services within NLP. All three of the systems satisfy the criteria listed in Figure 3.1 and offer the most necessary NLP tasks for conveying and "understanding" the meaning of text. They can therefor all potentially be used in this project. However, they are slightly different both in what tasks and services they provide and how they are used.

If the company or organization that wants to implement competence mining already utilizes several of Google Clouds other services and have all or most of their data stored in Google Cloud Storage, then Cloud Natural Language is the best option. This is because Cloud Natural Language is a part of the Google Cloud platform and offers direct connections to other services in the platform. If the competence-mining solution is to be implemented in Pyhton, then NLTK may be the most fitting option because it functions as a Python library. In additon, NLTK is free as opposed to Cloud Natural Language. Likewise, if the solution is to be developed in Java, then Apache OpenNLP is the best alternative because it functions as a Java library and is also free.

If no programming language or other development requirements are set, and the developers are free to choose the most fitting text analysis system regardless of technical restrictions, then Apache OpenNLP will be the best choice. This is because it offers the most adaptability and best support for custom training of models based on custom-annotated corpora through built-in support for the Brat format. OpenNLP also offers all the NLP tasks as separate components making it more modular, as well as offering language detection. As revealed by the analysis, both NLTK and Cloud Natural Language have some restrictions regarding custom-training of ML models.

In this project I was going to be working with potentially large amounts of data from Bouvet. To accommodate this and make it easier and faster to work with and process, I needed to store the data in a cloud system. Bouvet ASA normally utilizes both Google Storage through the Google Cloud Platform (GCP) and Azure storage as their main cloud systems.

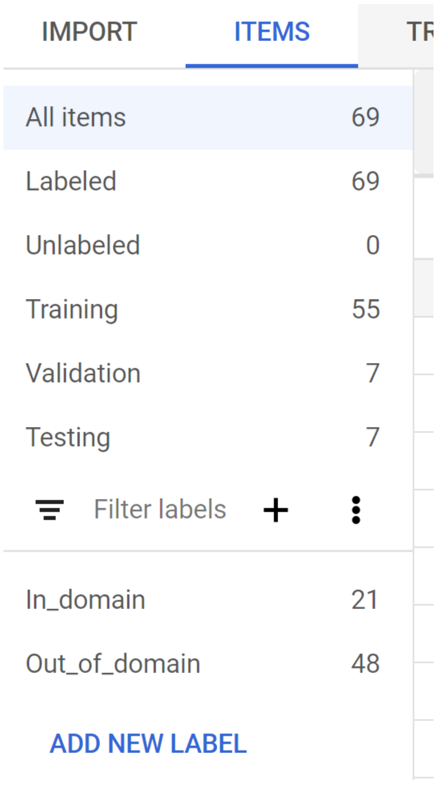
However Sesam, the data integration platform and subsidiary company of Bouvet, mostly utilizes GCP and could send the data required for this project directly to buckets in Google Cloud Storage. Based on this, combined with the analysis above, Google Cloud Natural Language was selected as the text mining system for the project. This is because Natural Language is also a part of the GCP and can be directly connected to Cloud Storage to analyse the documents stored there. AutoML was chosen over NL-API because of the possibility to use custom labels as well as better language support. AutoML supports text classification in both English and Swedish (*Language Support | AutoML*, n.d.), whereas NL-API only offers in English (*Language Support | API*, n.d.). Neither of the systems offer support for Norwegian.

4.2 Training Classification Models

As mentioned in chapter 3, Cloud AutoML provides an automatic evaluation of the models performance when training a ML model (*AutoML Natural Language Beginner's guide*, 2020). The evaluation is given based on the 10% of the documents AutoML used for testing the model after it is trained. This evaluation provides lists of true positives, false negatives and false positives within each label. It creates a confusion matrix for the model and calculates precision and recall for each label and all labels in total. Together, the evaluation statistics provide a quantitative assessment of each model trained.

Figure 4.2a shows the distribution of the labels as well as the number of documents chosen by AutoML for training, validation and testing in test one of the filtering model. Test one had 69 labeled documents from all the different buckets mentioned in chapter 3, while test two had 50 additional labeled documents specifically from "bouvet-deler_eu" making it 119 labeled documents in total. Figure 4.2b shows the distribution of documents in test two.

A single label classification model assigns one label to each classified document. For each document the model calculates a score for each label between 0 and 1 that indicates the confidence the model has that the document should be classified with that label. When using single label classification, the sum of the scores for all possible labels for each document will be 1. For example, if a document in my test is given a score (or probability) of 0.67 that the document should be labeled "In_domain", it will have a score of 0.33 that



IMPORT ITEMS TR

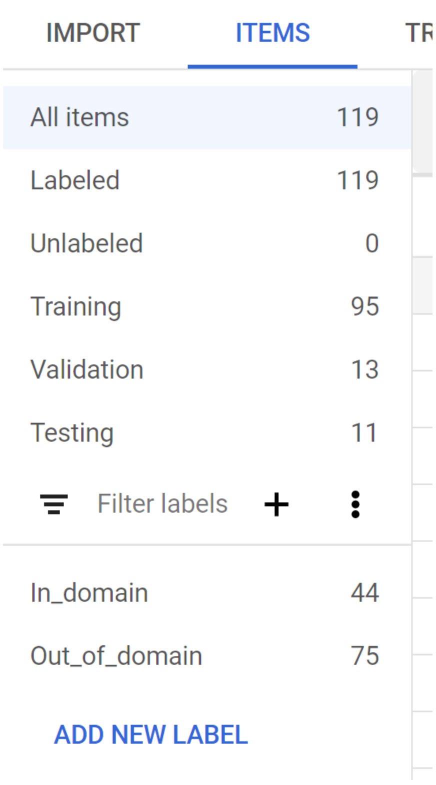
All items	69
Labeled	69
Unlabeled	0
Training	55
Validation	7
Testing	7

Filter labels

In_domain	21
Out_of_domain	48

ADD NEW LABEL

(a) Test one.



IMPORT ITEMS TR

All items	119
Labeled	119
Unlabeled	0
Training	95
Validation	13
Testing	11

Filter labels

In_domain	44
Out_of_domain	75

ADD NEW LABEL

(b) Test two.

Figure 4.2: Information about the distribution of documents and labels in test one and two of the filtering model.

it should be labeled "Out_of_domain". The document will then be labeled "In_domain" with a confidence of 0.67. When using multi label classification, each label is considered individually and the sum of the scores for each document will not necessarily be 1. In this case it would be relevant to chose a confidence threshold between 0 and 1 that is the minimum score a label must have to be assigned to that document. However, when using single label classification, the model will always assign the document with the label with the highest score regardless of the confidence threshold. In both the tests for the filtering model the only relevant threshold is therefor 0.5 since we have two possible labels.

The evaluation metrics compares the labels the model predicted for the test documents with the true labels they actually have. Based on this, it depicts lists of all the test document within each label that were true positives, false positives and false negatives. Figure 4.3

gives a graphical illustration of what this means. Relevant elements refers to all documents that are actually a given label (e.g. "In_domain") and selected elements refers to all documents that were labeled with that label. True positives are then all documents that were correctly labeled "In_domain", false positives are all documents that were incorrectly labeled "In_domain" and false negatives are all documents that were incorrectly *not* labeled "In_domain".

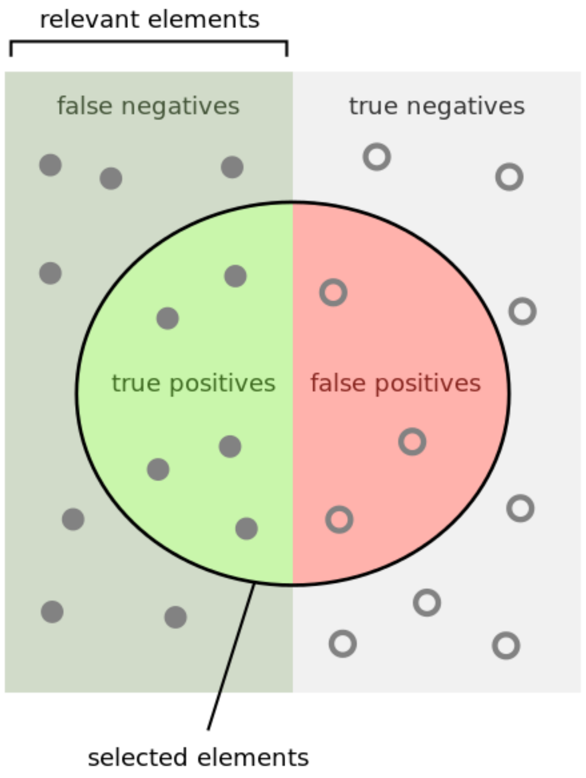


Figure 4.3: Graphical illustration of true and false positives, and true and false negatives for a label (Walber, 2014). Relevant elements refers to all documents that are actually that label (e.g. "In_domain") and selected elements refers to all documents that were classified with that label.

Table 4.3 describes how many of the tested documents were in each list for each test. The results show that both test only had two incorrectly labeled documents (the document labeled "False positive" for "In_domain" is the same document labeled False negative for "Out_of_domain" and vice versa). However, both test also had a very limited number of tested documents. If we calculate the percentages, we see that test one had 28.6%

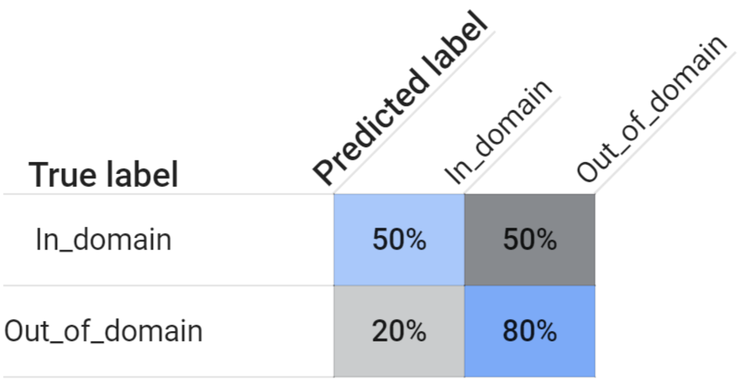
mislabeled documents and test two had 18.2% mislabeled documents.

Table 4.3: How many documents were predicted correctly and incorrectly for each label for each test of the filtering model.

		True Pos	**False Pos**	**False Neg**	Tested documents
Test one	"In_domain"	1	1	1	7
	"Out_of_domain"	4	1	1	
Test two	"In_domain"	3	1	1	11
	"Out_of_domain"	6	1	1	

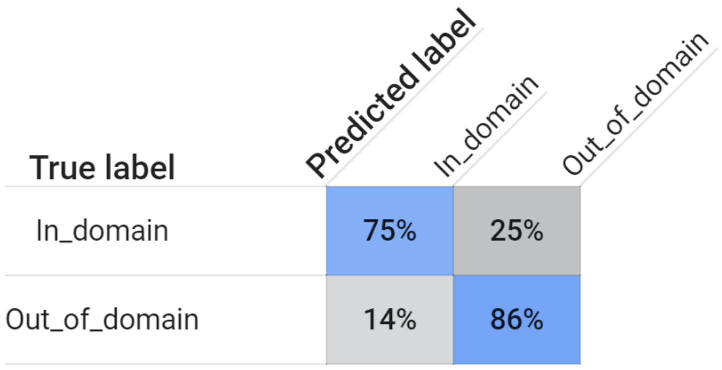
One of the other evaluation metrics given by AutoML that tells us something about the correctness percentages of the model is the confusion matrix. Figure 4.4 and Figure 4.5 depicts the matrices for test one and two. The confusion matrix shows how often each label is assigned correctly in the blue diagonal and what other label the incorrectly labeled documents are given instead. In this case there is only one other option if the document is labeled incorrectly since there are only two labels in total. The matrix for test one (Figure 4.4), show that documents that should be labeled "In_domain" are classified correctly only 50% of the time and documents that should be labeled "Out_of_domain" are classified correctly 80% of the time. The matrix for test two in Figure 4.5 shows that this has improved a little with more trained and tested documents with 75% classified correctly for "In_domain" and 86% for "Out_of_domain".

The final metrics given by AutoML is precision and recall for each label and for all labels in total. Equation 4.1 and Equation 4.2 gives the formulas for calculating precision and recall based on the terms used in Figure 4.3. In more layman's terms we can say that precision tells us how many selected documents are relevant and recall tells us how many relevant documents are selected (Walber, 2014). The precision and recall values for each label for both tests are depicted in Figure 4.6 and Figure 4.7. Both precision and recall were slightly improved by the second test with more annotated documents, especially for the label "In_domain".



True label	Predicted label: In_domain	Predicted label: Out_of_domain
In_domain	50%	50%
Out_of_domain	20%	80%

Figure 4.4: The confusion matrix of the model trained from test one.

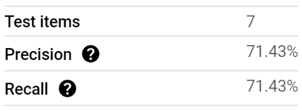


True label	Predicted label: In_domain	Predicted label: Out_of_domain
In_domain	75%	25%
Out_of_domain	14%	86%

Figure 4.5: The confusion matrix of the model trained from test two.

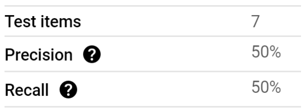
$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} = \frac{TruePositives}{Selected} \tag{4.1}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} = \frac{TruePositives}{RelevantElements} \tag{4.2}$$



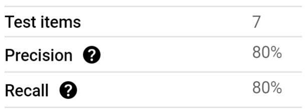
Test items	7
Precision	71.43%
Recall	71.43%

(a) All labels



Test items	7
Precision	50%
Recall	50%

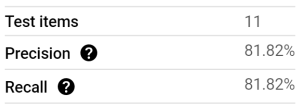
(b) In_domain



Test items	7
Precision	80%
Recall	80%

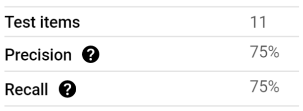
(c) Out_of_domain

Figure 4.6: Precision and recall for test one.



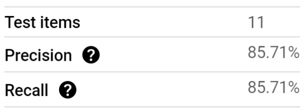
Test items	11
Precision	81.82%
Recall	81.82%

(a) All labels



Test items	11
Precision	75%
Recall	75%

(b) In_domain



Test items	11
Precision	85.71%
Recall	85.71%

(c) Out_of_domain

Figure 4.7: Precision and recall for test two.

Chapter 5

Discussion

In this chapter, I discuss the challenges and limitations that have arisen throughout the work and research for this study. I first discuss the challenges regarding the quality and availability of the data, as well as the technology used, in the project. Finally, I review the results of the evaluation of the trained ML models.

5.1 Challenges and Limitations

5.1.1 Data Quality and Availability

As I have touched on several times before, there are several possible challenges with regards to the quality of the data available. In the following paragraphs, I will discuss challenges with some of the most relevant information quality dimensions for this project as described in chapter 2. Out of the dimensions described, only 8 are relevant for this project:

Intrinsic Quality of Information:	Accuracy
Representational Quality of Information:	Consistency of Representation
	Ease of Understanding

Contextual Quality of Information:	Completeness
	Relevance
	Appropriate amount of Data
Accessibility of Information:	Access control
	Access time

Firstly, a challenge with the quality of the data I received that falls under several of the dimensions in Table 2.1 is the problem with the formatting of the documents shown in Figure 3.4 in chapter 3. This problem leaves many potentially important words and phrases unreadable to both a human and a machine learning system. This may cause the ML models to be trained incorrectly and can cause the model to mislabel or circumvent classifying a label to a document entirely. This is especially the case regarding the formatting of Norwegian and Swedish specific letters, like "Æ Ø Å" and "Å Ä Ö", but also sometimes concerns other characters like " ' * / ". There are also several html tags present in most of the documents. It is uncertain whether or not this could possibly be fixed in the future because it has to be done internally before the documents are sent to Google Storage. As mentioned in chapter 3, the microservice used by Sesam did not provide a field for setting file encoding to UTF-8 when sending the documents, so this would have to be solved in a different way. Cloud Storage gives the opportunity to change the metadata-field "encoding" to UTF-8, but only manually on one document at a time. Removing the html tags may also be possible to do for the communications department, but would require a lot of work because each type and case of html tags would have to be removed separately. This aspect of the data affects the information quality regarding the accuracy, consistency of representation and ease of understanding.

Another challenge regarding the consistency of the representation of the data is the fact that the buckets consists of a mix of Norwegian, Swedish and English documents. Bouvet publicly have their home page in those three languages; `http://bouvet.no`, `http://bouvet.se` and `http://en.bouvet.no`. I tried to separate them by filtering which page the different public documents came from, but this proves to not always be correct. There are some English and Swedish texts on the Norwegian page and vice versa. The same goes for the documents from the internal pages, the news and the articles, but these can not be separated based on language at all. Google Cloud AutoML suggests training models on documents that are all the same language for a higher quality model, but this is not possible based on the data available without somehow potentially filtering them based on language first.

There are also several contextual challenges and limitations present in the data. There is firstly a potentially huge challenge regarding the lack of metadata concerning the author of the texts. This affects the completeness dimension of the information quality. The field "Creator" mentioned in chapter 3 only exists as part of the documents, and not as a separate information stored in connection to the documents which means it may be difficult to separate it from the texts. While I annotated the training data, I also discovered that the creator is often either not filled, filled wrong, or the text is written or created on behalf of someone else. There are many examples of employees in the communications department writing about other employees' experiences and achievements. This implies that even if it is possible to train a model to classify the competences in the documents, I could not use this information to connect back to which employee has that competence. Seeing as this is the main purpose of the competence mining functionality suggested in this project, this may be a crucial hindrance if it is not possible to get the correct author in some other way. If that is possible, it would have to be connected before the data is sent to Google Storage. In addition to this, not all employees writes articles for publication, either internally or publicly. Some employees, specially heads of departments or employees with a technical advisor position, may write many articles on a regular basis, whereas others may not write any at all. This means that we potentially loose information about competences on some employees all together if this is the only method used.

Finally, most of the documents, especially the internal ones, turns out to be irrelevant in regards to finding competences. Examples are texts about renting the company cars and cabins, covid-19 regulations and general information about employee rules, etiquette and benefits. Some documents are even almost completely empty. This problem with relevance is attempted solved by introducing the filtering model as a step before the classification of competences as suggested on *Preparing your training data | AutoML Natural Language Documentation* (2020). However, as discussed in the end of chapter 3, this left a too limited amount of relevant documents for the classification of competences. It is so limited in fact, that even if I annotated and trained a competence ML model based on *all* the relevant data, this would result in either a quite poorly trained model with many labels or a possibly good model with very few unique competence labels. Either way, this would not satisfy the competence mining functionality discussed in this project. In addition, I would then have used all the data to train a model and would therefor have no documents left to use the model on, thereby rendering it virtually unused.

When it comes to the accessibility of the data, Bouvet does not utilize the data that I have used in this project for any form of analysis today. The data is stored in different places and in different ways and is only used as publications on the internal pages and on `http://bouvet.no`. This means that the raw text data and meta information about them are not readily available. Both the communications department at Bouvet and Sesam had to work to modify the data and make it available for my analysis. The fact that the quality of the data I ended up with is still far from perfect for an analysis shows that Bouvet would potentially have to make substantial changes to their data and how they store and utilizes it if they want to implement competence mining based on these documents in a bigger scale and more permanently.

5.1.2 Additional Potential Data Sources

It is not just the data quality that presents challenges in this project. The availability of other data that could potentially be useful is also an important aspect.

In regards to other possibly useful data, Bouvet does not operate with any kind of project database that could potentially document project topics, technologies, results and employees for each project. This means that there is no way of using the projects the employees have worked on and the technologies they used as a part of the competence mining. Today, the only place we can find information about this is in the internal project CV's that the workers write themselves as mentioned earlier. Using this data would defeat the purpose of looking at what people have done instead of what they themselves say they know.

Other potentially relevant data that were considered in this project were internal communications channels like Slack (Slack, n.d.) and Teams (Teams, n.d.). In Bouvet, these channels are used for all daily communication together with email. Here, everything from academic gatherings, lectures and technical questions to birthdays and company parties are being discussed. This raises a potential problem regarding privacy and GDPR-concerns when going through all of this data. It is vital that personal information about family, health and terms of employment among other things are not available to anyone that should not have access to that information. This was therefor considered out of scope for this project, but is something that could potentially be explored further in the future.

Bouvet also utilizes SharePoint (SharePoint, n.d.) in addition to their internal pages for document sharing and information about events and so on. This contains mostly documents meant for employees like information about insurance, setting up mail signatures and how to report work hours. However, SharePoint is also used for sharing power point presentations from course's and lectures. Because collecting the data stored in SharePoint could not be done together with collecting the articles from the Bouvet webpage and would have required more workhours from the employees at Bouvet's communications department and Sesam, the data from Sharepoint was also not included in this project, but is something that could be considered for future work.

Finally, one of Bouvet's most important visions is competence sharing both across departments and regions internally and externally. Bouvet therefore hosts regular technological and professional events, lectures and courses for and by their employees. Information about these events; who hosted them, summaries and/or titles and possibly even video recordings could also potentially be explored further with regards to competence mining. However, this is not included in this project because this information is either not stored after the event or not stored in a way that makes it possible to gather it and use it for analysis today. The events are published on an internal website where employees can sign up for different events and then receive information and links directly in their email.

5.1.3 Technical Limitations

The third and final aspect of this feasibility study is the possibilities, challenges and limitations presented by the technology used. Chapter 3 and chapter 4 described the process of analysing and choosing the text mining system to be used in this project. However, all systems have their limitations and here are some of the challenges I met while working with Google Cloud Platform and Cloud AutoML.

Firstly, as I have touch upon before, GCP AutoML classification does not have built in support for Norwegian texts (*Language Support | AutoML*, n.d.). Bouvet, being a Norwegian company, has Norwegian as their main communications language. Most of the documents both on their public website and internally is therefor in Norwegian. However, while I researched different text mining systems I found non that offered built in support for Norwegian classification or any other forms of natural language processing for Norwegian texts. AutoML does nevertheless offer support for classification in several other languages, including English and Swedish. Since these are the other two languages the texts from Bouvet are written in, and because the Swedish and Norwegian languages are so similar and have many of the same words, I chose to test AutoML on the Norwegian documents regardless of this limitation.

I also met some challenges when it came to the documentation for Google Cloud Platform and especially AutoML. GCP offers a lot of different services, with a correspondingly huge amount of user documentation and guides. Most of this documentation is organized and easy-to-follow guides and tutorials like the ones listed on *How-to Guides | AutoML Natural Language Documentation* (2020). However, this is not the case for all of the documentation. Because there are so many documents, quick-guides and tutorials divided on so many different web pages, subpages and separate links, it is hard to know if or when I have read all available documentation that is relevant for my task. Some of the documentation is also showing signs of not being updated along with changes and new deployments of the products. Particular information about how to use the systems are sometimes mentioned in one tutorial, but not another. Buttons and fields sometimes have different names, placements and even functionality in the documentation and in the actual product. This led to several problems and confusions during the project. One example concerns the requirements for the buckets in Google Storage mentioned in chapter 3. While I worked I used the tutorials from the Google Storage documentation to make the buckets and add the data (*Bucket locations | Cloud Storage*, 2020; *Creating storage buckets | Cloud Storage*, 2020), and then I used the tutorials for AutoML to prepare my training data and import it to AutoML (*Preparing your training data | AutoML Natural Language Documentation*, 2020). Neither of these guides mentioned that there were any requirements for the buckets if they were to be used by AutoML, so I did not discover this until I tried to import the training data to AutoML and got an error. This error message was also so vague that I did not understand it at first. Because of this error and confusion, I had to create new buckets with the correct locations requirements and have all the data sent to me one more time.

5.2 Training Results

The automatic evaluation results of the training of the two filtering models depicted in chapter 4 suggests that the ML model becomes better with more annotated documents. This is in line with the documentation on AutoML that states that there has to be at least 10 documents per label, and preferably around 100 documents per label for the best results. The fact that the results for test two is better overall then for test one also suggest that some of the challenges with the data quality mentioned above might not be as big as expected. If the quality of the data was so poor that AutoML was not able to "read" the documents at all, we could expect both the tests results to classify correctly only about 50% of the time on average. However, even though test two has quite a lot more annotated documents then test one, the 10% of the documents used for testing both models are still too few to get a significant and trustable result. Test one only allocated 7 documents for testing and test two allocated 11. In addition, the apparent improvement from test one to test two does not prove that the suggested next step, a multi-label classification model on competences, would work equally well. This is because the filtering models only have two labels and will always have access to all the documents available, whereas the competence model could potentially have many labels and can only be used on the relevant documents. There will therefor always be less documents per label in the second ML model and thereby a poorer quality on the competence model. Additionally, the different competence labels may require AutoML to be able to "read" and understand more details in each text in order to see the specific competence word or phrase presented in the document that is relevant.

Chapter 6 Conclusion

In this feasibility study I have endeavoured to assess, discuss and map the opportunities and impossibilities of a competence mining feature based on the data available at Bouvet and the technology used to create and train ML models. In this final chapter, I will attempt to answer the research questions presented in chapter 1 by reviewing the findings throughout the project.

6.1 Conclusion

I started my research by gathering information on and analysing different text mining systems to be potentially used to mine competences from natural language texts. Based on this analysis, I chose to test Google Cloud Platform's solution AutoML Natural Language. Next, I annotated and trained articles, news and blog posts from Bouvet's internal and public pages.

The results of the training shown in chapter 4 and discussed in chapter 5 suggests that it may be possible to mine competences from the data given by Bouvet using GCP AutoML, which was the first and main research question. However, there are substantial challenges and limitations that would have to be solved or changed in order for this feature to work optimally and as intended and initially described in chapter 1. The most preeminent challenges are presented and discussed in chapter 5.

The apparent answer to the second research question also raises a potential problem as I found several examples of employees writing about other employees' knowledge and achievements. This suggests that the vital assumption that the author of a text always inhabit the competences described is not true. If the competence mining functionality were to be implemented it could be possible to include an extraction of the names of other employees present in the text and/or ask automatic validation questions to each employee about the competences that the system "thinks" they inhabit.

6.2 Limitations of the Study

As with any research study and project, there are always things that could have been done differently. There will always be possible improvements to the method and research that is conducted. In addition, there will always be some future work and/or alternative methods, choices and technologies that could potentially be researched and included in a study. Here are some of the most prominent limitations to the study conducted in this project.

As touched on throughout the thesis, I should ideally have had the opportunity to annotate and train at least 100 documents per label for the filtering models. In order for this to be possible, I would have to read and annotate approximately 300 documents given the uneven distribution of the labels "In_domain" and "Out_of_domain". Due to the challenges mentioned in chapter 5 regarding the accessibility of the data I received from Bouvet, and the documentation for GCP AutoML, I was not able to receive the final usable data before I only had two months left of the project time. Due to the time-constraint that arose from that, combined with the time-consuming act of custom-annotating the data, I was only able to label 123 documents (4 of which received an error on import giving test two 119 documents). However, I was still able to show an improvement in the ML models when training based on a larger amount of annotated data from my test one to test two. This is the reason why annotating more data was not prioritized higher.

With the benefit of hindsight, I see that I could have read the documentation on the different GCP services more comprehensively and conducted more end-to-end tests before starting the second part of the project. This could potentially have evaded some of the confusions and challenges stemming from the GCP documentation. In addition, this may have reviled further possibilities regarding analysing and/or parsing of the texts before training the ML models.

I could also potentially have done more research into the possibility of utilizing existing pre-annotated corpora or even pre-trained ML models instead of attempting to custom-annotate and train models. Instead, this is considered one of the possible future steps after this study.

6.3 Future Work and Prerequisites for Implementing

So what should Bouvet or another company do if they wish to implement and utilize the competence mining functionality discussed in this project? I have assessed that the company should optimally have easy access to as many documents and texts written by the employees or about the employees' work as possible. This data preferably needs to be in the form of clean text to be used for classification connected to an identifier for the employee it concerns. To cover as many employees as possible, different sources and types of documentation like the ones mentioned in chapter 5 should be included. For Bouvet, this would mean a substantial change from how relevant data for competence mining is stored today. If they were to utilize the data they have today that is described in this project, they should work to improve the quality of the raw data by removing the problems with the formatting and finding a different way of connecting the texts back to the correct author. They would also have to try to make more potentially relevant data available as clean texts in Google Storage, like the sources suggested under data availability in chapter 5. In addition, it is possible for Bouvet, and other Norwegian companies, to try to find general Norwegian annotated corpora or even pre-trained Norwegian models in order to use these directly. If this exist in a form that is useful, meaning the annotation includes competences, this could potentially solve the problem of having to few relevant documents to train the models.

The potential next step after this study, could be to research the possibility of using existing Norwegian corpora or models as mentioned above. It could also be to assess the possibilities of accessing and utilizing the supplementary data-sources referred to and evaluate whether or not they can contribute to the competence mining corpora and functionality.

References

ACM Digital Library. (n.d.). Retrieved 2020-10-10, from https://dl.acm.org/

Aika: An Artificial Intelligence for Knowledge Acquisition. (n.d.). Retrieved 2020-03-18, from http://aika.network/index.html

Alpaydin, E. (2020). *Introduction to Machine Learning*. MIT Press. (Google-Books-ID: tZnSDwAAQBAJ)

Alter, S. (2006). *The Work System Method: Connecting People, Processes, and IT for Business Results*. Work System Method.

Apache OpenNLP | Home. (n.d.). Retrieved 2020-03-17, from https://opennlp.apache.org/ (Repository: https://github.com/apache/opennlp)

Apache Solr. (n.d.-a). Retrieved 2020-03-17, from https://lucene.apache.org/solr/

Apache Solr. (n.d.-b). Retrieved 2020-03-17, from https://lucene.apache.org/solr/

API Client Libraries. (n.d.). Retrieved 2020-05-21, from https://developers.google.com/api-client-library (Library Catalog: developers.google.com)

AutoML Natural Language Beginner's guide [Guide]. (2020, December). Retrieved 2021-03-14, from https://cloud.google.com/natural-language/automl/docs/beginners-guide

AutoML Natural Language documentation. (n.d.). Retrieved 2020-05-05, from https://cloud.google.com/natural-language/automl/docs (Library Catalog: cloud.google.com)

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python* (1st ed ed.). Beijing ; Cambridge [Mass.]: O'Reilly. (ISBN: 978-0-596-51649-9)

Bird, S., Loper, E., & Klein, E. (2001). *Natural Language Toolkit | nltk.* USA. Retrieved 2020-03-18, from http://www.nltk.org/ (Repository: github.com/nltk/nltk)

Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M. A., Maynard, D., & Aswani, N. (2013). TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, 8.

Bouvet Norge. (n.d.). Retrieved 2020-12-04, from https://www.bouvet.no/

Bucket locations | Cloud Storage. (2020). Retrieved 2020-11-03, from https://cloud.google.com/storage/docs/locations

Cambria, E., & White, B. (2014, May). Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article]. *IEEE Computational Intelligence Magazine*, *9*(2), 48–57. (Conference Name: IEEE Computational Intelligence Magazine) doi: 10.1109/MCI.2014.2307227

Cloud Natural Language API documentation. (n.d.). Retrieved 2020-05-05, from https://cloud.google.com/natural-language/docs (Library Catalog: cloud.google.com)

Craft, R. (2016, November). *Google Cloud Machine Learning family grows with new API, editions and pricing.* Retrieved 2020-05-06, from https://cloud.google.com/blog/products/gcp/cloud-machine-learning-family-grows-with-new-api-editions-and-pricing/ (Library Catalog: cloud.google.com)

Creating storage buckets | Cloud Storage. (2020). Retrieved 2020-11-03, from https://cloud.google.com/storage/docs/creating-buckets

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., … Derczynski, L. (2014, November). ANNIE: a Nearly-New Information Extraction System. In *Developing Language Processing Components with GATE Version 8.* United Kingdom: University of Sheffield Department of Computer Science. Retrieved 2020-03-17, from https://gate.ac.uk/sale/tao/splitch6.html#chap:annie

Curran, R., & Gualtieri, M. (2016, May). Brief: Cognitive Search Is Ready To Rev Up Your Enterprise's IQ. *Forrester*, *The Forrester Wave: Cognitive Search And Knowledge Discovery Q2 2017*. Retrieved 2020-08-28, from https://go.forrester.com/blogs/17-06-12-cognitive _search_is_the_ai_version_of_enterprise_search/

Definition of Skill Mining - Gartner Information Technology Glossary [Glossary]. (n.d.). Retrieved 2020-08-21, from https://www.gartner.com/en/information-technology/glossary/skill-mining

Demo | Cloud Natural Language. (n.d.). Retrieved 2020-04-20, from https://cloud.google.com/natural-language#natural-language-api-demo (Library Catalog: cloud.google.com)

De Smedt, T., & Daelemans, W. (n.d.). *CLiPS - Pattern.* Belgium: Computational Linguistics Research Group. Retrieved 2020-03-18, from https://github.com/clips/pattern (original-date: 2011-05-03T15:29:01Z)

Fehrenbacher, D. D., & Helfert, M. (2012). Contextual Factors Influencing Perceived Importance and Trade-offs of Information Quality. *Communications of the Association for Information Systems*, *30*(8), 111–126. doi: 10.17705/1CAIS.03008

Feldman, R., & Sanger, J. (2006). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data.* Cambridge: Cambridge University Press. Retrieved 2020-06-18, from https://www.cambridge.org/core/books/text-mining-handbook/0634B1DF14259CB43FCCF28972AE4382 doi: 10.1017/CBO9780511546914

Ge, M., & Helfert, M. (2007). A Review of Information Quality Research - Develop a Research Agenda. In *International Conference on Information Quality.*

Getting started with authentication | Authentication. (n.d.). Retrieved 2020-09-23, from https://cloud.google.com/docs/authentication/getting-started

Girard, J., & Girard, J. (2015). Defining knowledge management: Toward an applied compendium. , *3*(1), 20.

Google Cloud Natural Language. (n.d.). Retrieved 2020-05-05, from https://cloud.google.com/natural-language (Library Catalog: cloud.google.com)

Higuchi, K. (n.d.). *KH Coder.* Retrieved 2020-03-18, from http://khcoder.net/en/

How-to Guides | AutoML Natural Language Documentation. (2020). Retrieved 2020-11-05, from https://cloud.google.com/natural-language/automl/docs/how-to

IEEE Xplore. (n.d.). Retrieved 2020-10-10, from https://ieeexplore.ieee.org/Xplore/home.jsp

Jiang, J. (2010, May). *Google Storage for Developers: A Preview - The* *official Google Code blog.* Retrieved 2020-07-02, from http://googlecode.blogspot.com/2010/05/google-storage-for-developers-preview.html

Jin, C. X., Li, F. C., Zhang, K., Xu, L. D., & Chen, Y. (2020, January). A cooperative effect-based decision support model for team formation. *Enterprise Information Systems*, *14*(1), 110–132. Retrieved 2020-08-28, from https://doi.org/10.1080/17517575.2019.1678071 (Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/17517575.2019.1678071) doi: 10.1080/17517575.2019.1678071

Jordan, M. I., & Mitchell, T. M. (2015, July). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260. Retrieved 2021-03-23, from https://science.sciencemag.org/content/349/6245/255 (Publisher: American Association for the Advancement of Science Section: Review) doi: 10.1126/science.aaa8415

KNIME Analytics Platform. (n.d.). Retrieved 2020-03-18, from https://www.knime.com/knime-analytics-platform

Kofod-Petersen, A. (2014, October). *How to do a Structured Literature Review in computer science.* (Version 0.2)

Language Support | API. (n.d.). Retrieved 2020-05-06, from https://cloud.google.com/natural-language/docs/languages (Library Catalog: cloud.google.com)

Language Support | AutoML. (n.d.). Retrieved 2020-05-06, from https://cloud.google.com/natural-language/automl/docs/languages (Library Catalog: cloud.google.com)

Liu, L. (2019, December). *Discover insights from text with AutoML Natural Language, now generally available.* Retrieved 2020-05-07, from https://cloud.google.com/blog/products/ai-machine-learning/machine-learning-feature-automl-natural-language-generally-available/ (Library Catalog: cloud.google.com)

Macintosh, A. (1999, July). *Knowledge Management.* Retrieved 2020-12-18, from http://www.aiai.ed.ac.uk/~alm/kamlnks.html

Maffeo, L. (2019, October). *4 Easiest-to-Use Free and Open Source Text Analysis Software.* Retrieved 2020-03-17, from https://www.softwareadvice.com/resources/easiest-to-use-free-and-open-source-text-analysis-software/ (Library Catalog: www.softwareadvice.com

Section: Business Intelligence)

Mayring, P., Fenzl, T., & Lemke, S. (n.d.). *QCAmap | Qualitative Content Analysis.* Germany: Association for Supporting Qualitative Research ASQ. Retrieved 2020-03-17, from https://www.qcamap.org/

McDonald, P. (2008, April). *Introducing Google App Engine + our new blog.* Retrieved 2021-03-23, from http://googleappengine.blogspot.com/2008/04/introducing-google-app-engine-our-new.html

NL-API | How-to Guides. (n.d.). Retrieved 2020-06-25, from https://cloud.google.com/natural-language/docs/how-to (Library Catalog: cloud.google.com)

NLTK Demos for NLP. (n.d.). Retrieved 2020-04-26, from http://text-processing.com/demo/

Oxford Dictionary on Lexico.com. (n.d.). Retrieved 2020-12-09, from https://www.lexico.com/

PATresearch. (2019, October). *Top 26 Free Software for Text Analysis, Text Mining, Text Analytics.* Retrieved 2020-03-17, from https://www.predictiveanalyticstoday.com/top-free-software-for-text-analysis-text-mining-text-analytics/ (Library Catalog: www.predictiveanalyticstoday.com Section: Random)

Preparing your training data | AutoML Natural Language Documentation [Guide]. (2020, November). Retrieved 2020-11-05, from https://cloud.google.com/natural-language/automl/docs/prepare

Products and Services. (n.d.). Retrieved 2020-07-02, from https://cloud.google.com/products

Razno, M. (2019, April). Machine learning text classification model with NLP approach. *Computational Linguistics And Intelligent System, 2 : Proceedings of the 3nd International conference, COLINS 2019.*, 71–73. Retrieved from http://ena.lp.edu.ua:8080/handle/ntb/45487 (Publisher: Lviv Politechnic Publishing House)

Release notes | AutoML Natural Language. (n.d.). Retrieved 2020-05-14, from https://cloud.google.com/natural-language/automl/docs/release-notes (Library Catalog: cloud.google.com)

Release notes | Cloud Natural Language API. (n.d.). Retrieved 2020-05-14, from https://cloud.google.com/natural-language/docs/release-notes (Library Catalog: cloud.google.com)

Rodrigues, S., Oliveira, J., & de Souza, J. M. (2006). Recommendation for Team and Virtual Community Formations Based on Competence Mining. In W.-m. Shen, K.-M. Chao, Z. Lin, J.-P. A. Barthès, & A. James (Eds.), *Computer Supported Co-operative Work in Design II* (pp. 365–374). Berlin, Heidelberg: Springer. doi: 10.1007/11686699_37

Rodrigues, S., Oliveira, J., & Souza, J. M. d. (2004, July). Competence mining for virtual scientific community creation. *International Journal of Web Based Communities*, *1*(1), 90–102. Retrieved 2020-08-21, from `https://doi.org/10.1504/IJWBC.2004.004801` doi: 10.1504/IJWBC.2004.004801

Sannsyn. (n.d.). Retrieved 2020-03-17, from `https://sannsyn.com/`

Sarl, L. (n.d.). *Textable.* Retrieved 2020-03-18, from `http://textable.io/`

Sesam - Democratising Data. (n.d.). Retrieved 2020-08-27, from `https://sesam.io/index.html`

SharePoint. (n.d.). *SharePoint, Team Collaboration Software Tools.* Retrieved 2021-03-18, from `https://www.microsoft.com/en-us/microsoft-365/sharepoint/collaboration`

Slack. (n.d.). *Where work happens.* Retrieved 2021-03-18, from `https://slack.com/`

Teams, M. (n.d.). *Video Conferencing, Meetings, Calling | Microsoft Teams.* Retrieved 2021-03-18, from `https://www.microsoft.com/en-ww/microsoft-teams/group-chat-software`

Text Mining: The Beginner's Guide. (n.d.). Retrieved 2020-06-18, from `https://monkeylearn.com/text-mining`

Walber. (2014, November). *English: Precision and recall.* Retrieved 2021-03-14, from `https://commons.wikimedia.org/wiki/File:Precisionrecall.svg`

Wang, R., & Strong, D. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. *J. Manag. Inf. Syst.*, *12*(4), 5–34. doi: 10.1080/07421222.1996.11518099

Weiss, D., & Osinski, S. (n.d.). *Carrot2 - Open Source Search Results Clustering Engine.* Retrieved 2020-03-17, from `https://project.carrot2.org/`

What is Text Analysis. (n.d.). Retrieved 2020-06-18, from `https://www.ontotext.com/knowledgehub/fundamentals/text-analysis/` (Library Catalog: www.ontotext.com)

Řehůřek, R., & Sojka, P. (2009). *Gensim: topic modelling for humans.* Czech Republic: RARE Technologies Ltd. Retrieved 2020-03-17, from `https://`

radimrehurek.com/gensim/ (Repository: https://github.com/RaRe-Technologies/gensim)

NTNU
Norwegian University of
Science and Technology

bouvet