Utrecht University

Master Thesis

Business Informatics

Department of Informatics and Computing Sciences

ANALYSIS OF QUESTION TYPE CLASSIFICATION AND DISAMBIGUATION

Author:
A.L.L. Nijhuis (SID: 6962262)

Supervisors:
1st: Dr. M. J. S. Brinkhuis
2nd: Dr. G. Chen
Daily: C.J. van Toledo

July 3, 2023

ANALYSIS OF QUESTION TYPE CLASSIFICATION AND DISAMBIGUATION

Antoine L.L. Nijhuis
Business Informatics Department, Utrecht University
July 3, 2023
Supervised by Dr. M. J. S. Brinkhuis, Dr. G. Chen & C.J. van Toledo.

Abstract

This thesis proposes a research on question context analysis, utilizing Natural Language Processing (NLP) techniques. In past time, Question & Answer interactions were handled manually. Recent advancements in NLP and Machine Learning (ML) have created the opportunity to implement a plethora of Question & Answering Systems (QAS). These systems often utilize a classifier to classify the questions into types before answering them. The two most prominent classification methods historically are Rule-based models and Machine Learning models. Rule-based models utilize grammar rules and a dictionary to classify questions into categories and map these to the correct answer. Machine Learning models, such as neural networks, utilize mathematical equations to learn from a Question & Answer data set, with the intention of minimizing classification errors when matching question and answer pairs. This research aims to discover how popular classification techniques perform in a restricted domain environment, with regards to question type recognition and question enrichment. These two tasks are performed on a large structured Dutch data set and a public English data set. To determine how the classifiers score on these two tasks, two rigorous metrics are applied to determine classification power; F1 Score & Area under the ROC surface (AUC). Results suggest that question ambiguity can be recognized with an F1-score upwards of 90%. ML techniques featuring deep learning perform best across both question type detection and question enrichment.

Acknowledgements

I would like to thank my daily supervisor for his involvement and guidance in this research project. Although it was a long and difficult process, I am happy with what I have learned along the way. I would also like to thank my first and second supervisor for being patient with me and for their constructive feedback. Finally, I would like to thank my fellow students for their support and feedback during this journey.

Contents

---

1 Introduction

Reliable customer service is vital for the success of an organization. It strengthens the relationship between company and customer/employee, and it ensures the longevity of these relationships (Lewis & Mitchell, 1990). It is often one of the most resource intensive departments for a given organizational structure. Furthermore, delivering consistent, high quality customer service is one of the most challenging jobs today (Derrar, 2019).

In the past, for each question forwarded, a customer service employee was required to answer in real-time. This was due to the nature of these interactions being either in person or over the phone. With the rise of the internet many enterprises have moved their customer service branch to web-based online platforms (Naumov, 2019). Utilizing online platforms for customer service is more cost efficient for the enterprise, because of the significant decrease in required manpower. The digitization of customer service also means more stress and strain on the customer service department. Customer service providers are now expected to help multiple clients at the same time.

Due to the medium changing to online platforms, the customer service department spends a lot of time answering questions through e-mail or other textual formats. Online interactions are much less engaging to an individual, due to a lack of emotional stimuli. This can cause emotional fatigue as a result and tension in the customer service department (Ishii & Markman, 2016). Additionally, many of these questions are repetitive in nature and require an employee to be available to answer that particular question.

To combat this increased strain on the customer service department, Question Answering Systems (QAS) were introduced. These systems are retrieval systems that utilize a Question-and-Answer archive to find a good answer for a given question (Bouziane et al., 2015). This task can be roughly split into the two sub-tasks of classifying a question to a certain class and matching this question to the correct answer for that given class. If these models are successful at recognizing similar questions and classifying them as such, the customer service department is alleviated as a result.

In order to classify these questions into classes, it can be useful to recognize its type beforehand (Hien et al., 2020) (Garg et al., 2008). We define a question "type" as the way in which a question is formulated. The specificity and detail of the required answer relies on the question type (Moldovan et al., 2003). A question such as "Who was the first man on the moon?", is very different from a question such as "Why did dinosaurs go extinct?". The first question can be answered directly and holds a single "correct answer", whereas the second question can be answered in many different ways. The types of questions that are encountered rely heavily on the domain. A question type can be considered an abstract concept. The following set of "question types" serves as an example and is by no means exhaustive:

(1) Closed binary: Questions that can only be answered with "Yes" or "No".
(2) Closed non-binary: Questions that have a concise correct answer.
(3) Open: Questions that require a more elaborate answer, often consisting of multiple sentences.
(4) Disjunctive/Tag: Questions that consist of multiple parts (or sentences) (Mishra & Jain, 2016) (Adelaide, 2022; White, 2019).

Question type	Example sentence
Closed (binary)	“Have you done your homework?” “Are you cold?” “Do you like ice cream?”
Closed (non-binary)	“Who broke this window?” “What is your favorite snack?” “Did you order pizza or pasta?”
Open	“Why did dinosaurs go extinct?” “What was it like to go through a divorce?”
Disjunctive/Tag	“Your friend is an experienced technician, isn’t he?” “I would like to know how I can apply for this job. Can you help me with this?”

Table 1: Question type examples

Classifying questions into types can help as a first step in the classification process. The possible question types depend on the domain and can be used to shape a local dictionary (Cruz-Blandón et al., 2019). The application of question type annotation and the use case of a local dictionary is addressed in Section 4.A relevant form of question type recognition that is more domain-specific is categorization by question topic. Each question is linked to a topic. This approach can be interesting to gain more insight into frequently occurring question topics and can also be used as a coarse classification process (Garg et al., 2008). The notion of question types is especially interesting when looking at one of the most prominent challenges in Natural Language Processing today, structural ambiguity (Jusoh, 2018). Structural ambiguity arises when a body of text can be interpreted in multiple different ways. In the context of QAS, structural ambiguity occurs when a question can be interpreted or answered in multiple different ways. In this paper these questions shall be referred to as “ambiguous” type questions.

1.1 Problem Statement

In an ideal situation, the QAS is fed a question as input that directly maps onto the correct answer in a data archive. QAS are excellent at handling clear, concise and recognizable questions that can be directly linked to an answer in the knowledge archive/data base (Lende & Raghuwanshi, 2016).

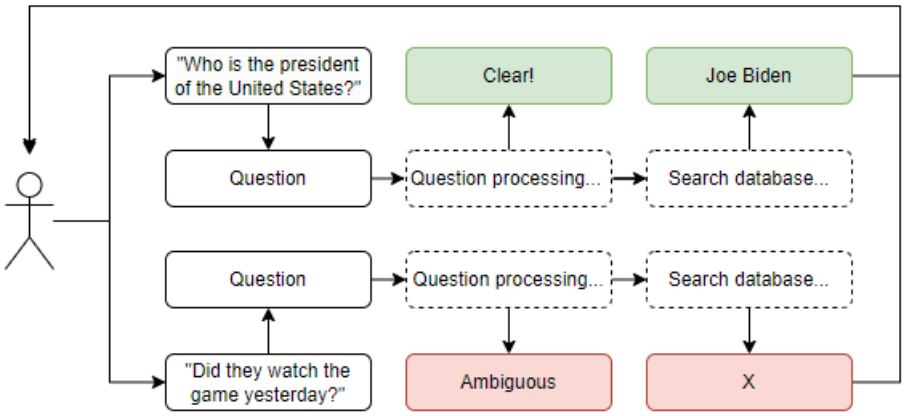

Figure 1: Simplified question answering process

When these systems are deployed in real life contexts however, it is very hard to regulate this process. More specifically, the processing of ambiguous type questions (such as Disjunctive/Tag questions) poses a significant challenge today (Jusoh, 2018). Every NLP technique to solve this issue can be sub-divided into two categories; Rule-based classifying techniques and Machine learning-based classifying techniques (Minaee et al., 2021). Research has been done to measure the effectiveness of Rule-based approaches (Tayyar Madabushi & Lee, 2016). However, this research focuses on characterizing questions based on the answer's content by applying an existing question taxonomy (X. Li & Roth, 2002). Moreover, the data set used in their experiment consists of clear questions in a public domain. There is research about Machine Learning approaches for QAS (González-Carvajal & Garrido-Merchán, 2021), but consider only clear question formats.

Research focused on question ambiguity specifically is lacking. Especially the comparison between Rule-based and Machine Learning-based techniques in closed/restricted-domain environments is scarce. Most recent research that has been conducted focuses on the comparison between different Machine Learning approaches on a large open-source data set (Kowsari et al., 2019), or looks at rule-based approaches for a closed-domain environment (Derici et al., 2015). Moreover, a large survey on question analysis work in the last decade reveals that most of the research in this field is conducted on English data sets (Q. Li et al., 2020).

1.2 Research Objective

This research sets out to add onto existing research by conducting a large Dutch case study. The Dutch data set used in this paper consists of over 240.000 e-mail exchanges in the past year. Although textual analysis has been conducted on Dutch data sets (Rouws et al., 2021), research on question type classification specifically is hard to find. This research focuses on the difficulties in classifying "ambiguous type" questions. The results give insights into the viability of detecting ambiguous type questions. Moreover, the findings serve as a reference points for customer service department with regard to question type distribution. This way, organizations that are similar to the one where this research is conducted, can get an idea of the structure of questions. Finally, this research intents to add onto existing research with regards to question disambiguation.

1.3 Research Questions

The central question this research aims to answer is the following:

- ***MRQ:*** *To what extent is it possible to enrich ambiguous type questions in a restricted domain environment?*

In order to answer this question in a structured manner, there are several sub questions that are relevant to this topic:

- ***SRQ1:*** *What are the characteristics of ambiguous type questions?*
  In order to disambiguate ambiguous type questions, the first task is to recognize these question types. After careful data exploration a protocol can be constructed to realize this. This protocol serves as guideline for characterizing ambiguous questions. The extent to which this is possible will dictate the direction of future research.

- ***SRQ2.1:*** *To what extent is it possible to automatically detect ambiguous type questions in a restricted domain environment?*
  Given a structured labeling protocol, how well can Machine Learning classifiers recognize the different question types? Is it possible to detect ambiguous type questions specifically? This is the first step of the two step process to answering the main research question. The second step being the enrichment of ambiguous type questions.

- ***SRQ2.2:*** *How does a Rule-based approach compare to a Machine Learning approach for question type classification in a restricted domain environment?*
  This question is relevant, because this research wants to add onto existing research by providing a comparison between Machine Learning and Rule-based approaches for question (type) classification.

- ***SRQ3:*** *How is question type classification in a restricted domain influenced by the language of the data?*
  Since we are conducting a large case study in Dutch, it is important to ensure generalizability of the research. That is why the same experimental steps are conducted on an English data set. The purpose of this is to strengthen the validity of the experimental results.

- ***SRQ4.1:*** *To what extent is it possible to enrich ambiguous type questions through context analysis?*
  Being able to enrich ambiguous type questions accurately allows customer service departments to fluently handle complex questions. Now these questions can be forwarded to the correct people or possibly be answered automatically. Furthermore the answer to this question adds onto existing research surrounding textual ambiguity.

- ***SRQ4.2:*** *How does a Rule-based approach compare to a Machine Learning approach for ambiguous type question enrichment?*

We hypothesize that Rule-based classifiers are better suited for both coarse and detailed question type classification in a restricted vocabulary space. This includes the recognition of question types. The resources available in a restricted domain are limited and the impact of expert knowledge is more directly applicable through this approach. However, we expect Machine Learning models, specifically those utilizing a "deep learning" approach, to facilitate a better disambiguation process. This is because context analysis is a more complex task in and of itself.

1.4 Thesis structure

This thesis is structured according to the following sections. In Section 2, the systematic literature review methodology is discussed. Section 2 covers introduces the reader to Question Answering Systems and NLP. In this Section the related work for this project is also discussed. Finally Section 2 gives an overview of the classification metrics applied in this research. Sections 3 provides an overview of the data sets used in this project. Section 4 covers the research methodology in detail. In Section 5 the results gathered are presented and discussed briefly. In section 6 the results of this research are interpreted and discussed in more detail. Furthermore the limitations and possible future work is discussed in this Section. See Figure 2 below for an overview of how this research is structured:

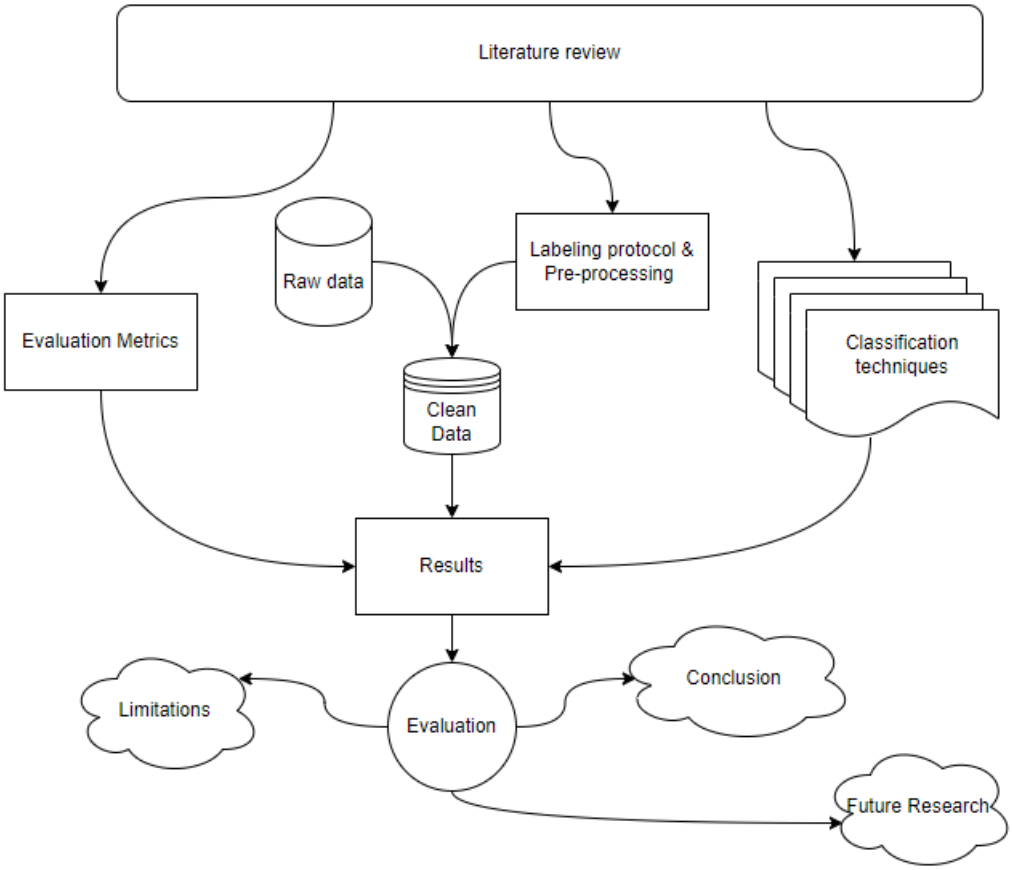

Figure 2: Research overview

2 Related Literature

2.1 Question Answering Systems

Question Answering Systems generally operate along three different phases (Allam & Haggag, 2012):

1. Question processing (extract question from question text)
2. Document processing (link the question to corresponding answer in the knowledge base)
3. Answer processing (answer question)

In question processing, we extract the "question" from the query text. The same question can be formulated in many different ways (Chowdhary, 2020). Some ways more ambiguous than others. A question can also be "hidden" in a larger text-body. Questions can be asked implicitly, for instance without a question mark "?" at the end of the question sentence. This is why it is vital to extract the question accurately in order to classify the question.

After having classified the question to a certain class, document processing is the act of linking the question to the corresponding answer text. The knowledge archive is screened for relevant documents based on keywords in the question. For the resulting documents, each paragraph/sentence is ranked by their probability of containing the correct answer. The result of this step is a set of possible answer sentences.

In the final step, the correct answer is extracted from the target document. A set of heuristics is applied to the set of possible answer to identify the correct answer. This answer is then carefully formulated back to the client in a way that easily understood. This research focuses on the first step (question processing) of Question Answering Systems. For an overview of the QAS phases, see figure 3 (Allam & Haggag, 2012).

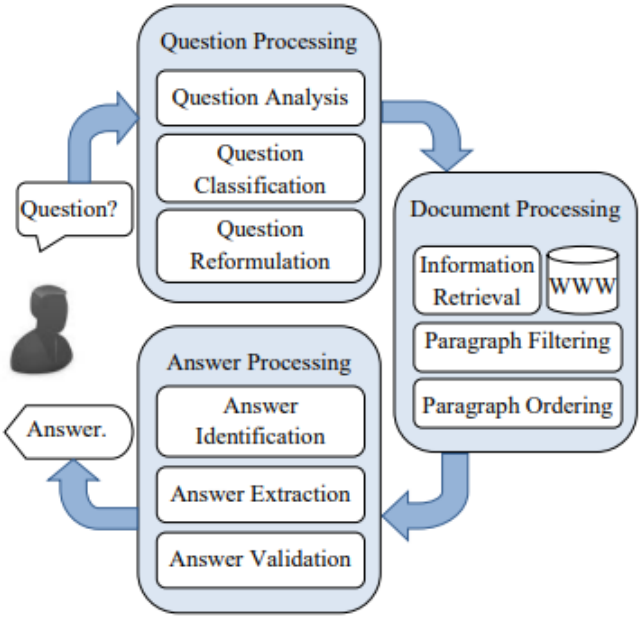

Figure 3: A diagram depicting the 3 phases for Question & Answer Systems, inspired by Allam et al, 2012

2.2 Natural Language Processing

Questions asked to customer service are in the form of human dialogue. A model can process this question by utilizing Natural Language Processing (NLP). NLP serves as a gateway between human dialogue and computers. NLP originated as an intersection between Artificial Intelligence and Linguistics (The scientific study of human language and its structure). It allows the model to characterize and explain the multitude of linguistic meanings of a question (Nadkarni et al., 2011). An example of an early NLP adaptation is a translator. This is a very simple one-to-one mapping where each word is translated into the designated language. A big obstacle for this approach is the existence of homonyms (a word with multiple meanings/translations) for instance.

It became evident that language is a hard concept to understand for computers. Language is ambiguous, language consists of very sparse date (many words are rarely used) and the way in which humans process language is not clearly understood (A. K. Joshi, 1991). Language can be analyzed in many different ways. We can analyze the words themselves, the tonality, the sentence structure (syntax). All these different lenses through which language can be observed attribute to the meaning of a sentence (semantics). To facilitate this process of language understanding, not in a human way, but in a computer-understandable way, we must look at language as data.

2.3 Rule based Question Classification

A Rule-based classifier utilizes rule-based grammar to imitate human understanding of sentences. Similarly to the example shown for Context Free Grammar, these rules consist of $if \rightarrow then$ statements. This usually implies that a human is directly involved in the step-wise development of such a model. This approach facilitates easy verification and maintenance of the model (Biswas et al., 2014).

A Rule-based classifier has a coverage level. In the case of a QAS, this is the percentage of "answers" in a body of text that are covered by a particular rule. These rules are not mutually exclusive. If multiple rules (queries) point toward the same answer, the correct class is usually determined either by a rule priority scheme (highest ranked rule determines the class), or by assigning weights to the rules (certain rule features carry more weight) (Mansoori et al., 2007).

Through these rule schemes, individual changes and updates are easily implemented without affecting the model as a whole. In the case of QAS this is an important feature, since the contextual vocabulary and knowledge base consistently changes throughout time. Rule based models can be altered and adjusted to the situation at hand, which makes this approach excellent for dealing with in-house, context dependent situations (Chiticariu et al., 2013). The downside of this flexibility, is that the solution is indeed context dependent and not easily scalable as a result (Diao et al., 2009). The rule-based approach does not require a large training set to be employable, but the development of the parser can be quite intensive and requires direct involvement of experts. (Gupta & Gupta, 2012). The flexible nature of this approach comes with costs.

A recent study on clinical phenotyping yielded strong results for Rule-based classifiers (Oleynik et al., 2019). It also implied that shallow machine learning techniques are more desirable than deep learning in a restricted domain. A rule-based system relying on question types to construct an initial classification shows that, supported by expert level knowledge on the

subject domain, rule base classifiers can be very powerful (Tayyar Madabushi & Lee, 2016). This research shows that a carefully constructed RB classifier can heavily outperform a variety of standardized ML-techniques (Convolutional Neural Networks & Support Vector Machines). What is particularly interesting is the absolute classification accuracy upwards of 97% that is achieved. A study on Mathematical type question classification did a comparison on rule-based classification between five different rule-mining techniques (Yuhana et al., 2019). Results show minimal differences in performances across sophisticated rule-mining techniques.

2.4 Machine Learning Question Classification

Most modern Machine Learning approaches in QAS utilize large globally available datasets to build a universal classifier through *deep learning* (Minaee et al., 2021). This is primarily because ML classifiers, unlike to Rule-based systems, require large amounts of training data to perform well.

The Stanford Question Answering Dataset (SQuaD) was designed for reading comprehension (Rajpurkar et al., 2018; Rajpurkar et al., 2016). This data set was trained on Wikipedia articles and contains answerable as well as unanswerable questions. The Compositional Freebase Questions (CFQ) data set was designed using automated rule-based question generation to minimize human error (Keysers et al., 2019). It was designed to measure compositional generalization as a measure of good question & answer pairs. The MS MARCO data set contains questions asked through Bing & Cortana (Nguyen et al., 2016). This data set also contains queries with answers manually typed by humans instead of only containing answers extracted from a passage (knowledge base). The TriviaQA dataset published by (M. Joshi et al., 2017), collected question & answer pairs from 14 different trivia websites. It holds textual evidence for the given answers from various sources such as Wikipedia articles.

.

Recent advancements in semi-supervised learning that focus on pre-training models bidirectionally are especially relevant in the Question Answering domain (Devlin et al., 2018). BERT is considered State-Of-The-Art when considering the domain of Question Classification within the NLP landscape. Unlike directional models, which process the text corpus sequentially (from left to right), BERT utilizes a Transformer mechanism that learns relationships between words bi-directionally. Adaptations of the BERT model exist for many languages, including Dutch (Delobelle et al., 2020b; de Vries et al., 2019). A deeper explanation of BERT and the transformer mechanism follows in the next section.

2.5 Hybrid Approach

A Hybrid approach to question classification is essentially the act of combining both Rule-based and Machine Learning techniques to build a classifier. This can be done in both orders. A study on the combination of associative rule mining with Naïve Bayes shows that a hybrid approach of these two techniques outperforms all of the regular techniques tested for (such as NB, J48, Associative based classification) (Hadi et al., 2018). A recent study, on the classification of questions posted by first year medicine students, using a relatively small data set, is also in favor of a hybrid approach (Harrak et al., 2019). It emphasises that complementing an existing RB classifier with ML techniques outperforms each of these techniques applied individually. Another study reveals the validity of unsupervised learning with minimal assistance of experts to form a supervised baseline (Haj-Yahia et al., 2019). A successful hybrid approach utilizes the

flexibility of RB techniques and the scalability of ML. Unfortunately due to the limited resources available for this project, we shall not attempt at constructing such a hybrid system.

2.6 State-Of-The-Art Machine Learning Techniques

In this section several popular Machine Learning techniques for text classification will be described. This includes textual embedding as well as language models.

Word2Vec

Back in 2013 the first technique for efficiently representing words as continuous vectors was introduced by Mikolov et al. This technique that utilizes neural networks to learn word associations is called Word2Vec (Mikolov et al., 2019). Word2Vec uses a sequence of numbers (vector) to represent each distinct word. Then a cosine similarity formula is used to measure the level of semantic similarity between words. It learns these word vectors in two ways:

1. **Continuous Bag-of-Words (CBOW):**
   The CBOW method is a neural network that tries to predict a word by looking at the context of surrounding words. The context is used as input, now the model tries to predict the target word. The projection layer takes the weighted sum of the context word vectors ($w_{t-2}...w_{t+2}$) and tries to "project" this sum into a continuous vector (output). For example, the CBOW model tries to predict the word "toe" from the context "I hit my " on the doorstep".

2. **Skip-Gram Method:**
   The Skip-Gram method works in the opposite way. It tries to predict the surrounding context words by looking at the input word vector. This method takes the dot product between a weight matrix and the input vector (w_t). Now the output layer takes this result to compute the probability of words appearing in the surrounding context. Take a look at the figure below for a visual representation of each method **??**.

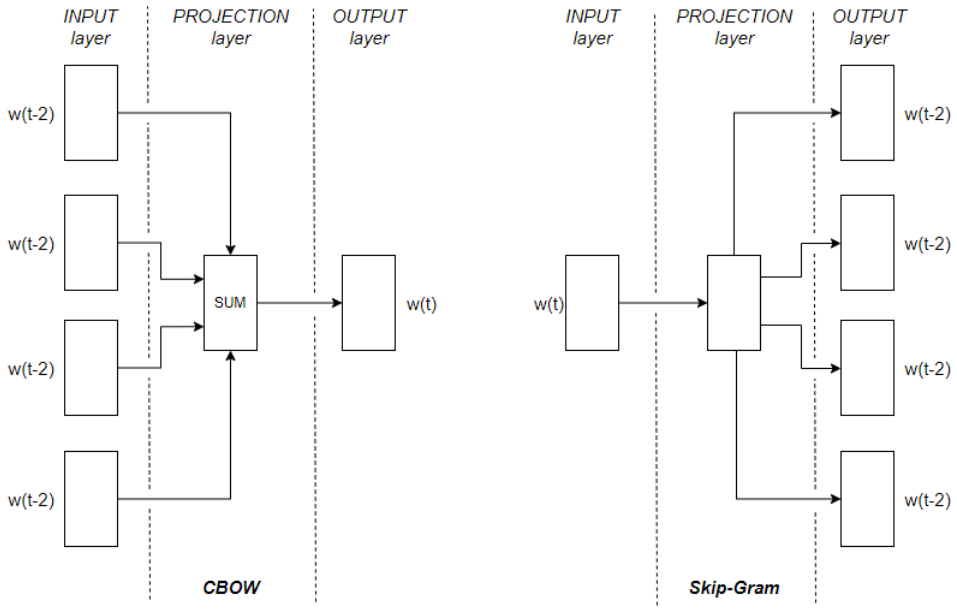

Figure 4: Word2Vec model

GloVe

GloVe (Global Vectors for Word representation) is an unsupervised learning algorithm used for obtaining vector representations for words (Pennington et al., 2014). This algorithm derives the relation between words statistically. It uses a co-occurrence matrix to track how often a pair of words occur together. The way it differs from Word2Vec is that it looks at this co-occurence globally instead of locally like Word2Vec does. Despite this difference, recent research suggests that both techniques perform similarly for major NLP tasks (Dharma et al., 2022). The difference in performance is mostly dependent on the type of data used.

Bi-directional Language Models

ELMo (Embeddings from Language Models) is a predecessor of BERT and features a deep contextualized word representation (Peters et al., 2018). It computes "character" vectors on top of a two-layer bidirectional language model (BiLM). It utilizes Convolutional Neural Networks to represent words as vectors. Unlike regular word vectors, character vectors capture the inner structure of words. This way it can relate similar words much more rapidly.

These vectors are the input for the first layer of the BiLM. In this first layer the model utilizes context before the word vector (forwards pass) and context after the word vector (backwards pass). The results from this first layer can be interpreted as "intermediate" word vectors. These intermediate word vectors are then fed into the second layer for the BiLM. Here the process is repeated one more time to achieve an additional intermediate word vector. Then the weighted sum is taken from both of these intermediate vectors as well as the original word vector. This is the final result of the ELMo model 5.

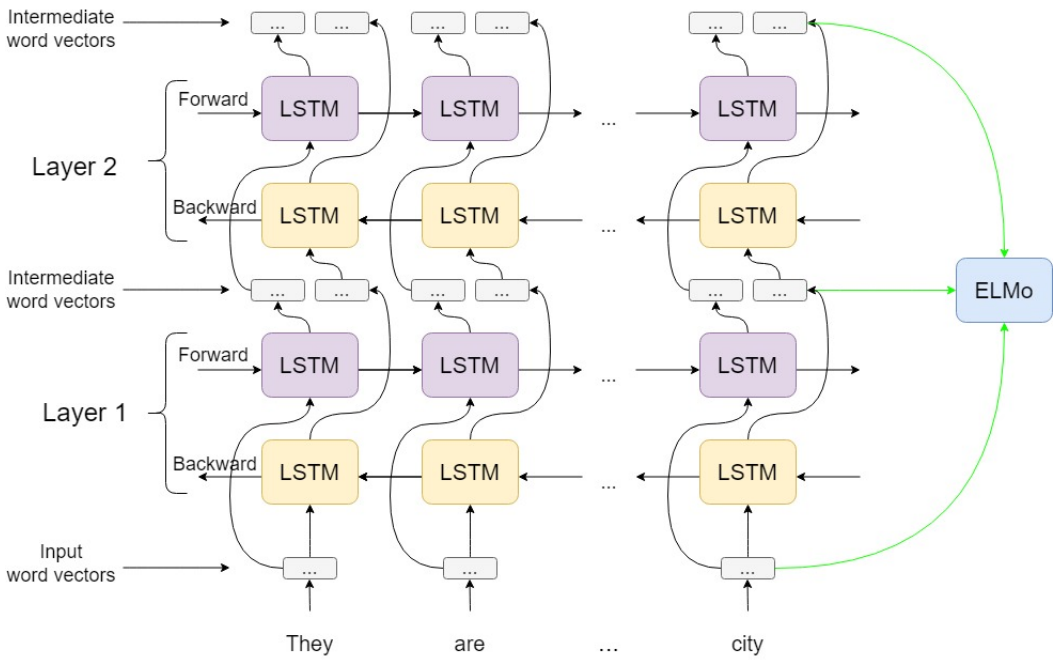

Figure 5: Embeddings from Language Models (ELMo) explained, inspired by (Huang & Zhao, 2020)

The way BERT differs from ELMo is that it does not use character vectors as input, but so-called "subwords" instead. This strikes a unique balance between the granular character vec-

tors and coarse word vectors, allowing for smaller vocabulary space than ELMo. A recent study implies there is a clear advantage to using word vectors instead of character vectors (Al-Rfou et al., 2019). Another difference is that BERT is based on the deep learning model "Transformers". It connects and understands the relationship between all words in a sentence, rather than processing them one at a time (ELMo).

Transformers

Transformers are language models that were originally used by Google (Vaswani et al., 2019). These models are a significant improvement over Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) used at the time for NLP tasks. Transformers do need require sequences of data to be processed in any specific order. By finding patterns mathematically, transformers eliminate the need for a labeled data set. This enabled these models to handle much larger data sets and facilitate pre-trained models such as BERT.

Transformers make use of a ML concept known as "attention". This can be seen as the ability to highlight and use salient parts of information. In NLP tasks this is the process of dynamically highlighting and using important parts of the word context to derive its meaning. Some early successful adaptations of this attention mechanism date back as early as 2014 (Bahdanau et al., 2018) (Luong et al., 2015). Because of the attention concept, the dependency on the in-between distance between words is longer a limitation. A formulaic expression can be seen below.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

If a transformer model is given a sentence and it tries to give meaning to a word, the attention mechanism works as follows. Each word in the sentence is attributed its own query, key and value. Essentially these variables are separate linear transformations of the input sentence. It uses these input variables to decide which part of the context to focus on. For a more thorough explanation, consider the paper published by Chan, 2021.

BERT

A new transfer learning technique called BERT (Bidirectional Encoder Representations for Transformers) has arisen as popular state-of-the-art language model. BERT has an entirely different approach to classification then regular machine learning. BERT is given millions of sentences at training time. It then tries to predict a random sample of missing words from these sentences. After repeatedly training on this huge text corpus, BERT learns how words and sentences fit together grammatically. It is able to cope with a multitude of Natural Language Processing challenges of today, such as Question Answer, abstract summarization, sentence predicting and much more (Devlin et al., 2018) (Liu & Lapata, 2019). There are a number of factors why BERT is able to perform so well on many of these tasks.

Firstly, it is pre-trained on a lot of data. The original model comes in two sizes: BERT-base and BERT-large. BERT-base was trained on a large data set of books with approximately 800 million words. BERT-large was trained on English Wikipedia with about 2500 million words.

Both of these "pre-trained models" are made publicly available. Because of these astonishingly big data sets, these models have an incredible "memory" to draw from.

Secondly, BERT is able to utilize a word's context very well. So instead of simply returning the same vector for a word (like most Machine Learning models), BERT returns a different vector depending on how the word is used. This is why BERT excels at context-rich language problems. BERT is the first model to rely fully on the "self-attention" mechanism introduced by Transformers.

GPT

The third generation of OpenAI's language model GPT was released in 2020. What makes GPT3 unique, is that it was the first model that can generate large texts without supervision. It learns language through repeated exposure like a child would. GPT3's results can be hard to distinguish from real human writing at times (Elkins & Chun, 2020). InstructGPT is a modified version of the original Generative Pre-trained Transformer 3 (GPT-3) model (Ouyang et al., 2022). InstructGPT excels at following human-written instructions. Its original purpose was to provide enterprises with a powerful tool for NLP tasks. It works in a very similar way to GPT-3, except it is more aligned with human preferences. Its core difference with GPT3 is that InstructGPT uses human feedback to fine-tune its process. It is trained on specific datasets to perform specialized tasks.

Most recently, toward the end of 2022, ChatGPT was introduced by OpenAI. ChatGPT is a sibling model to instructGPT and their underlying structure is very similar. ChatGPT however, is optimized for conversational contexts. It has the ability to access outside API's. This allows ChatGPT to provide accurate and informative answers on nearly any topic. Its application domain is very wide-spread. An extensive empirical experiment on 20 popular data sets studies ChatGPT's performance (Qin et al., 2023). This study revealed that ChatGPT is very proficient at NLP tasks based on dialogue and reasoning, but still struggles with more specific tasks like sequence tagging or sentiment analysis (Kocoń et al., 2023).

2.7 Question Detection

There is not much research on the topic of question detection. This is most likely due to the fact that most of the research surrounding questions in NLP is related to question answering. These experiments assume that the input is a question sentence, not a corpus of text. In this research the input is comprised of e-mails. This implies that the question is somewhere within a larger corpus of text. Moreover, it is not certain that there is a question present whatsoever.

In 2004 research by Shrestha et al. looks into ways to automatically detect questions in e-mail context (Shrestha & McKeown, 2004). Their methods comprised of Part-of-speech (POS) tagging and a rule-based algorithm achieved an F_1 score of 0.82. Each sentence is classified as a "question" or "non-question" based on POS features. The rule-based algorithm is then used to learn these rules (Cohen, 1995). This approach is still widely considered today.

Research that focuses on questions posted in online forums looks into the structure of questions (Cong et al., 2008). This study concluded that many questions are formulated in a non-trivial way. Simply looking at keywords like "What/Why/When/Who/Where/How" (5W1H)

or a question mark did not suffice. It concludes that 30% of questions posted did not include a question mark. They use a rule-based approach for question detection (Cohen, 1995). Labeled Sequence Patterns (LSP) is used to characterize questions instead of only using Part Of Speech (POS) tagging. The discovered patterns are used to train a classification model for question detection. Their study concludes the LSP approach for question detection outperforms the POS tagging approach for question detection in online forums. Another more sophisticated approach is introduced by (Wang & Chua, 2010), where different variations of pattern mining are used. Results of pattern mining are much better than using just regular expressions with (5W1H). But when evaluating question marks in the regular expressions, we can see that the difference is not that significant.

A study performed at TU/Delft that compares a plethora of question detection methods, such as naïvely looking at regular expressions (5W1H) and more complex methods proposed by (Shrestha & McKeown, 2004) was published by (Kwong & Yorke-Smith, 2012). This study also proves that using a naïve method of looking at question marks and keywords performs very adequately. Moreover this naïve approach results in a low run time, which is crucial for large messy data sets.

2.8 Question Ambiguity

An ambiguous question in NLP terms, is a question that does not map onto a specific query. More specifically, it could be interpreted in multiple ways, map onto several different answers or its subject is unclear altogether. It is essentially a semantic gap between the question intention and how this is conveyed. A recent study on the current challenges in NLP separates between several ambiguity types, such as structural ambiguity and lexical ambiguity (Jusoh, 2018). It is described in this study that knowledge graphs (constructed with help of experts) can be effective for dealing with textual ambiguity. A recent study proposes the idea of using weighted representation of words to disambiguate questions computationally (Zhu et al., 2019). This is similar in effect to Named Entity Recognition "NER", which is applicable to both Rule Based and Machine Learning classifiers. In the RB approach, a pre-defined set of rules is used to determine the entity of a word. Whereas in ML, statistical models are used for entity recognition (Goyal et al., 2018). A comparison of these two approaches show that both are perfectly viable (Gorinski et al., 2019). An interesting alternative approach for dealing with ambiguity is proposed by (Min et al., 2020). First the ambiguous question is split up into all its different possible interpretations. Then each interpretation is suggested to the client paired with its respective answer. ChatGPT uses a similar approach, where it asks a clarifying "follow-up" question to the user in the case of question ambiguity.

2.9 Evaluation Metrics

This research focuses on the comparison of classification techniques. A variety of evaluation metrics are applied to evaluate these classification techniques. Two metrics that are commonly used for testing the effectiveness of classifiers are the AUC, F1-score (Grandini et al., 2020).

Aurea Under the Curve / Confusion Matrix

The first method of evaluation is based on the "Area under the curve" (AUC) statistic (Vanderlooy & Hüllermeier, 2008). A receiver operation characteristic curve (ROC) shows the performance of a binary (positive or negative) classifier at all classification thresholds. It produces an aggregate measure of performance for all the different classification thresholds. It is a scale-invariant and threshold-invariant measure which is useful for generalizing and anonymizing the experiment results (Ling et al., 2003). It plots a True Positive rate (TPR) and False positive rate (FPR) onto a graph. These rates rely on the following parameters:

- True Positive (TP), the number of positive classifications that are correct.
- True Negative (TN), the number of negative classifications that are correct.
- False Positive (FP), the number of positive classifications that are incorrect.
- False Negative (FN), the number of negative classifications that are incorrect.
- $TPR = TP/(TP + FN)$, also known as "Recall"
- $FPR = FP/(FP + TN)$

These rates can be plotted into a confusion matrix 6 and can be used to calculate the Accuracy, Recall and Precision. These are famous classifier metrics to measure performance. Naturally, as the classification threshold rises, more items are categorized as positive, increasing both the TPR and FPR as a result. To evaluate all of these threshold values simultaneously we utilize AUC. (Ling et al., 2003). Essentially, the AUC method can be interpreted as the degree to which a classifier can distinguish between two classes. This metric ranges from 0.0 to 1.0 (0% - 100%). It captures the entire two dimensional space under the curve.

		Actual Class	
		Positive **(P)**	Negative **(N)**
Predicted Class	Positive **(P)**	*True Positive* ***(TP)***	*False Positive* ***(FP)***
	Negative **(N)**	*False Negative* ***(FN)***	*True Negative* ***(TN)***

Figure 6: Confusion matrix

Multi-class

A large portion of classification metrics are defined for binary classification problems. Since we are working with categorical data, this can be considered as a multi-class classifying problem. Fortunately the Area Under the Curve (binary-)metric can be extended to account for multi-class problems. Also referred to as "Volume Under the ROC Surface" (VUS) (Ferri et al., 2003). A rigorous way of doing this is by constructing a vector $d = k(k-1)$ accounting for the degrees of freedom (where $k =$ the number of classes in the set). This requires a total of $\frac{K(K-1)}{2}$ binary classifiers. When an instance is to be classified, a (weighted) voting procedure is performed among each of these binary classifiers (Aly, 2005).

Although this approach is very accurate, it becomes increasingly computationally complex as the number of classes rises (Ferri et al., 2003; Landgrebe & Duin, 2006). The cost of such classifiers can be estimated as follows; Let R denote the confusion matrix, Let $p(i)$ denote the frequency of class i and let $C(i, j)$ denote the cost matrix for classes i and j‘. The total cost can be approximated as follows:

$$\text{Cost} = \sum_{i,j,i \neq j} p(i) \cdot C(i,j) R(i,j)$$

A less computationally complex way of doing this is by considering the classifying problem over multiple classes as "One vs All" (Galar et al., 2011; Narkhede, 2018). This means that for each class in the set, we measure the classifier's ability of distinguishing between a particular class and the remainder of classes. Consider a multi-classification problem example with the set {A, B, C, D}. This multi-class instance is broken down into four binary classification tasks as follows:

[1] {A} vs {B,C,D}
[2] {B} vs {A,C,D}
[3] {C} vs {A,B,D}
[4] {D} vs {A,B,C}

Each task is performed by the same classifier. For each task the performance is then measured according to the binary classification metrics mentioned earlier (accuracy, precision, recall). Because we iterate the binary classifier over multiple tasks, we end up with multiple metrics (one for each task). To average these results into a single metric, we can use the (weighted) arithmetic mean (Tuychiev, 2021).

F1-score

Using an AUC is useful in cases when the exact threshold is not guaranteed in real application, so it gives you an aggregate metric across all threshold values. The AUC takes scores as input, whereas F1 takes predicted classes as input. This implies that in an imbalanced dataset, the F1 score performs better in general. The F1 score is applicable for any particular point on the ROC curve. Instead of using the FPR and TPR, it uses precision and recall. the F1 score is a number between 0 and 1, describing the harmonic mean of precision and recall (Fujino et al., 2008).

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

To adapt the F1-score to multi-class classification, we use the same approach as for AUC. Consider the following confusion matrix for the following set of features $\{A, B, C, D, E\}$.

Predicted Class \ Actual Class	A	B	C	D	E
A	43	3	6	0	0
B	0	57	1	8	0
C	0	4	61	4	0
D	14	2	3	66	0
E	26	5	3	3	34

Table 2: Confusion matrix example for calculating F1-score

The prediction success for each class is measured separately. The F1 score for any given class can now be calculated as follows (we take class B as an example):

Precision (class = B)	0.86
Recall (class = B)	0.80
F1 Score (class = B)	0.83

Table 3: F-1 Score calculation for class B

When using a weighted alternative, prioritizing either precision or recall, it is straightforward to fine tune and optimize this metric. Another advantage of this classifier is that it is easy to explain the results it provides. In this results section, we use the values provided in a confusion matrix to construct classification reports. These reports cover the precision, recall and f1 scores for a classifier. Furthermore they tell us about the Accuracy (Percentage of correct classifications of all classified samples).

3 Data

This Section provides an overview of the data, including information about the data sources, the data format and some key characteristics of the data. This experiment uses two data sets, a Dutch data set and an English data set. First, the Dutch data set and the English data set are described. Afterwards, the labeling process is introduced briefly and some differences between the Dutch and English data sets are discussed. A detailed description about the construction of the labeling protocol is discussed in the Methodology section.

3.1 P-direkt data set (Dutch data set)

This research is carried out at the Analytics Lab department within the P-direkt organization. P-direkt concerns itself with human resources and acts as a service provider for the Dutch Ministry of Internal Affairs. This department specializes in employee data processing and ensures privacy preservation of its clients. This implies that part of the data set used in this project is anonymized for publication purposes. This is to prevent privacy infringement. The context of this experiment is a customer service interaction. The client is an employee of a governmental department (the person asking a question), the respondent is an employee within P-direkt (the person answering the question). Here follows an altered example snippet, displaying the data's structure:

Data sample (Translated)	
ID	1
Time	11:48:12
Date	03.04.2021
Class (documented)	Question → Finances → IKB*
Class (code)	A0435**
Question subject	Monitor to work from home
Contact medium	E-mail
Question content	Is the consumption of a new computer monitor for my workplace at home included in the IKB for teleworking space? Can I submit a bill for this? I would like to hear from you. [SIGNATURE]
Answer subject	Monitor to work from home
Answer content	Dear [PERSON], On the 3rd of April, 2021, you have contacted P-direkt. You wonder whether the consumption of a computer monitor can be included in the IKB-submittal. Can I include my monitor in my IKB-submittal? .....[ANSWER***]..... You can reach us from 08:00 AM to 22:00 PM on weekdays. [SIGNATURE]

Table 4: P-direkt Data sample (anonymized)

* *This is a budget that is part of employment agreement. It usually consists of a holiday allowance, lifecourse allowance, end-of-year bonus and extra-statutory leave.*
** *This is a code referring to the department responsible for handling this request.*
*** *The full answer message in included in appendix C*

The data set available for this experiment consists of over 140.000 e-mail exchange instances between clients (the person asking a question) and respondents (the person answering the question). These question and answer (Q&A) exchanges can consist of multiple back and forth messages. All records are written in Dutch. A large portion of these records have been tagged based on topic of the question. Each topic is related to the department within P-direkt HR services that is best suited to answer this question. This tagging process is done manually by the first respondent. This respondent is only responsible for tagging the question and forwarding it to the responsible department. Furthermore each mail record holds various features of the interaction such as the date, content and the labels mentioned earlier. Here follows an altered example snippet, displaying the data's structure:

3.1.1 Descriptive Analysis

Before making any type of changes to the raw data set or doing any type of pre-processing, a Descriptive Analysis is performed on the data set. Some of the key characteristics of the data are summarized. From this we obtain a number of measurable characteristics of the data. We can measure the central tendencies such as the mean, median and mode. Furthermore we can obtain the variability of the data. Also the frequency within the distribution can be measured by keeping a count of certain features, such as the word frequency, which is discussed later.

3.1.2 Descriptive statistics

When looking at the complete data set consisting of mails in raw textual format, a number of interesting observations stand out. Although these mails represent "questions", forwarded to the customer service department, only 170.000 out of the 240.000 instances contain the question mark "?" symbol (roughly 70%). The questions are initially given a tag by the client. This tag corresponds to one of the following six question topics: "Mijn gegevens", "Verlof", "Verzuim", "Financiën", "Werk", "Overig".

Question Tag (translation)	Total Percentage
Financiën (finances)	31%
Werk (Work)	19%
Verlof (Leave of absence)	19%
Mijn Gegevens (Personal information)	13%
Overig (Other)	12%
Verzuim (Absence)	6%

Table 5: Question Tag Distribution (Dutch data set), translated

Out of these main topics, approximately 1/3rd of all questions forwarded are finance related. Whereas only 19% is related to absence from work. This is related to the type of questions that occur most frequently. For instance, many financial questions are formulated according to the "Open Question" type, characterized by the keywords "why"/"how"/"what". Furthermore, approximately 35% of financial related questions are complaints/requests formulated as questions. An example of such a question is the following: "Could it be that there was a mistake in calculating this month's salary?". This may seem like a question at first, but it is actually a request to fact-check some information. In Section 3.3.1 a more complete overview with examples of the different types of questions is provided.

A large majority of questions contain "reference words", pointing to either personal or factual information. More often than not this personal information can be extracted through the sender address. However, more trouble arises when factual information is being referenced. This is because this information can be in any of the sentences either following or preceding the question mark "?". After having inspected the questions containing reference words more clearly, the following was observed. If the factual information required is present in the mail, in nearly all cases this factual information can be found prior to the question sentence. In other words, when scanning for factual context surrounding a question, it is an effective strategy to consider only the previous sentences. This approach also aligns well with the resources available for this project.

3.2 Enron Data set (English data set)

In this research, a number of experiments are repeated on an English (public) data set (Enron Corpus) (Klimt & Yang, 2004). The Enron Corpus consists of a large set of email exchanges. After a generic cleaning procedure, this data set holds approximately 200.000 usable email messages. This data set was chosen, because it is somewhat similar in nature to the Dutch data set used in this experiment, especially after the necessary cleaning steps. Since the data is labeled manually, a large data set is preferable. This way extensive cleaning can be done without compromising the number of absolute samples used.

The purpose of this data set is to strengthen the validity of the experimental results. Due to the anonymization of the Dutch data set, it is not possible to show all the intermediary steps that lead to the result. Since the Enron data set is publicly available, it is possible to show much more of the experimental process. This data set also ensures reliability of the experiment, since it is now possible for others to repeat the experiment. This data set also ensures external validity, since the results are more generalized because English data is included. Given the scope of this project and the fact that this data set is complementary to the Dutch data set, the observed statistics for the Enron data set are left out.

3.3 Labeling protocol

The data for this experiment has been manually annotated. To do this, a suitable labeling protocol is required, to ensure reproducibility and validity of the experiment. This labeling protocol has been set up with the help of domain experts in the context of question answering in a restricted domain. In this section the labeling protocol is only introduced briefly. This is helpful in order to investigate some interesting differences between the Dutch data set and English data set. The extended labeling protocol can be found in Section 4.1.1.

Label	Reasoning
Clear [1]	no context necessary
Clear [2]	one previous sentence required
Clear [3]	two previous sentences required
Ambiguous	too complex to answer, regardless of context
Other	not relevant for this research

Table 6: Final labeling protocol, used to annotate both the Dutch and English data sets

3.3.1 Annotation examples (English data set)

To clarify how this protocol is applied in practice, here follow some examples using this protocol to label real data points used in this experiment. These are arbitrary annotation examples from the English data set. There are always edge cases that do not clearly map onto of the labels, this is even the case for experts with good domain knowledge. In this case, the label "Other" is applied as well. Consider the table below, with examples of each label type.

Question	Label	Reason
You did what just about every one of us would have done I'm guessing. It's part of the game. Is it frustrating?	Ambiguous	Context Missing [3]
Now add another 1000. Now add 10. What is the total?	Ambiguous	Context Missing [3]
I'll watch it on TV. I'm already buying a plane ticket to go home, I'm not buying another to fly to Atlanta. When do you leave for San Diego?	Clear[1]	One sentence required
That won't last until Christmas. I understand that there may be a deal on it at a large warehouse store, but I'm not allowed to tell you where. Do you remember the name of the protagonist of "Green Eggs and Ham"?	Clear[1]	One sentence required
I would like to firm a date for our follow up meeting. We had discussed your coming to Houston to meet at our office on Tuesday July 24. Does that date work for you ?	Clear[2]	Two sentences required
I have residents on the west side of the village who need someone to come over and check on her cat, house, get mail, etc when they have to leave town. They could have to leave on short notice and would probably not be gone for more than a week. Any suggestions or volunteers?	Clear[3]	Three sentences required
Danny Clark, who used to work in Deal Clearing & Documentation for ECT a couple of years ago, is in EES as far as I know. He was knowledgeable and meticulous, understood trading as he was a risk book administrator in a previous life. Have you considered him?	Clear[3]	Three sentences required
Check out Half.com for HALF price on Music, Books, Movies & Games! Win The Fantasy Football Trip Of A Lifetime! Love to surf the cbs.sportsline.com site?	Other	Advertisement
(1) I haven't seen you two in forever! (2) P.S. (3) - Eric, do you remember the name of the hotel that the four of us stayed at when we went to the Texas game last year?	Other	Parsing Error

Table 7: Annotation examples

3.3.2 Data distribution

Because of differences in the origin of the data sets, there is a difference in the spread of the labels. Noticeably, there are more ambiguous questions among the Dutch data set. This is most likely to the degree of formality. The Dutch data set is much more formal and structured. It is also noteworthy that there is a much larger set of label "Other" for the English data set. This is mainly due the fact that there are many more advertisements in this data set. This makes sense because the English data set covers personal e-mail accounts. Also this data set has more edge cases than the Dutch data set, partly because the labeling protocol was originally designed for the Dutch data set. The table below gives an overview of the (normalized) label distribution for both data sets, based on 1000 labels each:

Label	Percentage (Dutch)	Percentage (English)
Ambiguous	31,6%	12,3%
Clear[1]	25%	18,5%
Clear[2]	19,4%	9,3%
Clear[3]	12%	5,7%
Other	12%	54,2%

Table 8: Data label distribution

To give a clear visual representation of these differences, consider the pie-chart comparison below. It should be clear that the English data set includes many more label type "Other" samples than the Dutch data set. This is expected, since the data set is not as structured as the Dutch data set. The questions analyzed in the Dutch data set are largely set up in structured forms, whereas the questions in the English data set are informal mail exchanges between colleagues.

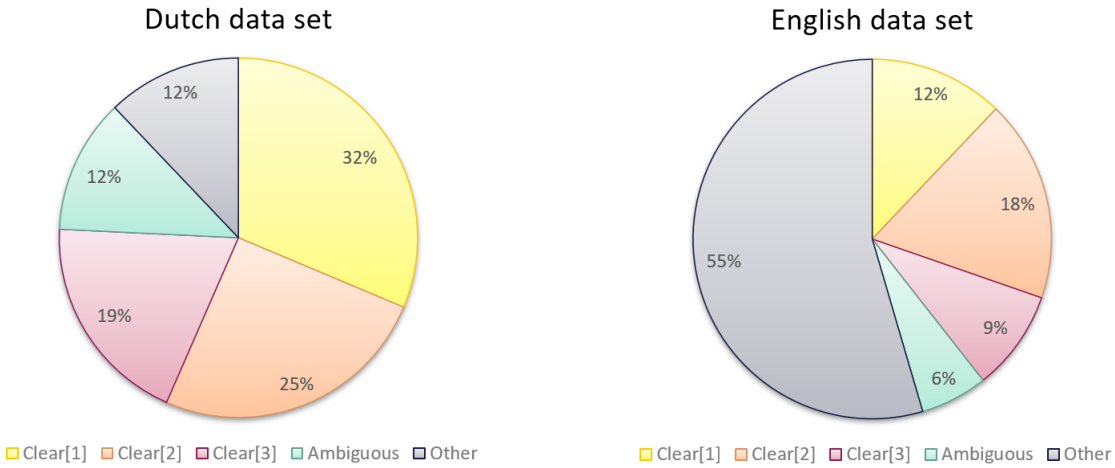

Figure 7: Label distribution pie-chart comparison between Dutch and English data

4 Research Methodology

This Section describes the experimental approach for the question type classification, the comparison of Rule-based classification and Machine Learning-based classification as well as the question enrichment step. A labeling exploration is used to construct a labeling protocol. This is a lengthy and iterative process, which will be discussed in section 4.1. The results of the labeling exploration (extended labeling protocol) are used to answer SRQ1. Furthermore, a research experiment is conducted. This experiment is a comparative analysis between a set of different classifiers. An overview of the classifiers used in this experiment is presented in table 9. Each of these classifiers will be tested for predictive power with respect to two steps, ambiguous question detection & ambiguous question enrichment. So this research experiment consists of two separate steps, question detection (Step 1) and question enrichment (Step 2). In Step 1 the classifiers try to detect ambiguous questions. In Step 2 the classifiers try to detect previously ambiguous type questions, that are now clear due to the inclusion of extra context. This extra context consists of either one or two prior sentences to the question sentence. Afterwards, the evaluation metrics discussed in Section 2.9 are applied to gather meaningful results to answer the remaining sub research questions. The table below depicts how the research questions relate to the different research methods.

Research questions		Research methods		
		Experiment Step 1	Experiment Step 2	Labeling Exploration
MRQ	To what extent is it possible to enrich ambiguous type questions in a restricted domain environment?	✓	✓	✓
SRQ1	What are the characteristics of of ambiguous type questions?			✓
SRQ2.1	To what extent is it possible to automatically detect ambiguous type questions in a restricted domain environment?	✓		
SRQ2.2	How does a Rule-based approach compare to a Machine Learning approach for question type classification in a restricted domain environment?	✓		
SRQ3	How is question type classification in a restricted domain influenced by the language of the data?	✓		
SRQ4.1	To what extent is it possible to enrich ambiguous type questions through context analysis?		✓	
SRQ4.2	How does a Rule-based approach compare to a Machine Learning approach for ambiguous type question enrichment?		✓	

Table 9: Research methods used for answering the research questions

Additionally this Sections covers how the data is pre-processed and prepared for the classification models. All the different pre-processing steps are explained and motivated. Afterwards each of the classifiers used in this research are mentioned briefly. This report does not cover an extensive explanation of how these classifiers work, as this is beyond the scope of this project. Consider the flowchart below, depicting a complete overview of the experimental process of this research 8.

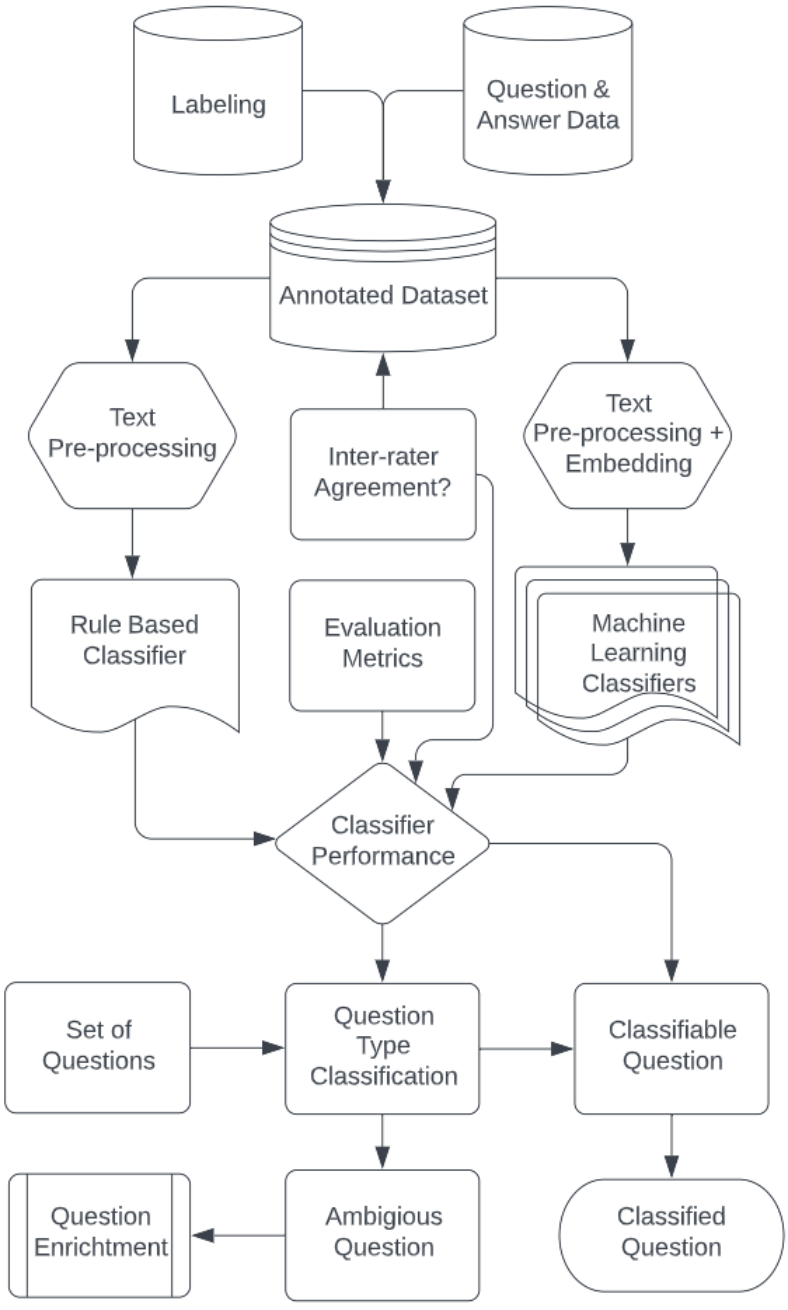

Figure 8: Question classification procedure

4.1 Labeling Exploration

Both data sets are annotated to have an approximation of what the classifiers should output. In the case of this research, a set of questions taken directly from the data set is given a label that determines whether this question is considered ambiguous (not directly answerable) or clear (directly answerable). Cohen's Kappa is used as an inter-rater agreement measure to validate this "Golden Standard" (McHugh, 2012). To facilitate an inter-rater agreement score as well as reproducibility of the experimental results, a protocol is set up to annotate the data. Although the resulting labeling protocol has been applied to both data sets, it was specifically constructed with the Dutch data set in mind. As can be seen in the labeling distribution in figure ENTER, the labels are more evenly distributed in the Dutch data set.

Assumptions

Before constructing a protocol, the following assumptions are made about the data set. These assumptions are made based upon an interview conducted at P-Direkt with experts in the field:

1. All abbreviations that occur in the email message are known information.
2. All terms and language related to P-Direkt & Enron specifically is known information.
3. There is a standard procedure in place for frequently occurring problems (such as the procedure for getting a new password to the P-Direkt portal).

4.1.1 Extended Labeling Protocol

After thoroughly exploring the data set with the help of experts, a number of differentiable labels were identified. This sub-division is used as a basis for the labeling of question types for both the Dutch and English data sets. Because the labeling process was done manually, the size of the data set is limited. This means that not all identifiable label types can be applied. Instead the identified labels are used as "sub-labels" to make a set of "main" labeling categories. The main labels can be seen in front "Clear", "Ambiguous", "Request", "Other". For instance, the labels "Unclear Information" or "Personal Information Missing" are instead labeled as "Ambiguous". Any question labeled as "Ambiguous" is therefore a sub-type of the Ambiguous class. Consider the extended labeling protocol on the next page:

Question Type	Explanation
Clear (Factoid / Closed):	A clear question that has a singular correct answer.
Clear (Open):	A clear question that can be answered in multiple ways.
Ambiguous (Context Missing [1]):	There is context missing required to answer the question. By looking at one prior sentence, this question is now clear.
Ambiguous (Context Missing [2]):	There is context missing required to answer the question. By looking at two prior sentences, this question is now clear.
Ambiguous (Context Missing [3]):	There is context missing required to answer the question. Even by looking at prior sentences, this question remains "Ambiguous"
Ambiguous (Multiple Questions):	This is a question that consists of multiple questions in one sentence and can therefore not be directly answered.
Ambiguous (Confirmation):	The question makes an assumption and then asks whether this assumption is correct (I have to contact HR about a work laptop, right?).
Ambiguous (Unclear Information):	There are grammar mistakes in the sentence or the question is not clearly formulated.
Ambiguous (Personal Information Missing):	To answer this type of question, personal information is required. This is for instance the name of a person.
Ambiguous (Complex):	This question is very complex and hard to understand. Usually these questions consist of multiple lines and are formulated in a way that is hard to understand.
Other (Request):	Mails that ask the respondent to undertake action (Could you deliver this file to Bob?). These are very tricky, because although they might look like a question at first glance, there is no clear answer available.
Other (Filtering Error):	Mails that are a response to an original mail. These are mails that ideally would be removed during the pre-processing stage already.
Other (Advertisement):	Mails that are automated advertisements. This was only the case in the English data set.
Other (Parsing Error):	These are mails that are parsed incorrectly. Due to the fact that the data had to be anonymized, many steps have been performed to alter the data. In some cases, this resulted in parsing errors where sentences are tokenized incorrectly. This could also be the result of grammar mistakes on the user end.

Table 10: Extended labeling protocol

Clear type questions are categorized as questions that should be directly answerable by a "machine", assuming a suitable knowledge base is readily available. Ambiguous type questions are questions that introduce structural ambiguity, or are too complex and therefore not directly answerable. The goal is to label each data point (e-mail) to one of these 4 labels (Clear, Ambiguous, Request, Other). If a data point is does not clearly belong to one of the classes, the data point is given the label "Other". This extended labeling protocol is meant to characterize ambiguous type questions.

As mentioned earlier, this extended labeling protocol is not well suited to the scope of this

research. This is why the "main" categories are used to annotate both data sets. This abstraction of the extended labeling protocol is a difficult aspect of this research, because the labeling strategy for question type classification has no clear guidelines founded in academic research. This is why each way of categorizing the data set into "main categories" was tested for predictive power (by looking at the distribution of the categories). It is important that the labeling protocol does not create an imbalanced data set for classifiers to be trained effectively. However, to be representative of real world data, some imbalance is to be expected. The labeled data set was tested for non-ambiguity. The labeling protocol was evaluated by using a measure of inter-rater agreement with an expert in the field. An overview of the final labeling protocol can be found in table 6, as well as some examples using this protocol in table 7, in section 6

Cohen's Kappa

To validate this annotation protocol, Cohen's Kappa is used as a measure of inter-rater agreement (McHugh, 2012). Cohen's Kappa results in a score between 0 (random agreement) and 1 (perfect agreement). The labeling protocol has been tested by experts working with the P-Direkt data set on a daily basis. Both the researcher and the expert manually labeled 100 samples from the question data set, according to the 5 labels discussed above. In this test the Dutch data set achieved an annotation score of **0.82** whereas the English data set achieved one of **0.79**. Both these scores signify a substantial agreement (McHugh, 2012).

4.2 Data Pre-processing

In Natural Language Processing, classifiers are not classifying raw text. Instead the text is first processed and embedded into a format where it can be recognized by a computer. The steps below summarize the processing of the data for this project. Where it is relevant, the step of pre-processing is shown, using the English data set.

Cleaning up raw data

In this step, we filter out all of the data that is not relevant to this research. Several question detection principles have been tested. Since labeling is done manually, we do not need to fetch a large data set. This is why a naïve approach for question detection is applied, scanning solely for the "?" symbol. After having tried including the "5W1H" in this filtering step, some interesting question were left out. This simple approach left us with a more diverse set of questions.

Here follows an overview of the data cleaning process. More specifically, the number of mails that are left after each pruning step. In this example we shall use a sample size of 10.000 mails in the Dutch data set as well as the English data set. Consider Appendix Section D for an overview of the code used for this process:

Tokenization

This is the initial step in pre-processing the textual format. This is the process of breaking down documents into smaller units called "tokens" (words or sentences for example). Punctuation is included in this step. Python's NLTK (Loper & Bird, 2002) and spaCy (Vasiliev, 2020) facilitate this process as well as many of the pre-processing steps listed below.

Cleaning step	Dutch data	English data
Raw data	10.000	10.000
Containing "?"	7.009	3.322
Remove responses	6.046	2.839
Remove short mails	2.879	1.654
Remove duplicates	2.832	1.033
Remove special symbols & Pruning errors	357	221

Table 11: Cleaning raw data process

```
Before processing & tokenizing:


Lucy, I want to speak to Wade myself. He can call me at work or home. Or if you email me his number I
will call him. I would like Gary to direct Wade on renovation tasks and you can give him work orders
for normal maintenance. I will call you tomorrow to discuss items from the office. Do you need Mary t
o come in on any more Fridays? I think I can guess your answer. I might stop by this Friday. Phillip


After processing & tokenizing:


Lucy, I want to speak to Wade myself.
He can call me at work or home.
Or if you email me his number I will call him.
I would like Gary to direct Wade on renovation tasks and you can give him work orders for normal main
tenance.
I will call you tomorrow to discuss items from the office.
Do you need Mary to come in on any more Fridays?
```

Punctuation

In this step the text is marked at certain special characters such as "?", "!", "," to signify a pause in the flow of text or address sentiment to a sentence. In the case of question classification, the question mark symbol ("?") in particular is of interest.

In this step the data is also stripped of capitalization. This is a useful step due to the nature of the data. The data is comprised of human written mails. This means that capitalization is not following a strict protocol. A possible concern when converting all text to lowercase is the removal of accents. However, for both the English language and Dutch language, accents do not play an important role ($\acute{e} -> e, \ddot{e} -> e.\hat{i} -> i, \tilde{o} -> o$).

To test the importance of capitalization, each classifier was tested with and without converting the data to lowercase. The lower-cased iterations performed marginally, but consistently better. Although capitalization may hold relevant information, given the size and nature of the data set, it is considered as a type of noise for this research.

Stopword removal

It is often the case that certain words appear commonly throughout all samples within our data set. Naturally this does not help in differentiating between the different categories. These words hold little to no information gain for a classifying system, so these can be removed as part of the pre-processing step.

```
Before stopword removal:


['I would like Gary to direct Wade on renovation tasks and you can give him work orders for normal ma
intenance.', 'I will call you tomorrow to discuss items from the office.', 'Do you need Mary to come
in on any more Fridays?']


After stopword removal:


['would like Gary direct Wade renovation tasks give work orders normal maintenance.', 'call tomorrow
discuss items office.', 'need Mary come Fridays?']
```

Lemmatization

This is the process of aligning all semantically identical forms of a word to their base meaning. For example, the words "beauty" and "beautiful" are considered the same after lemmatizing.

```
Before lemmatizing:


['I would like Gary to direct Wade on renovation tasks and you can give him work orders for normal ma
intenance.', 'I will call you tomorrow to discuss items from the office.', 'Do you need Mary to come
in on any more Fridays?']


After lemmatizing:


['I would like Gary to direct Wade on renovation task and you can give he work order for normal maint
enance .', 'I will call you tomorrow to discuss item from the office .', 'do you need Mary to come in
on any more friday ?']
```

However, after having tested this pre-processing step, it is evident that lemmatization does not enhance performance of the tested classifiers in this project.

For this step specifically, the BERT model uses a slightly altered version. BERT models use Byte-Pair Encoding to shrink the vocabulary size. This is done by creating a complete vocabulary consisting of all characters (bytes) in the text corpus. Now the most frequently occurring pairs of consecutive characters (byte-pairs) are merged into sub-words in descending order.

Embedding

To ensure that words that have a completely different syntax but are semantically similar are grouped together, embedding techniques are applied. This is the process of representing words with similar meaning under the same grouping. By process of word vectorization this can be achieved. Considering many words are used infrequently, this would result in a very sparse matrix. However, due to the limited size of the training data, these sparse matrices do not pose a hindrance.

To represent words more densely, Word2Vec is introduced (Rong, 2014). This technique uses a neural network to learn word associations. An extension of Word2Vec is Doc2Vec, which computes a feature vector for each document instead of each word. However, this embedding technique is not used during this research. After testing and comparing this embedding technique to the more "simple" TF-IDF representation, it became quickly evident that the TF-IDF

count matrix performs much better.

TF-IDF stands for "Term frequency-inverse document frequency". It is a numerical statistic that reflects a word's importance relative to other words in the text corpus (Robertson, 2004). TF-IDF is known to perform well at tasks such as "topic selection/sentiment analysis" (Cahyani & Patasik, 2021). Furthermore, the sparse nature of TF-IDF matrices are especially suited to smaller data sets, such as the one in this research. A recent study, directly comparing Word2Vec and TF-IDF for sentiment analysis shows the potency of TF-IDF (Cahyani & Patasik, 2021). Therefore TF-IDF is used in combination with a vectorizer as embedding technique for every classifier, except for BERT.

BERT, which does not construct an embedding for single tokens (words), instead uses a sequence (sentence) as input to generate a contextual embedding (Devlin et al., 2018). BERT's vocabulary extends beyond the training corpus. A simple vectorization process can be seen below, transforming the words into integer values, displaying their frequency ranking.

```
from:  do you want me to fax this to you? | len: 9
to:  [ 16   2  84 449  20 450  12 451   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0] | len: 50
```

Dictionary

Pre-processed words are put into a dictionary. A mapping is constructed between the words and the different IDs (corresponding to the different categories). This dictionary serves as a feature selection process for the question text categorization. Each dictionary ID holds a set of words (features) contrasting from the other IDs. The dictionary itself is not directly used. This dictionary is merely shown below as an intermediate step to creating a TF-IDF representation of the data, to facilitate better understanding of the embedding process. Research has shown that this data conversion technique can significantly boost classifier performance (Seger, 2018).

```
vocabulary:  {'NaN': 1, 'you': 2, 'the': 3, 'to': 4, 'do': 5, 'is': 6, 'of': 7, 'a': 8, 'what': 9, 't
his': 10, 'are': 11, 'and': 12, 'that': 13, 'for': 14, 'have': 15, 'can': 16, 'or': 17, 'how': 18, 'w
e': 19, 'with': 20, 'on': 21, 'in': 22, 'your': 23, 'be': 24, 'i': 25, 'it': 26, 'will': 27, 'would':
28, 'any': 29, 'me': 30, 'get': 31, 'does': 32, 'at': 33, 'could': 34, 'if': 35, 'need': 36, 'going':
37, 'when': 38, 'about': 39, 'want': 40, 'think': 41, 'us': 42, 'week': 43, 'an': 44, 'why': 45, 'tim
e': 46, 'there': 47, 'up': 48, 'work': 49, 'these': 50} ... (padding element, 0)
```

4.3 Rule-based classifier

In this project a Rule-based classification approach and Machine Learning classification approaches are evaluated and compared. To do this, a rule-based system shall be designed for this restricted domain. A rule based approach is especially well-suited for restricted domain, professional QAS environments, as is the case for the Question & Answering department in P-direkt. (Cai et al., 2020).

With the help of experts (HR department P-direkt) a suitable knowledge graph can be constructed to map the knowledge archive to the questions. The resulting type of system can be regarded as part of an "expert system", designed to thrive in a restricted domain (Khella, 2017). When finalized, this system should ideally be able to handle the Q&A process in its entirety, emulating an expert service provider. This project is primarily focused on the first step of this

Q&A process, implying that we are not focused on generating an automated answer. Careful integration of expert opinions will help shaping this rule based system to classify questions effectively. The first set of rules for this classifier is set up with the "RIPPER" algorithm (Cohen, 1995). Afterwards, an additional set of rules is added onto RIPPER algorithm induced rule set to complete the Rule-based classifier.

RIPPER Algorithm

As a direct method, sequential pattern mining is used, utilizing the "repeated incremental pruning to produce error reduction" (RIPPER) method (Cohen, 1995). RIPPER used the majority class as default class and learns rules for identifying the minority classes. In the case of multi-class classification, this algorithm iterates over the ordered classes (in terms of frequency), distinguishing the current class from the remaining classes. This process is repeated until the least frequently appearing class is the only class left. This makes the RIPPER algrorithm especially suitable for imbalanced class distributions. It also works well with noisy data. Since our data set consists of messages written by humans, the data can be considered noisy.

RIPPER is widely used as rule induction method and has numerous successful prior use cases in question type classification (Aubaid & Mishra, 2020; Radev et al., 2002; Sangodiah et al., 2015). RIPPER has also been used for question detection, but as discussed earlier this is not a necessity for this research due to the manual labeling process in combination with large data sets.

Indirect construction of a Rule-based classifier is the application of a different classification to the training data before constructing the rule based classifier (This can be considered a Hybrid approach). In this project, a direct method for constructing the classifier is used. The set of rules is directly extracted from the training data. Several techniques for rule-induction exist, such as OleX and RIPPER (Cohen, 1995; Rullo et al., 2008). This research shall use the RIPPER algorithm, as this algorithm has proven to be one of the most accurate amongst Rule-based algorithms.

4.4 Machine Learning classifiers

Recent surveys on text classification techniques provide a broad overview of all ML methodologies used in text classification tasks (Gasparetto et al., 2022; Mirończuk & Protasiewicz, 2018). These surveys highlight the potency and popularity of Support Vector Machines(SVM) in the area of text classification. A recent study comparing the effectiveness of various machine learning methods for multi-class text classification concluded Logistic regression to be the most effective out of the common set of methods used for text classification (Pranckevičius & Marcinkevičius, 2017). Logistic regression is a discriminative classification method, which implies that this method works well for large annotated data sets specifically. We shall also test a generative classification method, Naïve Bayes (NB). NB is well suited for multi class classification problems and generates a model from the data itself. Lastly we shall be testing deep learning methods very commonly used in text classification, Recurrent Neural Networks & a dutch adaptation of BERT. In figure 9 you will find an overview of the methods used in this experiment.

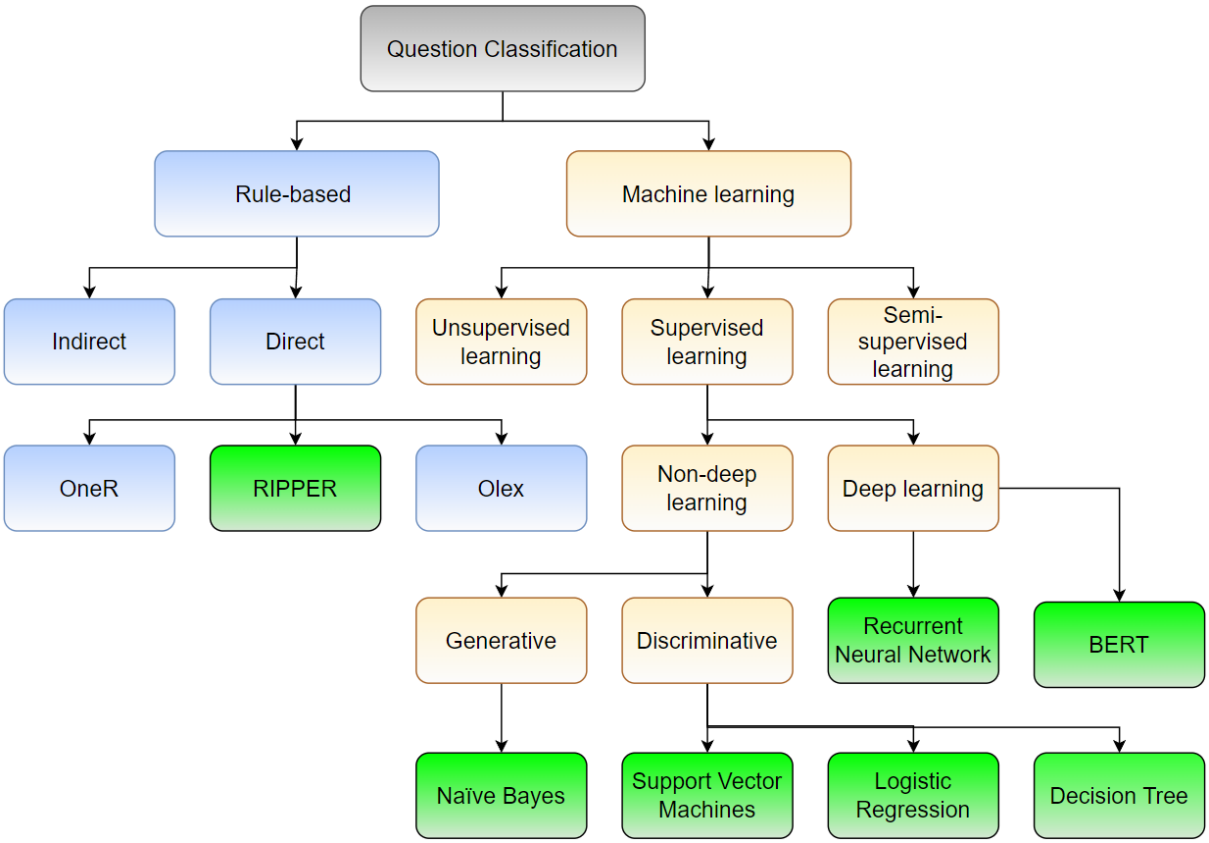

Figure 9: Overview of the Question Classification landscape

Naive Bayes

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other (Abbas et al., 2019). It is based on the Bayes Theorem (Insert Bayes Source). It uses this famous equation to find the probability of an event occuring, to predict context for a given data point.

Naïve Bayes is one of the fastest ML-based classification techniques. Moreover it performs very well on categorical data. That is why this ML technique is used for this experiment as a representative of generative models.

Support Vector Machines

A Support Vector Machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each class, it is able to categorize new text (Cervantes et al., 2020).

Compared to newer algorithms based on neural networks, SVMs have two main advantages: higher speed and better performance with a limited number of samples (in the thousands). This makes the algorithm very suitable for text classification problems, where it's common to have access to a data set of at most a couple of thousands of tagged samples. That is why this ML technique is used for this experiment.

Logistic Regression

Logistic regression as a classifying technique has a number of advantages. It is fast at classifying unseen records and makes no assumptions about class distributions (good for imbalanced data sets). It is not very prone to over-fitting and can easily extend to multi class classification.

In a study that is focused on comparing several text classification techniques for BBC news text, Logistic Regression performed the best (Pranckevičius & Marcinkevičius, 2017). This experiment used a data set with 5 classification classes. This is quite similar to the nature of the data used in this experiment. In their experiment, Logistic regression scored an accuracy of 97% and also performed best out of all classifiers on smaller data sets.

Decision Tree

Decision trees have also proven to be good at handling text classification tasks in the past. More importantly, they provide a clear overview of performance outcomes. In a survey on Decision Tree applications, recent usage of Decision Trees in the medical field has yielded an accuracy upwards of 99% (Charbuty & Abdulazeez, 2021).

Recurrent Neural Networks

Recurrent Neural Networks (RNN) are based on the "Feed Forward" mechanism with an internal memory component. Except, unlike Feed Forward Neural Networks that only allow signals to travel in one direction, RNNs allow for signals to travel in both direction with the introduction of loops. This makes RNNs good at handling sequential data, like sentences.

BERT

As discussed in the Related Work section, a novel transfer learning technique called BERT (Bidirectional Encoder Representations for Transformers) made big waves in the NLP research space. At its core BERT excels at "context-rich" language problems (Devlin et al., 2018). It excels at finding relationships between words and sentences.

BERT is given millions of sentences at training time. It then tries to predict a random sample of missing words from these sentences. After repeatedly training on this huge text corpus, BERT learns how words and sentences fit together grammatically. For this research, BERT is an excellent contender. In this experiment we try to understand ambiguously formulated questions through context-analysis. This is precisely what BERT excels at.

4.4.1 Training

To track the training process of the deep learning classifiers, we use Cross Entropy. It measures the similarity between two distribution functions. Categorical Cross Entropy is a type of loss function that can be considered as a measure of error for categorical multi-class classification where a score of 0 signifies a perfect cross-entropy value. It is based on the assumption that for each instance there is only one correct class. This is how it differs from Binary Cross Entropy, where each instance can belong to multiple classes. Consider c to be the set of classes, $p(c)$

the true probability distribution and $q(c)$ the model's predicted probabilistic distribution. The Cross-Entropy loss is defined as follows (Grandini et al., 2020):

$$H(p,q) = -\int_{C_c} p(c) \log q(c)$$
$$H(p,q) = -\sum_{C_c} p(c) \log q(c)$$

Perhaps this metric is not very well suited for the multi-class classification task in this research, since the data used for this research can include multi-labelled data. However, since a question is only forwarded to a single specialization department within the P-direkt, the task can be interpreted as multi-label classification instead.

Another possible concern for this method is that it uses $SoftMax$ to normalize the output probabilities. This can pose a problem in the scenario we want to track the training progress of Rule-based classifiers. Rule-based classifiers operate by $if \rightarrow then$ schemes, the output probabilities are 0 or 1. But since these rules are determined in a probabilistic manner, categorical cross entropy can be measured indirectly. For the deep learning approaches this metric is rather straightforward, since the class assignment of an instance is probabilistic by itself.

By tracking the training process, we can tune the hyper-parameters more effectively. This way a suitable number of epochs and batch size is determined.

4.5 Question Enrichment

To enrich ambiguous question types, the context will be screened for relevant information. To ensure this is done justly, an inter-rater agreement is performed to determine to conditions for context extraction. Part of Speech (PoS) tagging and NER have been applied in previous research to attempt query generation to tackle question ambiguity (Azevedo et al., 2020) (Alazani & Mahender, 2021). The approach for this research is to look at prior context to the question sentence. In this case we shall attempt to gather information in the question context to "enrich" the previous prediction by including a larger feature space.

Since there is not much research on question ambiguity, the purpose of this enrichment step is to add onto existing research about question clarification. Furthermore, if this enrichment step can be executed successfully, these results can be useful for the question answering landscape. More specifically, it gives a rough estimate of the degree to which ambiguous type questions can be utilized. When interpreting these results, it can give enterprises an idea of the percentage of incoming questions that are automatically processable. Following this, it also gives enterprises an idea of the percentage of questions that need to be looked at in more detail instead.

4.6 Tools

Python

This research follows a quantitative study design. The data set mentioned in 3.1 is used to rigorously measure the performance of Machine Learning and Rule-based classification. Both these classification designs are set up in Python. Python has a variety of packages available

to help with setting up and training of classifiers. This programming language also has access to power deep learning libraries such as TensorFlow and Keras (Sarkar et al., 2018). Python is well suited for building classifiers in general, making it suitable for Rule-based classification too. The data set itself is stored in Excel sheets. This is easily converted into a CSV-format in Python, so that it can be imported and used directly.

NumPy

Python provides Numpy, a comprehensive mathematical library (Harris et al., 2020) which includes ML libraries such as Sci-Kit Learn (Pedregosa et al., 2011) to help train ML models. This a well known tool in the Machine Learning and Artificial Intelligence domain. Together these packages facilitates linear & logistic regression and they make it easy to set up a training and testing set split.

NLTK & spaCy

The Natural Language Toolkit (NLTK) is a famous platform within python that includes pretty much all text pre-processing steps necessary (Loper & Bird, 2002). spaCy is a free open source library for NLP tasks in Python, featuring NER, POS tagging and much more. It has a pipeline specifically trained and optimized for Dutch sentences and text interpretation. spaCy is implemented for Python and uses a convolutional neural network model and has proven to be succesful for multiple Dutch NLP tasks (Nobel et al., 2020; Tian et al., 2022; Trienes et al., 2020)

RobBERT

RobBERT is the state-of-the-art Dutch BERT model (Delobelle et al., 2020a). It is a large pre-trained general Dutch language model that can be fine-tuned on a given dataset to perform any text classification, regression or token-tagging task. As such, it has been successfully used by many researchers and practitioners for achieving state-of-the-art performance for a wide range of Dutch natural language processing tasks, including: sentiment analysis, POS tagging and most relevant NLP downstream tasks.

This pre-trained classifier is fine-tuned on the data sets in this experiment to build a classifier that can do both ambiguous question detection and question enrichment.

Huggingface

Implementations for both deep learning models (RNN & BERT) are facilitated by the Huggingface library (Wolf et al., 2019). This library allows for the training of both the RNN and BERT models. It provides the user with trainer classes. These trainer classes provide an API that is highly customizable with respect to the model setup (such as adding the different layers). Furthermore these classes facilitate the entire training process and have built-in functions such as “trainer.evaluate” and “trainer.predict” to fetch model outcomes.

The Huggingface library also holds very up to date pre-trained versions of BERT models, such as the Dutch variations of the RobBERTa model for sequence classifications mentioned earlier (Liu et al., 2019).

5 Results

In this section, the gathered results and analysis metrics are presented and evaluated. This is accomplished by presenting the measurement statistics and relevant figures & tables. For each classification technique repeated measurements have been conducted. The True Positive, True Negative, False Positive and False negative rates are calculated by comparing the predicted classes to the actual classes. Hence a confusion matrix can be constructed. From these aggregated matrices, the accuracy, recall, F1-score and precision are extracted for each classification technique.

Since the data we are working with in this experiment is imbalanced due to class imbalance, the F1 statistic is preferable over the Area Under the Curve statistic (The reasoning behind this decision is further clarified in Section 2.9). Since we are primarily concerned with the detection of the majority class (ambiguous type), there is no need for excessive re-balancing of the data sets. The training data sets emulate the real world environment accurately for this reason. Because the results section is quite elaborate, the results for the Dutch data set and the English data set are split up. For each metric, a short explanation is given in the Dutch results section. Classification reports for every classifier can be found in Appendices E (Dutch) and F (English).

5.1 Dutch results

5.1.1 AUC & ROC Curve

In this Section the AUC metric is defined for each classification technique. Each score measure depicts the ability of a given classifier to distinguish between a given label and the rest at a given threshold. This Section will also include a graph displaying the ROC curves of all tested classifiers.

Step 1 [Detection]

First, the results for the detection of ambiguous type questions are presented. This set consists of an aggregation of labels "Ambiguous", "Clear[2]" and "Clear[3]". The set of clear type questions is represented by label "Clear[1]". So in essence we are looking at how classifiers perform at differentiating between questions that are directly clear and questions that are ambiguous (without context analysis). The table below depicts the AUC score for each classifier for the question detection step.

Classifier	*AUC*
Logistic Regression	0.8683
Support Vector Machines	0.8867
RIPPER	0.5561
Naïve Bayes	0.8390
Decision Tree	0.8035
RobBERT	0.9399
Recurrent Neural Network (LSTM)	0.8149

Table 12: AUC metric for the classification of Clear [1] against Ambiguous and Clear[2+3]

The closer the AUC score is to 1, the better the classifier performance is on this task. As can be seen by the table above, the AUC metric favours the more complex models over simple

rule based models for this task. This is somewhat expected as the more complex models tend to be less biased than their rule-based counter parts. The AUC metric measures the capability of each model to differentiate between classes, which implies that a biased model is punished more heavily. Bias refers to the tendency of a model to favour the majority class during classification. In the case of this research, models tend to "over classify" questions as ambiguous, because this is the majority class. If we plot the TPR and FPR at different thresholds for each model in a graph, this effect can be seen more clearly 10. According to the AUC metric, the RobBERT model scores best amongst all classifiers.

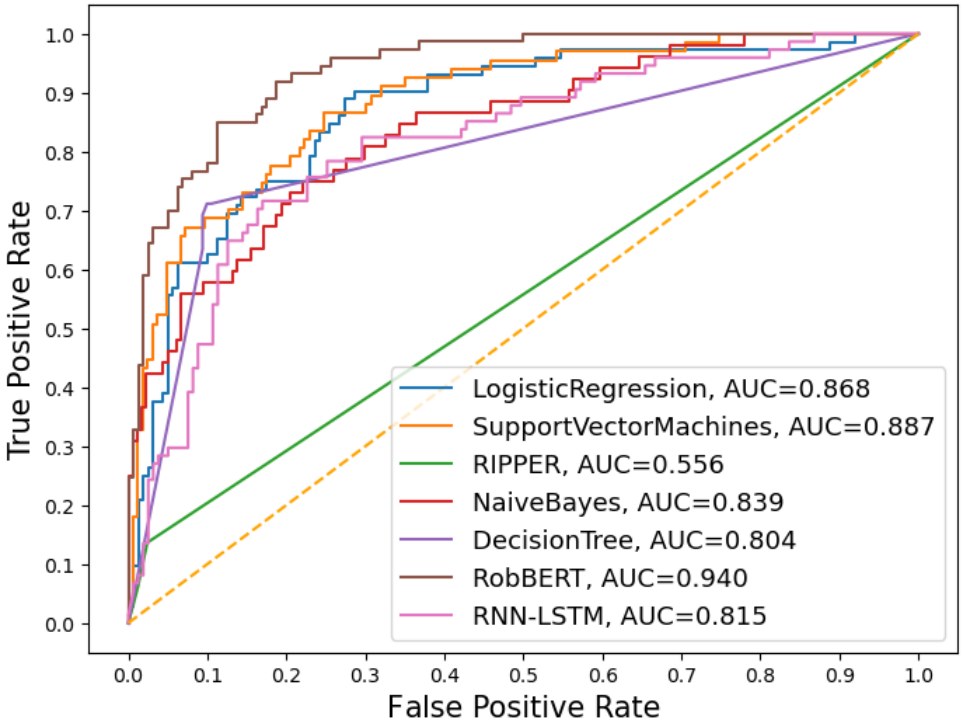

Figure 10: AUC metric for the classification of Clear [1] against Ambiguous and Clear[2+3]

Step 2 [Clarification/Enrichment]

Now let us continue by expressing the AUC metric for the task of question enrichment, separated into categories: [1] one extra sentence required for clarification and [2] two extra sentences required for classification.

Classifier	$AUC_{1sentence}$	$AUC_{2sentences}$
Logistic Regression	0.7326	0.7327
Support Vector Machines	0.6977	0.7115
RIPPER	0.5833	0.5142
Naïve Bayes	0.7028	0.7270
Decision Tree	0.6748	0.6436
RobBERT	0.8316	0.7130

Table 13: AUC metric for the question enrichment with one or two prior sentences

Due to restricted access to the Dutch data set, it was not possible to evaluate RNN model performance for the enrichment step. In this step the Clear[1] labels are removed from the

data set, because these questions do not require enrichment as they are already clear. Thus, the resulting data set is smaller as a result, leading to an overall worse performance for most classifiers in comparison to step 1.

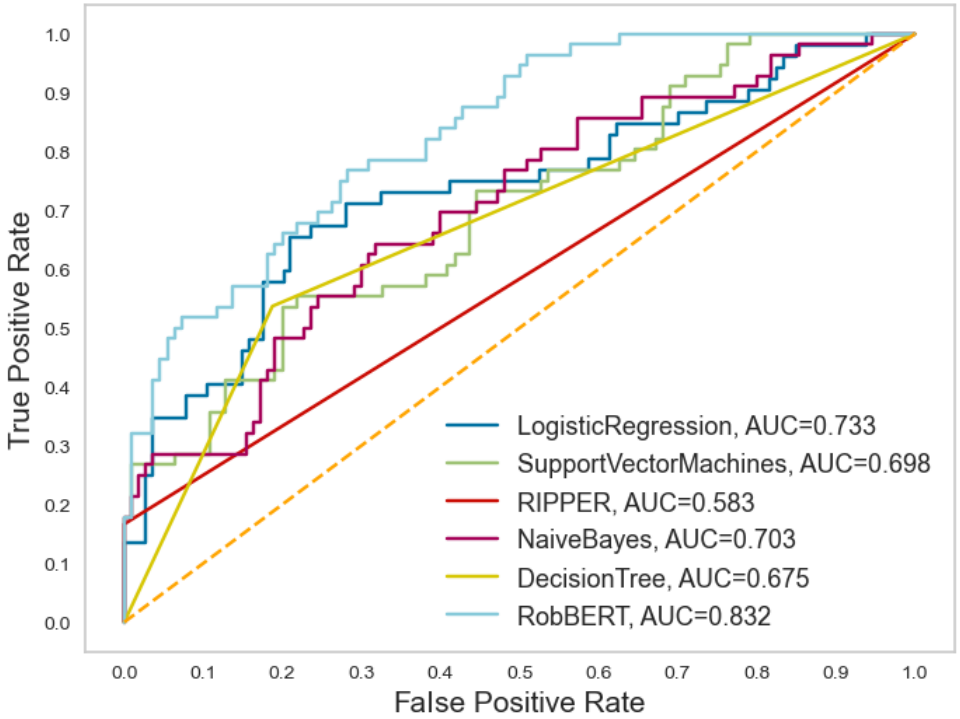

Figure 11: AUC metric for question enrichment with 1 prior sentence

When plotting the TPR and FPR rates, it becomes evident that the rule based classifier gets outperformed by the more complex Machine Learning techniques in both enrichment steps. Especially the BERT model is performing well.

Some classifiers perform better on smaller data sets as can be seen in figure **??**. For instance the Naïve Bayes classifier and the SVM classifier both have a higher AUC score. Why this is the case is not clear, especially since the data set is getting smaller at this step.

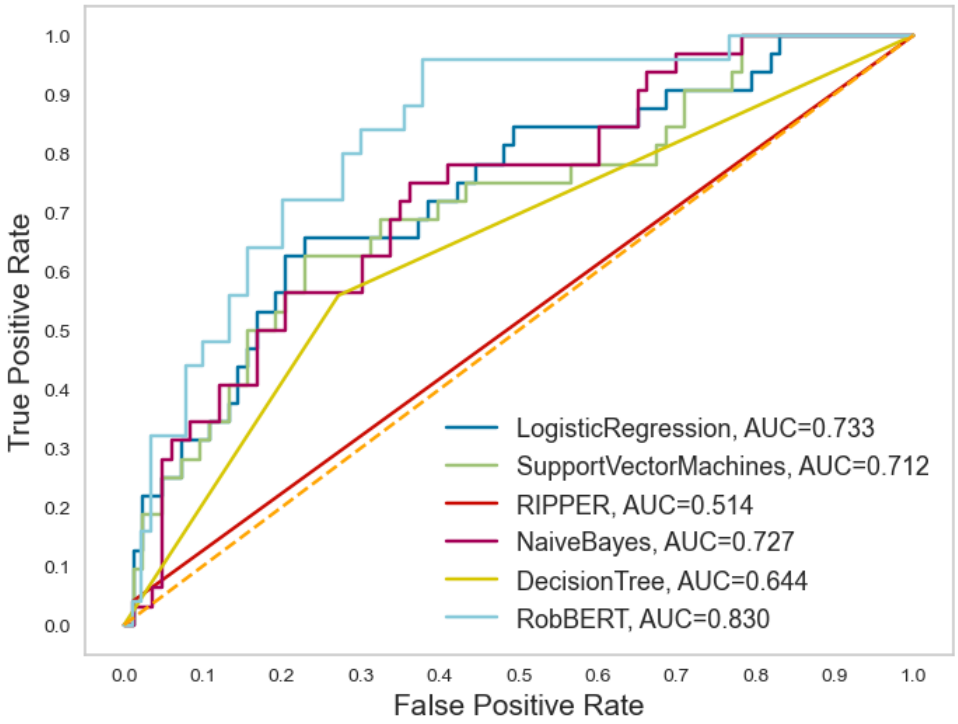

Figure 12: AUC metric for question enrichment with 2 prior sentences

5.1.2 Classification reports

In this Section we take a look at model performance through classification reports. As explained in the evaluation section, classification reports are derived from confusion matrices. These reports display model performance on a number of metrics, such as precision, accuracy and F1-score. Furthermore it supplies the weighted and macro averages of these scores for the predicted classes. The macro average is the arithmetic mean over all classified labels. For the weighted average we do the same, except we consider the support for each label as well. The classification reports are also color encoded to give a quick overview of the strengths and weaknesses of each classifier.

For each step (detection step and enrichment steps), only the best performing classifier is shown, according to the F1-score metric. Please refer to the Appendix for a complete overview of classifier performance. A classification report for every classifier can be found there.

Step 1 [Detection]

From the classification reports, we can also extract some meaningful metrics such as the Accuracy, F1-score, Precision and Recall. A table is provided with classifier performances over these metrics as well.

Classifier	*Accuracy*	*Precision*	*Recall*	$F1_{score}$
Logistic Regression	0.84	0.84	0.84	0.84
Support Vector Machines	0.85	0.85	0.85	0.85
RIPPER	0.8	0.77	0.8	0.76
Naïve Bayes	0.84	0.83	0.84	0.84
Decision Tree	0.83	0.82	0.83	0.82
RobBERT	0.90	0.90	0.90	0.89
Recurrent Neural Network (LSTM)	0.79	0.77	0.80	0.79

Table 14: Classifier performance metrics for question Detection (Dutch)

The metrics here display the *averages* over all label classifications. When looking at the classification reports, it can be clearly seen that there is a bias toward the majority class (ambiguous questions). This is to be expected due to class imbalance.

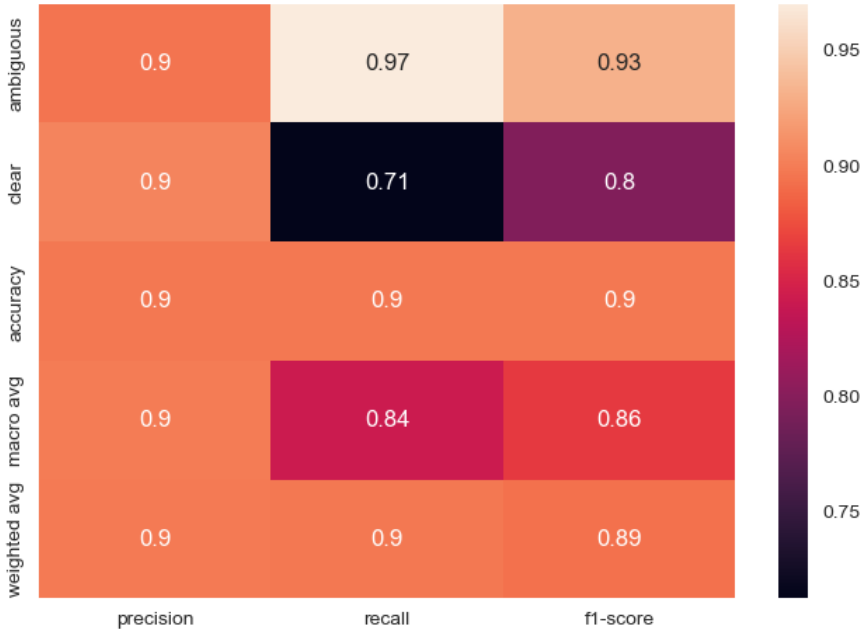

Figure 13: Classification report for RobBERT classifier for ambiguous question detection

As can be seen in the classification reports, there is a bias toward the majority class for pretty much each and every classifier. This is an expected result since we opted against class re-balancing due to the small data set. Since the primary goal of this step was to detect ambiguous type questions, it is acceptable to have a slight bias in that direction.

The bias of favoring the ambiguous question type has been heavily reduced through parameter tuning and by introducing weights during the training process. This was especially relevant for the (non-deep) Machine Learning classifiers such as "Naïve Bayes" and "Logistic Regression".

Step 2 [Clarification/Enrichment]

Now we shall look at the results for the enrichment of ambiguous type questions through context analysis. Here the the ability to differentiate between labels "Clear[2]", "Clear[3]" and "Ambiguous" is measured for each classifier. This is a measurement of how well classifiers can detect questions that are clarified/enriched through the inclusion of extra context.

Question enrichment with 1 extra sentence

Classifier	*Accuracy*	*Precision*	*Recall*	$F1_{score}$
Logistic Regression	0.75	0.76	0.75	0.75
Support Vector Machines	0.76	0.78	0.76	0.72
RIPPER	0.76	0.82	0.76	0.69
Naïve Bayes	0.71	0.69	0.71	0.69
Decision Tree	0.72	0.72	0.72	0.72
RobBERT	0.73	0.73	0.73	0.73

Table 15: Classifier performance metrics for question enrichment with 1 extra sentence (Dutch)

The questions labeled as "ambiguous" by the classifiers in the previous step are now forwarded to the next task. Here the classifier tries to enrich these questions by looking at question context. The results display the classifiers' ability to recognize context that holds enough information to clarify the previously marked "ambiguous" questions. In simple terms, it shows the ability of a classifier to enrich an ambiguous question.

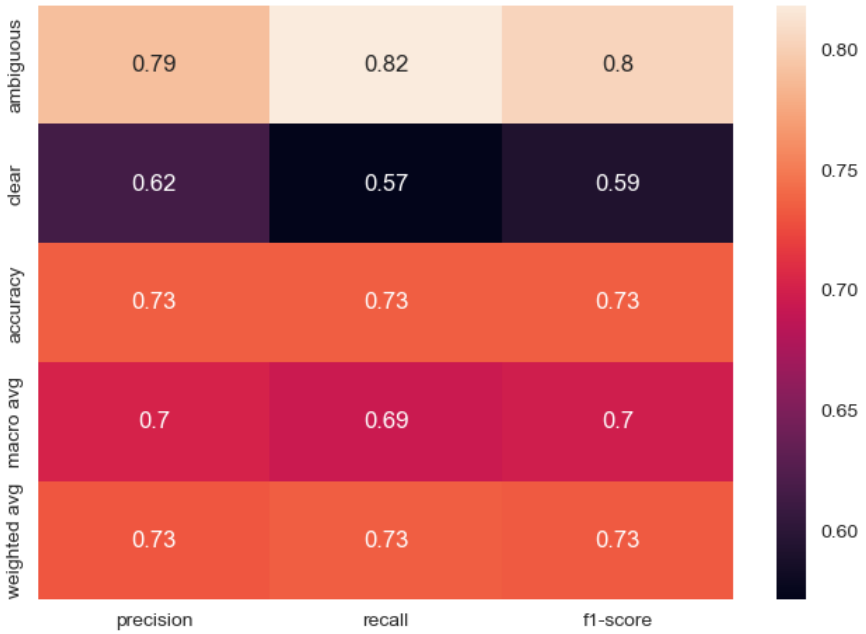

Figure 14: Classification report for RobBERT classifier for ambiguous question enrichment (1 sentence)

Question enrichment with 2 extra sentences

Classifier	*Accuracy*	*Precision*	*Recall*	$F1_{score}$
Logistic Regression	0.73	0.75	0.73	0.74
Support Vector Machines	0.71	0.73	0.71	0.72
RIPPER	0.78	0.73	0.78	0.70
Naïve Bayes	0.71	0.73	0.71	0.72
Decision Tree	0.72	0.71	0.72	0.71
RobBERT	0.80	0.79	0.80	0.78

Table 16: Classifier performance metrics for question enrichment with 2 extra sentences (Dutch)

As can be seen. enrichment becomes increasingly difficult with the inclusion of more context. However, the decrease in predictive power of these classifiers may also be influenced by the decrease in size of the data set for each subsequent step. Somehow the BERT classifier performs better with the inclusion of more context. Perhaps this is due to its proficiency is deriving meaning from context.

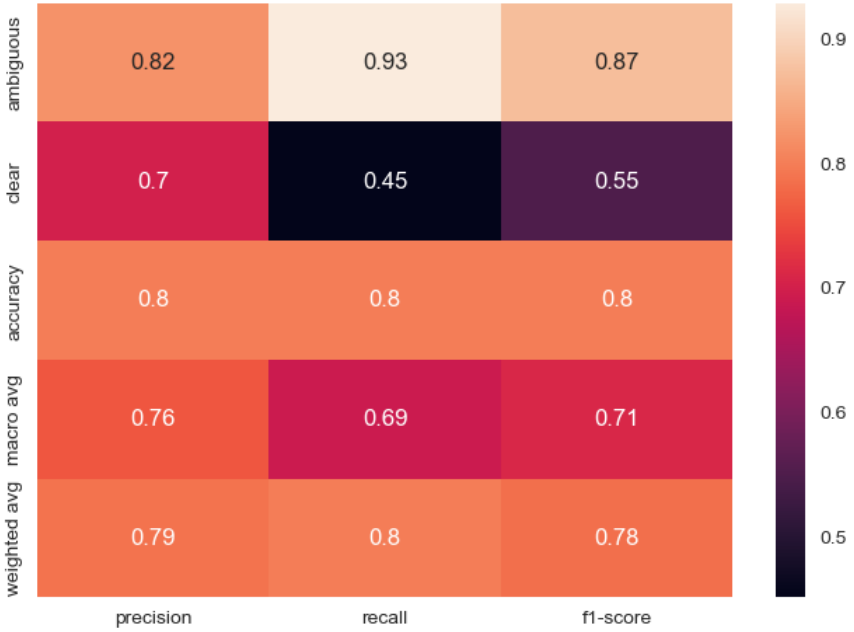

Figure 15: Classification report for RobBERT classifier for ambiguous question enrichment (2 sentences)

5.2 English results

The English results section follows the same structure as the Dutch results section. If you need a more detailed explanation of a certain metric, please refer to the Dutch results section where all metrics are explained. Furthermore, the enrichment step for 2 extra sentences is omitted. This is because the number of training samples for this step was too small to produce meaningful results (training set has below 200 data points, thus results are very biased for every classifier).

Furthermore, this section features an Error Analysis to better understand classifier performance. The chosen classifiers for this are the Support Vector Machines classifier and the RIPPER classifier. The SVM classifier was chosen because it represents an average performance of ML-based classifiers.

5.2.1 AUC & ROC Curve

In this Section the AUC metric is defined for each classification technique. Each score measure depicts the ability of a given classifier to distinguish between a given label and the rest at a given threshold. This Section will also include a graph displaying the ROC curves of all tested classifiers.

Step 1 [Detection]

First, the results for the detection of ambiguous type questions are presented. This set consists of an aggregation of labels "Ambiguous", "Clear[2]" and "Clear[3]". The set of clear type questions is represented by label "Clear[1]". So in essence we are looking at how classifiers perform at differentiating between questions that are directly clear and questions that are ambiguous. The table below depicts the AUC score for each classifier for the question detection step. The closer the AUC score is to 1, the closer the classifier is to a perfect classification result.

Classifier	*AUC*
Logistic Regression	0.7769
Support Vector Machines	0.7419
RIPPER	0.6309
Naïve Bayes	0.7646
Decision Tree	0.6769
RobBERT	0.9553
Recurrent Neural Network (LSTM)	0.7256

Table 17: AUC metric for the classification of Clear [1] against Ambiguous and Clear[2+3]

Here we can see that results follow a similar trend to the those of the Dutch data set. However, the RIPPER algorithm is much closer in performance to other classifiers. Perhaps rule based classifiers perform better on smaller data sets. Additionally, the BERT classifier performs exceptionally well on this data set.

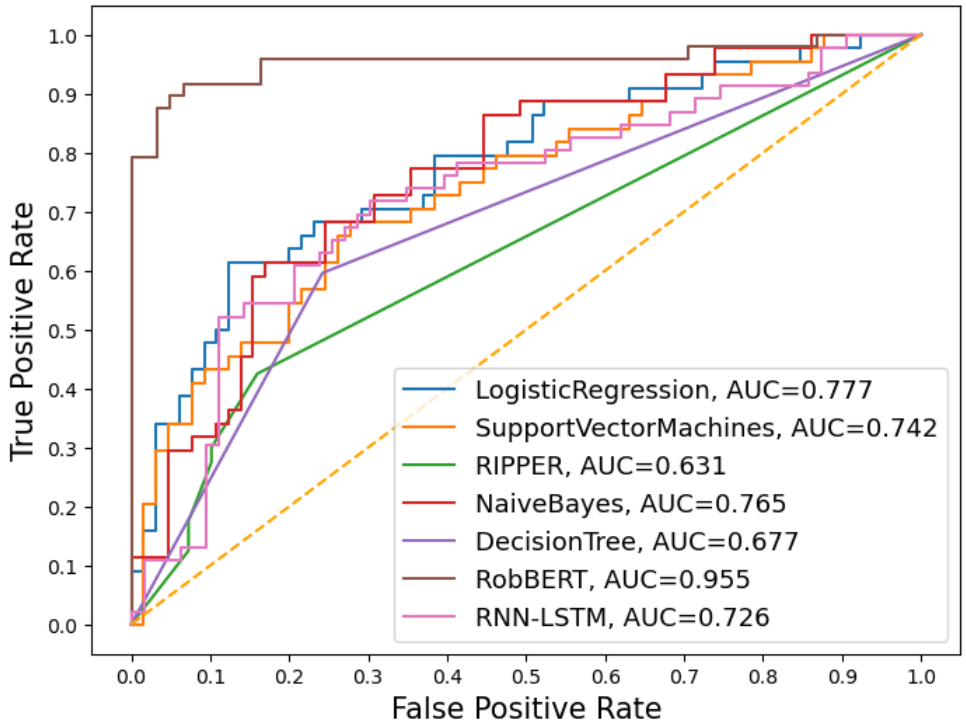

Figure 16: AUC metric for the classification of Clear [1] against Ambiguous and Clear[2+3]

When plotting the ROC Curve for each classifier, it can be seen that the RIPPER algorithm is much closer to the Machine Learning methods

Step 2 [Clarification/Enrichment]

Here the AUC metric for the task of question enrichment is shown. For this data set it is not separated into categories, instead only the results of "one extra sentence required for clarification" are shown. This is because the data set is too small to perform the second enrichment step. Also, because the English data set is complementary to the Dutch data set and used for comparison, the RNN classifier has not been applied to the enrichment step either (because it was not available for the Dutch data set).

Classifier	$AUC_{1sentence}$
Logistic Regression	0.5799
Support Vector Machines	0.5842
RIPPER	0.5659
Naïve Bayes	0.6834
Decision Tree	0.6518
RobBERT	0.7577

Table 18: AUC metric for the question enrichment with one prior sentence

As we can see, for the enrichment step, the BERT classifier, as well as all the other classifiers, performs much worse. Unfortatuntely this is most likely due to the decrease in data size, rendering the English data set a bit too small for this step. We can still see a similar trend however in classifier performance. In all situations, the BERT classifier outperforms the other classifiers, as its AUC score is closest to 1.

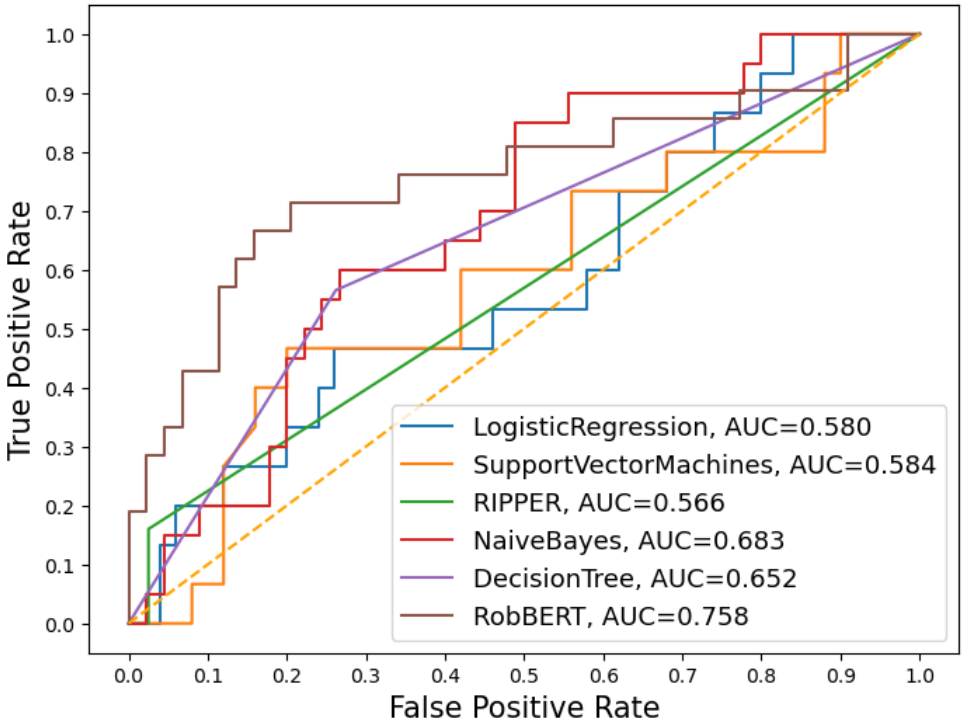

Figure 17: AUC metric for question enrichment with 1 prior sentence

5.2.2 Classification reports

Step 1 [Detection]

If we look at the classification report for the detection step of the best classifier for the English data set, we can see that the BERT classifier actually scored an accuracy of over 90%. This is accompanied by an F1-score of over 0.9, which is very impressive considering the small data set.

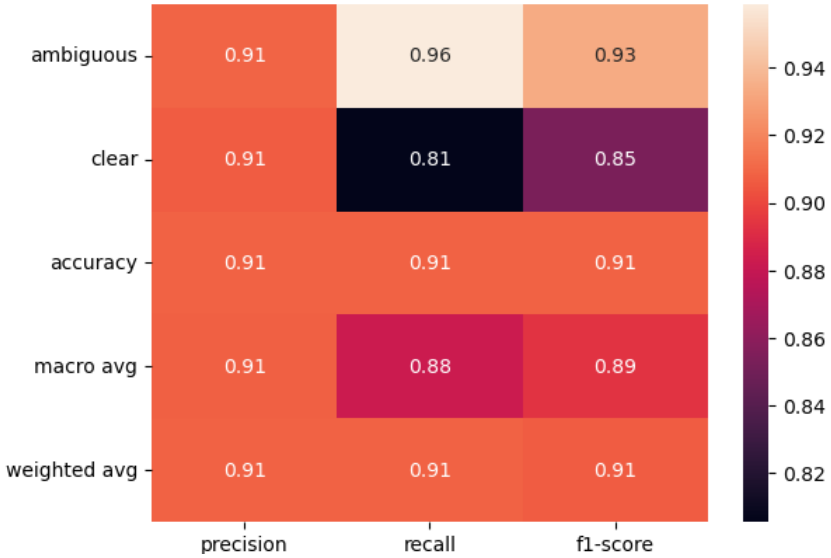

Figure 18: Classification report for BERT classifier for ambiguous question detection

Step 2 [Clarification/Enrichment]

For the enrichment step, the performance of the classifier is much less impressive. Once again this is most likely due to the lack of data samples for this step. An F1-score of nearly 0.8 is still an adequate result however.

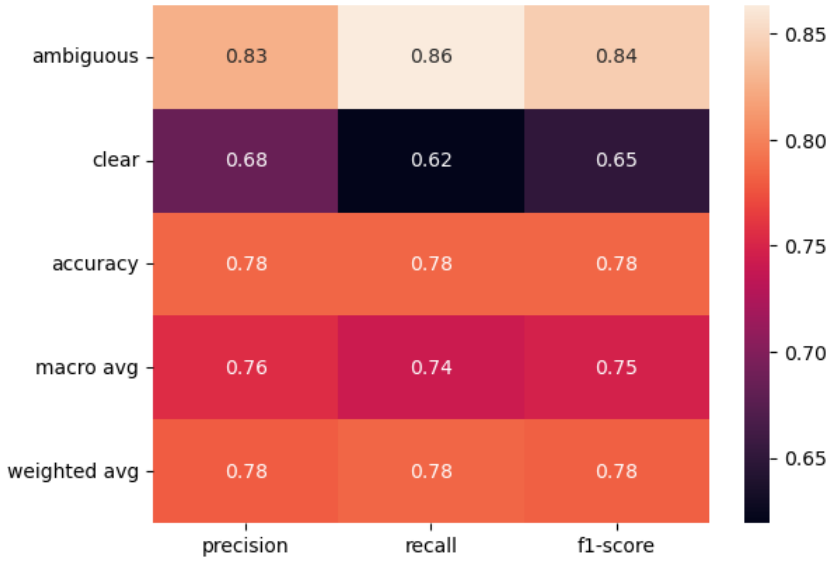

Figure 19: Classification report for BERT classifier for ambiguous question enrichment (1 sentence)

5.2.3 Error Analysis

To better understand and interpret the classifier results, some of the wrongly classified samples are analyzed. To keep things organized, we consider 5 correct and 5 incorrect classifications for a shallow ML technique (Support Vector Machines) and the RB classifier (RIPPER). This process is done for both the question detection, as well as the question enrichment step. These examples are from the English data set.

For each wrongly classified data point (Question), the classifier prediction is shown (Prediction) as well as the corresponding ground truth label (Target). Since these models are trained on a limited data set, their rules are not as elaborated as the human understanding goes. When putting these results next to the labeling protocol, it makes sense why the classifiers made these mistakes. Under each table you will find a short explanation of some of the errors that occurred.

Step 1 [Detection]

Question	**Prediction**	**Target**
That, and his observations on everything from gasoline to telecom to Winston Churchill, formed the basis for an optimistic economic outlook he recently shared with participants at the Wharton Investment Management Finance and Investment Should Hong Kong Worry When China Joins the WTO?	*Clear*	*Ambiguous*
Write to Christina Cheddar at Copyright (c) 2001 Dow Jones & Company, Inc. All Rights Reserved Copyright?	*Clear*	*Ambiguous*
Are you continuing to lose trading?	*Clear*	*Ambiguous*
Can you let me know if you have any problems or concerns with the length of my planned holiday?	*Ambiguous*	*Clear*
But does Webvan's collapse mean that shoppers dislike buying groceries online?	*Ambiguous*	*Clear*

Table 19: Faulty classifications for Support Vector Machines classifier (Detection)

The first question in Table 11 was predicted by the Support Vector Machine classifier to be Clear, even though it had been previously labeled as Ambiguous. It is likely that this is because the last part of the sentence posts a clear question. However because of the length and complexity of the question it was manually labeled as ambiguous, there were too many factors to consider for it to be a clear question. The second mistake in the table above was likely because it contains a full name. A lot of questions were labeled Ambiguous because the writer referred to him/her/they without clarifying who this subject was. The third error likely occurred because the sentence structure is clear and there are no reference words. However during the labeling process this label was classified as ambiguous because it is unclear what kind of trading the writer refers to. This question can only be answered when more context is available.

Question	Prediction	Target
What is going on with these files?	*Clear*	*Ambiguous*
Are you in the business of brokering properties or contacts?	*Ambiguous*	*Clear*
WHO'S GETTING MARRIED?	*Ambiguous*	*Clear*
Did you ever meet my cousin Chuck and his wife Shelley?	*Ambiguous*	*Clear*
Why don't we set up a lunch so that you and I can have the opportunity to talk separate from immediate issues at hand?	*Ambiguous*	*Clear*

Table 20: Faulty classifications for RIPPER classifier (Detection)

As discussed before, the RIPPER-model is biased and tends to predict questions to be Ambiguous more often than needed. The table above clearly showcases this. However the first question has been classified as Clear when it should actually be classified as Ambiguous because the "these files" refers to something that is not specified in this text. However the sentence structure is clear and "these" could also be interpreted to refer to "files". Questions four and five are both classified as ambiguous when they are actually clear. This error occurred because one of the RIPPER rules is to classify questions with "or" and "and" as ambiguous because it often indicates a double question.

Step 2 [Clarification/Enrichment]

[1] Context + [2] Question	Prediction	Target
[1] I will be there at about 7. [2] Where are you going to be, that puts you on that side of town?	*Clear*	*Ambiguous*
[1] Some analysts believe that assets no longer give "competitive advantage" to energy companies, as the focus shifts towards the end-user and convergence of gas and electricity. [2] TXU's principal production assets comprise the operating licence and 64.2% of the 83bn ft?	*Clear*	*Ambiguous*
[1] I know Naomi Connell has put together the SOAP which her group in London uses for this purpose which might be useful. [2] Naomi - could you send the three of us a copy of this document?	*Ambiguous*	*Clear*
[1] I'm having trouble setting a syncopated basis child of a syncopated basis child (grandchild). [2] Is this not currently allowed by the system?	*Ambiguous*	*Clear*
[1] For some reason, HR has confused the issue with the Pulp & Paper employees by putting them under Global Products rather than ENA, a step they took I believe without any outside direction. [2] Does this make sense to you?	*Ambiguous*	*Clear*

Table 21: Faulty classifications for Support Vector Machines classifier (Enrichment)

The table above shows some of the errors the Support Vector Machine classifier made during the enrichment phase. The phase in which the classifier not only looks at the question but also at the previous sentence for more context. For example, Question 3 has been predicted to be Ambiguous by the classifier when it should have been clear. This error likely happened because the sentence before still has an undefined reference word "this". However, the definition of "this" is not needed to answer the question which is why the manual label is Clear.

[1] Context + [2] Question	Prediction	Target
[1] Stocks, Bonds and commodities market have all thrown in the towel for a V shape recovery. [2] What's the probability that all three markets are wrong?	*Clear*	*Ambiguous*
[1] On the cable side, Excite@Home recently filed for bankruptcy, and numerous broadband wireless providers including Winstar and Metricom have also dissolved. [2] But what lies ahead for those that survive?	*Ambiguous*	*Clear*
[1] Besides that day there are many smaller differences. [2] Can you help me by pulling the P&L packets from some of the larger days?	*Clear*	*Ambiguous*
[1] Other wine auctions charge as much as 12% to the buyer AND 12% to the seller. [2] What are we going to do with the commission?	*Ambiguous*	*Clear*

Table 22: Faulty classifications for RIPPER classifier (Enrichment)

The table above shows faulty classifications with the RIPPER classifier on the enrichment step. The last sample is labeled as ambiguous due to the "and" keyword. But again, in this context it did not signify a double question. This also likely happened in the second sample. The second to last question is quite tricky. The entire question is clear, except for the day which is not clarified.

6 Discussion & Future Work

In this Section the evaluated results are discussed. To do this in a structured manner, we shall go over each sub research question one by one. Afterwards, the limitations of this research are reflected upon and future work is discussed.

6.1 Discussion

SRQ1: *What are the characteristics of ambiguous type questions?*
Since this research is primarily concerned with "ambiguous" questions, the first question that arises is "What exactly is an ambiguous question?". To answer this question, we constructed an extensive labeling protocol. This protocol defines what an "ambiguous" type question is in the question answering domain. It combines the definition of structural& lexical ambiguity with the way in which humans understand questions in a practical context. This implies that questions that can be interpreted in multiple ways are considered ambiguous, regardless of causation. In the extended labeling protocol 4.1.1, ambiguity is therefore defined in multiple ways. It includes structural ambiguity (multiple questions, unclear information, context missing), but also a different type of ambiguity (confirmation, personal information, complexity). We therefore define an ambiguous type question as a question that for a variety of reasons is hard to understand.

Without setting clear guidelines to characterize ambiguous type questions, no measurements can be done and thus no research can be conducted. This is why the extended labeling protocol is so important and why this research question has been tackled and analyzed so extensively. Throughout the research experiments, this definition sometimes had to be updated, which meant that all results gathered up till that point had to be discarded. Only after several iterations of re-labeling the data and testing and reflection upon the labeling distribution, the (extended) labeling protocol was complete.

SRQ2.1: *To what extent is it possible to automatically detect ambiguous type questions in a restricted domain environment?*
The first goal of this project was to try and automate the process of ambiguous type question detection. As we can see between all classifiers, the fine-tuned BERT model has an AUC score of 0.940. This is very high, considering the data set used is imbalanced. The AUC metric is quite sensitive to imbalanced data. So this score is excellent and shows that it is most definitely possible to automatically detect ambiguous type questions in a (professional) restricted domain environment. Although the questions in this data set followed a structured manner, there was still a large variety in data. The people that submitted these questions used an e-mail format. There are no guidelines to how these e-mail are written, meaning that they have complete freedom. The only part that is structured is the fact that the questions pertain to a certain subject.

When considering the accuracy and F1 scores of the best scoring model, once again the BERT model scores best. This model scored an accuracy of over 90%, implying that out of all classifications, 90% are correct. For ambiguous type questions specifically we see an f1 score of 0.93 for the Dutch data set, which is a very good score considering the model is not too biased in its results. The minority class still has an f1 score of 0.8 in this case. The English data set results show very similar AUC and F1 scores.

SRQ2.2: *How does a Rule-based approach compare to a Machine Learning approach for question type classification in a restricted domain environment?*
When looking at the AUC scores for both sets of classifiers, we cannot easily draw a conclusion.

This is because Rule-based classifiers follow a crisp decision making process when classifying. Rule-based classifiers make their predictions based on a set of predefined rules or conditions, often in the form of "if-then" statements. As a result, their decision boundaries can be discrete and often do not yield a continuous probability distribution that produces a smoothened ROC curve. This can be due to incomplete coverage. Rule-based classifiers may not cover all edge cases, resulting in regions where no rules apply. This is why we can see abrupt changes in the AUC score at different thresholds. Furthermore the output of the rule based classifiers used in this experiment is discrete and not probabilistic. This is why looking at the classification reports is much more relevant to answering this research question.

As we can see, the Decision Tree based classifier is actually very close in performance to our fine-tuned BERT model. This is an interesting result, since the fine-tuning of the already pre-trained BERT costs almost a full day of training. This is excluding the pre-training process that all BERT models have undergone. Considering the difference in resources allocated toward these classifiers, the result of the Decision Tree classifier is exceptional for this task. What must be stated is that the variance for the Decision Tree classifier was much greater. A tree depth of approximately 50 nodes was used for this experiment.

Regardless of the impressive result by the Decision Tree classifier, the BERT model still comes out on top. What might be interesting to point out is the fact that apart from the heavily pre-trained and fine-tuned BERT model, every other model scored worse than the Decision Tree classifier.

SRQ3: *How is question type classification in a restricted domain influenced by the language of the data?*
If we consider the results gathered in the Enron data set, can it be said that question type classification is simply much easier for Dutch data sets? This would be quite the ignorant conclusion. Unfortunately this research has not been able to answer this question confidently. The data set in English was much different in nature. This data set simply consisted of e-mail exchanges between colleagues. Therefore it is to be expected that the data is much more chaotic in nature. This makes the process of automated question type detection much more difficult. After removing the data points labeled as irrelevant to this research, half of the data set was lost. So apart from the already chaotic nature of this data set, it now also a more sparse data set, making a direct comparison even harder. The English data set was still a necessary component to this research, since the Dutch data set is anonymized. With the help of the English data set, it was possible to show all the pre-processing and labeling steps.

Does this mean no conclusions can be drawn from this extra data set? Interestingly enough, we can see that the BERT model performed the best on the English data set. It performed better than the BERT model on the Dutch data set. It scored better in both the AUC metric as well as precision, recall and F1-score. How this is possible is not entirely clear. Considering the data set is both smaller and more chaotic, this is quite unique. Does this mean that the pre-trained BERT model is better at performance than its Dutch counterpart? Perhaps the chaotic nature of the data suits the BERT model quite well. It can be the case that it can pick up on informal language easier than the other classifiers. Every other classifier nearly performed worse on the English data set.

SRQ4.1: *To what extent is it possible to enrich ambiguous type questions through context analysis?*

This proved a difficult problem for all classifiers. A steep decline can be observed across all metrics for this task. Every classifier had a much harder time differentiating between questions that are enriched versus questions that are still ambiguous. The first obvious source of this problem is the decrease in data size. This led to more bias across all models. This can be observed quite clearly by looking at the classification reports. Even the best classifier (BERT model) had a F1-score of 0.62 for clear type question classification.

Interestingly enough, when looking at the results for enrichment with two prior sentences, the BERT model suddenly performs better. Although it is slightly more biased, it scored an accuracy of up to 80%. This follows the observation of the English data set. With the inclusion of more sentences, the data can be considered more chaotic in nature. BERT once more proves its strength in this kind of environment.

SRQ4.2: *How does a Rule-based approach compare to a Machine Learning approach for ambiguous type question enrichment?*
The results regarding this research question are according to our hypothesis. We can see that the Rule-based classifier perform poorly in the AUC metric. As stated before this is to be expected, given the deterministic nature of these models. However, when we look at the classification reports, the results are not much different. The RIPPER algorithm was completely biased toward ambiguous type questions and did not manage to enrich the ambiguous type questions whatsoever. With a highest AUC score of 0.583 and highest F1-score of 0.70 for the enrichment step, the Rule-based classifier scored lowest amongst all classifiers in the task of question enrichment. Even the Decision Tree classifier, which has a categorical target variable, was still more biased toward ambiguous type questions than both shallow and deep machine learning classifiers.

6.2 Validity threats

Since this is quite a large scale comparison, there are some relevant details that may influence results and pose a threat to the overall validity of the research design.

Firstly, there are some minor differences in set-up between the different classifiers used in this experiment. The Dutch implementation of BERT (RobBERT) is a large pre-trained language model. Since it utilizes deep learning to derive meaning from each sentence, it uses one system for both lemmatizing and tokenizing the text. BERT uses a "WordPiece" tokenizer (Devlin et al., 2018), that splits words into full forms or so called "word-pieces". It is a more detailed-approach to find relations between words. The BERT tokenizer used for this experiment was built by utilizing a large data set (pre-training). This implies that there is essentially an extra "pre-processing" step for the BERT model. Moreover, BERT generates different output vectors for each word, depending on the word context. The TF-IDF embedding provides a single, context-independent vector instead. This poses a threat to the internal validity of the comparison of the different classifiers. The Naïve Bayes classifier uses a weight function when fitting the model on the training data. Without introducing weights to the labels, this classifier was not able to make meaningful predictions. The Rule-based classifier required a different pre-processing step as input for the RIPPER algorithm. To introduce new rules to the decision process, it was not possible to convert each word with the TF-IDF transformer, unlike most other classifiers. This means that the embedding step for the Rule-based classifier is less sophisticated than for the other classifiers.

This implies that most classifiers use a slightly different setup. This is largely due to programming limitations, but also due to study design. This research primarily aims at discovering

the ability of automatically recognizing ambiguous question types. Therefore, we want to ensure that each classifier is performing to the best of its ability. If we force each classifier to utilize label weights during training, or we strip all classifiers of their embedding step. This would be a sacrifice in overall performance for the sake of uniformity and integrity of the classifier comparison, which is not the primary goal of this research.

The data is also quite imbalanced. Due to the sparse nature of the data, it was not possible to construct a sufficiently large and balanced data set. Since the imbalance of this data is a true representation of real data, this is considered preferable over artificially re-balancing the data. Since re-balancing might lead to external validity of the experiment. This threat is discussed in more detail in the Discussion section of this report.

6.3 Limitations & Future work

The research is conducted in Dutch & English, but not in other languages. Furthermore, the comparison was done within a Restricted domain and the data sets are quite specific. The Dutch data set only covers government related questions for instance. Future research should try to either replicate this study in a different language or different domain. Using a different data set could confirm or deny the results gathered in this research. It is interesting to see to what degree the predictive power is influenced by these factors.

Another great limitation was the size of the data set. This was due to the fact that labeling was mostly done manually by experts. This gave insurance on the quality and accuracy of these labels, but it also made the labeled data set relatively small in size. In the future, this experiment would greatly benefit from a large group of experts to label the data. This way a more detailed labeling protocol can be used for more accurate results during classification.

Another limitation of this experiment is the lack of resources available. Most notably, it was not possible to perform extensive parameter tuning for all of the models used in this experiment. It was possible to implement a data re-balancing strategy, however this was also a limitation in itself due to the decrease in size of an already small data set. Apart from this, different weights for feature importance were applied for the non-deep learning machine learning methods. But it was not possible to extensively fine-tune the parameters for the RNN and BERT deep learning models. This was due to the run-time of these models being way too high to run locally. Therefore in future research it would be very interesting to see more resources allocated to parameter tuning of the deep learning methods.

Another limitation was the unrefinedness of the Rule-based classifier. It was very hard for me to construct a solid Rule-based classifier in this project. Perhaps this is due to a lack of programming skills. Personally I think this is mostly due to the nature of the data. During this project I realised that constructing a Rule-based classifier for a complex question such as recognizing question ambiguity is nearly impossible. Now consider step 2, the enrichment process, which is even more complex. After re-visiting related literature, it becomes increasingly evident that Rule-based classification is mostly performed on tasks such as sentiment analysis. This way key-words can be included in the rule set.

In future research it would be very interesting to look deeper into the question enrichment process. Due to a limitation in resources it was evident that the data sets got too small for the question enrichment process. Also it was not feasible to train separate deep learning models for each of the enrichment steps. In future research it is interesting to see how this enrichment

process can perform under larger data sets with more granular fine tuning of the BERT models.

During this research it was evident that the data set did not follow a common structure, meaning that the data was quite sparse in a way. This posed a limitation for the learn-ability of the data set through machine learning techniques. Due to the large variety in questions in this experiment, following the labeling protocol proved difficult at times. Perhaps in a different domain, there is a more structured set of questions to be analyzed. Some extra formatting steps beforehand would have been very useful before developing a labeling protocol.

Since this research covers both Dutch and English data sets, the results of this experiment can be useful for future research surrounding question type recognition specifically. The second step of this research, question enrichment, was only executed in a very simplistic manner. Due to the time frame and resources allocated to this project, the results are not conclusive. Perhaps with a larger data set, different enrichment techniques can be extrapolated.

6.4 Conclusion

This research has attempted to make a clear distinction between ambiguous question types and clear question types within a formal setting. To define this distinction, an extensive labeling protocol has been constructed. This protocol has been applied to measure the predictive power of Rule-based classifiers and Machine Learning classifiers with regards to question type detection and question enrichment. Results indicate that question ambiguity can be detected automatically. The rigorous metrics (AUC-score and F1-score upwards of 0.90) validate this claim. Given the relatively small size of the data set, these results are impressive. Regarding the comparison of Rule-based techniques versus Machine Learning techniques, this research was not conclusive. However, Machine Learning techniques did outperform Rule-based classifiers for both tasks, question detection and question enrichment respectively.

References

Abbas, M., Memon, K. A., Jamali, A. A., Memon, S., & Ahmed, A. (2019). Multinomial naive bayes classification model for sentiment analysis. *IJCSNS Int. J. Comput. Sci. Netw. Secur*, *19*(3), 62.

Adelaide, A. (2022). The 4 main types of questions in english + examples. *preply*.

Alazani, S. A., & Mahender, C. N. (2021). Rule based question generation for arabic text: Question answering system. *Proceedings of the International Conference on Data Science, Machine Learning and Artificial Intelligence*, 7–12.

Allam, A., & Haggag, M. (2012). The Question Answering Systems: A Survey. *International Journal of Research and Reviews in Information Sciences*, *2*, 211–221.

Al-Rfou, R., Choe, D., Constant, N., Guo, M., & Jones, L. (2019). Character-level language modeling with deeper self-attention. *Proceedings of the AAAI conference on artificial intelligence*, *33*(01), 3159–3166.

Aly, M. (2005). Survey on multiclass classification methods. *Neural Netw*, *19*(1-9), 2.

Aubaid, A. M., & Mishra, A. (2020). A rule-based approach to embedding techniques for text document classification. *Applied Sciences*, *10*(11), 4009.

Azevedo, P., Leite, B., Cardoso, H. L., Silva, D. C., & Reis, L. P. (2020). Exploring nlp and information extraction to jointly address question generation and answering. *IFIP International Conference on Artificial Intelligence Applications and Innovations*, 396–407.

Bahdanau, D., Cho, K., & Bengio, Y. (2018). Neural machine translation by jointly learning to align and translate (2014). *arXiv preprint arXiv:1409.0473*.

Biswas, P., Sharan, A., & Malik, N. (2014). A framework for restricted domain Question Answering System. *2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, 613–620. https://doi.org/10.1109/ICICICT.2014.6781351

Bouziane, A., Bouchiha, D., Doumi, N., & Malki, M. (2015). Question answering systems: Survey and trends. *Procedia Computer Science*, *73*, 366–375.

Cahyani, D. E., & Patasik, I. (2021). Performance comparison of tf-idf and word2vec models for emotion text classification. *Bulletin of Electrical Engineering and Informatics*, *10*(5), 2780–2788.

Cai, L.-Q., Wei, M., Zhou, S.-T., & Yan, X. (2020). Intelligent Question Answering in Restricted Domains Using Deep Learning and Question Pair Matching [Conference Name: IEEE Access]. *IEEE Access*, *8*, 32922–32934. https://doi.org/10.1109/ACCESS.2020.2973728

Cervantes, J., Garcia-Lamont, F., Rodrıguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, *408*, 189–215.

Chan, L. (2021). Google's rfa: Approximating softmax attention mechanism in transformers. *Towards Data Science*.

Charbuty, B., & Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, *2*(01), 20–28.

Chiticariu, L., Li, Y., & Reiss, F. (2013). Rule-based information extraction is dead! long live rule-based information extraction systems! *Proceedings of the 2013 conference on empirical methods in natural language processing*, 827–832.

Chowdhary, K. R. (2020). Natural Language Processing. In K. Chowdhary (Ed.), *Fundamentals of Artificial Intelligence* (pp. 603–649). Springer India. https://doi.org/10.1007/978-81-322-3972-7_19

Cohen, W. W. (1995). Fast effective rule induction. *Machine learning proceedings 1995* (pp. 115–123). Elsevier.

Cong, G., Wang, L., Lin, C.-Y., Song, Y.-I., & Sun, Y. (2008). Finding question-answer pairs from online forums. *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 467–474.

Cruz-Blandón, M.-A., Minnema, G., Nourbakhsh, A., Boritchev, M., & Amblard, M. (2019). Toward dialogue modeling: A semantic annotation scheme for questions and answers. *arXiv preprint arXiv:1908.09921*.

Delobelle, P., Winters, T., & Berendt, B. (2020a). RobBERT: A Dutch RoBERTa-based Language Model. *Findings of the Association for Computational Linguistics: EMNLP 2020*, 3255–3265. https://doi.org/10.18653/v1/2020.findings-emnlp.292

Delobelle, P., Winters, T., & Berendt, B. (2020b). Robbert: A dutch roberta-based language model. *arXiv preprint arXiv:2001.06286*.

Derici, C., Çelik, K., Kutbay, E., Aydın, Y., Güngör, T., Özgür, A., & Kartal, G. (2015). Question Analysis for a Closed Domain Question Answering System. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing* (pp. 468–482). Springer International Publishing. https://doi.org/10.1007/978-3-319-18117-2_35

Derrar, H. M. (2019). CLUSTERING FOR THE AUTOMATIC ANNOTATION OF CUSTOMER SERVICE CHAT MESSAGES, 75.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G., & Nissim, M. (2019). Bertje: A dutch bert model. *arXiv preprint arXiv:1912.09582*.

Dharma, E. M., Gaol, F. L., Warnars, H., & Soewito, B. (2022). The accuracy comparison among word2vec, glove, and fasttext towards convolution neural network (cnn) text classification. *Journal of Theoretical and Applied Information Technology*, *100*(2), 31.

Diao, Y., Jamjoom, H., & Loewenstern, D. (2009). Rule-based problem classification in it service management. *2009 IEEE International Conference on Cloud Computing*, 221–228.

Elkins, K., & Chun, J. (2020). Can gpt-3 pass a writer's turing test? *Journal of Cultural Analytics*, *5*(2).

Ferri, C., Hernández-Orallo, J., & Salido, M. A. (2003). Volume under the roc surface for multi-class problems. *European conference on machine learning*, 108–120.

Fujino, A., Isozaki, H., & Suzuki, J. (2008). Multi-label text categorization with model combination based on f1-score maximization. *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, *44*(8), 1761–1776.

Garg, D., Kambhatla, N., Vukovic, M., & Pingali, G. (2008). Mining top issues from contact center logs for self help portals. *2008 IEEE International Conference on Services Computing*, *2*, 171–178.

Gasparetto, A., Marcuzzo, M., Zangari, A., & Albarelli, A. (2022). A survey on text classification algorithms: From text to predictions. *Information*, *13*(2), 83.

González-Carvajal, S., & Garrido-Merchán, E. C. (2021). Comparing BERT against traditional machine learning text classification [arXiv: 2005.13012]. *arXiv:2005.13012 [cs, stat]*. Retrieved March 24, 2022, from http://arxiv.org/abs/2005.13012

Gorinski, P. J., Wu, H., Grover, C., Tobin, R., Talbot, C., Whalley, H., Sudlow, C., Whiteley, W., & Alex, B. (2019). Named entity recognition for electronic health records: A comparison of rule-based and machine learning approaches. *arXiv preprint arXiv:1903.03985*.

Goyal, A., Gupta, V., & Kumar, M. (2018). Recent named entity recognition and classification techniques: A systematic review. *Computer Science Review*, *29*, 21–43.

Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: An overview. *arXiv preprint arXiv:2008.05756*.

Gupta, P., & Gupta, V. (2012). A survey of text question answering techniques. *International Journal of Computer Applications*, *53*(4).

Hadi, W., Al-Radaideh, Q. A., & Alhawari, S. (2018). Integrating associative rule-based classification with naıve bayes for text classification. *Applied Soft Computing*, *69*, 344–356.

Haj-Yahia, Z., Sieg, A., & Deleris, L. A. (2019). Towards unsupervised text classification leveraging experts and word embeddings. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 371–379.

Harrak, F., Bouchet, F., & Luengo, V. (2019). Categorizing students' questions using an ensemble hybrid approach. *Educational Data Mining*.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy [Number: 7825 Publisher: Nature Publishing Group]. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hien, L. T., Ly, L. T. T., Pham-Nguyen, C., Le Dinh, T., Gia, H. T., et al. (2020). Towards chatbot-based interactive what-and how-question answering systems: The adobot approach. *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, 1–3.

Huang, Z., & Zhao, W. (2020). Combination of elmo representation and cnn approaches to enhance service discovery. *IEEE Access*, *8*, 130782–130796.

Ishii, K., & Markman, K. M. (2016). Online customer service and emotional labor: An exploratory study. *Computers in Human Behavior*, *62*, 658–665. https://doi.org/10.1016/j.chb.2016.04.037

Joshi, A. K. (1991). Natural language processing. *Science*, *253*(5025), 1242–1249.

Joshi, M., Choi, E., Weld, D. S., & Zettlemoyer, L. (2017). Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Jusoh, S. (2018). A study on nlp applications and ambiguity problems. *Journal of Theoretical & Applied Information Technology*, *96*(6).

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., et al. (2019). Measuring compositional generalization: A comprehensive method on realistic data. *arXiv preprint arXiv:1912.09713*.

Khella, R. (2017). Rule Based System for Chest Pain in Infants and Children. *International Journal of Engineering and Information Systems*, *1*(4), 138–148. Retrieved March 24, 2022, from https://hal.archives-ouvertes.fr/hal-01562357

Klimt, B., & Yang, Y. (2004). Introducing the enron corpus. *CEAS*, *45*, 92–96.

Kocoń, J., Cichecki, I., Kaszyca, O., Kochanek, M., Szydło, D., Baran, J., Bielaniewicz, J., Gruza, M., Janz, A., Kanclerz, K., et al. (2023). Chatgpt: Jack of all trades, master of none. *arXiv preprint arXiv:2302.10724*.

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text Classification Algorithms: A Survey [Number: 4 Publisher: Multidisciplinary Digital Publishing Institute]. *Information*, *10*(4), 150. https://doi.org/10.3390/info10040150

Kwong, H., & Yorke-Smith, N. (2012). Detection of imperative and declarative question-answer pairs in email conversations. *AI Communications*.

Landgrebe, T., & Duin, R. (2006). A simplified extension of the area under the roc to the multiclass domain. *Seventeenth annual symposium of the pattern recognition association of South Africa*, 241–245.

Lende, S. P., & Raghuwanshi, M. M. (2016). Question answering system on education acts using NLP techniques. *2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave)*, 1–6. https://doi.org/10.1109/STARTUP.2016.7583963

Lewis, B. R., & Mitchell, V. W. (1990). Defining and Measuring the Quality of Customer Service [Publisher: MCB UP Ltd]. *Marketing Intelligence & Planning*, *8*(6), 11–17. https://doi.org/10.1108/EUM0000000001086

Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P. S., & He, L. (2020). A survey on text classification: From shallow to deep learning. *arXiv preprint arXiv:2008.00364*.

Li, X., & Roth, D. (2002). Learning question classifiers. *COLING 2002: The 19th International Conference on Computational Linguistics*.

Ling, C. X., Huang, J., Zhang, H., et al. (2003). Auc: A statistically consistent and more discriminating measure than accuracy. *Ijcai*, *3*, 519–524.

Liu, Y., & Lapata, M. (2019). Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Loper, E., & Bird, S. (2002). Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

Luong, M., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *CoRR*, *abs/1508.04025*. http://arxiv.org/abs/1508.04025

Mansoori, E. G., Zolghadri, M. J., & Katebi, S. D. (2007). A weighting function for improving fuzzy classification systems performance. *Fuzzy sets and systems*, *158*(5), 583–591.

McHugh, M. L. (2012). Interrater reliability: The kappa statistic. *Biochemia medica*, *22*(3), 276–282.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2019). Efficient estimation of word representations in vector space. arxiv preprint (2013). *arXiv preprint arXiv:1301.3781*, *10*.

Min, S., Michael, J., Hajishirzi, H., & Zettlemoyer, L. (2020). Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning–based text classification: A comprehensive review. *ACM Computing Surveys (CSUR)*, *54*(3), 1–40.

Mirończuk, M. M., & Protasiewicz, J. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, *106*, 36–54.

Mishra, A., & Jain, S. K. (2016). A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, *28*(3), 345–361.

Moldovan, D., Paşca, M., Harabagiu, S., & Surdeanu, M. (2003). Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, *21*(2), 133–154.

Nadkarni, P. M., Ohno-Machado, L., & Chapman, W. W. (2011). Natural language processing: An introduction. *Journal of the American Medical Informatics Association*, *18*(5), 544–551. https://doi.org/10.1136/amiajnl-2011-000464

Narkhede, S. (2018). Understanding auc-roc curve. *Towards Data Science*, *26*(1), 220–227.

Naumov, N. (2019). *The impact of robots, artificial intelligence, and service automation on service quality and service experience in hospitality*. emerald publishing limited.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., & Deng, L. (2016). Ms marco: A human generated machine reading comprehension dataset. *CoCo@ NIPS*.

Nobel, J. M., Puts, S., Bakers, F. C., Robben, S. G., & Dekker, A. L. (2020). Natural language processing in dutch free text radiology reports: Challenges in a small language area staging pulmonary oncology. *Journal of digital imaging*, *33*(4), 1002–1008.

Oleynik, M., Kugic, A., Kasáč, Z., & Kreuzthaler, M. (2019). Evaluating shallow and deep learning strategies for the 2018 n2c2 shared task on clinical text classification. *Journal of the American Medical Informatics Association*, *26*(11), 1247–1254.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, *35*, 27730–27744.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, *12*, 2825–2830.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *EMNLP*, *14*, 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237. https://doi.org/10.18653/v1/N18-1202

Pranckevičius, T., & Marcinkevičius, V. (2017). Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification. *Baltic Journal of Modern Computing*, *5*(2), 221.

Qin, C., Zhang, A., Zhang, Z., Chen, J., Yasunaga, M., & Yang, D. (2023). Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.

Radev, D., Fan, W., Qi, H., Wu, H., & Grewal, A. (2002). Probabilistic question answering on the web. *Proceedings of the 11th international conference on World Wide Web*, 408–419.

Rajpurkar, P., Jia, R., & Liang, P. (2018). Know What You Don't Know: Unanswerable Questions for SQuAD [arXiv: 1806.03822]. *arXiv:1806.03822 [cs]*. Retrieved March 24, 2022, from http://arxiv.org/abs/1806.03822

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation - J DOC*, *60*, 503–520. https://doi.org/10.1108/00220410410560582

Rong, X. (2014). Word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*.

Rouws, N. J., Vakulenko, S., & Katrenko, S. (2021). Dutch squad and ensemble learning for question answering from labour agreements. *Benelux Conference on Artificial Intelligence*, 155–169.

Rullo, P., Policicchio, V. L., Cumbo, C., & Iiritano, S. (2008). Olex: Effective rule learning for text categorization. *IEEE Transactions on Knowledge and Data Engineering*, *21*(8), 1118–1132.

Sangodiah, A., Muniandy, M., & Heng, L. E. (2015). Question classification using statistical approach: A complete review. *Journal of Theoretical & Applied Information Technology*, *71*(3).

Sarkar, D., Bali, R., & Ghosh, T. (2018). *Hands-on transfer learning with python: Implement advanced deep learning and neural network models using tensorflow and keras*. Packt Publishing Ltd.

Seger, C. (2018). An investigation of categorical variable encoding techniques in machine learning: Binary versus one-hot and feature hashing.

Shrestha, L., & McKeown, K. (2004). Detection of question-answer pairs in email conversations. *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, 889–895. https://aclanthology.org/C04-1128

Tayyar Madabushi, H., & Lee, M. (2016). High Accuracy Rule-based Question Classification using Question Syntax and Semantics. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 1220–1230. Retrieved March 24, 2022, from https://aclanthology.org/C16-1116

Tian, X., Vertommen, I., Tsiami, L., van Thienen, P., & Paraskevopoulos, S. (2022). Automated customer complaint processing for water utilities based on natural language processing—case study of a dutch water utility. *Water*, *14*(4), 674.

Trienes, J., Trieschnigg, D., Seifert, C., & Hiemstra, D. (2020). Comparing rule-based, feature-based and deep neural methods for de-identification of dutch medical records. *arXiv preprint arXiv:2001.05714*.

Tuychiev, B. (2021). Comprehensive guide to multiclass classification metrics. *Towards Data Science*.

Vanderlooy, S., & Hüllermeier, E. (2008). A critical analysis of variants of the AUC. *Machine Learning*, *72*(3), 247–262. https://doi.org/10.1007/s10994-008-5070-x

van Haastrecht, M., Sarhan, I., Yigit Ozkan, B., Brinkhuis, M., & Spruit, M. (2021). Symbals: A systematic review methodology blending active learning and snowballing. *Frontiers in research metrics and analytics*, *6*, 33.

Vasiliev, Y. (2020). *Natural language processing with python and spacy: A practical introduction*. No Starch Press.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2019). Attention is all you need (2017). *arXiv preprint arXiv:1706.03762*.

Wang, K., & Chua, T.-S. (2010). Exploiting salient patterns for question detection and question retrieval in community-based question answering. *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, 1155–1163. https://aclanthology.org/C10-1130

White, M. G. (2019). Examples of open-ended vs. closed-ended questions. *yourdictionary*.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. (2019). Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yuhana, U. L., Rochimah, S., Yuniarno, E. M., Rysbekova, A., Tormasi, A., Koczy, L. T., & Purnomo, M. H. (2019). A rule-based expert system for automatic question classification in mathematics adaptive assessment on indonesian elementary school environment. *Int. J. Innov. Comput. Inf. Control*, *15*(1), 143–161.

Zhu, Y., Shen, J., & Zhang, M. (2019). Learning to answer ambiguous questions with knowledge graph. *arXiv preprint arXiv:1912.11668*.

Appendices

A Systematic Literature Review

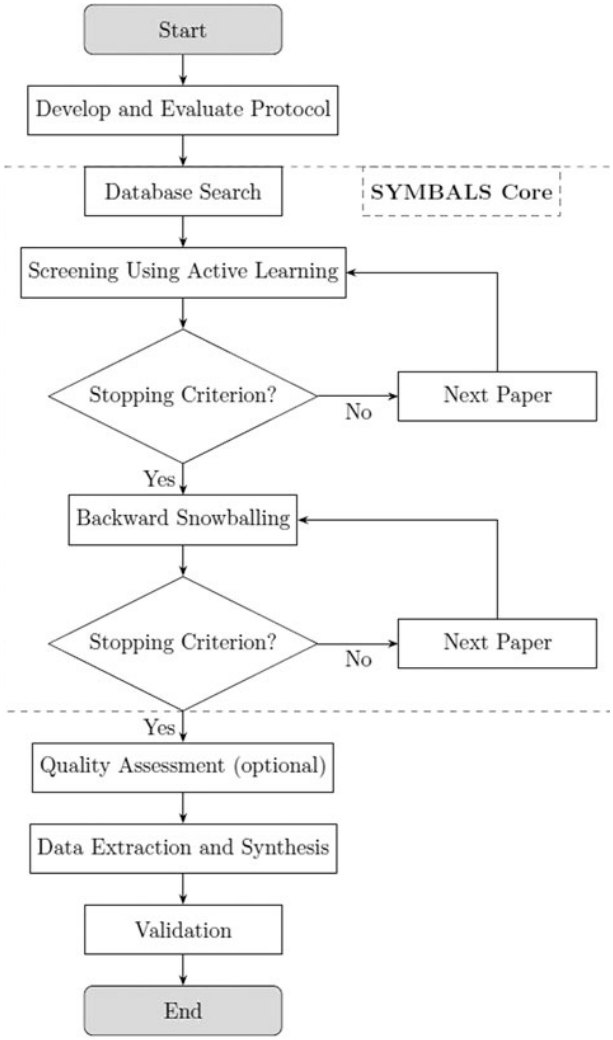

Figure 20: SYMBALS: Review methodology (van Haastrecht et al., 2021)

The literary review surrounding the Related Work Section of this report is carried out systematically, following the SYMBALS methodology (van Haastrecht et al., 2021). It is comprised of an extensive data base search, following a carefully chosen protocol. Here follows a brief summary of the SYMBALS method 20.

First a protocol is selected, by investigating background information and formulating a research rationale. In this step the motivation and goal for the literature review is established. The objective for this research was to specify the existing research questions and formulate subquestions. The criteria for what kind of sources are deemed relevant for this research and the desired number of sources is (stopping criterion), were specified. A more detailed description of the steps carried out for this research specifically can be found in the appendix A

Next up an extensive database search is carried out. Here the stopping criteria from the previous are applied to extract a bulk of sources. This research works with Google Scholar and

ProQuest. Both of these academic databases have many high quality sources related to the topic of Natural Language Processing. By utilizing multiple data bases we cover a wider variety of fields. Zotero was used as a data management system to extract and store the chosen articles in the right format. Here we also ensure that duplicates are deleted.

After reaching the stopping criterion we stop the data base search and start screening the found sources. Afterward a number of sources is screened to estimate the number of relevant sources among the bulk. 5 out of 30 sources were deemed to be relevant for my research (approximately 17%) which mapped onto approximately 500 relevant sources. We then use this estimated number of relevant sources as a stopping criterion for the Active Learning step. ASReview was used for the active learning step. Now follows a brief overview of the steps taken during this research:

This Section describes the protocol of the systematic literature review for this paper. The systematic literature review follows the SYMBALS methodology (van Haastrecht et al., 2021). It is comprised of 7 steps:

1. Motivation
2. Database search
3. Active Learning
4. Backward Snowballing
5. Quality Assessment
6. Data Extraction & Synthesis
7. Validation

A.1 Motivation

A.1.1 Why perform a systematic literature review?

To ensure that the research question is not already studied rigorously and answered accordingly.
To ensure that the research question is currently relevant in its respective research domain.
To ensure that contextual information that is already published can be utilized.
To reflect on the spectrum of research and validation methods that have been used in similar research.
To place this research well in its respective field.
To validate this research and ensure its reproducibility.

A.1.2 What are you trying to achieve with a systematic literature review?

To clarify the problem statement and formulate it concisely.
To align the research questions with the context & relevant literature.
To determine a suitable research and validation method.

A.1.3 How are you going to accomplish these goals?

The systematic literature review starts with a basis of 10 research papers that are fundamental to my research subject. These papers serve as the starting point of the literary review. From these initial papers keywords should be extracted for the search strings. These papers are also used as a guideline for the inclusion and exclusion criteria in the Database search.

After having completed the Database search, relevant literature will be categorized in to sub-domains. These subcategories are used in the next step, the Active Learning component. For each sub-domain Active Learning will be applied to screen the papers' abstracts and titles. This will result in a set of relevant/noteworthy sources for each sub-domain of this research. A stopping criterion is here is applied based on the number of expected relevant sources.

Next, backward snowballing (and optionally forward snowballing) is applied to gather relevant supportive literature to finalize and complete the literary review for each sub-domain. Appropriate stopping criteria are applied during snowballing to ensure this step does not take too long and to ensure a relevant and manageable amount of sources.

After this step, the absolute number sources is reviewed. If there are too many sources, a Quality assessment is applied to trim down this number to a suitable amount. The Quality assessment is to be carried out manually, unless the number of sources is too large.

Afterward, the gathered sources will be formatted in a structured manner to provide an overview of the research field. During the course of this project more sources might be added when new relevant information is discovered, including grey literature.

A.2 Database Search

The ProQuest and GoogleScholar databases were utilized for the database search. The ProQuest database utilizes 15 different databases simultaneously to process each query, these include:

Coronavirus Research Database
Digital National Security Archive
Documents on British Policy Oversears
Early Modern Books
Ebook Central
Entertainment Industry Magazine Archive
ERIC
International Bibliography of Art (IBA)
Latino Literature: Poetry, Drama and Fiction
Linguistics and Language Behavior Abstracts (LLBA)
Periodicals Archive Online
ProQuest Historical Newspapers: The Guardian and The Observer
PTSDpubs
Publicly Available Content Database
Sociological Abstracts

A.2.1 Querying

Multiple search queries were used to prevent missing out on too many relevant articles and to make it easier to categorize the literature into sub-domains.

To finalize each search string, start with an initial search string and select the first 100 sources. Then, alter the search string (usually by adding a word) and look at the first 100 sources again. Now the previously selected sources are still highlighted and the sources that are not selected yet are there because of the search query change. Look at these new sources and by screening the title decide whether the change in search query is desirable. For each search string, it is possible to include/exclude certain research subjects. Down below the exclusions for each search string in this research is listed:

Search String:

Question Answering Systems
Period: Last 5 years
Excluding: none
Total Results: 38,734
Included first 300, sorted by relevance

Search String:

Machine Learning Question Classification text
Period: Last 5 years
Excluding:
NOT (covid-19 AND political science AND sustainability AND climate change AND religion AND proteins AND genomes AND coronaviruses AND politics AND pandemics AND laboratories AND philosophy AND gene expression AND language arts AND medical research AND sensors AND mental disorders AND mental health AND epidemiology)
Total Results: 17,714
Included first 500 search results (Sorted by Relevance)

Search String:

Rule based classification Question text
Period: Last 5 years
Excluding:
NOT (covid-19 AND political science AND sustainability AND climate change AND religion AND proteins AND genomes AND coronaviruses AND politics AND pandemics AND laboratories AND philosophy AND gene expression AND language arts AND medical research AND sensors AND mental disorders AND mental health AND epidemiology)
Total results: 27,924
included first 500, sorted by relevance

Search String:

Question classification rule based machine learning
Period: Last 10 years
Including:
(machine learning OR algorithms OR artificial intelligence OR engineering OR classification OR linguistics OR learning algorithms OR decision making OR language OR natural language processing) NOT 20th century
Total results; 13,909
Included first 200 search results

Search String:

Question Answering natural language processing
Period: Last 5 years including:
(machine learning OR algorithms OR artificial intelligence OR engineering OR classification OR linguistics OR learning algorithms OR decision making OR language OR natural language processing) NOT 20th century
Total results: 3,842
Including first 500 based on relevancy

Search String:

Question Answering text
Period: Last 2 years
excluding:
NOT (covid-19 AND questionnaires AND coronaviruses AND pandemics AND behavior AND systematic review AND climate change AND sustainability AND medical personnel AND families & family life AND severe acute respiratory syndrome coronavirus 2)
Total results: 8,650
Including first 500 based on relevancy

Search String:

Natural Language Processing Classification
Period: Last 2 years
including:
(machine learning OR algorithms OR neural networks OR deep learning OR classification OR datasets OR artificial intelligence OR natural language processing OR accuracy OR artificial neural networks OR semantics OR automation OR linguistics OR information processing OR natural language)
Excluding:
NOT (sensors AND coronaviruses AND climate change AND proteins AND image classification AND genomes AND gene expression AND medical imaging AND image processing AND computer vision AND digital media AND public health)
Total Results: 7,721
Including first 1300 based on relevancy

Search String:

Question classification model text
Period: last 2 years
Excluding:
NOT (coronaviruses AND pandemics AND climate change AND proteins AND genomes AND mental disorders AND cognitive ability AND clinical trials AND religion)
Total Results: 24,544
Included first 806 based on relevancy

A.3 Active Learning

Tool: ASReview
Databases used: Google Scholar, ProQuest
Export format: RIS (Zotero)
Active Learning Tool: ASReview

Stopping criterion:

Let R be the number of estimated Relevant papers among all papers collected in the Database search. After evaluating 30 randomized papers in step 2, 5 of these were found to be relevant, so R is [5/30 x 3800] = 630. This research paper stops evaluating sources after reaching 80% of my estimated number of relevant sources. 0.8*630 = 500. As a backup stopping criterion, after reaching 100 relevant sources, this research paper stops after evaluating 10 consecutive irrelevant sources. Since the set of sources turned out to be large after this step, an additional stopping critirion was set after reaching 200 sources instead. Many articles covered similar topics and thus held minimal information gain.

Inclusion & Exclusion criteria:

Source must be released in the last 5 years.
A source is relevant if it covers one of the following topics:
Machine Learning in Question Answering (QA)
Rule based classification in Question Answering (QA)
Natural Language Processing Question Classification

Important sources:

If a source covers multiple topics described in step 3c or is very high in quality (For instance it is very recently published or has a high number of citations), This source is marked.

A.4 Backward Snowballing

Step 4 starts by backward snowballing on the "Important sources" derived in step 3d. When reaching at least [0.5 x R] sources from this step, this step will be ended. (The reasoning for stopping is because after having reached such a high number of relevant papers, there is most likely a lot of information overlap within the database already. Continuing to add more sources

will only exacerbate this problem.) If among the next 5 papers AND 100 references there are less than 4 relevant papers, backwards snowballing will end. If backwards snowballing derives less than [0.3 x R] sources, the process of Forward snowballing will be used on the sources derived from Backward snowballing.

A.5 Quality Assessment

If the total number of sources is above 50, the disposable sources will be removed manually from the set of collected sources based on relevancy to this research. The reasoning for this is because if there are more than 50 sources, it signals to me that the research topic may be too broad and can be tightened. Deleting sources essentially narrows down the research topic.

B Original example data snippet

Data sample (Original)	
ID	1
Tijd	11:48:12
Datum	03.04.2021
Categorie (gedocumenteerd)	Vraag → Financiën → Individueel Keuzebudget (IKB)
Categorie (code)	A0435
Vraag onderwerp	Beeldscherm voor thuiswerken
Contactvorm	E-mail
Vraag content	Valt het aanschaffen van een nieuw beeldscherm voor mijn thuiswerkplek onder het IKB van telewerkruimte? Kan ik hiervoor een rekening insturen? Ik hoor het graag van jullie. [HANDTEKENING]
Antwoord onderwerp	Beeldscherm voor thuiswerken
Antwoord content	Geachte [PERSOON], U heeft op 3 april 2021 contact opgenomen met P-Direkt. U vraagt of u de aanschaf van een beeldscherm voor uw computer ingediend kan worden bij uw IKB-aanvraag. Mag ik mijn monitor indienen bij mijn IKB-aanvraag? U kunt eenmaal per vijf kalenderjaren maximaal 1.815 euro van uw IKB-budget inzetten voor het inrichten van een telewerkruimte. U kunt dit bedrag gebruiken voor de meubilering en stoffering van de werkruimte, maar niet voor de aanschaf van een computer of randapparatuur. Heeft u nog vragen over deze e-mail?. Reageert u dan op deze e-mail via 'Reply' of 'Beantwoorden'. Hierdoor behoudt u het registratienummer 123 en kunnen wij u sneller helpen. Heeft u andere vragen? Neem dan contact op met het contactcenter. Wij zijn op werkdagen bereikbaar van 8.00 uur tot 22.00 uur. [HANDTEKENING]

C Translated example data snippet

Data sample (Translated)	
ID	1
Time	11:48:12
Date	03.04.2021
Class (documented)	Question → Finances → IKB*
Class (code)	A0435**
Question subject	Monitor to work from home
Contact medium	E-mail
Question content	Is the consumption of a new computer monitor for my workplace at home included in the IKB for teleworking space? Can I submit a bill for this? I would like to hear from you. [SIGNATURE]
Answer subject	Monitor to work from home
Answer content	Dear [PERSON], On the 3rd of April, 2021, you have contacted P-direkt. You wonder whether the consumption of a computer monitor can be included in the IKB-submittal. Can I include my monitor in my IKB-submittal? You are allowed to use up to 1.815 euro of your IKB-budget every 5 calender-years for the arrangement of a teleworking space. You can use this amount for furnishing and upholstery of the working space, but not for the purchase of a computer or peripherals. Have you got any questions regarding this e-mail?. You can respond to this e-mail by "Reply". This way you retain the registration number 123 and we can help you faster. Do you have any other questions? Contact the contactcenter. You can reach us from 08:00 AM to 22:00 PM on weekdays. [SIGNATURE]

D Code

D.1 Functions

Stopword Removal

```
def lemmatizer(dataText):
    dataTextLemmatized = []
    for s in dataText:
        sentence = str(s)
        doc = nlp(sentence)
        finalSentence = ' '.join([word.lemma_ for word in doc])
        dataTextLemmatized.append(finalSentence)
    return dataTextLemmatized
```

Lemmatizer

```
def remove_stop_words(sentenceText):
    dataTextClean = []
    for sentence in sentenceText:
        stop_words  = nltk.corpus.stopwords.words('dutch')
        word_list=sentence.split()
        clean_sentence=' '.join([w for w in word_list if w.lower() not in stop_words])
        dataTextClean.append(clean_sentence)
    return dataTextClean
```

D.2 Text Pre-processing

Raw text formatting

```
# Fetch enron dataset (pre-cleaned) and store in dataframe
df = pd.read_csv('enron_data.csv')
mails_2 = df['content']

# Initialize empty lists for storing the filtered data
mails = []
mailsCompleet = []

# Find mails with question mark (question detection) and filter out response e-mails
for index, mail in enumerate(mails_2[]):

    # Split mail into words
    q1 = str(mail).split(" ")
    if "Forwarded" in  q1: continue
    if "re:" in q1: continue
    if  "RE" in q1: continue
    if "Re" in q1: continue
    startIndex = 0
    endIndex = 0

    # Check for "?" symbol in each mail (stored as list of words)
    for index2, i in enumerate(q1):

        # If question mark is found, append mail to list
        if "?" in i:
            endIndex = index2
            mails.append(" ".join(q1[startIndex:endIndex+1]))
            mailsCompleet.append(" ".join(q1))
            break

        # If no question mark is found, go to next mail and ignore the mail
        else:
            endIndex = 0
            continue
```

Tokenizing SpaCy & NLTK

```
# Tokenizing the mails into sentences
for i, s in enumerate(mails):

    # Tokenizing with SpaCy V, needs to be unpacked (object.text)
    #doc = nlp(s)
    #mails[i] = doc.sents

    # Tokenizing with NLTK
    mails[i] = nltk.sent_tokenize(s, language = "dutch")
    if type(mails[i]) is not list:
        mails[i] = [mails[i]]

# Filter out mails with less than 3 sentences.
mailsClean = [i for i in mails if len(i) > 2]
```

Data set filtering

```
# Hier halen we alles weg behalve de zin met het vraagteken erin.
vraagzin = []
xxvraagzin = []

for u, s in enumerate(mailsClean):

    # zin met vraag eruit filteren
    r = filter(lambda x: "?" in x, s)

    # troep eruit filteren
    r = filter(lambda x: "/" not in x, r)
    r = filter(lambda x: "@" not in x, r)
    r = filter(lambda x: "--" not in x, r)
    r = filter(lambda x: "[" not in x, r)
    r = filter(lambda x: "]" not in x, r)
    r = filter(lambda x: "~" not in x, r)
    r = filter(lambda x: "=" not in x, r)
    r = filter(lambda x: "_" not in x, r)
    r = filter(lambda x: ":" not in x, r)
    r = filter(lambda x: ">" not in x, r)
    r = filter(lambda x: "<" not in x, r)

    temp1 = list(r)

    # VERWIJDEREN VAN MAILS MET MEER DAN 2 VRAAGZINNEN & MAILS ZONDER VRAAGZIN
    if 2 > len(temp1) > 0:

        # VERWIJDEREN VAN LEGE ZINNEN
        if 120 >len(str(temp1)) > 15:
            vraagzin.append(temp1)

    # VERWIJDEREN VAN DUBBELE ZINNEN
    if len(vraagzin) > 2:
        if (vraagzin[len(vraagzin)-1] == vraagzin[len(vraagzin)-2]):
            vraagzin.pop()
```

```
    # zin met vraag eruit filteren, plus de twee zinnen ervoor (MUST HAVE)
    for i, v in enumerate(s):
        boole = False
        if "?" in v:
            if "/" in v: break
            if "@" in v: break
            if "--" in v : break
            if "]" in v : break
            if "[" in v : break
            if "~" in v : break
            if "=" in v : break
            if "_" in v : break
            if ":" in v : break
            if ">" in v : break
            if "<" in v : break

            # Hele korte mails (minder dan 15 leestekens) eruit filteren
            if len(v) > 15 and i > 2:
                boole = True
                tripleSent = []

                # Filter alle 3 de zinnen op ongewenste leestekens
                for j in reversed(range(3)):
                    if any(["/" in s[i-j], "@" in s[i-j], "--" in s[i-j], "]" in s[i-j], "[" in s[i-j], "~" in s[i-j],
                            "=" in s[i-j], "_" in s[i-j], ":" in s[i-j], ">" in s[i-j], "<" in s[i-j]]):
                        boole = False
                        break
                    tripleSent.append(s[i-j])
            else:
                boole = False
                break

        if boole == True:
            xxvraagzin.append(tripleSent)

# DF met alleen vraagzin
vraagzindf = pd.DataFrame(vraagzin, columns = ["vragen"])
#vraagzindf['mail'] = mails

# DF met vraagzin en 2 zinnen ervoor
xxvraagzindf = pd.DataFrame(xxvraagzin, columns = ["zin1", "zin2", "zin3"])
xxvraagzindf['text'] = xxvraagzindf['zin1'] + ' ' + xxvraagzindf['zin2'] + ' ' + xxvraagzindf['zin3']

# DF met de gehele vraagmail
vraagmaildf = pd.DataFrame(mailsCompleet, columns = ["mail"])
```

Duplicate removal

```
res = []
tempResult = [res.append(x) for x in mailsClean if x not in res]
mailsClean = list(res)
```

```
#VERWIJDEREN VAN DUBBELE ZINNEN
if len(xxvraagzin) > 2:
    if (xxvraagzin[len(xxvraagzin)-1] == xxvraagzin[len(xxvraagzin)-2]):
        xxvraagzin.pop()
```

D.3 Models

Naïve Bayes

```
X_train, X_test, y_train, y_test = train_test_split(labelSet['vragen'],
                                                    labelSet['labels'],
                                                    test_size = 0.25)

y_train=y_train.astype('int')
y_test=y_test.astype('int')

# BUILDING THE MODEL
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
X_train_counts.shape

# Get the TF-IDF vector representation of the train data
from sklearn.feature_extraction.text import TfidfTransformer
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

# Weight function for fitting process
def weightDefiner(label_term):
    result = 1
    if label_term == 0: result = 5
    if label_term == 1: result = 5
    if label_term == 2: result = 5
    if label_term == 3: result = 5
    if label_term == 4: result = 5
    return result

weights = [weightDefiner(x) for x in y_train]

# Build Multinomial Naïve Base Classifier
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB().fit(X_train_tfidf, y_train, sample_weight = weights)


# Get the TF-IDF vector representation of the test data
X_new_counts = count_vect.transform(X_test)
X_new_tfidf = tfidf_transformer.transform(X_new_counts)

# Predict on test data
predicted = clf.predict(X_new_tfidf)
```

Logistic Regression

```
from sklearn.model_selection import StratifiedShuffleSplit
X_train, X_test, y_train, y_test = train_test_split(labelSet_1['vragen'],
                                                    labelSet_1['labels'],
                                                    test_size = 0.25,
                                                    stratify = labelSet_1['labels']
                                                   )


# BUILDING THE MODEL
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
X_train_counts.shape

# Get the TF-IDF vector representation of the train data
from sklearn.feature_extraction.text import TfidfTransformer
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

# Build Logistic Regression Classifier
from sklearn.linear_model import LogisticRegressionCV, LinearRegression
clf = LogisticRegressionCV(cv = 5, random_state=0, max_iter=1000).fit(X_train_tfidf, y_train)

# Get the TF-IDF vector representation of the test data
X_new_counts = count_vect.transform(X_test)
X_new_tfidf = tfidf_transformer.transform(X_new_counts)

# Predict on test data
predicted = clf.predict(X_new_tfidf)
```

Support Vector Machines

```
X_train, X_test, y_train, y_test = train_test_split(labelSet['vragen'],
                                                    labelSet['labels'],
                                                    test_size = 0.25)

y_train=y_train.astype('int')
y_test=y_test.astype('int')

# BUILDING THE MODEL
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
X_train_counts.shape

# Get the TF-IDF vector representation of the train data
from sklearn.feature_extraction.text import TfidfTransformer
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

# Weight function for fitting process
def weightDefiner(label_term):
    result = 1
    if label_term == 0: result = 5
    if label_term == 1: result = 5
    if label_term == 2: result = 5
    if label_term == 3: result = 5
    if label_term == 4: result = 5
    return result

weights = [weightDefiner(x) for x in y_train]

# Build Multinomial Naïve Base Classifier
from sklearn.naive_bayes import MultinomialNB
clf = SVC().fit(X_train_tfidf, y_train, sample_weight = weights)


# Get the TF-IDF vector representation of the test data
X_new_counts = count_vect.transform(X_test)
X_new_tfidf = tfidf_transformer.transform(X_new_counts)

# Predict on test data
predicted = clf.predict(X_new_tfidf)
```

RIPPER Algorithm

```
# Vectorize the (text) data set
count_vect = CountVectorizer(min_df=1)
X_train_counts = count_vect.fit_transform(labelSet_1['vragen'].tolist())

# Create bag of words representation & save into data frame
df_BoW = pd.DataFrame(X_train_counts.toarray(),
                      columns=count_vect.get_feature_names_out())
result = pd.concat([labelSet_1['labels'], df_BoW], axis=1)

# Create train and test sets
X_train, X_test, y_train, y_test = train_test_split(result.drop(columns=['labels']),
                                                    result['labels'],
                                                    test_size = 0.25)

# Create & fit Rule-based model on the training data
ripper_clf = lw.RIPPER() # Or lw.IREP()
ripper_clf.fit(X_train, y_train)

# Display generated ruleset
ripper_clf.ruleset_
```

Gensim

```
#Installeren GENSIM - preprocessen text
import gensim
from gensim import corpora, models, similarities
from gensim.models import Word2Vec
from gensim.models import KeyedVectors

# Build a Gensim model, based on the entire Enron corpus.
import re
x = EnronData
x = lemmatizer(x) # Only use this lemmatizer for smaller data sets (<1000)

# Pre-process text using regular expressions
import re
for i, sen in enumerate(x):
    #remove extra characters
    sen = re.sub(r"[[0-9]*\]", " ", sen)
    #remove the extra spaces between words
    sen = re.sub(r"\s+", " ", sen)
    #convert all letters to lowercase
    sen = sen.lower()
    x[i] = sen

#GENSIM WORD2VEC MODEL BOUWEN (VOOR DATASET VAN HIERBOVEN)
tok_corp = [nltk.word_tokenize(sent) for sent in x]
model_gensim = gensim.models.Word2Vec(tok_corp, min_count=1)

#GENSIM WORD2VEC MODEL BOUWEN (MANIER 2)
nlp_gensim = gensim.models.word2vec.Word2Vec(tok_corp, vector_size=300,
            window=8, min_count=1, sg=1, epochs=30)
```

D.4 Visualization

Confusion Matrix & Accuracy

```
score = accuracy_score(y_test, predicted)
print(score)
print(classification_report(y_test, prediction))
```

Confusion Matrix Heatmap

```
#Visualization
labels_vis = ['ambiguous','clear']
clf_report = classification_report(y_test,
                                   predicted,
                                   labels=[0,1],
                                   target_names=labels_vis,
                                   output_dict=True)
print(sns.heatmap(pd.DataFrame(clf_report).iloc[:-1, :].T, annot=True))
```

3d visualization of similarity

```
from sklearn import (feature_extraction, model_selection, naive_bayes,
pipeline, manifold, preprocessing)

# Keyword to make a centralized 3 model around
word = "fiets"
fig = plt.figure()

# Make embedding for the "most similar words" to the "Keyword"
tot_words = [word] + [tupla[0] for tupla in
                 model_gensim.wv.most_similar(word, topn=5)]
X = model_gensim.wv[tot_words]

# Pca to reduce dimensionality from 300 to 3
pca = manifold.TSNE(perplexity=5, n_components=3, init='pca')
X = pca.fit_transform(X)

# Create Data frame for visualization of the 3d scatterplot
dfv = pd.DataFrame(X, index=tot_words, columns=["x","y","z"])
dfv["input"] = 0
dfv["input"].iloc[0:1] = 1

# Make 3d plot
from mpl_toolkits.mplot3d import Axes3D
ax = fig.add_subplot(111, projection='3d')
ax.scatter(dfv[dfv["input"]==0]['x'],
           dfv[dfv["input"]==0]['y'],
           dfv[dfv["input"]==0]['z'], c="black")
ax.scatter(dfv[dfv["input"]==1]['x'],
           dfv[dfv["input"]==1]['y'],
           dfv[dfv["input"]==1]['z'], c="green")
ax.set(xlabel=None, ylabel=None, zlabel=None, xticklabels=[],
       yticklabels=[], zticklabels=[])
for label, row in dfv[["x","y","z"]].iterrows():
    x, y, z = row
    ax.text(x, y, z, s=label)
```

Sequence padding

```
# Initialize a keras tokenizer, with possibility for filtering
tokenizer = kprocessing.text.Tokenizer(lower=True, split=' ',
                                       oov_token="NaN",
                                      # filters='!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~\t\n'
                                      )

# Fit tokenizer on the train and test data
tokenizer.fit_on_texts(X_train)
tokenizer.fit_on_texts(X_test)


# Create sequence from train and test data
lst_text2seq = tokenizer.texts_to_sequences(X_train)
lst_text2seq2 = tokenizer.texts_to_sequences(X_test)

# Padding sequence with empty characters to appropriate length
X_train = kprocessing.sequence.pad_sequences(lst_text2seq,
                    maxlen=75, padding="post", truncating="post")

X_test = kprocessing.sequence.pad_sequences(lst_text2seq2, maxlen=75,
             padding="post", truncating="post")

sns.heatmap(X_train==0, vmin=0, vmax=1, cbar=False)
plt.show()
```

D.5 General library imports

```
import wittgenstein as lw
import pandas as pd
import pickle as pickle
import spacy
import nltk
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
import tqdm
import numpy as np
import gensim
from sklearn.model_selection import train_test_split
from spacy.lang.nl import Dutch
from spacy.lang.nl.examples import sentences
nlp = spacy.load("nl_core_news_sm")

# for deep learning
import tensorflow
from tensorflow.keras import models, layers, preprocessing as kprocessing
from tensorflow.keras import backend as K

import gensim
from gensim import corpora, models, similarities
from gensim.models import Word2Vec
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from gensim.models import KeyedVectors
nltk.download('stopwords')
```

```
import re

import tensorflow
from tensorflow.keras import models, layers, preprocessing as kprocessing
from tensorflow.keras import backend as K

import mpl_toolkits.mplot3d

from sklearn import (feature_extraction, model_selection, naive_bayes,
pipeline, manifold, preprocessing)
from mpl_toolkits.mplot3d import Axes3D

from sklearn.datasets import make_classification
from sklearn import svm
from sklearn.svm import SVC
import matplotlib.pyplot as plt
from gensim.corpora import Dictionary
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import MinMaxScaler #fixed import
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

E Dutch data set (P-direkt) results

Following are the extended results (Classification Reports) regarding the Dutch data set for each step.

Step 1 [Detection]

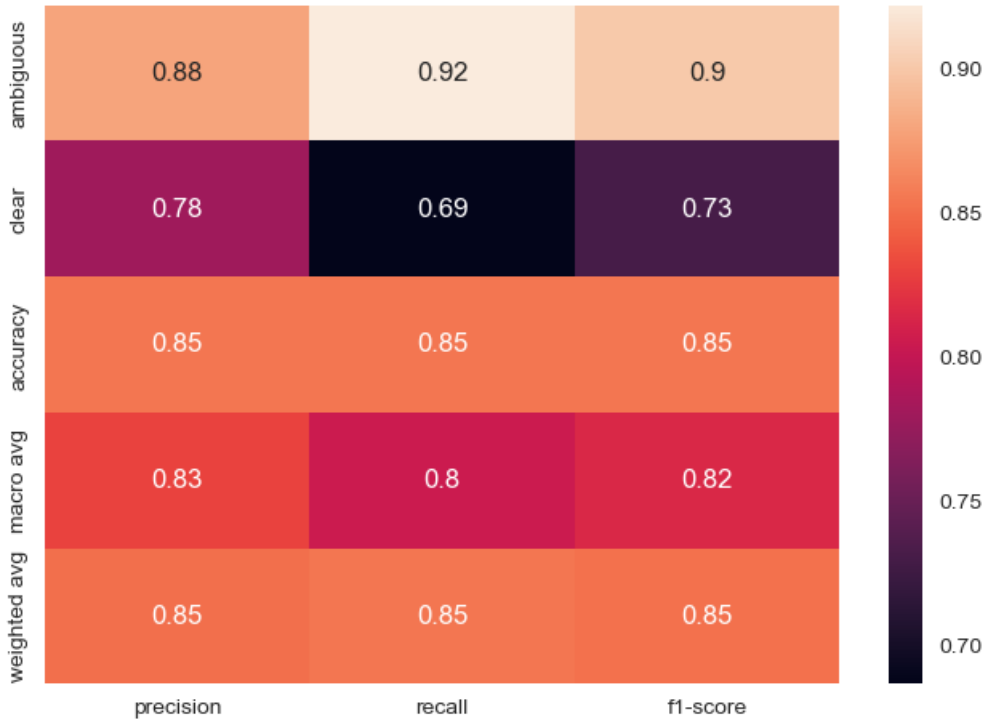

Figure 21: Classification report for Support Vector Machines classifier

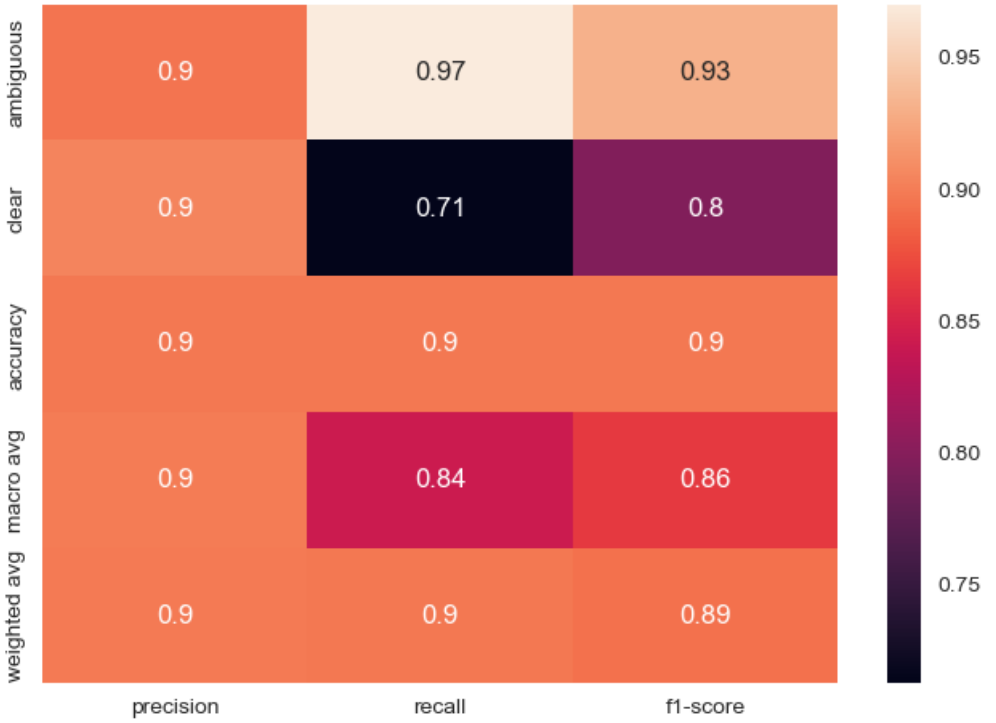

Figure 22: Classification report for fine-tuned BERT classifier

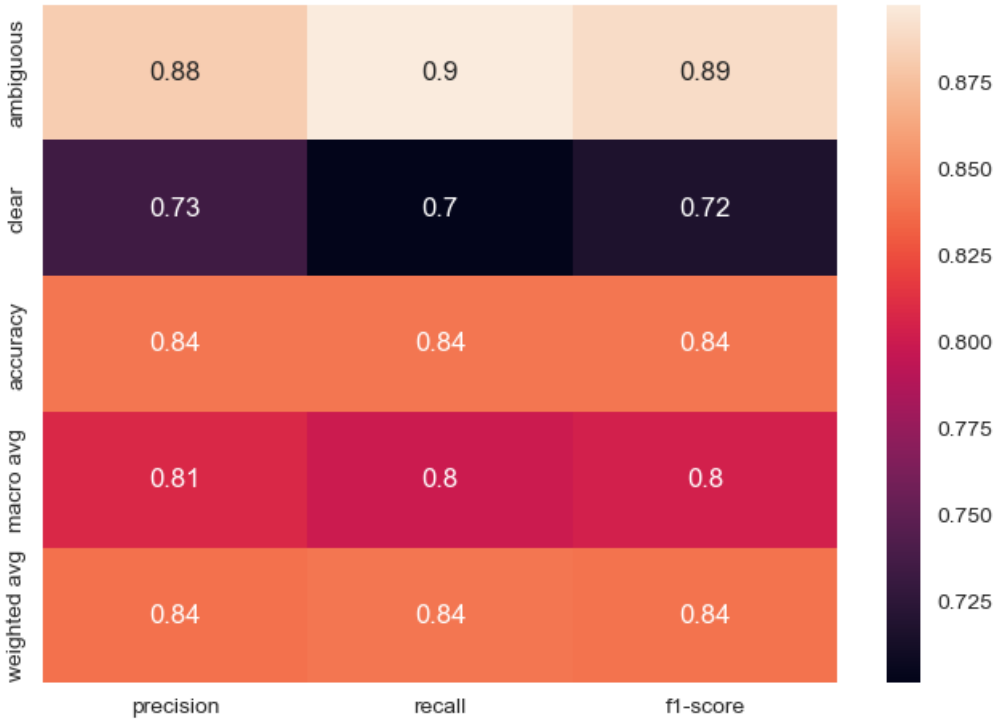

Figure 23: Classification report for Logistic Regression classifier

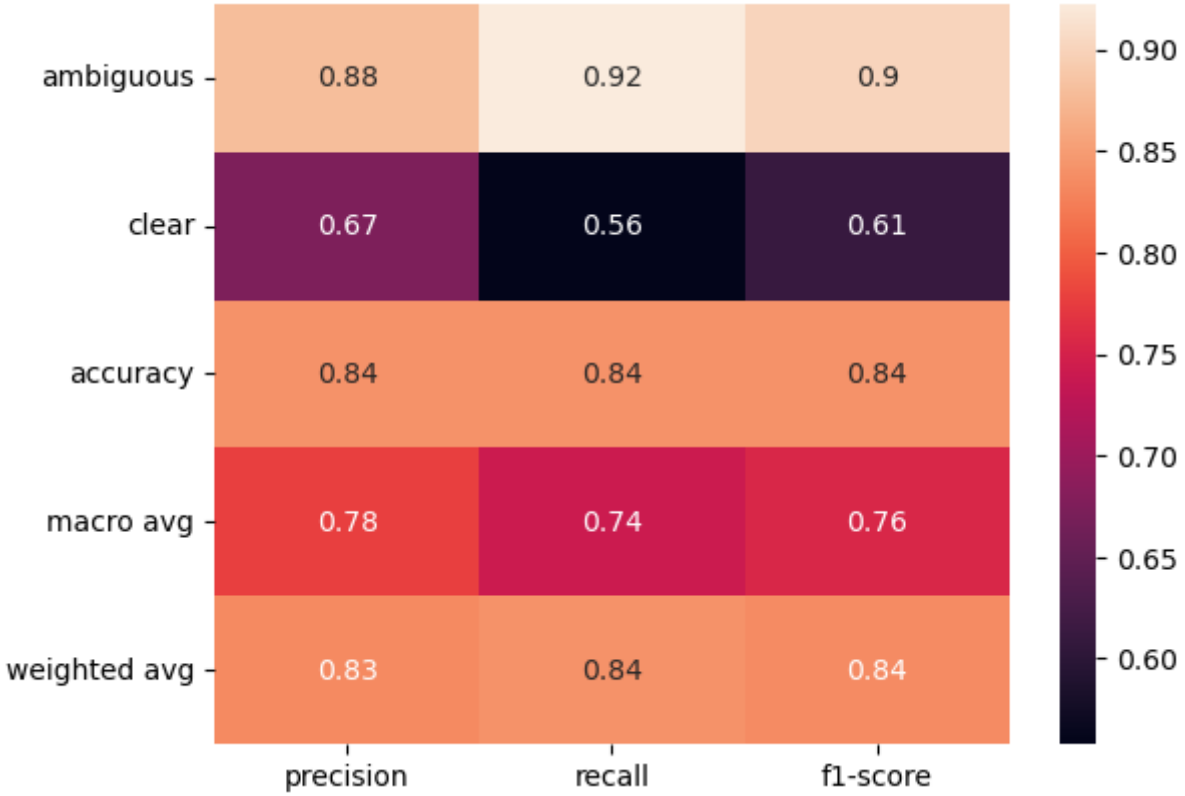

Figure 24: Classification report for Naïve Bayes classifier

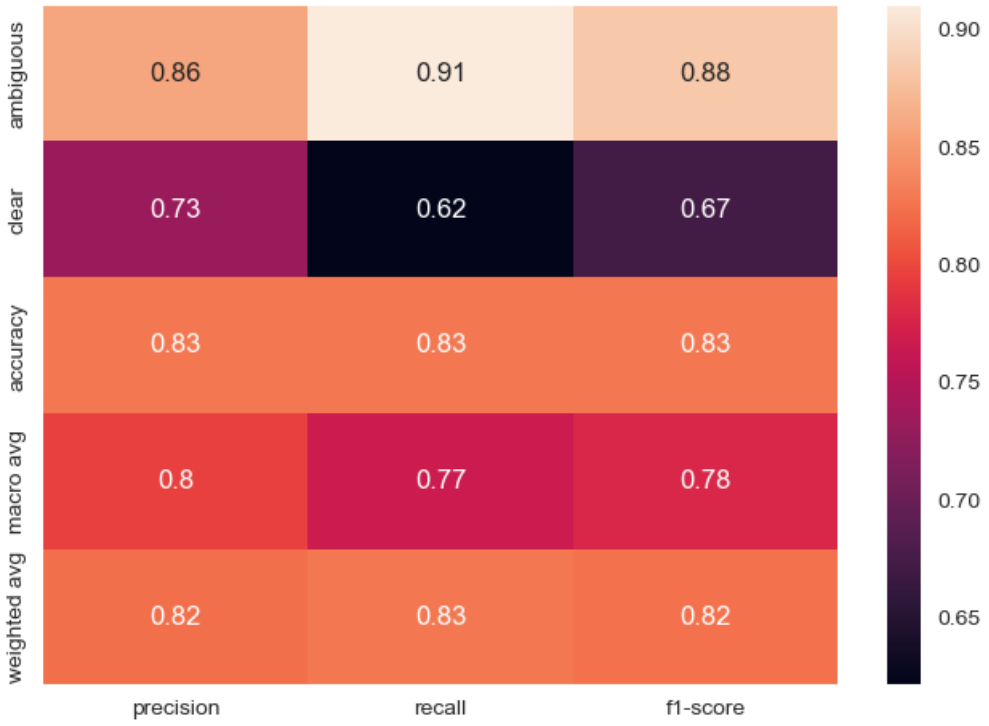

Figure 25: Classification report for Decision Tree classifier (node depth 10)

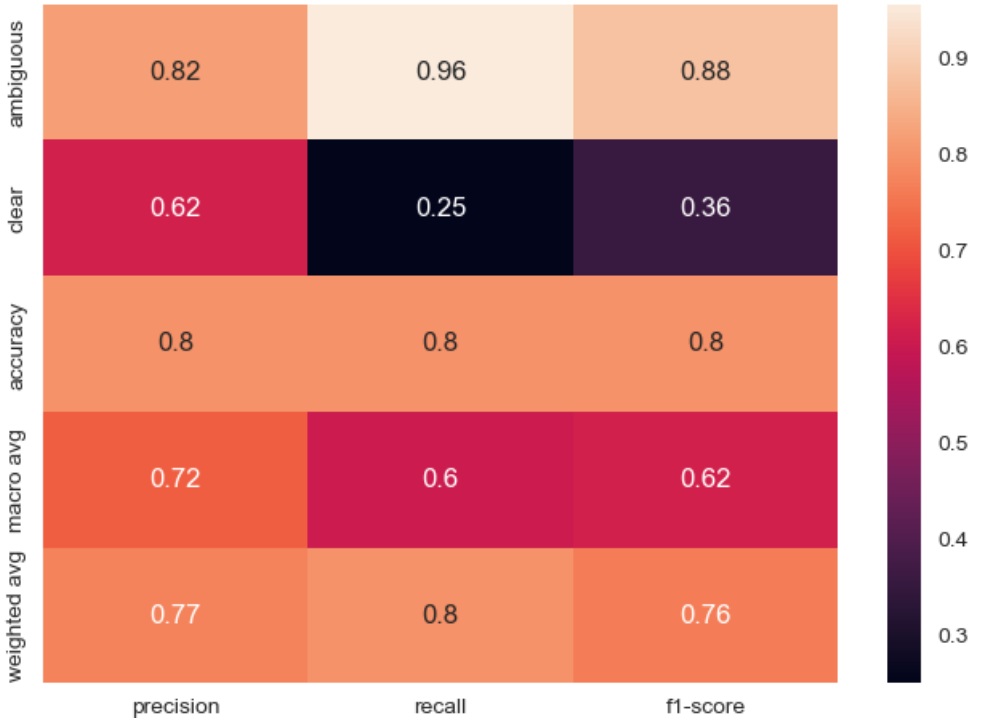

Figure 26: Classification report for RIPPER classifier

Step 2 [Enrichment with 1 prior sentence]

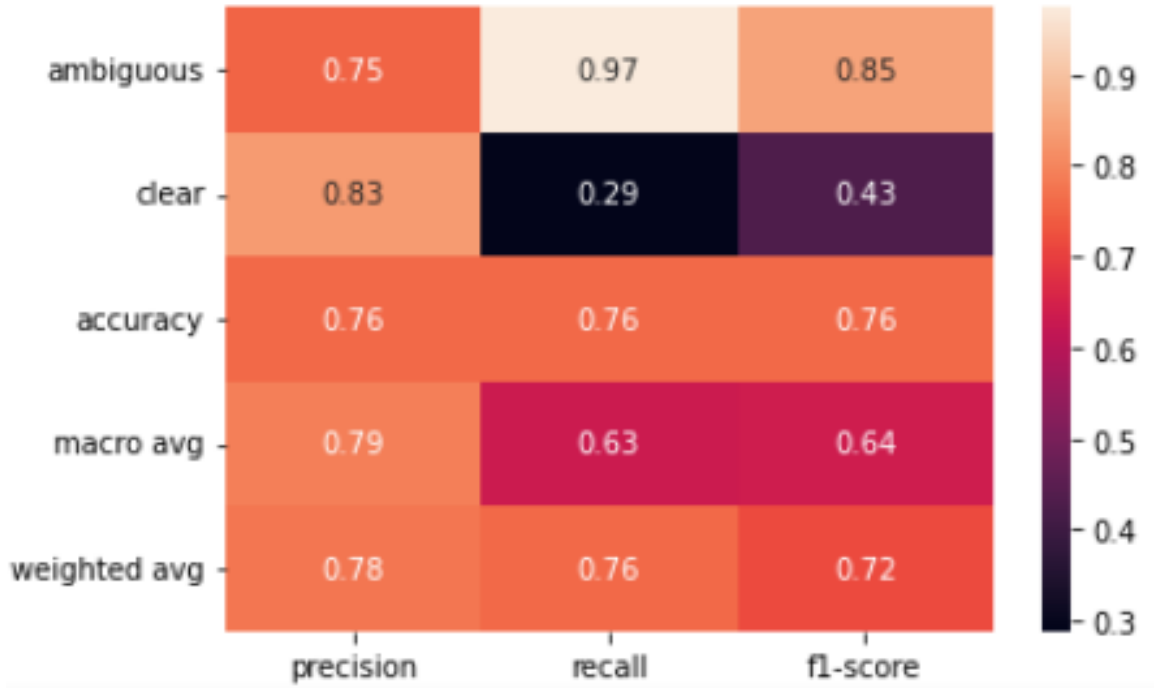

Figure 27: Classification report for Support Vector Machines classifier

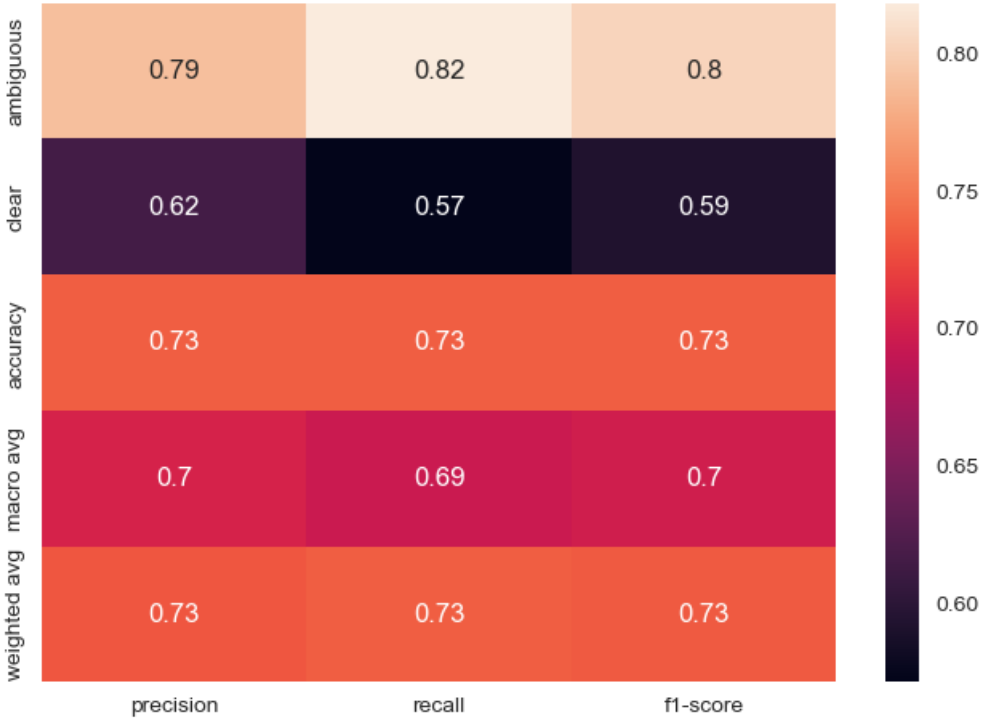

Figure 28: Classification report for fine-tuned BERT classifier

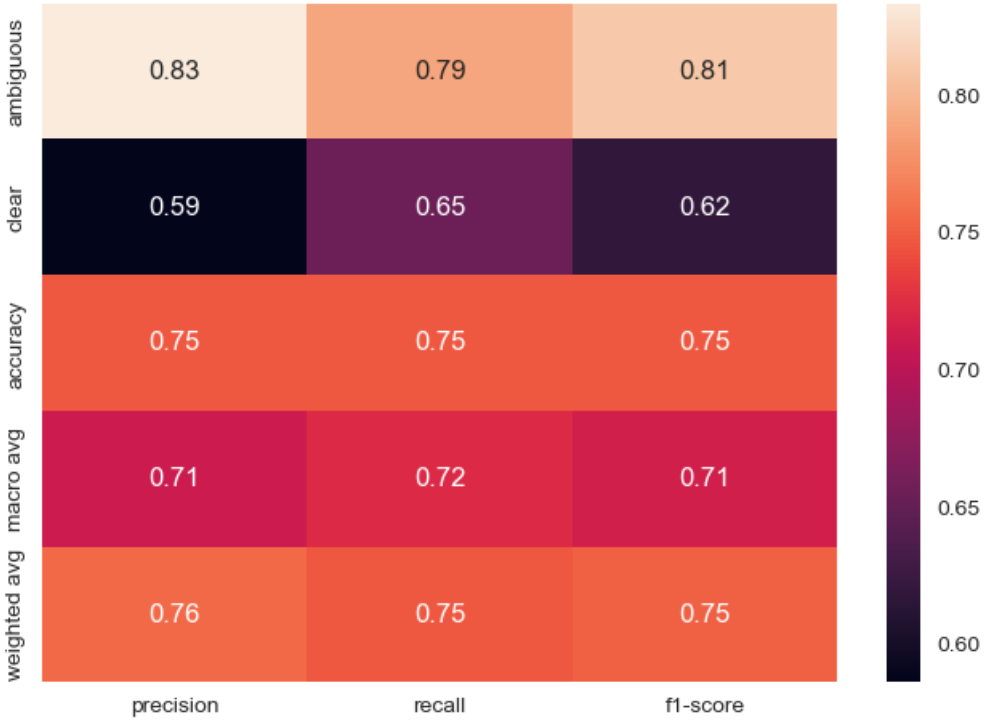

Figure 29: Classification report for Logistic Regression classifier

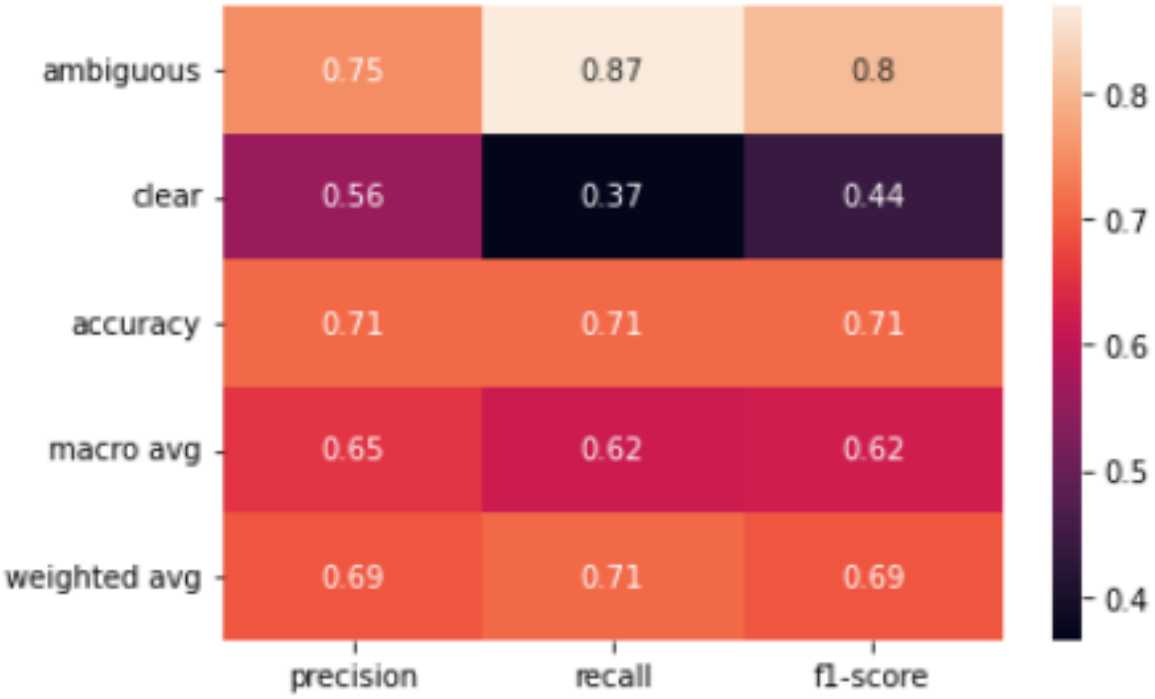

Figure 30: Classification report for Naïve Bayes classifier

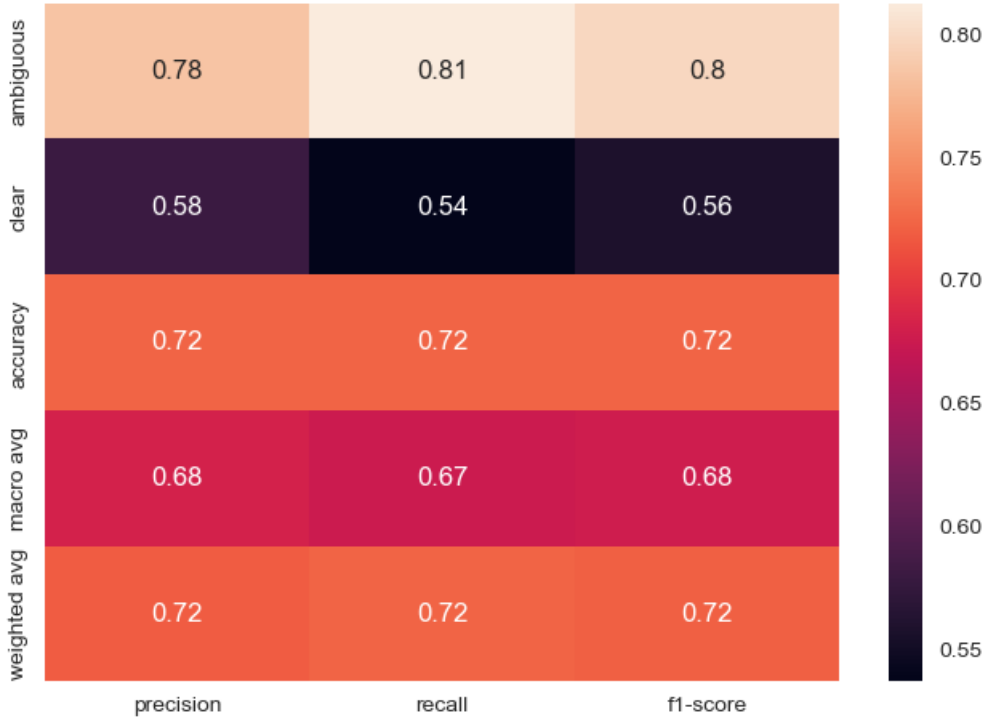

Figure 31: Classification report for Decision Tree classifier (node depth 10)

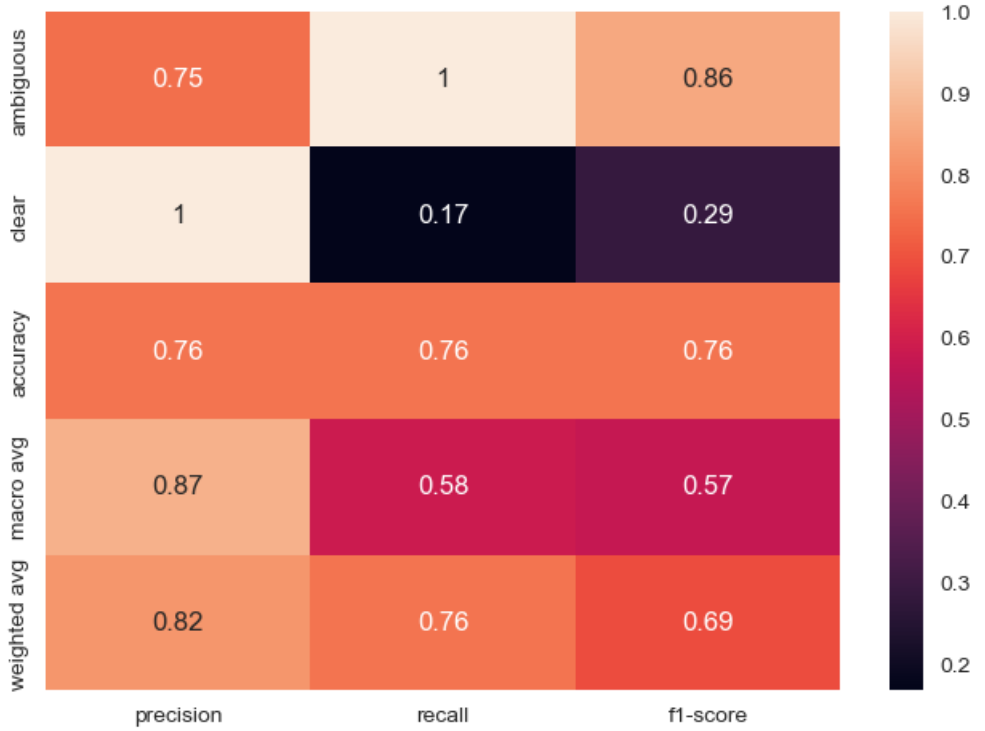

Figure 32: Classification report for RIPPER classifier

Step 2 [Enrichment with 2 prior sentences]

Figure 33: Classification report for Support Vector Machines classifier

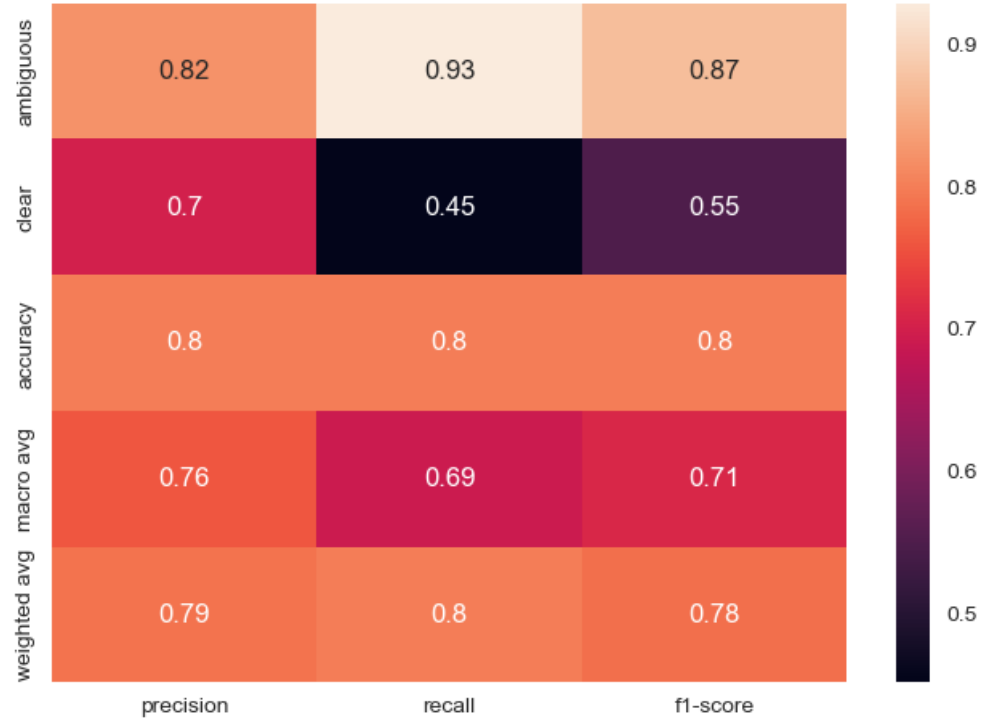

Figure 34: Classification report for fine-tuned BERT classifier

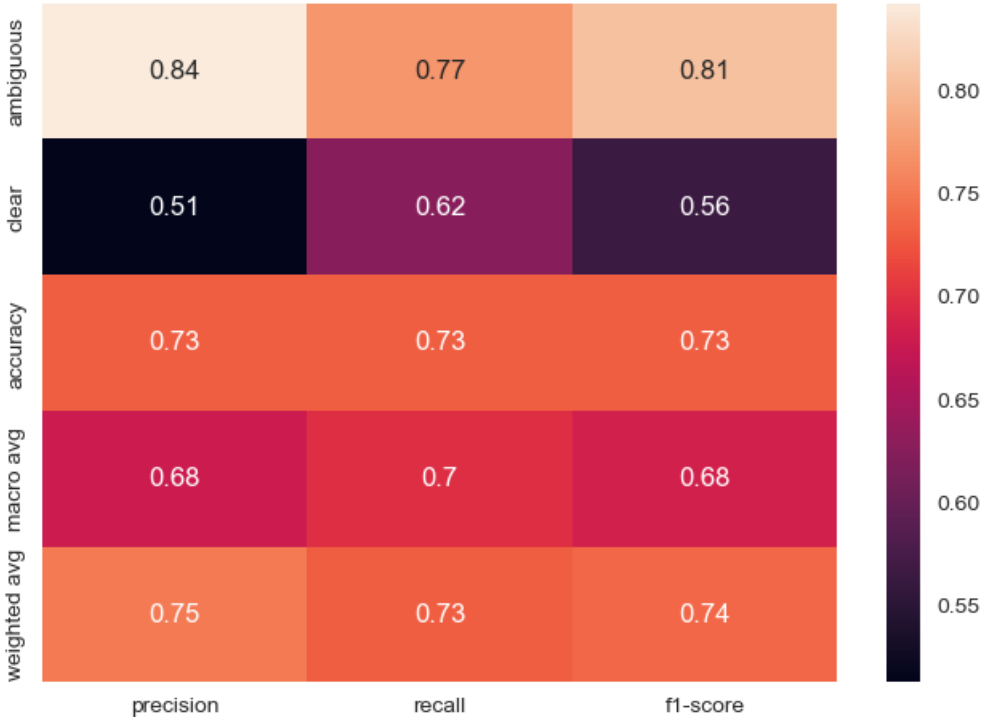

Figure 35: Classification report for Logistic Regression classifier

Figure 36: Classification report for Naïve Bayes classifier

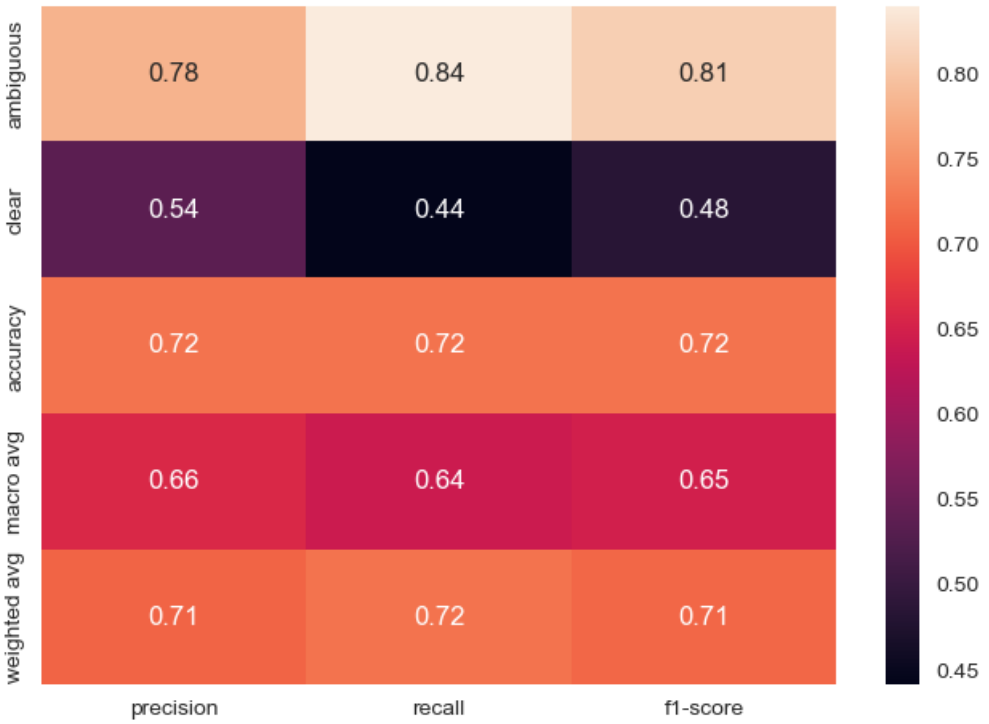

Figure 37: Classification report for Decision Tree classifier (node depth 10)

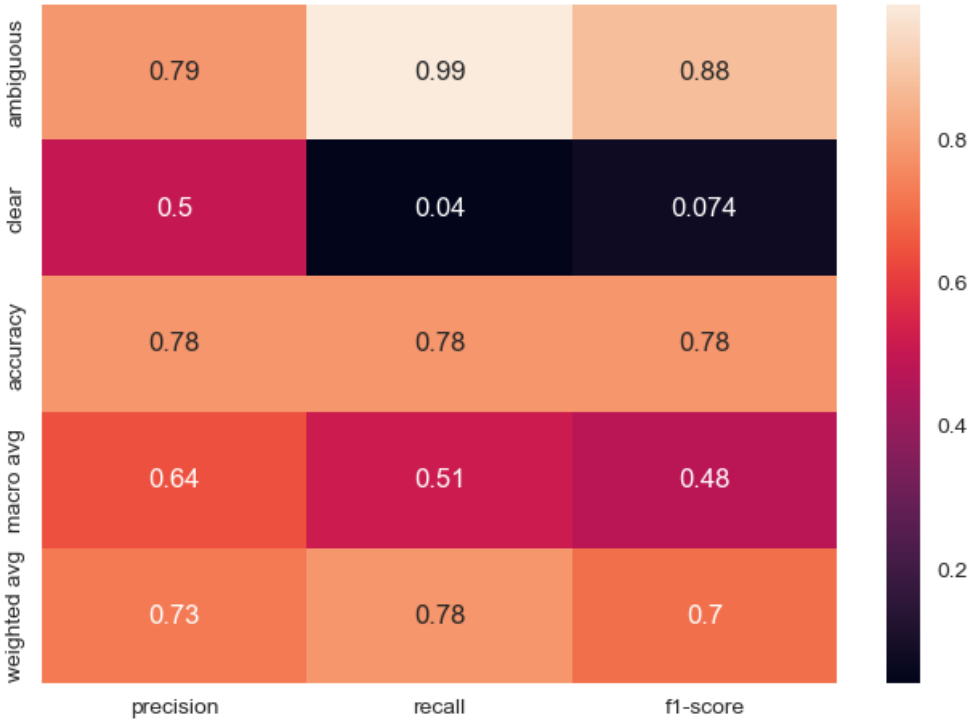

Figure 38: Classification report for RIPPER classifier

F English data set (Enron) results

Following are the extended results (Classification Reports) regarding the English data set for each step.

Step 1 [Detection]

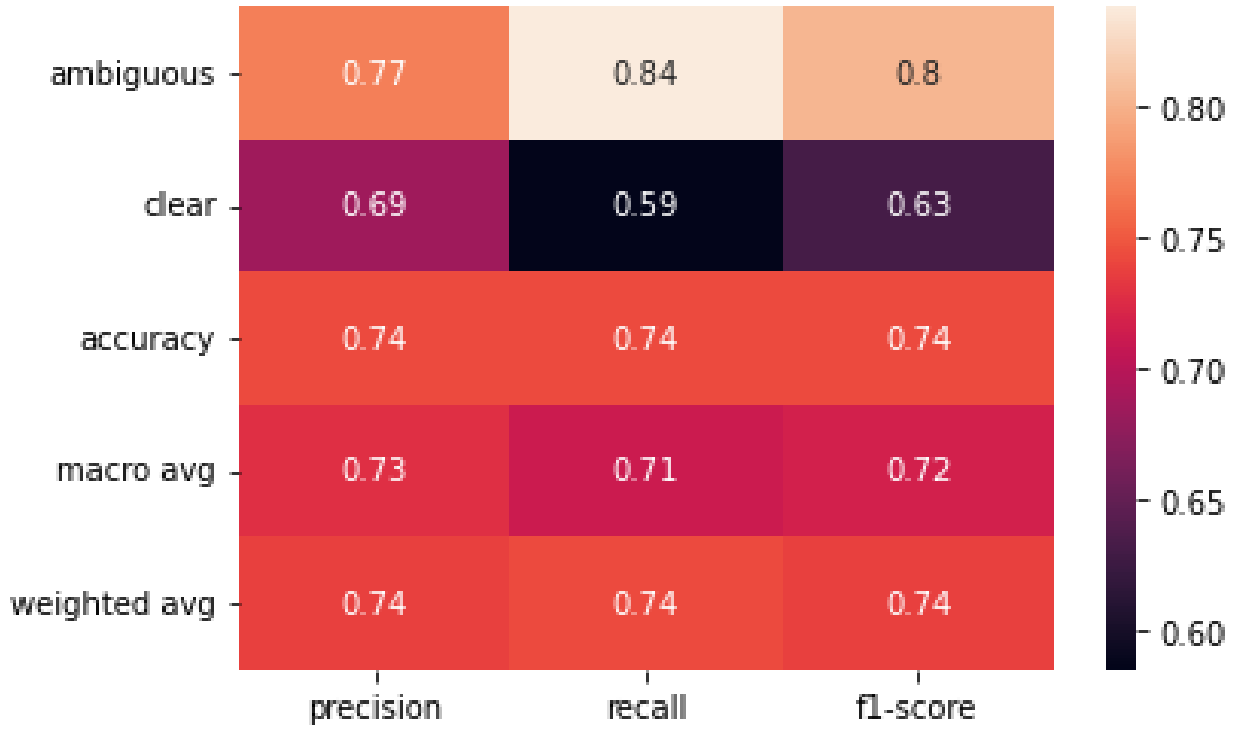

Figure 39: Classification report for Support Vector Machines classifier

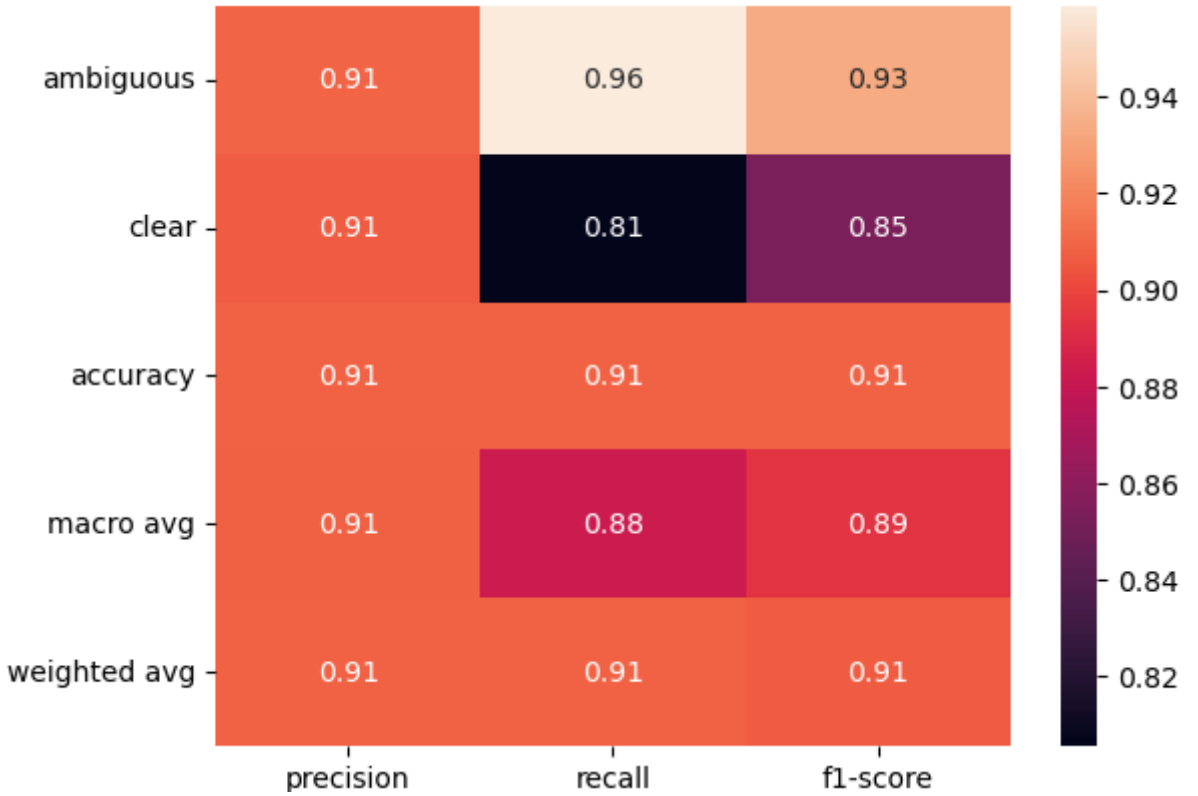

Figure 40: Classification report for fine-tuned BERT classifier

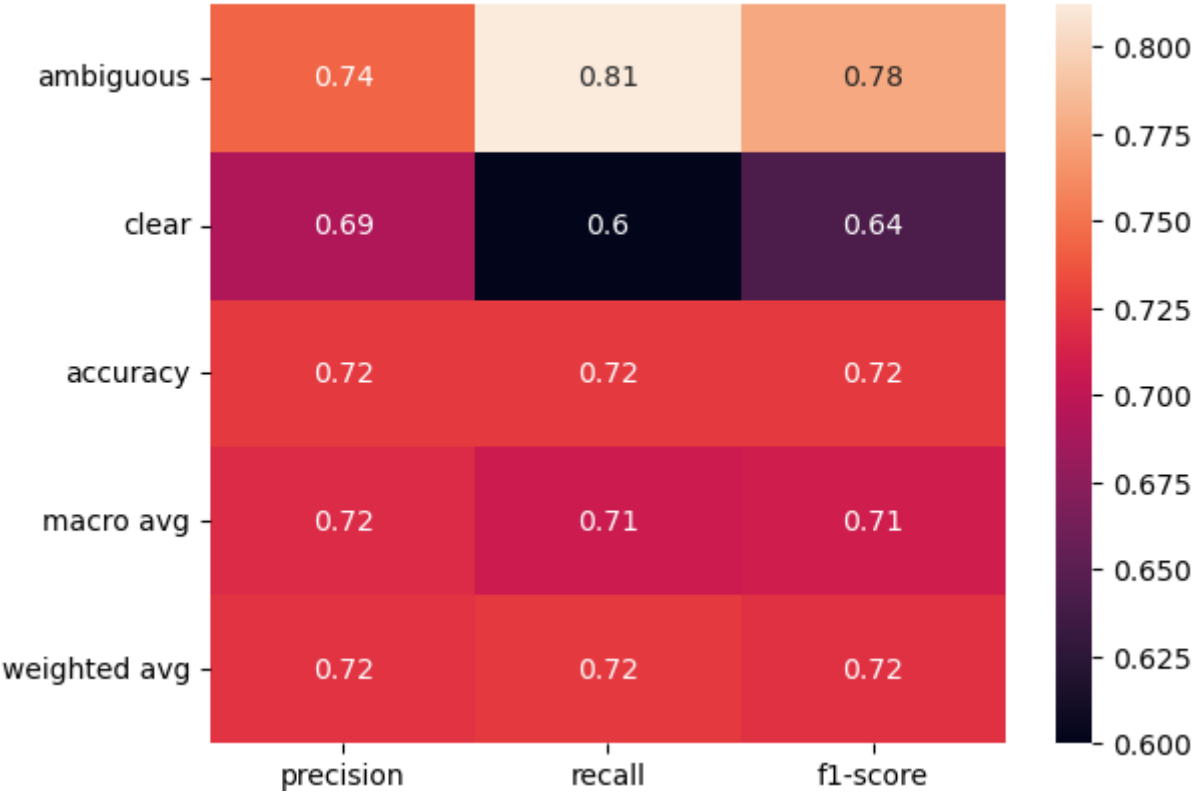

Figure 41: Classification report for Logistic Regression classifier

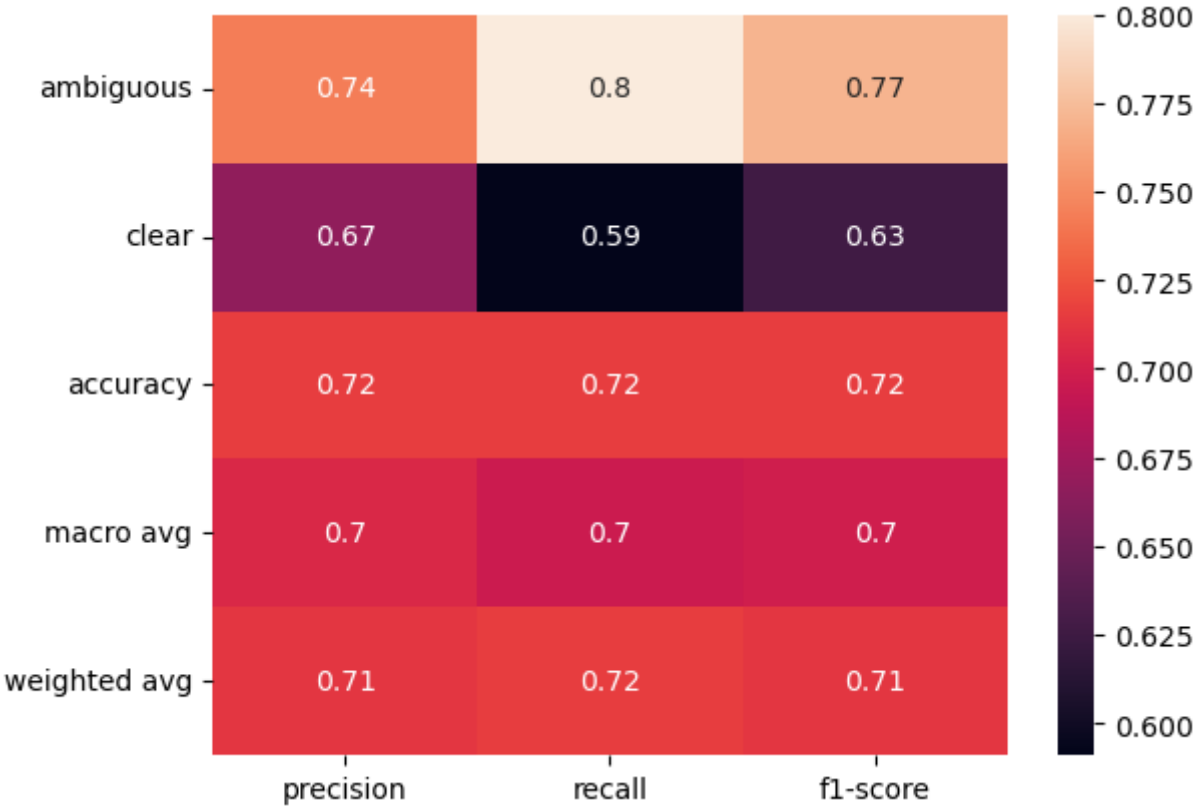

Figure 42: Classification report for Naïve Bayes classifier

Figure 43: Classification report for Decision Tree classifier (node depth 10)

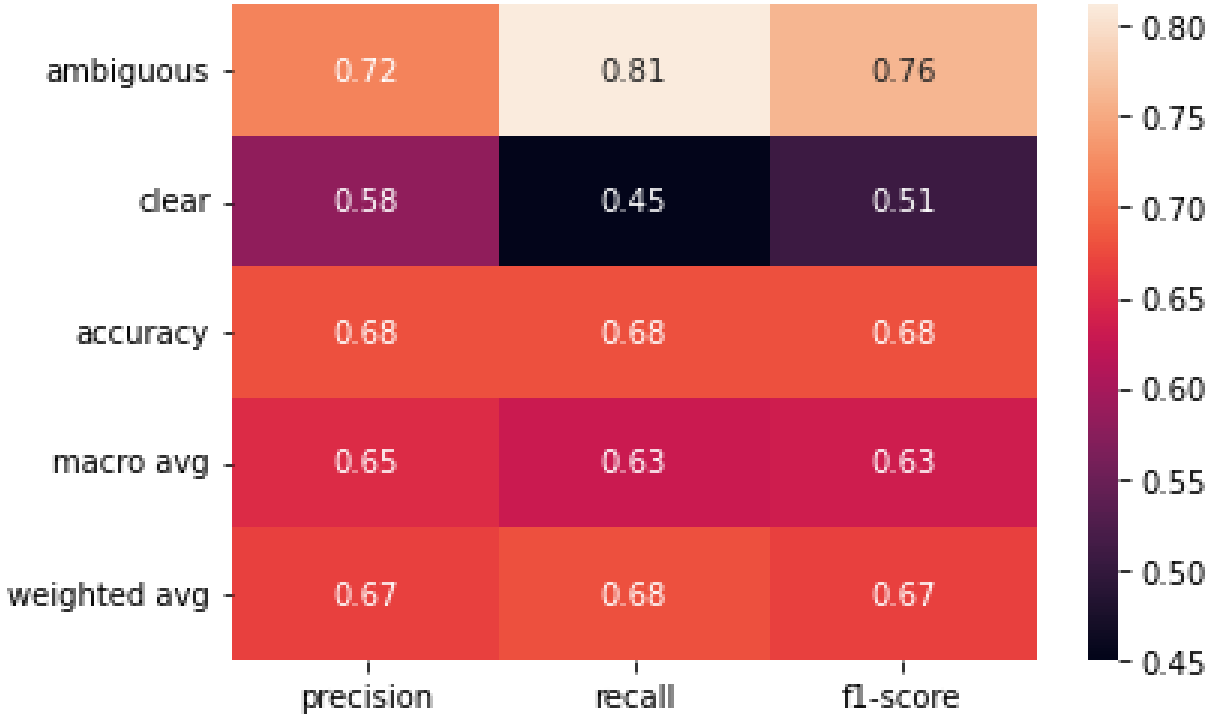

Figure 44: Classification report for RIPPER classifier

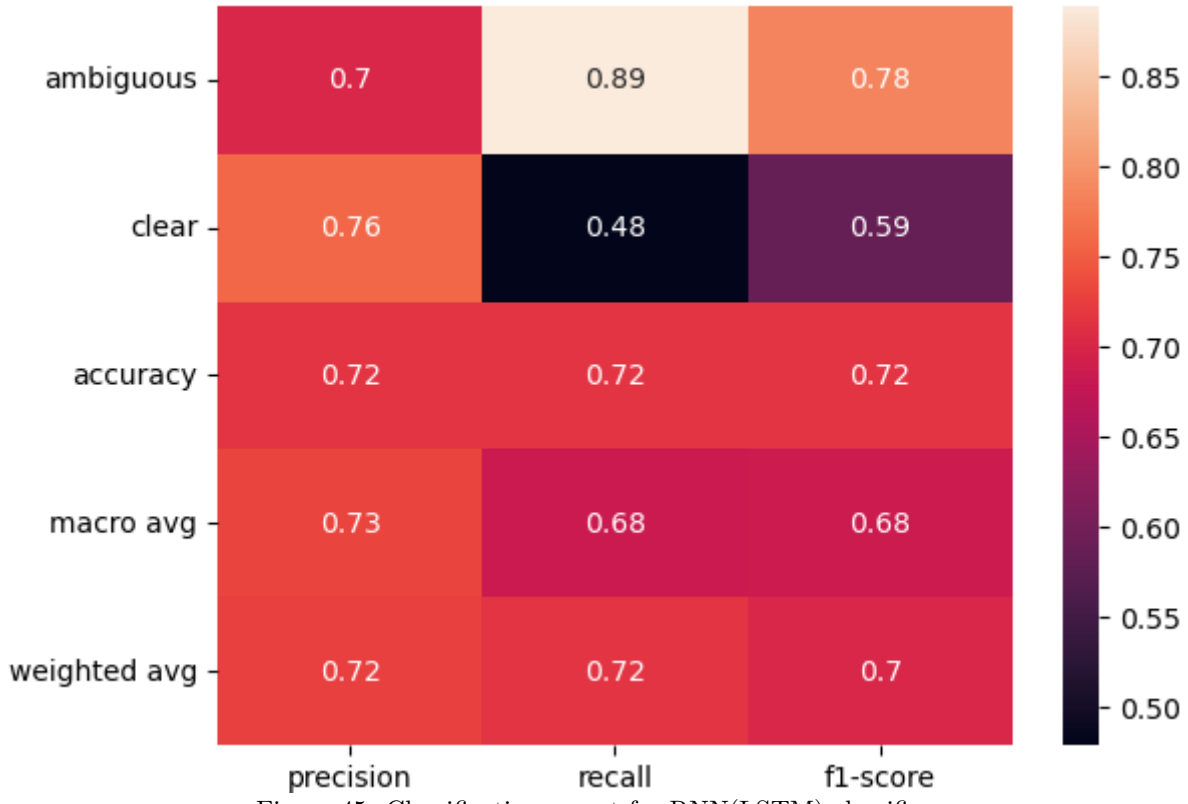

Figure 45: Classification report for RNN(LSTM) classifier

Step 2 [Enrichment/Clarification]

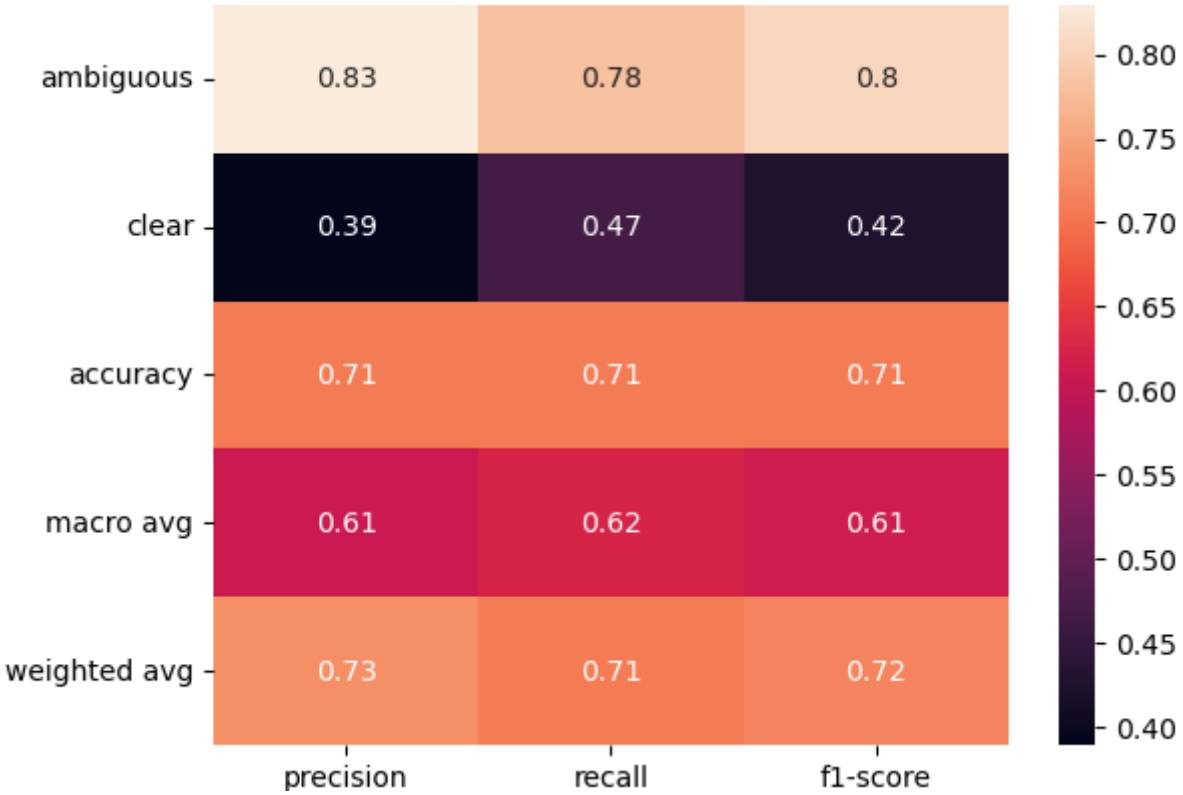

Figure 46: Classification report for Support Vector Machines classifier

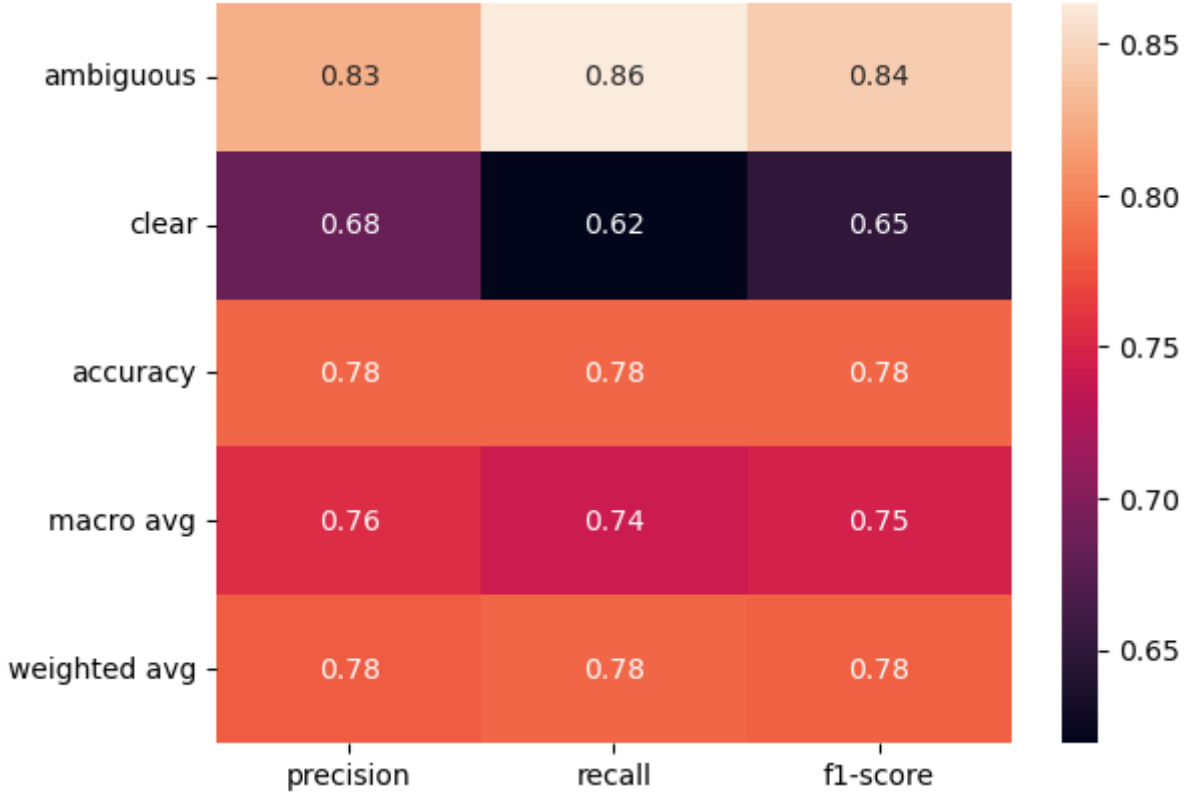

Figure 47: Classification report for fine-tuned BERT classifier

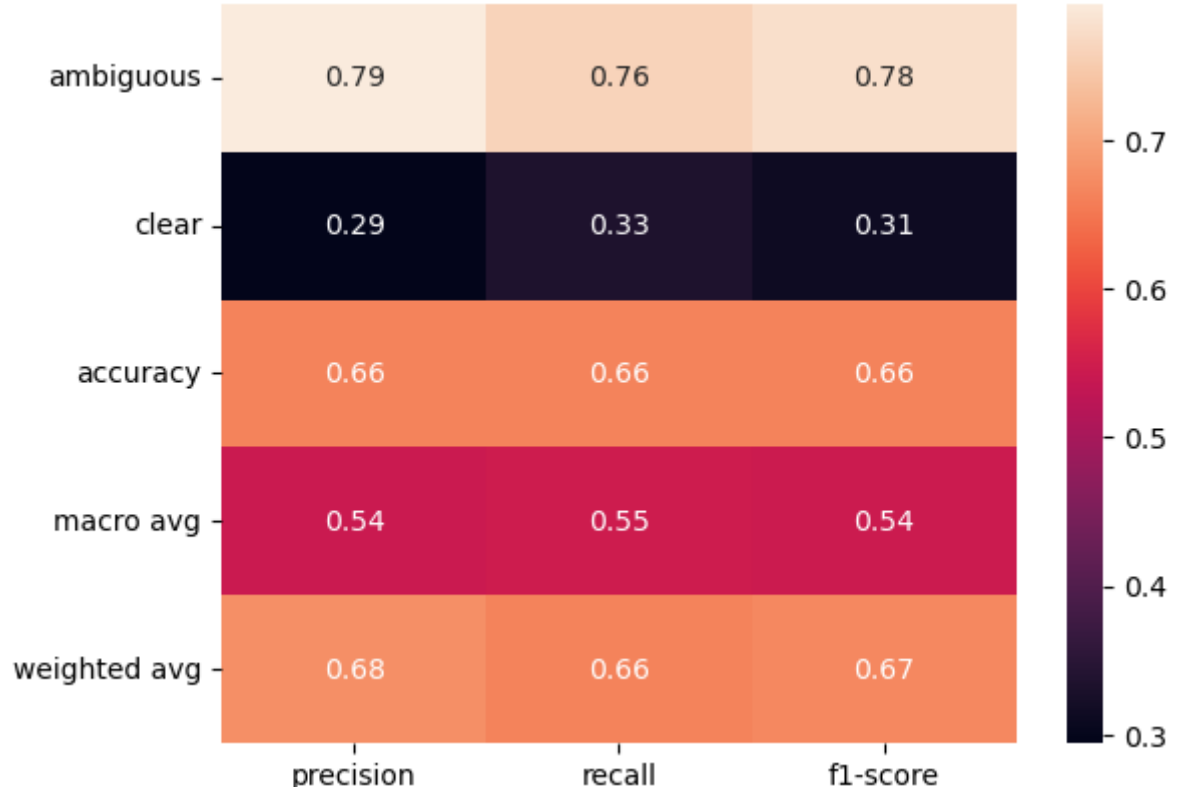

Figure 48: Classification report for Logistic Regression classifier

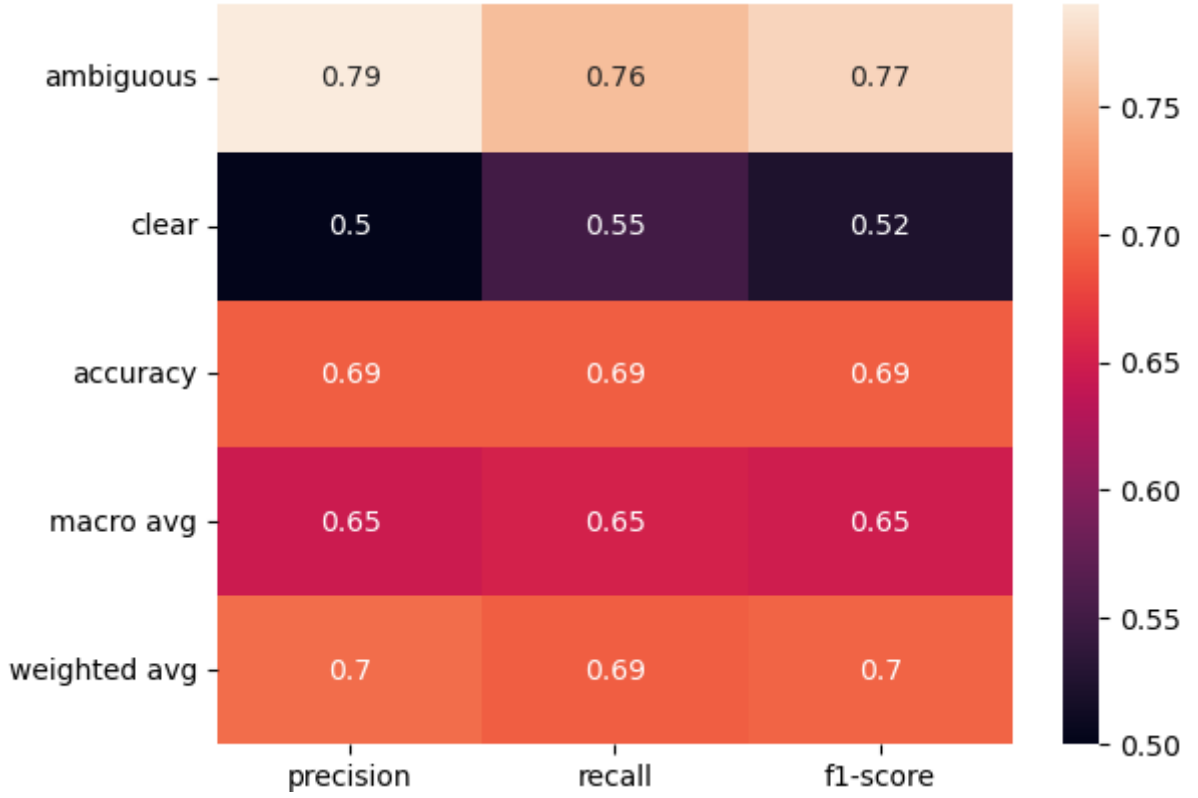

Figure 49: Classification report for Naïve Bayes classifier

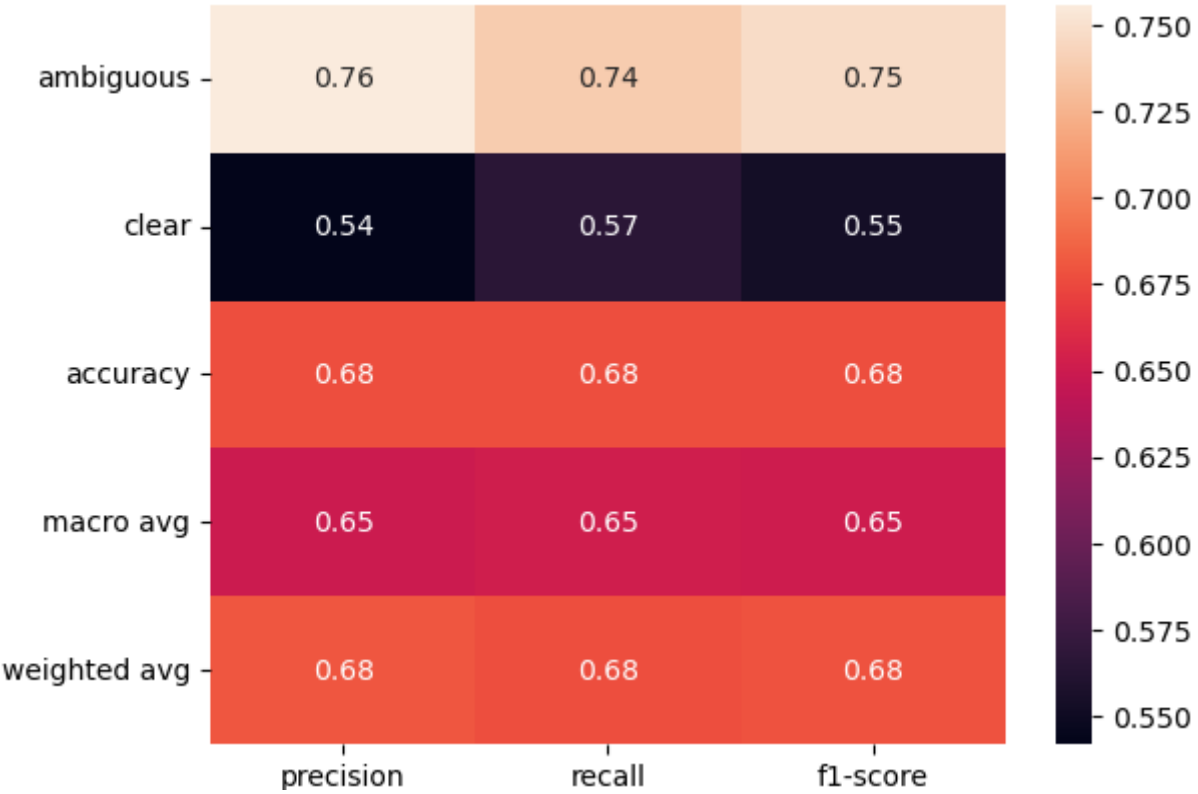

Figure 50: Classification report for Decision Tree classifier (node depth 10)

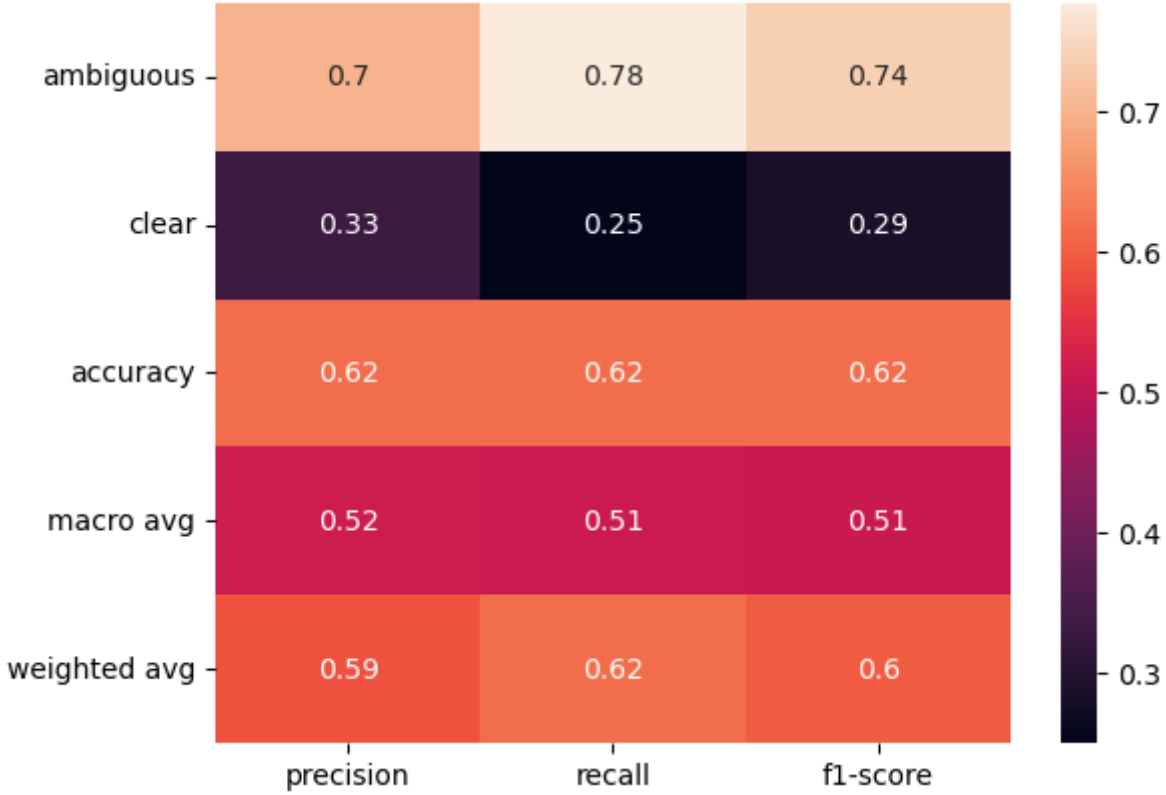

Figure 51: Classification report for RIPPER classifier