Extracting information from funding opportunities using state-of-the-art NLP method BERT with minimized data annotation

Daphne C.J.C Rietvelt

Committee:
Dr. M. Theune
Dr. S. Wang
Dr. N. Strisciuglio
Dr. M. Doornenbal

UNIVERSITY OF TWENTE.

ELSEVIER

Acknowledgments

This master thesis would never have come about without Elsevier. I want to thank Elsevier for enabling me to focus on such an interesting project relevant to literature and business processes. Thank you for all the support and excellent cooperation despite the setbacks health-wise. I especially want to thank Marius Doornenbal for his continued support and interesting insights that shaped this master thesis. Moreover, I want to thank my supervisors from the University of Twente, Shenghui Wang, and Mariët Theune, who helped me with support, interesting insights that deepened my research, and feedback. Also, I want to thank Nicola Strisciuglio for being part of my examination committee. Lastly, I wish to thank my family and friends who helped me through this difficult time. Particularly, I wish to thank my brother Mark Rietvelt, who was of great help during the writing process.

Abstract

Elsevier is an information analytics business that helps researchers advance science. One of the services Elsevier offers, is to allow researchers to find suitable funding opportunities. Information on the eligibility of potential applicants is crucial to recommend suitable funding opportunities. Currently, eligibility criteria extraction is performed manually within Elsevier. This thesis is a first step towards automating eligibility criteria extraction from funding data. It researches BERT as a method to extract information on eligibility from funding data. The focus is on BERT specifically, since it shows state-of-the-art results in various natural language understanding tasks, including information extraction. Furthermore, BERT is a pre-trained neural network, which requires less labeled training data to perform well compared to other neural networks. Combined with the fact that BERT was not yet used on funding data, these findings made it interesting to research BERT in information extraction from funding data. Literature shows that a three-step approach with a segmentation step and two classification steps, where BERT is deployed as a classifier, is common in information extraction. In this work, this approach is tested on funding data from Elsevier. As the creation of training and evaluation datasets is a labor-intensive task, an approach with minimized data annotation is applied by using a small dataset and balancing the data before data annotation. The three-step approach, using BERT BERT as classifier, is compared to a BiLSTM and Conventional Machine Learning implementation. The results show that BERT outperforms the two other implementations. It is suspected that the BERT embeddings are crucial to outperforming the different implementations. An additional experiment explored this and found that BERT embeddings indeed play an essential role. Although BERT outperforms the two other implementations, its performances are too low to be viable in an actual situation. It is suspected that the suboptimal performances can be attributed to the data (quality, size, and distribution) used in this case study. Future work would need to be done to confirm that the data limited the performance of this study. If so, a three-step approach with BERT as a method for the classification tasks is a viable approach to extract information from funding opportunities with minimized data annotation, as long as the data is of high quality and the resources to run a deep neural network are available.

Acronyms

BERT Bidirectional Encoder Represenations from Transformers

BiLSTM Bidirectional Long Short-Term Memory

CNN Convolutional Neural Network

CRF Conditional Random Field

CV ML Conventional Machine Learning

DOM Document Object Model

GPU Graphics processing unit

HMM Hidden Markov Model

IE Information Extraction

LSTM Long Short-Term Memory

MEMM Max Entropy Markov Model

ML Machine Learning

NER Named Entity Recognition

NLP Natural Language Processing

RAM Random-access memory

RNN Recurrent Neural Network

SVM Support Vector Machines

Contents

Chapter 1

Introduction

Funding appears in all levels of society, from research to festivals, and from businesses to sport societies. People in all levels encounter the phenomenon of funding: 'money given by a government or organization for an event or activity'. as Cambridge Dictionary defines it [94]. For example, in exchange for financial resources, logos are printed on sport shirts in your local sport club. Also, lots of large organizations are built up and run based on funds such as the World Health Organization (WHO) [88], which is funded by their Member States, or the Formula 1 racing teams, which are funded by their partners [97]. These money exchanges for activities or events are a crucial part of the financial health of organizations.

Particularly research organizations have funds as their main income and therefore regularly search for funding. Funds can be found on fund organizations websites', such as the website of the European Commission [26]. However, searching multiple websites is a time-consuming process. A number of companies, including Elsevier [37], identified this problem. "Elsevier is a global information analytics business that helps institutions and professionals progress science, advance healthcare and improve performance" [36]. One of the services Elsevier offers, is gathering and structuring funding opportunities in one location, so that researchers and other professionals can quickly find funding opportunities that suit them. This allows researchers to focus more on their main activities rather than the search for funding. In order to allow researchers and research organizations to effectively search a funding opportunities website, a few characteristics to search on are crucial, such as field of research, location restrictions, group of researchers or single researchers, and many more. These characteristics need to be obtained from the funding opportunity, which is in text format. Although many of the characteristics to search on are extracted and structured in a large database, a preliminary experiment by one of Elsevier's employees indicates that the characteristics are not always extracted, even though the information is available in the funding opportunity. This is possible due to the fact that most of the characteristics are manually extracted and human mistakes are inevitable. Especially the characteristic 'Individual Applicant Type' is found to contain mistakes[1]. This, combined with the fact that it is a crucial characteristic in search-

[1]One out of ten randomly selected FOs which do not have the characteristic 'Individual Applicant Type' available (67% of all FOs) in the database, do have information for that characteristic in the pieces of original funding opportunity text available. This indicates that there is room for improvement.

ing funding opportunities, makes 'Individual Applicant Type' the characteristic of interest to improve the extraction coverage by automating the extraction process. 'Individual Applicant Type' is defined as the career stage(s) at which an individual applicant needs to be in order to be eligible for that funding opportunity. Improving the extraction coverage of 'Individual Applicant Type' can be done by extracting information on the eligibility of individuals based on career stage. For example, the sentence: "Bachelor students focusing on biomedical technology are eligible to apply", is a clear example of text that contains information on 'Individual Applicant Type'. Another example would be the sentence: "It is open to applicants of all nationalities and at all levels." Both examples provide information on 'Individual Applicant Type'. The first example provides information that 'bachelor students' are eligible to apply. The second example shows that applicants in any career stage may apply.

Extracting this kind of information from text uses Information Extraction (IE) methods. Information Extraction is a subfield of Natural Language Processing (NLP). In literature, a limited number of research papers can be found which focus on Information Extraction from funding data (text). A more extensive discussion of these research papers can be found in Chapter 3. The main methods used are SVM in combination with HMM, MEMM, CRF models, and rule-based approaches. The newer methods, machine learning methods and more specifically deep neural networks, are more commonly used in similar domains, including resume IE, due to their high performances as found in the Research Topics Report related to this study [101]. The drawback of deep neural networks is the extensive use of labelled training data, which can be labor-intensive [23]. BERT is a novel neural network specialized on natural language data (language model) introduced by Devlin et al. [34] in 2019 and is based on the concept of transfer learning. It is a large pre-trained language model and uses its pre-trained knowledge for all other tasks it encounters. This ensures that BERT has a better understanding of natural language data (semantics) which makes BERT require less training data compared to other neural networks [34] to achieve state-of-the-art results in various natural language understanding tasks including information extraction [107, 30]. This, combined with the fact that BERT has not yet been deployed on funding data, makes it interesting to research whether this new language model can be applied in information extraction from funding data. This results in the following main research question:

How can the state-of-the-art NLP method BERT be applied to extract information from funding opportunity data with minimized data annotation?

The main research question is answered in three parts. First, reviewing literature on BERT in information extraction to find suitable approaches to apply BERT in the funding domain. This translates to the question: (1) How is BERT applied in information extraction in similar domains?. This question is answered in Chapter 3: Related Work, which summarizes reviewed literature. It shows that a two-step or three-step approach where BERT is deployed in the last steps for either named entity recognition or text classification depending on the IE problem at hand, is a common approach for BERT in information extraction for similar do-

mains.

Second, the main research question is answered by assessing whether a two-step approach or three-step approach with BERT, as was found in literature, also applies to funding data in a real-world scenario. The approach is therefore tested on the funding opportunity data available within Elsevier by extracting information on 'Individual Applicant Type' from funding opportunity data. In order to assess how well BERT is performing, BERT is compared to BiLSTM and Conventional Machine Learning approaches (non-neural network machine learning algorithms). This answers the question: (2) *How well can BERT extract information from funding opportunities compared to BiLSTM and Conventional Machine Learning models?*. This question is supported by three sub-questions that reflect the two segment level steps in a three-step approach that involve BERT as well as the evaluation at funding opportunity level: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform: A) in determining whether segments contain information of interest?, B) in determining the type of information in a segment of interest?, C) at funding opportunity level?* Extensive answers to these questions are given in Chapter 8: Results.

Third, to answer the main research question, labelled funding data is required to train and evaluate BERT. As found earlier in this introduction, the drawback of machine learning approaches, especially neural networks, is the extensive labelling data required [23]. Although BERT requires less training data compared to other neural networks [34], data annotation is still a labor-intensive task. Therefore, minimizing data annotation is key to using this approach. This results in the sub-research question: (3) *How to acquire balanced labelled datasets of sufficient quantity and quality with minimized annotation?*. Chapter 6: Creation of Training and Evaluation Datasets provides an answer to this research question.

This master thesis report is structured as follows. In Chapter 2: *Background in IE* a closer look is taken at Information Extraction as a subfield of NLP. Chapter 3: *Related Work* discusses relevant literature related to Funding IE, Resume IE, and BERT in Information Extraction. Chapter 4: *Funding Data* discusses the data available within Elsevier for this study. Chapter 5: *Methods*, describes the main methods used to test the findings of the Related Work. Chapter 6: *Creation of Training and Evaluation Datasets*, discusses the creation and annotation of the training and evaluation datasets. Chapter 7: *Experiments* describes the experiments conducted in the case study. The results of the experiments are presented and discussed in Chapter 8: *Results and Discussion*. The Limitations and Future Work can be found in Chapter 9. Lastly, the Conclusion can be found in Chapter 10.

Chapter 2

Background in Information Extraction

Information Extraction (IE), a subfield of Natural Language Processing (NLP), is defined as: "*any process which selectively structures and combines data which is found, explicitly stated or implied, in one or more texts. The final output of the extraction process varies; in every case, however, it can be transformed so as to populate some type of database.*" [28]. In simpler words: "*Information extraction is the task of finding structured information from unstructured or semi-structured text*" [1]. In this report when referred to information extraction, Closed Information Extraction is meant. In contrast to Open Information Extraction [7, 6, 11], Closed Information Extraction requires upfront specification of the information that is to be extracted. where the information to be extracted is specified upfront. The first information extraction systems date back to the late 70's, such as the FRUMP Program [31]. However, IE got little attention until the 90's when the Message Understanding Conferences set out research challenges in this field [48, 67].

2.1 Methods in IE

In information extraction different methods are used to extract information from text. These methods are summarized below along with their benefits and drawbacks.

1. *Rule-based approaches* use linguistic patterns in text to extract information. For example, the expression "(watched or seen) <NP>", might capture the name of a movie (represented by the noun phrase (NP)) [128]. Rule-based methods were the main methods in the 90s [114, 102]. The main benefits of rule-based methods, as discussed in [23], are that they are easy to comprehend and implement and can easily incorporate domain knowledge. Furthermore, these methods can achieve high precision as shown in [125]. However the drawback of these methods is that, in order to achieve high precision, rule-based approaches require a high number of patterns. In addition, these rules and patterns are specific to a particular domain and require tedious manual labor, which results in an implementation that is not easily scalable and is labor-intensive [23]. Although rules can be semi-automatically generated as presented first by Califf et al.[14], this still requires domain knowledge and labor-intensive initial setting of rules.

2. *Gazetteers* are similar to rule-based methods, but instead of patterns they use characters, words or keywords to extract information. For example, the '@' character is helpful to detect and extract e-mail addresses from text. However, gazetteers can also help to detect features in the text. For example, Ku et al. [64] used a gazetteers list to detect crime act and the object used in the crime. This knowledge was then used by rule-based approaches to detect more complex information from the text, such as the clothes someone was wearing. As discussed in [113], the kind of gazetteer features as used by Ku et al. [64], are highly informative and result in a higher accuracy of information extraction. However, the major drawback of using gazetteers is that in case a gazetteer list is used, a large number of keywords, synonyms and word combinations impede the creation of complete gazetteers lists, which limits the ability to achieve meaningful information extraction [21]

3. *Sequential Labelling* is a group of methods which view the information extraction problem as a sequence of tokens which need to be given a label. A label is assigned based on the probability of that token following the previous token. The goal is to find the sequence labelling which provides the highest probability [119]. The most common sequential labelling algorithms are: HMM, MEMM, and CRF [136, 9, 105]. A paper by Zhang et al. [136], uses the sequential labelling algorithm HMM to extract information from restaurant reviews. Each token in the review was given one of the following labels: Prefix, Suffix, Target, and Background. The labels were given based on the probability of the next token to be of that state based on previous state. The results of this HMM showed that the probability of a token being a Target label increased when previous tokens consisted of HTML tags and the current token had a different font size. As stated in [119]: "sequential labeling enables describing the dependencies between target information. The dependencies can be utilized to improve the accuracy of the extraction." However, this comes at a cost: these algorithms require training data to obtain realistic probabilities of the different states (labels).

4. *Machine Learning* consists of algorithms that are able to learn and gradually improve through experience. Machine learning (ML) methods can be broadly subdivided into unsupervised and supervised approaches. The main difference is that supervised approaches classify data with the use of labelled training data, while unsupervised approaches try to find a pattern in the data without labels. Unsupervised ML methods are not used often in information extraction. This in contrast to supervised ML, for which multiple methods are regularly used. In the early 2000s, support vector machines were the best performing supervised ML algorithms for information extraction as stated by Li et al. [69]. This statement is supported by two papers published in 2002 and 2003 respectively [55, 78]. However, this was before neural networks became popular as of the 2010s. Neural networks are nowadays often used in information extraction. One of the common models used is the Long-Term-Short-Term Memory (LSTM), a recurrent neural network, which is adapted so that it can handle long-distance dependencies in text. For example, if in the beginning it is mentioned that the text is about the band: "ABBA" and

later in the same text the phrase “the band” occurs in the text, the LSTM will link “ABBA” to ‘the band”. An LSTM is able to capture information in one direction. To capture information to use context of the past (left side) and the future (right side), the BiLSTM combines a Forward LSTM with a Backward LSTM [108]. Because of this feature, the BiLSTM is widely used in information extraction. Chalapathy et al. [18] used LSTM in combination with a CRF model to extract clinical concepts from clinical records. The forward model is a LSTM, while the backward model is a CRF algorithm. This combination of BiLSTM and CRF model showed to be a top-ranked system in the 2010 i2b2/VA reference [18]. Jia et al. showed another combination with a CNN and CRF, where Chinese named entities were tagged [56]. Another group of Machine Learning methods is the transformer models. Since BERT was introduced [34], it has outperformed BiLSTM and LSTM on various natural language understanding tasks[40, 121] and has shown state-of-the-art results in a diverse set of natural language understanding tasks [10, 107, 91, 20, 57, 138]. BERT performs so well on numerous tasks, because of transfer learning. BERT is pre-trained on a large and diverse training dataset (3000 million words) and transfers this knowledge to all the other tasks it encounters. Thus, compared to other neural networks, BERT requires less training data while achieving similar results. More detailed background on (Bi)LSTM neural networks and BERT models can be found in Appendix A. The main benefits of Machine Learning approaches compared to rule-based approaches is that they require less domain knowledge, are easier to adapt and there are also scalable [23]. The major drawback of supervised ML methods is their need for labelled training data, which an be labor-intensive to gather [23].

2.2 The Five Tasks in IE

Almost any information extraction problem can be decomposed into five tasks or a subset of these five tasks as Simoes et al. proposed in 2004 [109]. Simoes et al. based their work on earlier published work by Mccallum [79]. This decomposition makes the problem less complex and allows different approaches to be taken for every task to fit the information extraction problem. The five different tasks are summarized below:

1. *Segmentation* is aimed at dividing text into segments. Segmentation can be performed at different levels of detail. Segments can be words, sentences, phrases or even paragraphs, depending on the task [109]. Based on segmentation size, text layout, and other task-related factors, different approaches are used. Possible approaches are lexical [68, 24], syntactic, heuristic [77] or supervised learning based segmentation [63, 62].

2. *Classification* involves determining the class (type) of the segments. One of the most common approaches in information extraction for classification is named entity recognition. In named entity recognition words or sets of words are given a label. For example, “James Bond” is an set of words (entity) that represents a British secret agent working for MI6 in a series of movies. Such an entity is recognized and tagged into a class. The most common named entity recognition algorithm classifies each segment as one

of the four classes: Location, Organization, Person or Other. Using this algorithm the entity 'James Bond' will likely be classified as a Person. Different approaches are suitable for named entity recognition, depending on the task at hand. For instance, tagging all e-mail addresses [72] is less complex than tagging deadlines for a specific project [82]. E-mail addresses can be easily tagged based on a rules. A single rule suffices: all strings that contains an 'at' (@) sign. In contrast, tagging a deadlie for a project requires a more advanced approach as it is not confined to a specific format. Suitable candidates might be sequential labelling approaches: HMM [136, 106], MEMM[9, 61], CRF [89, 105], but also Machine Learning approaches: SVMs [69] or newer approaches based on neural networks, such as CNN, (bi)LSTM or BERT [18, 56, 20]

3. *Association* is aimed at finding the relations between classified segments (entities) and is also called relation extraction. An example would be <Person> works in <Location>. From this segment, the relation between entities <person > and <Location > can be extracted and defined as 'works in'. Similar to the Classification task, supervised approaches, such as MEMM [76], CRF [13], but also SVMs [49] and newer approaches using neural networks [43, 90] or a combination of these methods can be used. HMM's are less suitable, because these models are suited for local problems and can not cope with long distances across entities [133]. In addition to these approaches, kernel methods are also often used in relation extraction [5]. Unlike well-known classification methods, kernel methods do not require a feature vector representation, but use sentences themselves as features [133]. Besides these supervised methods, semi-supervised methods, which require less training data, can be applied. Semi-supervised algorithms, such as DIPRE [12] or Snowball [2] use a small amount of tagged instances or tagged extraction patterns as input and find more patterns and general rules based on them [5]. Also newer approaches on neural networks such as BiLSTM, CNN, CRF, and BERT are often deployed to extract relations between entities [50, 131].

4. *Normalization* is a process that guarantees that extracted data is in the same format. For instance, dates may be formatted: mm/dd/yyyy (US format) or dd/mm/yyyy (European format. Rule-based approaches are often best suited to perform normalization [109].

5. *Coreference or Deduplication* focuses on entities that are referred to in multiple ways in the same text. An example is the entity 'Bill Gates', who could previously have been referred to as 'the richest man in the world' or the word 'He' in the sentence: 'He is the founder of Microsoft'. These words all refer to the same entity. In order to detect this, the semantic meaning of the entity is of use and only entities that have similar semantic meanings are seen as referring to the same entity. Similar to the Association task, different sets of methods can be used to perform Coreference. Possible methods include a combination of clustering and unsupervised Machine Learning (ML), supervised ML (both conventional ML, such as logistic regression as well as neural networks, such as (Bi)LSTM and BERT). [17, 45, 134, 59, 29].

Chapter 3

Related Work

The previous Chapter presented various methods in IE and a division of information extraction into five different tasks. Using this background knowledge, this Chapter will zoom in on information in the funding domain and similar domains with a focus on the transformer model BERT.

3.1 Funding Information Extraction

The main topics in the funding domain that are researched focus on how higher education institutions use their funds and the influence of funding on the performance of researchers [8, 60, 42, 47]. However, little research has been done on information extraction of funding data. To the best of my knowledge, three papers focus on information extraction in relation to funding. Two of these papers are closely linked, both focusing on tagging funding bodies and funding grants (ID's) from scientific articles [63, 62].The first of these two papers was published in 2017 [63] in response to BioASQ challenge task 5c. It evaluates three sequential labelling models: a Hidden Markov Model (HMM), a Max Entropy model (MEMM), and a Conditional Random Field (CRF). These models perform the task of tagging entities funding body and funding grants. In this paper, tagging entities (Named Entity recognition) is performed at the word level, meaning that for every word it is evaluated whether this is a Grant ID, Funding Body (FB) or neither of the two. The results show that a combination of these three models outperforms the best scoring model, the CRF model.

The second paper is a follow-up paper which uses scientific articles collected from ScienceDirect. From these articles two dataset were created, an automatically annotated 'silver set' and a smaller manually annotated 'gold set'. Different implementations of the three sequential learning models: HMM, CRF, and MEMM were evaluated which were either trained on the silver set, gold set or on pre-trained models. Again it was found that the combination of the three models outperforms the use of a single model both for tagging funding bodies and funding grants. This confirms the finding that the sequential learning methods HMM, CRF, and MEMM are complementary. Furthermore, it was found that the models trained on

the gold set outperformed the models trained on the silver set, showing that a smaller but a higher quality dataset is more suitable in tagging FB's and Grants. Both papers suggest to use a three-step approach, which first splits the text into smaller segments, in this case paragraphs. Secondly, the segments are classified based on whether these segments contain information of interest (contain information on funding information). Both papers presented use support vector machines (L2-SVM) to classify whether a paragraph contains information, and acquired an F1 score of 90% [62]. In the third step, the HMM, CRF, and MEMM models were deployed to label the words in each paragraph as different entities (FB's, Grants, or None). These three steps resemble the different tasks in information extraction presented in section 2.2. The first step is a segmentation task, where text is split into paragraphs. The second step is a classification task that determines whether segments contain infomraiton on funding data. The last step is also a classification task that determines the type of entity (FB's, Grants, or None) of the words in the relevant paragraphs using the HMM, CRF, and MEMM models. This three-step approach has the benefit that not all words in the text parts are processed which results in less computational costs.

Besides these two papers, only one paper was found that focuses on tagging entities in funding opportunities [103]. This paper uses an ontology to improve the quality of funding opportunities in the database (CIDEM portal) by using information on the funding opportunity from other sources. An ontology is "a formal and explicit specification of a shared conceptualization" [128]. In the funding domain, a widely used ontology is the DINGO ontology [22]. The ontology describes the concepts and its relations in the funding domain. For example, a fund has an amount and that a fund is coupled to a project which in turn is coupled to an organization or individual. The property of interest to this master thesis: Individual Applicant Type is one of these concepts that can be part of the ontology. The ontology itself cannot extract information. However, an information extraction module within an Ontology-Based Information Extraction System can be used to extract the information which is then linked to a property inside the ontology. In the paper on improving the quality of the CIDEM portal [103], the information extraction module is formalized in the form of a knowledge parser. This approach was obtained from a paper published in 2004 [27]. Their knowledge parser consisted of four complementary parts: 1) The Document Object Model (DOM), which is used for understanding HTML and the structure of the document, 2) The Text Model, which uses regular expression techniques, 3) The Layout model, which focuses on the structure of the document, and 4) The NLP model, which utilizes proper names, verbal phrases, and synonyms. Just like the three-step approach of the previous papers, the knowledge parser can also be mapped onto the five tasks in IE. The segmentation task is already performed, because the different parts in the HTML tags are the segments. The knowledge parser uses the a DOM to obtain segments of interest. A classification task uses the Text Model, Layout Model, and NLP model to actually extract the information.

These three papers show that information can be extracted from funding data using a three-step approach: 1) Split the text into segments (segmentation), 2) Select segments of

interest (classification), 3) Determine what type of information is available in the segment (classification). This three-step approach uses two out of the five tasks in information extraction, as laid out in Chapter 2: segmentation and classification. The first two papers use mainly sequential labelling approaches. The last paper uses a combination of Document Object Model for segmentation and gazetteers and rule-based approaches for the classification tasks. None of these three papers applied a BERT model to funding data.

3.1.1 Resume IE

Since there is limited research on information extraction from funding opportunities or funding data in the Research Topics Report related to this study, literature related to other domains was investigated as well [101]. The most similar domain to the funding domain was resume IE, because of the similar kind of information to be extracted, such as information on education and work experience. The main related work and it's findings in this domain are summarized below.

Although the information that is extracted is similar in both the funding domain and resume domain, there are some slight differences which make it a bit easier to extract information from resumes than from funding data. Especially the more frequent use of headers in resumes simplifies the task. Headers provide information regarding the topic of the text below the header which can be utilized for segmentation. Chen et al. [21] used this information to create segments. They deployed a classifier that determined whether a line is a header based on the length of the line and its position with respect to other lines. The text between headers was seen as segments (segmentation task). The headers were classified into different classes based on a synonym list (gazetteer list) to obtain segments of interest (classification task).The extraction of specific information was done by classifying segments into pre-defined classes using a Naive Bayes text classifier (classification task). Lumban et al. [75] also used gazetteers to both detect headers and classify them to obtain segments of interest. For the extraction, different methods were used depending on the specific entity. These included gazetteers (regex), clusters and their centroids. Unfortunately this approach with headers can not directly be applied to funding data, because text of funding opportunities are not that common in funding opportunities. As further explained in Chapter 4, no text of funding opportunities is available within Elsevier. However, the approaches used in these papers follow the three-step approach which could be applied to extract information from funding opportunities. The three-step approach used in these papers, uses different approaches for the last classification task, either text classification using Naive Bayes classifier or named entity recognition using a combination of gazetteers and clustering methods.

The three-step approach is also found in work by Zu et al. [140], where a combination of a line type classifier and line label classifier are used to extract information. The line type classifier classifies each line into one of the four classes "header, content, metadata, and footer", where neighbouring lines with the same class form a text block (segmentation task). There-

after, a line label classifier labels each line in the content segments with more fine-grained classes: “education, work experience, personal information, projects, skill, and publication” (classification task). On these text blocks a named entity recognizer is applied using a combination of BiLSTM and CNN to obtain certain specific entities from the data, such as the years of work experience (classification task). Also in work by Zhang et. al, 2020[137] a three-step approach was used. First segments of interest were obtained based on headers (segmentation task). Next a rule-based named entity recognizer was applied (classification task).

Most of the research identified in resume IE uses a three-step approach, where a segmentation and two classification tasks can be identified. However, there are methods in resume IE where a two-step approach with a segmentation task and only one classification task is used instead. For example, in work by Wang et. al [127] with as goal to extract keywords from resumes that depict competencies in three classes: functions (FUNC), specialities & skills (SEP), and none of the two (None). BERT was utilized as a named entity recognizer (NER) to perform this task. Compared to a three-step approach, this approach lacks a classification which selects segments of interest. This extra task is useful, however, when the number of segments is large and the proportion of segments of interest is low. By selecting the segments of interest, it saves some resources that would have been used on the more complex classification step that extracts information from the segment and thus save some computational costs.

In existing resume extraction literature, different methods for the classification step(s) of the two-step approach or three-step approach were found to tag entities within the text or classify text. Methods used ranged from rules or gazetteers [75, 21], sequential labelling methods, such as HMM [132] and CRF [4] to machine learning approaches such as Naive Bayes[21], SVM [132], and a combination of BiLSTM and CNN [140]. Besides these methods, as already found in research by Wang et. al [127], there is a recent trend in resume information extraction of transformer models and more specifically BERT models. Bhatia et al. [10] used BERT to determine which applicant is most eligible for the job based on required job experience. They trained BERT to perform sequence pair classification, where the professional job experience of the applicant in text format is compared to the job description. The better the job experience and the job description match, the more suitable an applicant is. BERT is not only used in this paper as part of the sequence pair classification step, but also in the classification step. This step determined the segments of interest by classifying non-LinkedIn resumes using the segment structure of LinkedIn and achieved an accuracy of 97%. For the sequence pair classification task an accuracy of 72.77% was achieved. For the segmentation step, heuristic rules on positions and lengths of lines were combined to split the text into segments (paragraphs). Su et al. [117] used BERT to determine the quality of a resume dataset. For this unstructured resume dataset, it was evaluated whether it was applicable as a dataset to automatically extract information for resumes of common types, such as names, gender, birthday, education, work experience and so on. The resumes were so unstructured that they were split into sentences. The ML model classified for each sentence

whether the sentence contained information on the common types. Four ML models were evaluated including BiLSTM, BERT, CNN, and BiGRU, with different kinds of embeddings (either BERT embeddings or Tencent AI Lab Embeddings). BERT in combination with BiLSTM and BERT embeddings outperformed all other models. The F1 score of 95.8% showed that the resume dataset was applicable to automatically extract the common types from resumes.

3.1.2 BERT in Other Domains

The trend of BERT transformer models is also found in other domains. For example, information extraction from business documents. Zhang et al. [138] used BERT to extract content elements from regulatory filings and property lease agreements. Furthermore, Ngyuen et al. [82] employed BERT and ALBERT for named entity recognition on Japanese bidding and sale business documents. The models show satisfactory results (90.62% for bidding business documents and 86.14% for sale documents) with limited training data (100 documents). This paper supports the statement made by the developers of BERT [34] that BERT can achieve high performances with a rather small training dataset, because of its transfer learning ability. In the tourism domain, Chantraporncha et al. [20] used BERT as a named entity recognizer to extract name, location and facility type from reviews of restaurants, hotels, shopping, and tourism locations. Furthermore, BERT was used to classify sentences of reviews into the class for which it provides information either: location (0), organization (1), facility (2), or None (3). These labels were thereafter used for summarizing the reviews. Sentences with a label were given more priority than sentences without labels. These papers show that BERT is utilized in extracting information from resumes, business documents, and reviews. In fact, BERT is deployed to extract information from text documents from many more domains, such as health care [30, 115] or robot navigation [107].

In literature on BERT in information extraction, BERT is often deployed as method for named entity recognition [82, 70, 138]. For example, as stated earlier in this section, Chantraporncha et al. [20] used BERT as a named entity recognizer to extract name, location and facility type from reviews of restaurants, hotels, shopping, and tourism locations. Solarte-Pabón et al. [115] also deployed BERT as a NER task in health care to extract information from Spanish Radiology Reports using Multilingual BERT. The papers that use BERT as method for named entity recognition (NER) use, similarly to the papers found in funding IE and resume IE, a two-step or three-step approach with a segmentation task and classification task(s), where the named entity recognizer is used as the last classification task [19, 30, 126, 70, 115, 107]. The combination of BERT in a two-step or three-step approach shows improvements over the baseline models tested in the papers, such as CRF-SVM [107], BiLSTM [30], and BiLSTM-CRF [19, 70].

BERT as named entity recognizer is often deployed in combination with other neural networks, such as the (Bi)LSTM, CRF, and CNN [32] or a combination of these models [19,

126, 70, 56]. For example, in work by Dai et al. [30] a combination of BERT, BiLSTM and CRF models was used to extract medical information from electronic health records. Dai et al. trained a neural network to perform a NER task. The neural network consists of three layers: 1) BERT layer, 2) BiLSTM layer and 3) CRF layer. The first layer (BERT) maps each word in the sentence to a word vector based on the pre-trained word embedding. The second layer (BiLSTM) extracts sentence features. The third layer (CRF) performs sequence labelling between sentences. This combined model outperformed simple BiLSTM and BiLSTM-CRF, by combining the strengths of each models. For example, the strength of BERT is its great understanding of natural language data and is used to convert natural language data to word vectors that can be understood by other models. The strengths of BiLSTM are the ability to detect long-term dependencies and the bidirectionality which allows the use of the whole context of a word, which is great for extracting sentence features. Details on BiLSTM and BERT can be found in Appendix A. The CRF model also has its strengths. It is a sequential labelling method that gives labels to tokens and is able to take into account context that was added by the bidirectionality of BiLSTM. This combination of (Bi)LSTM and CRF was introduced by Lample et. al [66] and since then often used for NER [19, 56].

Besides named entity recognition, BERT is often found as method for relation extraction [130, 71, 50, 131]. As explained in Chapter 2, relation extraction focuses on finding relations between entities. This step is implemented with an association task and follows after a named entity recognition process which consists of either a two-step or three-step approach. Relation extraction was not found in funding IE or resume IE, but shows usage in other domains, such as causalities extraction in the medical domain [50, 130].

3.2 Conclusion

This chapter showed the limited literature available on information extraction from funding data. In total three papers were identified, which used mainly sequential labelling methods and a combination of rules and gazetteers. None of the models utilized BERT as method, which supports the interest to research BERT on funding data. Because of the limited funding IE literature available, similar domains were investigated in the Research Topics Report related to this master thesis [101], including resume IE.

Both in Funding IE and Resume IE, a two-step or three-step approach approach with two out of the five common IE tasks was found to be a common approach. The two-step approach combines 1) a Segmentation step to split text in to segments with 2) a Classification step to determine the type of information available in the segment. In the three-step approach an additional classification step is performed between the segment and classification step of the two-step approach, which selects segments that contain information of interest. This additional step is useful when the number of segments is large and the proportion of segments of interest is low. By selecting the segments of interest, it saves some resources that would have been used in the more complex classification step that extracts information from the

segments.

Different methods for the classification steps were used in Funding and Resume IE, such as sequential labelling, but most machine learning approaches such as naive bayes classifier. Resume IE literature showed a recent trend in the use of BERT models. This trend was also found in other information extraction domains and showed start-of-the-art results [30, 107]. Specifically resume IE showed that BERT could be applied for both classification steps in a three-step approach by performing either text classification or named entity recognition depending on the IE problem at hand [127, 117, 10], where the BERT model was either used as text classification or named entity recognition.

Specifically research in domains other than funding and resume showed the use of BERT as named entity recognizer (NER). BERT for NER is often deployed in combination with other neural network models, such as CRF, CNN, and BiLSTM. The strength of each model is combined to achieve higher performances and outperforms singular usage of the models. Besides BERT for NER, BERT was also found to be used for relation extraction in domains besides funding and resume. BERT for text classification that was found in resume IE was not specifically found in other domains for information extraction. Furthermore, only in the resume domain BERT was found to be used for text classification. However, it has shown state-of-the-art results by outperforming traditional methods, such as LSTMs, CNN, and Logistic Regression [41, 118].

These findings provide an answer to the question: *How is BERT applied in information extraction in similar domains?*. The findings show that BERT is often applied in similar domains as method for the classification tasks in a two-step or three-step approach either for text classification or named entity recognition. Text classification is often used for the first classification task and both text classification, while named entity recognition is found to be used for the second classification task. Relation extraction using BERT is a common method in other domains, but not specifically in a domain similar to the funding domain.

Chapter 4

Funding Data

To test the applicability of BERT in a two-step or three-step approach in the funding domain, a case study is conducted to answer the question: *How well can BERT extract information from funding opportunities compared to BiLSTM and Conventional Machine Learning models?*. Elsevier's case, which was introduced in the introduction of this report is used as case study for this master thesis. The case introduction showed that information on 'Individual Applicant Type' is often available in text format, but not always found as characteristic in the funding database. Obtaining the exact number of funding opportunities to which this applies is impossible, since it would require manually reading thousands of funding opportunities. However, as mentioned in Chapter 1, a preliminary study by one of Elsevier's employees gives a fist estimate. One out of then randomly selected funding opportunities which do not have the characteristic 'Individual Applicant Type' available (67% of all FOs) in the database, do have information for that characteristic in the pieces of original funding opportunity text available. This indicates room for improvement. The goal of this case study is to test the applicability of BERT for the extraction of information on 'Individual Applicant Type' from funding text. Furthermore, it is a first step in automating the process of extraction of the characteristic 'Individual Applicant Type' within Elsevier to improve the availability (extraction coverage) of the characteristics.

'Individual Applicant Type' is defined as at what stage in their career an *individual* applicant needs to be in order to apply for a funding opportunity. For example, only early career researchers who have received a Ph.D in the last five years may apply or both undergraduate and graduate students may apply. As the name suggest, 'Individual Applicant Type' solely considers individuals and is not available to funding opportunities for which only organizations may apply.

This chapter focuses on what funding data within Elsevier is available from which information can be extracted on 'Individual Applicant Type'.

4.1 Database and a single Funding Opportunity

The funding data available in Elsevier consists of a database of over 200.000 funding opportunities which were gathered in 2020 and 2021. The funding opportunities in the funding opportunity database have a structured format of characteristics, which have been extracted manually from posts on their funder's website which are in text format. In this extraction process, employees of Elsevier watch the website of the funder, read the funding opportunity text and based on that information fill out the characteristics in the database. The structured format of characteristics allows customers of Elsevier to search the funding opportunities that suits them. An overview of the format of a funding opportunity in the funding opportunity database is visualised in Table 4.1[1].

A funding opportunity in the database consists of two parts: 1) characteristics, and 2) pieces of original funding opportunity text for every group of characteristics. The characteristics are marked in Italics in Table 4.1. The characteristics are grouped together in different columns (see columns Award, Eligibility and Application Procedure). In each group (column), a piece of original funding opportunity text is included which contains information on the extracted characteristics of that group. In Table 4.1, the original funding opportunity text is visualised by the underlined word 'Text' (Text). An example is the characteristic 'Amount' which is part of the group of 'Award' characteristics and has a piece of original Text. Note that these pieces of text vary in length from one sentence to multiple pages. Furthermore, the pieces of text can be incoherent as it can be composed of smaller pieces that are obtained from different locations in the original funding opportunity.

The characteristic 'Applicant Type' is visualised in Table 4.1, as part of the characteristic 'Individual Eligibility'. This shows 'Applicant Type' is only available for funding opportunities that allow individuals to apply. Besides 'Applicant Type', 'Individual Eligibility' has a characteristic 'Limitation', which has one of the following values:

a) Limited: there is evidence in the funding opportunity that there is a limitation in who as an individual can apply.

b) Not Limited: there is evidence in the funding opportunity that there is no limitation in who as an individual can apply.

c) Not Specified: the funding opportunity does not specify anything on individual eligibility.

[1]Note that this is a fictional snapshot to get an overview. The funding opportunities in the database contain more characteristics and groups. The full data set of funding opportunities in the database will not be shared here, because it contains confidential and/or proprietary information. More detailed information on the database and data can be found in the research topics report related to this master thesis [101]

FO ID	Award	Eligibility	Application procedure
300123	{*Amount*: 50.000, *Currency*: US Dollars, ..., Text: The applicant will be awarded $50.000 US dollar over a period of 4 months in which the work is required to be executed.}	{*Individual Eligibility*: {*Applicant Type*: ['GRADUATE', 'NEW-FACULTY'], *Degree requirement*: Bsc, Msc, *Limitation*: LIMITED}, ..., Text: Applicants who have a Bachelor and Master in the direction of Chemistry or are currently pursuing a Master in the direction of Chemistry may apply}	{*Deadline*: 1st of March 2023, *Products to hand in*: relevant research and CV, ..., Text: The applicant need to hand in relevant research and their CV via our website before the 1st of March 2023.}

Table 4.1: Snapshot of a Funding Opportunity in the Funding Opportunity Database of Elsevier

Career Stage	Explanation
Undergraduate	Opportunities that specifically mention that bachelor or undergraduate students may apply
Graduate	Opportunities that specifically mention that graduate students may apply. This is also used for undergraduate students who are completing a degree and seeking funding for continuing to graduate studies. This can include Master students and Doctoral students (PhD).
New Faculty	Opportunities for which new faculty who are considered inexperienced or emerging may apply. This can include postdoctoral applicants (obtained a PhD recently); new, young, emerging researchers, or early career investigators; or junior faculty applicants. This classification is used across all disciplines, such as an emerging artist or young investigator in cardo medicine.
Degree	Opportunities for which an applicant has obtained a degree or is experienced (researcher) (if stated in the opportunity)
Other	Opportunities for which individuals can apply but none of the above categories apply

Table 4.2: Career stages for 'Individual Applicant Type'

In the situations where 'Limitation' has value 'limited' the applicant type characteristic is filled, which means that only individuals with a certain career stage can apply. The example in Table 4.1 has a 'limited' value for characteristic Limitation, and a corresponding list of career stages in which an applicant is required to be to apply for the funding opportunity as 'Applicant Type' value (see ['GRADUATE', 'NEW FACULTY']). In total five different career stages

(levels of 'Individual Applicant Type') are used in the database for 'Individual Applicant Type' (see Table 4.2). For each funding opportunity, one or more career stages can apply.

In the situations where 'Limitation' is 'Not Specified' the applicant type value is empty, because there is no information on 'Individual Eligibility' and thus no information on 'Applicant Type'. 'Applicant Type' value is also empty for 'Not Limited' cases, because there is no limitation for individuals and thus no limitation for 'Applicant Type'.

An overview of the different characteristics and their corresponding values is visualised in Figure 4.1.

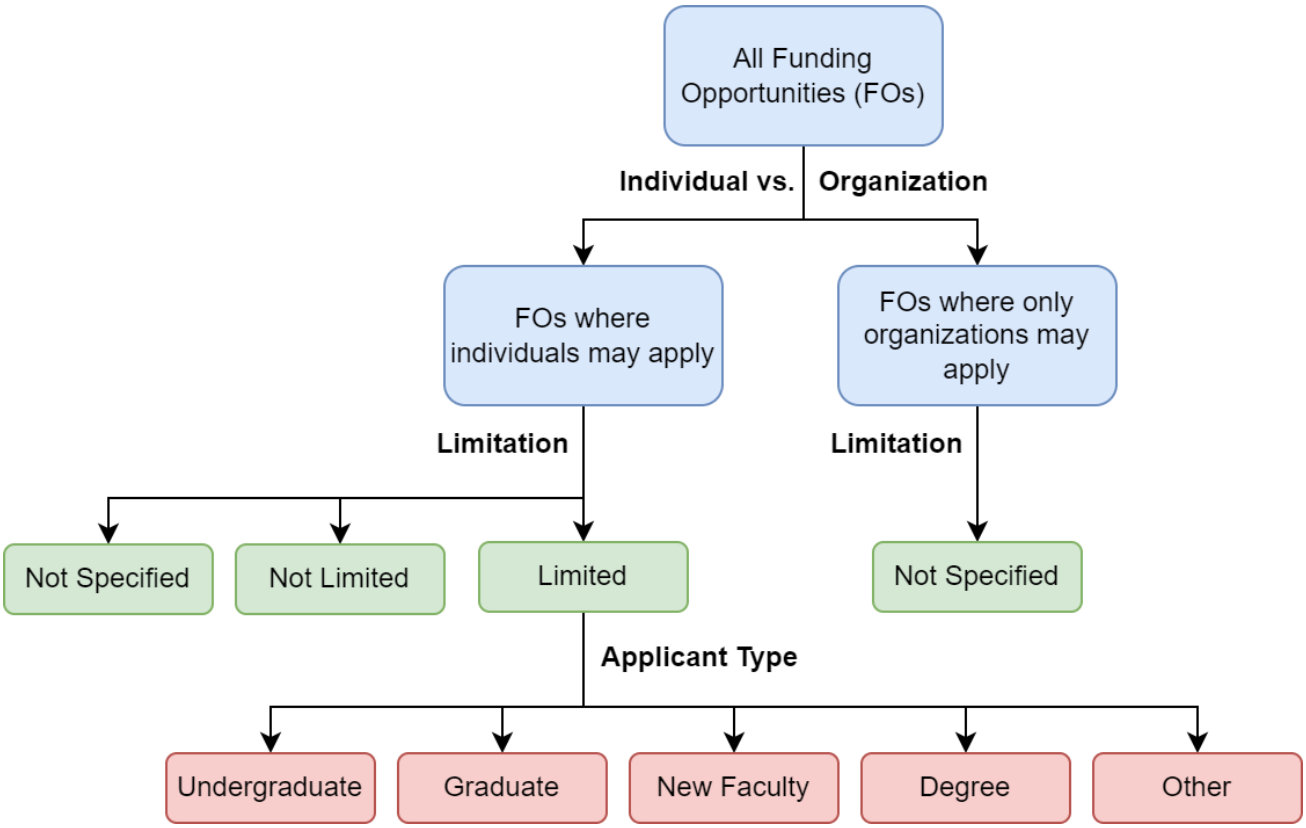

Figure 4.1: Overview of relevant characteristics and their correspondence

4.2 Data of Interest

Besides the characteristics and the pieces of original funding opportunity text, the database also contains some general information about the source of the funding opportunity. This includes the URL which redirects to the website where the full original funding opportunity text is posted. The original funding opportunity text is likely to contain more complete information on the funding opportunity, since it is the source of the funding opportunity in the database. However, this thesis focuses on the available funding opportunity database, rather than ob-

taining the full original funding opportunity. This has multiple reasons: 1) For more than 10% of the funding opportunities, the full original funding opportunity is not available. The URL is not accessible for these cases, because the funding opportunity is not active anymore, as applicants can no longer apply. 2) In case the full original funding opportunity is available, it needs to be scraped from a website. This process is complicated due to different websites and different layouts of full original funding opportunities. The scraping process, could be a master thesis project on its own and therefore it has been decided to rather focus on the available data in the funding opportunity database.

The data of interest in the database, to extract information on 'Individual Applicant Type' from, are the pieces of text from the original funding opportunities that are added per group of characteristics in the database. Not only pieces of text on eligibility criteria, but pieces of text from all different groups of characteristics are considered, because the preliminary study found that information on a characteristic sometimes is available in other pieces of original funding opportunity text[1]. After some exploration of the data and finding the most prominent words in each group of texts, it was found that pieces of text within the group of characteristics: 'Opportunity Date' contained mixed languages and lots of spelling mistakes. Therefore, the pieces of text of the group of characteristics on 'Opportunity Date' were discarded from the available texts. Furthermore, only the funding opportunities which had funding opportunity text parts written in English were considered in this case study. This was the case for 97% of the funding opportunities which resulted in a dataset of 195.558 funding opportunities.

The text parts of 195.558 funding opportunities can be used as data to extract information on 'Individual Applicant Type' from to determine the career stage(s) that may apply for each funding opportunity (see Table 4.2).

4.3 Data Descriptives

In the database, 33.6% of the 195.558 funding opportunities have the characteristic 'Individual Applicant Type' available. This means that 33.6% of the funding opportunities have the label 'Limited' with at least one career stage for which a limitation is posed. The label 'Not Limited' occurs in 0.2% of the funding opportunities and the majority of the funding opportunities 66.2% have the label 'Not Specified'. The 'Not Specified' funding opportunities can be divided into three types of funding opportunities: 1) Funding opportunities that allow individuals to apply, but do not specify anything on 'Individual Applicant Type', 2) Funding opportunities that allow individuals to apply and do contain information on 'Individual Applicant Type' in text format, but the information is not mapped to the database characteristic, and 3) Funding opportunities which do not allow individuals to apply (Organization funding opportunities) and directly assign a 'Not Specified' label to the funding opportunity. Determining to which type of funding opportunity a funding opportunity belongs is difficult, because in the

[1] Details on the preliminary experiment can be found in the Research Topics report related to this case study [101].

database there is no distinctive indication as to which type of funding opportunity belongs. The fact that also Organization funding opportunities have the value 'Not Specified' further complicates the process of obtaining an indication on the number of type 2 funding opportunities.

In case a funding opportunity has the value 'Limited', the specific career stage(s) on which a limitation is posed are defined as well. Multiple career stages may apply per funding opportunity. A short overview of the percentages of funding opportunities with a specific career stage as limitation is shown below in Table 4.3. The majority of the 195.558 funding opportunities have a limitation in the category 'Degree'. Furthermore, the category 'Graduate' is available in a large portion of funding opportunities as limitation. In contrast, the limitations 'Undergraduate' and 'Other' are not as common.

Career Stage	**Percentage of funding opportunities with this career stage**
Undergraduate	7.83%
Graduate	43.15%
New Faculty	16.00%
Degree	59.66%
Other	1.92%

Table 4.3: Percentage of funding opportunities with a specific career stage

4.4 Summary

This Chapter discussed the funding data available within Elsevier from which to extract information on 'Individual Applicant Type'. The text available in the database is in the form of a selection of text parts from the original funding opportunity text that was posted by the funder on their local website. The text parts are manually extracted by employees of Elsevier for each group of characteristics. A text part can vary in length from one sentence to multiple pages and can be composed from text of different locations in the original funding opportunity text. Thus, text parts may not be coherent. All text parts will be used, because it was found that information on eligibility of 'Individual Applicant Type' was sometimes available in text parts other than the text part on eligibility. [1] Only the text parts of the group on 'Opportunity Date' will not be used in this study, because these text parts are found to contain mixed languages and lots of spelling mistakes. Text parts of 195.558 funding opportunities written in English can be used in this study.

This Chapter also showed the use of different characteristics and how they relate to one

[1] Details on the preliminary experiment can be found in the Research Topics report related to this case study [101].

another in the database. The characteristics ‘Limitation’ and ‘Individual Applicant Type’ are both part of the characteristic ‘Individual Eligibility’, which is only available for funding opportunities which allow individuals to apply. The ‘Limitation’ characteristic value specifies whether the funding opportunity specifies anything on individual eligibility. In case the ‘Limitation’ value specifies that there is a limitation in who can apply as an individual, the characteristic ‘Individual Applicant Type’ is filled with that limitation. A total overview of these characteristics with their possible values and how they relate to one another can be found in Figure 4.1.

Lastly, section 4.3 showed the availability the ‘Limitation’ characteristic and ‘Individual Applicant Type’ characteristic and their possible values in the dataset of 195.558 funding opportunities. Approximately two thirds of the funding opportunities have ‘Not Specified’ as ‘Limitation’ value and one third has the value ‘Limited’. Only 0.2% of the funding opportunities have value ‘Not Limited’. The characteristic ‘Individual Applicant Type’ is available for one third of funding opportunities, as they have ‘Limitation’ value ‘Limited’. Multiple levels of ‘Individual Applicant Type’ may be allowed to apply for a funding opportunity, as funding opportunities could allow multiple career stages. The levels ‘Graduate’ and ‘Degree’ occur in a major part of the funding opportunities, while ‘Undergraduate’, ‘New Faculty’ and ‘Other’ occur in a range of 2% to 16% of the funding opportunities. These data descriptives prove useful for datasets creation in Chapter 6.

Chapter 5

Method

The related work revealed that BERT has succesfully been applied in similar IE domains as Named Entity Recognizer of Text Classifier. In this case study, BERT for Text Classification is the most suitable technique. After all, this study aims to classify funding opportunities based on their career stage and does not focus on entities or the relations between them.

Text classification is often used in the classification task(s) of a two-step or three-step approach, as found in Chapter 3. In this case study, a three-step approach is chosen over a two-step approach. As explained in chapter 3, this approach contains an additional classification step that classifies segments based on whether they contain information of interest. This is especially useful in this case study, due to the large number of segments and the small proportion of segments of interest. This imbalance was to be expected as the dataset includes all sentences of a funding opportunity and not only those specifically selected for the eligibility characteristics. This was confirmed by a random sample of 100 sentences which contained only six sentences with information on 'Individual Applicant Type'.

BERT for text classification will be used in both classification tasks of a three-step approach for this case study, similarly to work by Bhatia et. al [10]. The second step of the three-step approach is reflected in the sub-question: *How does BERT perform in determining whether segments contain information of interest compared to BiLSTM and Conventional Machine Learning approaches?*. The third step of the three-step approach is reflected in the sub-question: *How does BERT perform in determining the type of information in a segment of interest compared to BiLSTM and Conventional Machine Learning approaches?*. Each of the three-steps are explained in detail below.

5.1 Segmentation

Segments can be at different levels of detail, such as paragraphs, sentences, words depending on the task and data at hand [109]. The data used in this case study consists of the different pieces of original funding data text of a funding opportunity in Elsevier's database. The different pieces of original funding data text can be seen as segments themselves. How-

ever, as found in Chapter 4, the pieces of text vary in length from one sentence to multiple pages. Multiple pages of text are difficult for BERT to handle in the classification step, because BERT is trained to handle input segments of up to 512 tokens (words) [52]. Since this case study is focused on the applicability of BERT in the funding domain, these pieces of text need to be split into smaller segments.

Besides the technical limitations of BERT, certain properties of the data also provide reasons for further data segmentation. A piece of text in the funding opportunity database is not formatted, so syntactic information on where a new paragraph starts, based on headers or white lines, is not available in the text. Without further segmentation, this can create unintentional semantic interpretations. For example, consider this text part snippet: “The funding opportunity is open to young scientist who have a PhD in neuroscience. This also applies to undergraduate students and graduate students provided that they did not receive a PhD in the coming two months.” The second sentence seems to indicate that also undergraduate and graduate students may apply to the funding opportunity. However, this sentence was in fact from a different paragraph in the original funding opportunity text and undergraduate and graduate student were not allowed to apply.

These considerations resulted in the decision to split the pieces of text in small but still meaningful segments, namely sentences, similarly to Chantraporncha et al. [20] who used sentences to extract information from reviews. The drawback of the use of sentences as segments is that for the paragraphs in the text parts, the information (semantics) that is shared between sentences get lost. However, due to the structure and the lack of syntactic information on paragraphs in the data, this loss is inevitable.

The split of text parts into sentences is performed using the Sentence Tokenizer from the python package NLTK [84]. The NTLK package is often used for simple Natural Language Processing tasks, such as splitting text. The Sentence Tokenizer uses punctuation to split the text into sentences. Besides the split up, the text is also cleaned using the NLTK package. HTML tags and brackets are removed from the text using Regular Expressions. The creation of segments using the Sentence Tokenizer resulted in a dataset of 2.6 million sentences.

The sentence segments are used as input for the two classification tasks. The methods for the two classification step are shared in the coming two sections.

5.2 Segments that contain information of interest

The first classification step focuses on determining whether a segment contains information of interest. For this case study, the sentences of interest are the sentences that contain information on a limitation in the career stages of individual applicants. In other words, sentences that have the label ‘Limited’ as ‘Limitation’ value. This classification step is split in two classifi-

cation tasks, because distinguishing whether the sentence contains information on 'Individual Applicant Type' and distinguishing whether the sentence contains a limitation (should have value 'Limited') are two vastly different distinctions. These two classifications together obtain the sentences that have a limitation ('Limited') and are each summarized below.

1. *Selecting segments that contain information on 'Individual Applicant Type'*. - A binary classification task that determines whether a segment contains information on 'Individual Applicant Type' or not. It splits the data into the classes **Specified** (there is information in the segment on 'Individual Applicant Type') and **Not Specified** (no information in the segment on 'Individual Applicant Type').

2. *Selecting segments that contain information on a limitation*. - A binary classification task that determines whether a segment which contains information on 'Individual Applicant Type' (specified segments) also contain information on a limitation who can apply. It splits the data into the classes **Not Limited** (the segment contains information that there is no limitation in who can apply, so applicants in all career stages can apply) and **Limited** (the segment contains information that there is a limitation in the career stages who can apply).

5.3 Career stages that may apply

The second classification task focuses on the type of information in a segment of interest. The segments of interest are the segments that contain a limitation (value is 'Limited'). For these segments the career stages (the different levels of 'Individual Applicant Type') that may apply according to the segment are the different types of information in the segment. To obtain the information on what career stages may apply, similarly again, a classification task is deployed. The classification task is explained below.

3. *Determining the career stages that may apply* - For segments that are classified as having a limitation, the career stages of individual applicants that may apply according to the segment need to be obtained. This is formalized as a multilabel multiclass classification task, with the different levels of career stages as labels: **'Undergraduate'**, **'Graduate'**, **'New Faculty'**, and **'Degree'**. The 'Other' class was disregarded after data annotation due to misuse of this class as explained in Chapter 6.

An overview of the classification tasks and their relations can be found in Figure 5.1.

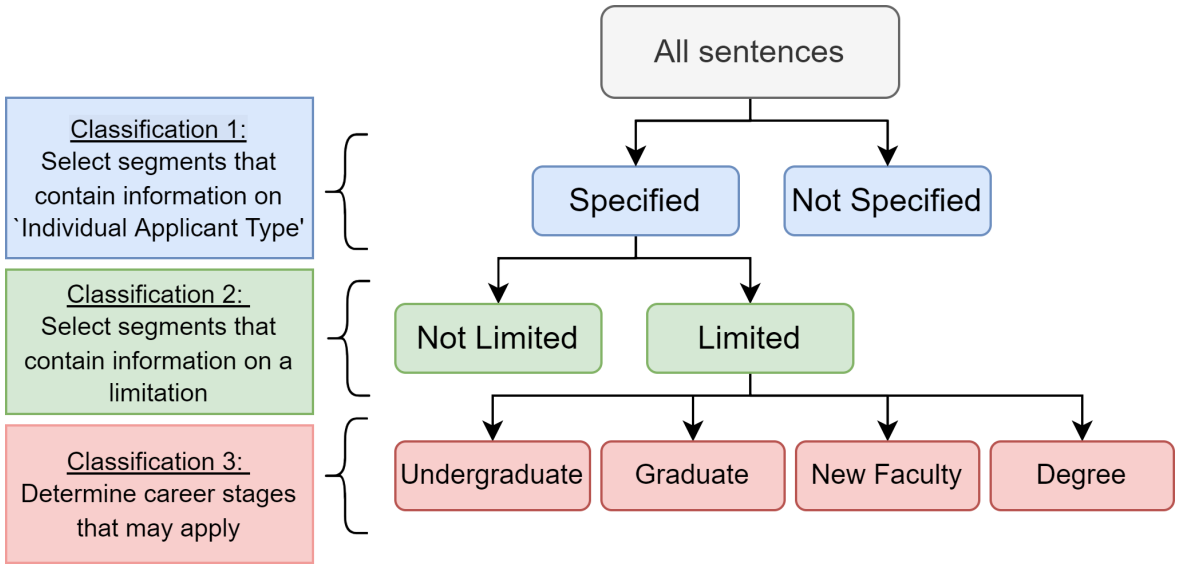

Figure 5.1: Visualization of the three classification tasks with their classes

5.4 From segment level to funding opportunity level

The three classification tasks together form an output at sentence level (for each segment). However, the 'Individual Applicant Type' characteristic is available in the database at funding opportunity level. Therefore, an output at funding opportunity level is required. Tackling this task contributes towards the third sub-question: *How well does the three-step approach with method BERT perform at funding opportunity level?*. To obtain information at funding opportunity level, the information of all sentences are gathered and used to determine the values at funding opportunity level. Determination of the values at funding opportunity level is performed using the two algorithms below. An overview of the full system at funding opportunity level is visualised in Figure 5.2.

Algorithm 1 determines limitation level with possible values:
'Not Specified', 'Not Limited', 'Limited'

Require: Q = list(limitation values of all sentences of a funding opportunity)
#If all limitation values are the same
if length of unique(Q) == 1 **then**
limitation = Q[0]
else if length of unique(Q) >1 **then**
if Limited **in** Q **and** Not Limited **not in** Q **then**
limitation = Limited
else if Not Limited **in** Q **and** Limited **not in** Q **then**
limitation = Not Limited
else
limitation = limitation value ('Not Limited' vs. 'Limited') that occurs most often in Q

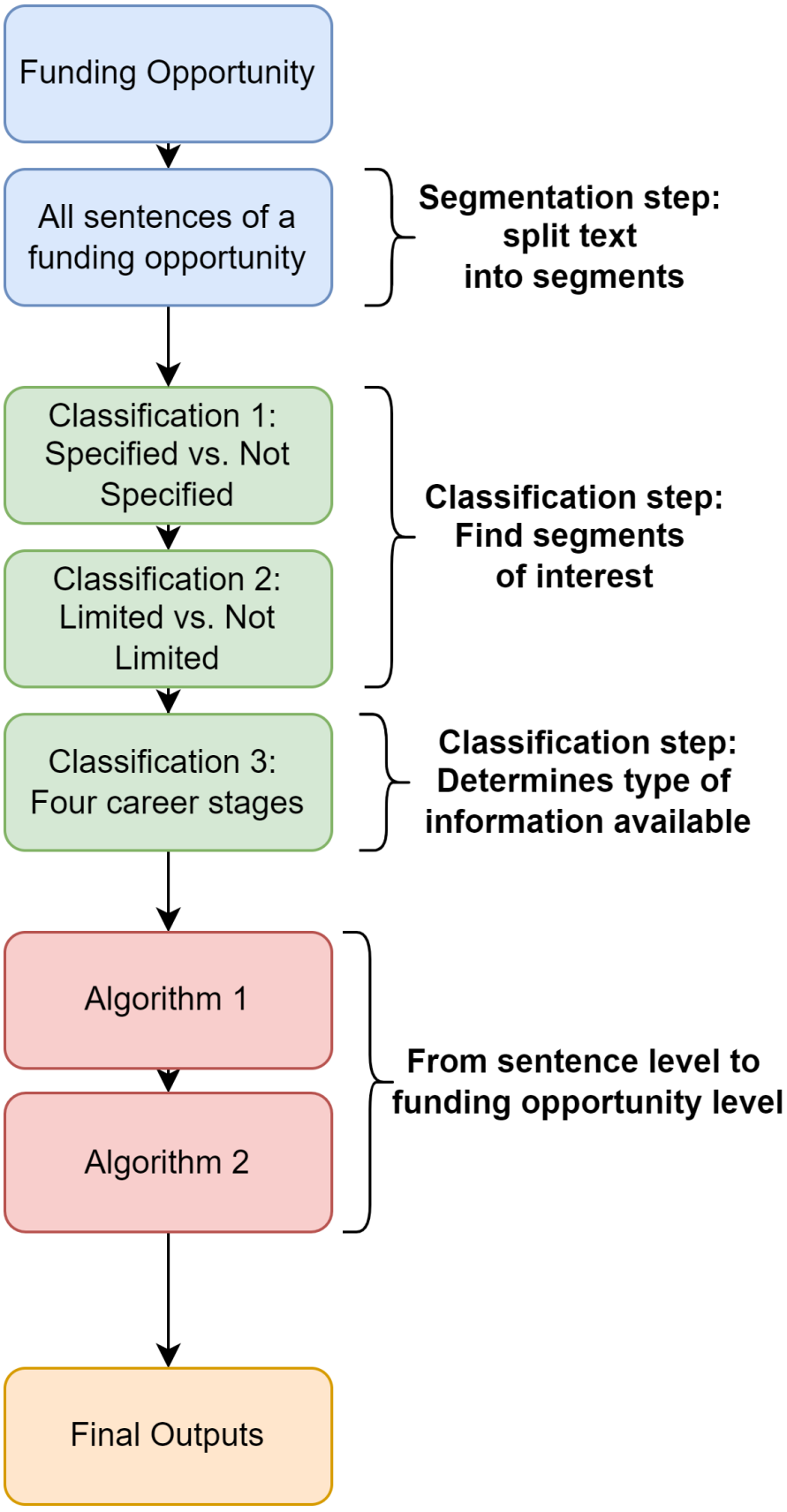

Figure 5.2: System at funding opportunity level

Algorithm 1 determines the limitation value at funding opportunity level based on all limitation values of all sentences of a funding opportunity. The 'Limited' and 'Not Limited' values are prioritized above 'Not Specified' values, because these indicate that there is information on 'Individual Applicant Type' in the funding opportunity available. A majority voting between the values 'Not Limited' and 'Limited' determines the final output value. In case only 'Not Specified' values are present, the funding opportunity is determined as 'Not Specified'. To the funding opportunities that are predicted to have a 'Limited' limitation, algorithm 2 is applied to obtain final values for the different career stages that may apply to that funding opportunity. All career stage values from all sentences of a funding opportunity are gathered in a list. If the list has at least one sentence that states the level is allowed to apply (indicated by 1), the level value is given the value allowed to apply (level value= 1), otherwise the level value is set to not allowed to apply (level value= 0). This process ensures that if the information is stated in one sentences, the information is received and added to the information at funding opportunity level.

5.5 Three models

As explained in Chapter 3, BERT is often combined with other models including BiLSTM, CNN, and CRF. However, to test the applicability of BERT in isolation, in this case study BERT is applied on its own. To evaluate how well BERT is performing, BERT is compared to BiLSTM models, which were commonly used before transformer-based models were introduced [135, 18]. In literature it was found that, just like BERT, BiLSTM is often combined with CRF, CNN, or another type of model (e.g. ese [18]). However, similar to BERT, a plain BiLSTM model is chosen for simplicity. Besides the BiLSTM neural network, the BERT model is also evaluated against a conventional machine learning approach. Conventional machine learning models have a lower space and time complexity when compared to deep neural network models such as BERT and BiLSTM. In case conventional machine learning models can achieve comparable results to the deep neural network approaches, the conventional machine learning approaches will be preferred as they are less resource intensive. This preference is also used in work by Goa et al [39] who evaluated their hierarchical neural network against both deep learning models and conventional machine learning approaches. For simplicity, simple conventional machine learning approaches are chosen, which is a Logistic Regression model for binary classification task and a Random Forest model for the multilabel classification task.

The implementations in this case study can all be divided into two parts: 1) the knowledge part, which converts the natural language into features (embeddings) 2) the model part, which uses these features to classify the segments. In conventional machine learning approaches this knowledge part (feature extraction) is separate from the model part. However, in neural network implementations, such as BERT and BiLSTM, the feature extraction is part of the model itself [85]. BERT uses its internal knowledge on which it was trained to convert natural

language to features. The other models use GloVe to create features. The BiLSTM uses GloVe knowledge as part of the model in the Embedding layer. In contrast, the conventional ML models use GloVe as a separate module in the implementation. A visualization of the implementations of the models in this case study can be found in Figure 5.3.

GloVe was a common approach [92] before the introduction of deep contextualized embeddings, such as BERT [73]. GloVe embeddings are pre-trained word vectors that were trained on Wikipedia 2014 and GigaWord5 [116]. Specifically BERT embeddings were not used in this study for the baseline implementations, to fully test BERT on its own by comparing a full BERT implementation to implementations that do not use BERT in any form. Furthermore, BiLSTM embeddings were not used as they require a large dataset to be trained. Unlike BERT and Glove, BiLSTM embeddings are not pretrained.

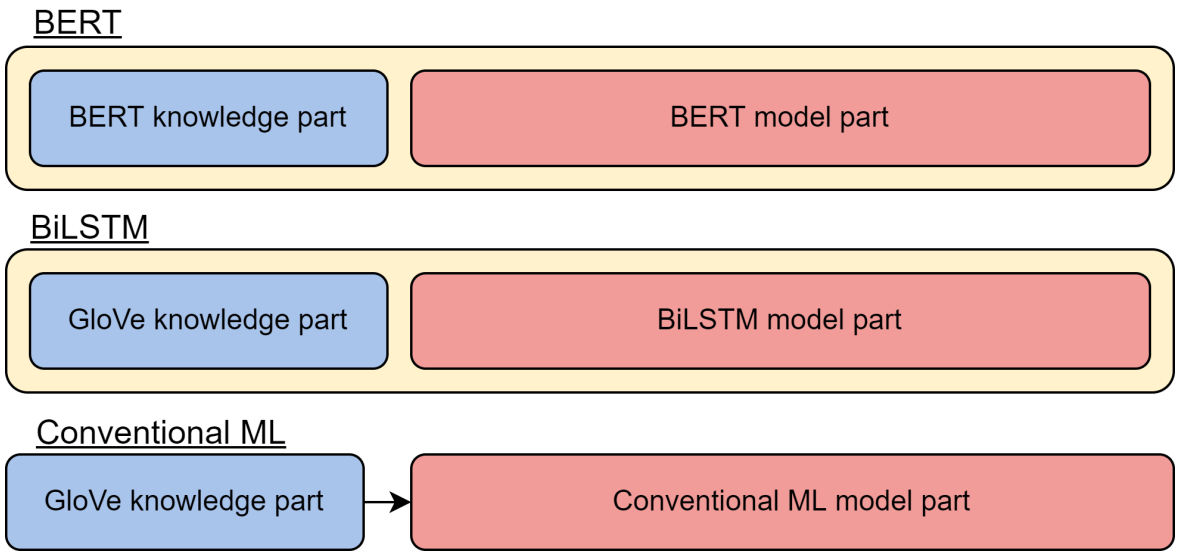

Figure 5.3: The knowledge parts and model parts of the three implementations tested in this case study

5.6 Datasets

As outlined earlier in this Chapter, the models perform three classification tasks. Each of these three clasification tasks requires a training dataset to train and an evaluation dataset to evaluate the performance of the models. Besides the three classification tasks, the whole system at funding opportunity level needs to be evaluated as well, which requires an evaluation dataset. The creation of the training and evaluation datasets is described in detail in Chapter 6. The created datasets are small ($<$1000 items per class) to minimize the data annotation. The small datasets are used both for training and evaluation. Because of the size of the datasets splitting into a training and evaluation dataset is costly. Preferably all data would

be used for training. However, the model needs to be evaluated as well. An easy solution to utilize the data both for training and testing, while keeping the training and test set separate is by applying k-fold cross validation. K-fold cross validation splits the dataset into K parts, and K-1 parts are used for training and one part is used for evaluation [98]. The process is repeated K times until every part has been used once for evaluation, which results in K evaluation outputs. The average of the K evaluation outputs is the final output of the model. A 5-fold cross validation is chosen over a standard 10-fold cross validation to limit the required computational power and have a sufficiently large evaluation set.

5.7 Evaluation

The evaluation process of the three classification tasks and the system at funding opportunity level consist of comparing the predicted labels to the actual labels in the evaluation dataset for all three models: BERT, BiLSTM, and Conventional ML. This results in situations where the data is correctly classified (either true negatives or true positives), and incorrectly classified (either false positives or false negatives). These four values are summarized in a confusion matrix. An example of a confusion matrix with the definitions is visualised in Figure 5.4. These four values are used to calculate the metric values to evaluate the performance of the classifier. Common performance metrics in Machine Learning are the precision, recall, and their harmonic mean the f1-score. Their definitions and formulas are discussed below.

- *Precision* is the proportion of results that are relevant and is calculated using the following formula:

$$\frac{TP}{TP + FP}$$

- *Recall* is the proportion of relevant results that are correctly classified by the model and is calculated using the following formula:

$$\frac{TP}{TP + FN}$$

- *F1-score* is the harmonic mean of precision and recall and is calculated using the formula:

$$2 * \frac{Precision * Recall}{Precision + Recall}$$

		Actual	
		Positive	**Negative**
Predicted	**Positive**	**True positives (TP)**: #items that are classified as class positive and are also positive	**False Positives (FP)**: #items that are classified as class positive, while actually belonging to class negative
	Negative	**False Negatives (FN)**: #items that are classified as class negative, while actually belonging to class positive	**True Negatives (TN):** #items that are classified as class negative and are also negative

Figure 5.4: Example Confusion Matrix with the definitions of FP, FN, TN, TP

These three metrics: precision, recall, and f1-score provide a way to compare the performance of BERT with BiLSTM and Conventional ML and are thus a key ingredient to answer the second sub-research question: ”How well can BERT extract information from funding opportunities compared to BiLSTM and Conventional Machine Learning models?” and its sub-questions. All three models: BERT, BiLSTM, and Conventional ML are compared based on these three metrics for all four evaluations. In the multilabel classification tasks (third classification task and system at funding opportunity level), the classes of career stages are separately evaluated using the three metrics. To obtain metric values over the whole task, the weighted average is taken based on their class frequency.

The precision and recall performance metrics are trade-off metrics. The higher the recall (low false negatives), the lower the precision (high false positives) and vice versa. For this case study, achieving low false positives (high precision) or low false negatives (high recall) is equally important. For example, for the first classification task, assigning the value ‘Specified’ (positive class) while the actual value is ‘Not Specified’ would result in the assignment of incorrect labels in the second and third classification task. The other way around, assigning ‘Not Specified’ while the actual label is ‘Specified’ results in information that is missed for that funding opportunity. Missing information (false negative) as well as assigning the wrong values at sentence level (false positive) both have an impact on the labels at funding opportunity level. It is difficult to state whether one of the two has a larger negative impact, as that depends on the values of the other sentences as well. The same holds for the second and third classification task.

At funding opportunity level, achieving low false positives (high precision) and low false negatives (high recall) are equally important as well. If a funding opportunity is assigned a career stage, while the funding opportunity is not open to that career stage, the funding opportunity incorrectly ends up in searches of Elsevier's customers. On the other hand, if information on a career stage is missed, the funding opportunity will not be shown to potential applicants. Because of the fact that achieving low false positives (high precision) and low false negatives (recall) are equally important for this case study, the most important performance metric is the f1-score which weighs precision and recall equally by taking the harmonic mean. The f1-score is chosen over the well-known accuracy metric, because the f1-score is less affected by the class imbalance that exists in a multilabel classification task [53].

Besides the three performance metrics, a resource metric is used in all four evaluations to determine the feasibility of deploying the models. The resource metric that is used for the three classification tasks (two classification steps) is the time taken by the model to train and evaluate over 5-fold cross validation combined with the required computational power. For the evaluation of the systems at funding opportunity level, instead of the training and evaluation time metric, the inference time, the time taken in seconds to predict labels for a funding opportunity, is used. Similar to the training and evaluation time of the three classification tasks, the time is combined with computational power used. Details on the four evaluations and their three implementations can be found in Chapter 7.

Chapter 6

Creation of Training and Evaluation Datasets

A training or evaluation dataset consists of segments and their corresponding labels. These segments and labels are used as examples by machine learning algorithms to learn from or as ground truth against which the predicted labels (values) can be evaluated. The creation and labelling of training and evaluation dataset at segment level (sentence level) in section 6.1 of this Chapter. Furthermore, the creation and labelling of the evaluation dataset at funding opportunity level is discussed in section 6.2. As discussed in the introduction of this report, neural network approaches such as BERT require labelled data, which can be labor-intensive to create. Therefore, a creation of training and evaluation datasets using minimized data annotation is preferred. In this chapter, multiple steps to minimize data annotation are taken that together answer the research question: *How to acquire balanced labelled datasets of sufficient quantity and quality with minimized annotation?* Section 6.4 summarizes the answer to this research question.

6.1 Training and Evaluation Dataset at Sentence level

For all three classification tasks at sentence level, a single dataset is created. This dataset contains labels for all three classification as can be found in Figure 5.1. Sentence segments for the dataset are available in abundance: 2.6 million sentence segments. However, the corresponding labels for these segments are not available. The process of manually giving the sentence segments their labels is called annotation. However, annotating all 2.6 million sentences is a time-consuming process and therefore not feasible. Annotating a subset of the dataset of a few hundred to a few thousand sentences is feasible. A random subset of the dataset would result in an unbalanced dataset for the first classification task, because there are more sentences in the dataset that do not contain information on ‘Individual Applicant Type’ than sentences that do. This intuitively makes sense as the dataset contains text snippets of all characteristics as explained in Chapter 4. Furthermore, this was also shown through a manual evaluation of the data. In a random sample of 100 sentences only six sentences were found that contained information on ‘Individual Applicant Type’. Models that are

trained on an unbalanced dataset are biased towards the over represented class. That is, when a new example is encountered by the model, the model would more likely consider it to be of the over represented class than the underrepresented class. In our situation, that would result in a bias towards sentences that do not contain information on 'Individual Applicant Type' in classification task 1.

Furthermore, the large imbalance for classification task 1 has an effect on the size of the datasets of the second and third classification. These datasets only consider sentences that do contain information on 'Individual Applicant Type' as visualised in Figure 5.1 of Chapter 5. This results in additional data annotation to get datasets of sufficient quantity for the second and third classification task.

To avoid a bias for the first classification task and additional data annotation to get datasets of sufficient quantity for the second and third classification task, the dataset needs to be balanced. Balancing a dataset is often performed after data annotation by either oversampling or undersampling. However, in this case the imbalance is large: 6 to 94. Applying oversampling techniques to create 88 new instances from the existing 6 would result in overfitting. Furthermore, combining oversampling techniques with cross-validation gives a distorted view of performance because cross-validation assumes independence between folds and applying oversampling creates new data in folds which are dependent on data from existing folds. The overfitting and distorted view of performance makes oversampling unsatisfactory. However, applying undersampling is also undesirable as it requires a large amount of data annotation to have a dataset of sufficient quantity.

To minimize the amount of data annotation, the dataset is, therefore, balanced before data annotation. This is performed by selecting sentences that are likely to contain information on 'Individual Applicant Type'. The whole process of balancing the dataset before data annotation is discussed in section 6.1.1. The annotation of the more balanced dataset and the techniques applied to minimize data annotation are discussed in section 6.1.2.

6.1.1 Create balanced dataset

In order to obtain a balanced dataset, sentences that are likely to contain information on 'Individual Applicant Type' need to be selected and added to the random sample. Selecting these sentences was done by grouping sentences with similar semantics. For each of the groups it is determined whether it is likely that the sentences in the group contain information on 'Individual Applicant Type'. The groups of sentences that are likely to contain information on 'Individual Appplicant Type' are selected and added to the the random sample. This process is split into three parts: 1) create sentence embeddings 2) group sentence embeddings 3) determine which groups contain information on 'Individual Applicant Type' based on the keyword ratio. Each of these steps will be discussed in more detail below.

Sentence Embeddings

A sentence embedding is a vector of real numbers that represents a sentence. Sentence embeddings are used mainly by machine learning algorithms to process textual information. The vector representations represent the semantics of the sentence and have the property that vectors (sentences) with similar semantics are closer together in the vector space. This property allows for easily grouping of embeddings with similar semantics and allows for separation of groups with a similar semantics, such as information on 'Individual Applicant Type'.

The creation of sentence embeddings is a task for which a great understanding of natural language is required. BERT has outperformed many natural language understanding tasks, including creating sentence embeddings. Reimers & Gurevych [99] evaluated different kinds of sentence embeddings, including the common used GloVe embeddings, Infersent embeddings, fast-text embeddings, and different kinds of BERT embeddings on seven different sentence classification tasks. Their results show that the BERT embeddings outperform the other approaches on most of the tasks. Especially, the new BERT approach (SBERT) presented in their paper [99], shows state-of-the-art results in 6 out of the 7 tasks. Unfortuantely, SBERT cannot deployed in parallel and thus is no feasible option for a datsaet of 2.6 million sentences. Instead, a Spark implementation, that can be easily deployed in parallel on a Spark Computer Cluster, is preferred. John Snow Lab [58] offers a Spark BERT sentence embedding implementation, called 'BertSentenceEmbeddings.pretrained()' and uses by default a BERT model that was fine-tuned on sentence embeddings, called 'sent_small_bert_L2_768' BERT model. The model outputs a 768 dimensional vector (sentence embedding) for each sentence. These embeddings are used in the next step to group sentence embeddings based on semantic similarity.

Cluster Sentence Embeddings

A set of unsupervised methods that group data points based on their proximity in vector space is called clustering. As Surya Priy [95] defines it: "Clustering is an approach that divides the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups". In literature many different kinds of clustering algorithms are presented together with their advantages and disadvantages [129]. Google Developers identified four groups as the most common approaches [33].

1. *Distribution-based algorithms* group data points together based on the likelihood of belonging to the same probability distribution (Gaussian, Binomial etc.).

2. *Centroid-based algorithms*, present data points as vectors and group data points together based on the proximity of their vector.

3. *Hierarchical Clustering algorithms* use one of two approaches: 1) bottom up: consider each data point as a cluster and group similar clusters together 2) top down: consider all

data points as one cluster and divide the most dissimilar data points into new clusters. At each grouping or division, the similarity and dissimilarity of clusters is determined, based on these values, the optimal set of clusters is determined.

4. *Density-based algorithms* use the density of data points, where clusters are dense areas separated by less dense areas.

Each of these groups of algorithms have advantages and disadvantages. For this case study, the clustering needs to be scalable to a large dataset. Furthermore, it is beneficial to have an algorithm that does not require ufpront specification of the amount of clusters and their shape. Due to the high dimensionality of sentence embedding vectors these values would be hard to estimate from a visualization. Another aspect that is beneficial for a clustering algorithm of natural language data is outlier handling. Natural language data is quite diverse, which can result in sentences that do not belong to a cluster. These requirements rule out both distribution-based algorithms, which require the shape to be specified, and centroid-based algorithms, which require the number of clusters to be set upfront [129]. Of the two remaining groups, density-based algorithms are preferred over hierarchical clustering algorithms, because density-based algorithms have a better scalability. Especially DBSCAN [129] and its successor HDBSCAN give great results on large datasets. An additional benefit of these algorithms is the fact that they can easily handle outliers. Data points outside the dense areas are not seen as a cluster, but are seen as outliers.

How do DBSCAN and HDBSCAN work? DBSCAN transforms the space according to the density of the data [15]. Data points in dense areas are kept in the same position, while data points in the sparse areas are positioned further away. In this new space, bottom-up hierarchical clustering is performed, which results in a dendogram (tree) of clusters. The dendogram is split at a specific point, based on the epsilon distance parameter. The point at which the dendogram is cut is horizontal and results in clusters with equal density levels. Any cluster with one data point is marked as ‘noise’ and left unclustered. In this approach the episol value needs to be set upfront, which can be quite difficult. Furthermore, it results in clusters with the same density, but clusters with varying densities that migh better reflect the actual structure of the data. The authors of DBSCAN identified this and created a different approach (HDBSCAN) [15] to cut the dendogram. Instead of epsilon, the persistence of clusters is used as indicator. The longer a cluster remains unchanged, the more coherent the cluster is. Using persistence of clusters instead of epsilon results in clusters of varying denseities and avoids the upfront specification that epsilon requires. These benefits resulted in the decision to deploy the HDBSCAN as clustering algorithm in this case study.

The HDBSCAN algorithm has a few parameters that are required to be set upfront. These parameters and their definitions are listed below:

- *Minimum cluster size* determines the minimum number of data points that have to be grouped together to be considered a cluster.

- *Minimum samples* determines the amount of noise (outliers). The larger the value of minimum samples, the more dense the areas have to be in order to consider it a cluster, resulting in more noise (outliers).

- *Cluster selection method* determines the method by which clusters are selected from the tree cluster hierarchy. Two different methods can be used: 'leaf' or 'eom'. 'Eom' (Excess of Mass) has the tendency to select one or two large clusters and some smaller clusters, while 'leaf' select small homogeneous clusters.

As a first step in clustering, a trial experiment was set up. For this trial experiment a dataset of 100.000 sentences was randomly selected out of the 2.6 million. The parameters were set as follows: 1) minimum samples parameter was set to 2 (low value), so that most data was not seen as noise 2) the cluster selection method was set to default 'eom' 3) two different values for minimum cluster size were set, 15 and 80. Remarkably, the results of the trial experiment showed that most of the data was seen as noise. As visualised in Table 6.1, the noise was respectively 78.7% and 87.3% of the data. Although it was expected that the data contained some noise, since it is natural language data, such a high percentage of noise was not expected.

Minimum Cluster Size	**Number of Clusters**	**Amount of Noise**
15	242	78.7%
80	118	87.3%

Table 6.1: Initial findings on clustering HDBSCAN

The documentation on HDBSCAN revealed two possible explanations for the high amount of noise [46]. The first explanation revolves around the parameter minimum samples that heavily affects the amount of noise. In case the minimum number of samples was set too high, the majority of the data would be considered as noise. This explanation was not valid for this trial experiment, since the minimum number of samples was set to its lowest value (min samples=2). The second explanation had to do with the dimensionality of the data that is clustered. As explained in [46], HDBSCAN gives good performances for data dimensionality of 50 or 100, but above a dimensionality of 100 the performance significantly decreases. Our data has a dimensionality of 768, which is way above the recommended data dimensionality. Therefore, a dimensionality reduction method had to be applied to reduce the dimensionality of our input data. Until 2018, t-SNE was the most commonly used dimensionality reduction method, but since 2018 UMAP took that position [80]. Although UMAP performance is a little bit lower ($<$1%), its lower time complexity and high scalability to large datasets made it popular. As presented in [3], UMAP improves clustering performances of various clustering algorithms, including HDBSCAN. These considerations were the deciding factor to apply UMAP as a pre-processing step for clustering.

Minimum Cluster Size	Amount of Clusters	Amount of Noise
15	1238	48.5%
75	186	59.4%

Table 6.2: Results of initial clustering using UMAP with a dimensionality of 50

As can be seen from Table 6.2, the results of the clustering with UMAP show a significant decrease in the amount of noise compared to clustering without dimensionality reduction. Interestingly, the table also shows that the dimensionality reduction affects the amount of clusters as well. For instance, the amount of clusters for minimum cluster size of 15 is multiplied with factor 5. This can easily be explained, as sentences that were considered noise in the previous trial experiment can now have formed new clusters or have joined existing clusters. Especially for the smaller minimum cluster sizes. In that configuration, the model prefers creating new clusters above merging existing clusters. The reverse is true for large minimum cluster sizes.

For UMAP there are also some parameters to be set upfront [122]:

- *Number of neighbors* determines how global or local the UMAP focuses on. This influences, for example, the amount of data points that are glued together.
- *Minimum distance* determines how closely packed the data points are allowed to be after reduction. Low values of minimum distance are great for clustering purposes. Therefore, minimum distance is set of 0.0.
- *Number of components* specifies the dimensionality of the data after dimensionality reduction.
- *Metric* determines how the distance is calculated of the input data. Distance metrics explored were canberra, cosine and euclidean distance.

Different combinations of UMAP and HDBSCAN parameters were tested. The UMAP parameter values that were tested: number of neighbors [15, 50, 100, 200], number of components [2, 10, 50], and metrics [canberra, cosine, euclidean]. The HDBSCAN parameter values that were tested were minimum cluster size [5, 15, 30, 50, 100, 200, 300], the minimum samples [2, 4, 15, 30, 50, 100], and the cluster selection method [‘eom’ or ‘leaf’]. Different combinations of parameters were tested using a grid search. The optimal set of parameters was determined using the relative validity index of HDBSCAN, which is a measure based on the euclidean distance between clusters. The larger the distance between clusters, the more cohesive the clusters are expected to be [51]. Due to limited computational power, the clustering was performed on a random sample of 100.000 sentence embeddings.

The optimal set of parameters for UMAP was: number of neighbors of 15, number of components of 2, and the ‘cosine’ metric. The optimal HDBSCAN parameters were a minimum

cluster size of 200, minimum samples of 4, and the default cluster selection method 'eom'. These combination of parameters achieved a relative validity index of 0.8232. The full results can be found in Appendix B. The obtained clusters of sentences are used in the next step, where the keyword ratio is used to determine which clusters are likely to contain information on 'Individual Applicant Type'.

Keyword Ratio

The third step is determining which clusters are likely to contain information on 'Individual Applicant Type'. The sentences in the clusters which are likely to contain information on 'Individual Applicant Type' are selected in combination with a few clusters that are less likely to contain information on 'Individual Applicant Type' to obtain a balanced dataset. The ranking of clusters on the likeliness of containing information on 'Individual Applicant Type' is performed using gazetteers (keywords).

Gazetteer list: [applicant, eligible, student, scholar, postdoc, senior, undergraduate, graduate, degree, postdoctoral, junior, medior, senior, researcher, early, phd, artist, investigator, unrestricted, masters, bachelor, experienced, bsc, msc, career]

For each cluster the ratio of words in the cluster that are in the Gazetteer list was determined, hereafter referred to as 'the ratio'. The higher the ratio of a cluster, the more likely that the sentences in the cluster contain information on 'Individual Applicant Type'. The use of a gazetteer list is a simple approach which does not require detailed knowledge about the sentences and gives an indication on what the cluster comprises. The top 400 clusters with a keyword ratio above 0.04 were combined with the lowest 300 clusters, so the dataset contained both sentences that are likely to contain information on 'Individual Applicant Type' (top clusters) and sentences that are not likely to contain information on 'Individual Applicant Type'. This results in a more balanced dataset of 4.983 sentences that can be annotated. The process of annotation is discussed in the section below.

6.1.2 Annotation of balanced dataset

Annotation is the process of giving a data point a label. The label is a representation of what class the data point belongs to and helps to train or evaluate a ML algorithm. Unfortunately however, the process of data annotation is labor-intensive. One way to minimize data annotation is by simply labelling a small amount of data points, resulting a small dataset ($<$1500 data points per class). As found in literature, small datasets can outperform a larger dataset as long as the quality of the small dataset is high compared to the larger dataset [63, 62]. In this case study, experts within Elsevier, who work on a daily basis with the funding data, were asked to provide labels for a subset of the 4.983 sentences. In total 9 different experts were recruited who together labelled a dataset of 1990 sentences[1]. Different experts contributed

[1] Details on the Data Annotation task are presented in Appendix C

a different amount of labelled sentences, ranging from 50 sentences to a few hundred sentences. The annotation of 60 sentences took an annotator approximately half an hour and three annotators reported that the task was rather simple. Each sentence was labelled by one expert. To verify the quality of the annotations, 50 sentences were annotated by multiple annotators (n>3). These sentences were only used to calculate the inter-annotator agreement which determines the quality of the dataset. The quality assessment is discussed below the header 'Inter-Annotator Agreeemnt'.

The annotation process went as follows. The experts were presented with a sentence and asked whether the sentence contained information on 'Individual Applicant Type' or not. The possible labels the experts could assign were: 1) Limited 2) Not Limited 3) Not Specified or 4) Organization. The values: 'Limited' and 'Not Limited' state that there is information on 'Individual Applicant Type'. 'Limited' is assigned to sentences that specifically state which groups of applicants may apply based on 'Individual Applicant Type', whereas 'Not Limited' is assigned when it specifies that anyone can apply based on 'Individual Applicant Type'. The value 'Not Specified' indicates that there is no information on 'Individual Applicant Type' in the sentence. The fourth value is a special case. As discussed in Chapter 4 of this report, 'Individual Applicant Type' is only available for funding opportunities that allow individuals to apply. In case the sentence indicates that only Organizations can apply, the 'Organization' value is given and the sentence is discarded from the dataset. The sentences annotated with 'Limitation' excluding the sentences with label 'Organization' amounts to 1877 sentences.

In case the sentence was labelled as 'Limited', the same expert who labelled the value 'Limited' was asked to specify what 'Limitation' is set. In other words, which applicant type(s) may apply to the funding opportunity? The possible labels the expert could assign were the different (career stages) levels of 'Individual Applicant Type': 'Undergraduate', 'Graduate', 'New Faculty', 'Degree', and 'Other'. For each sentence multiple labels could be assigned. For example, a sentence might specify that both 'Undergraduate' and 'Graduate' students may apply.

After manual evaluation of the annotations, it was found that the 'Other' class was misinterpreted by multiple experts. The experts used the 'Other' class for sentences that contained information on 'Eligibility' in general, instead of specific for 'Individual Applicant Type'. For example, the sentence: "An ideal candidate will have experience in biological data analyses.", specifies information on 'Eligibility'. However, it does not specify information on the type of career stage an individual is required to be in. Therefore, the 'Other' class was disregarded from the dataset. The sentences that only had the class 'Other' as limitation were removed from the dataset, resulting in a dataset of 1697 sentences. The detailed annotation process is described in Appendix C.1.

6.1.3 Imbalance for classification task 2

The annotated dataset consists of 1697 sentences, where 779 sentences were labelled as 'Limited', 39 sentences were labelled as 'Not Limited' and 818 sentences were labelled as 'Not Specified'.

Class	**Number of sentences**
Limited	779
Not Limited	39
Not Specified	818

Table 6.3: Distribution of the 'Limitation' values in the dataset for the three classification tasks

The dataset contains only 39 'unlimited' sentences. This creates a large unbalance for the second classification task where 'Limited' and 'Not Limited' sentences are distinguished. To create a more balanced dataset, undersampling was explored by undersampling to 39 items for each class. However, this resulted in predictions that were close to chance, because of the small dataset size. Therefore, an oversampling technique was used, namely data augmentation. Data augmentation is the creation of new data points by slightly altering the available data points and present them as new data points. Different augmentation techniques from the Python library nlpaug [83] were combined. The functions used are listed below.

- SynonymAug uses WordNet [81] to substitute a word in the sentence which results in a new sentence.
- ContextualWordEmbdsAug insert uses the bert-base-uncased model to insert a word into the sentence which results in a new sentence.
- ContextualWordEmbdsAug substitute uses the bert-base-uncased model to substitute a word in the sentence which results in a new sentence.
- BackTranslationAug uses two fine-tuned BERT models called 'facebook/wmt19-en-de' and 'facebook/wmt19-de-en', which translate the sentence to another language (German) and translate it back to English. The newly created English sentence is a new sentence that slightly differs from the original sentence.

The quality of the 'Not Limited' sentences was manually checked and it was found that half of the sentences did not contain information on 'Individual Applicant Type', but rather other types of eligibility, such as the citizenship of an individual applicant. In case these sentences would have been used for data augmentation, it would have created new data points that provide information on other types of eligibility. On the 15 sentences that are of high quality the data augmentation functions were applied. Each sentence produces four new sentences using the four techniques, resulting in 60 augmented sentences. The total amount of sentences with the class 'Not Limited' was increased to 99 sentences.

For the sentences with the limitation value: 'Limited', the different levels of 'Individual Applicant Type' (career stages that may apply) were assigned. The distribution of the different levels are visualised in Table 6.4. These classes are not balanced, because re-sampling the data so that the data is balanced is very difficult because of the class dependencies present in multilabel data. However, the fact that each class is represented in at least 120 sentences ensures that each class is represented sufficiently.

Level of 'Individual Applicant Type'	**Number of sentences**
Undergraduate	120
Graduate	406
New Faculty	210
Degree	319

Table 6.4: The distribution of levels of 'Individual Applicant Type'

6.1.4 Quality assessment annotations at sentence level

To verify the quality of the annotations, the 50 sentences that were annotated by multiple annotators (n>3) were used to calculate the inter-annotator agreement. The most well-known annotator agreement statistic is the Cohen's Kappa [25]. The drawback of this approach is that it is limited to two annotators who annotate all subjects. Fleiss et al. [38] identified this problem and presented the Fleiss Kappa statistic which allows for multiple annotators. A random sample of n annotations are selected per sentence (subject) and the agreement between these n annotators is determined as follows:

$$P = \frac{1}{Nn(n-1)}(\sum_{i=1}^{N}\sum_{j=1}^{k} n_{ij}^2 - Nn), where$$

N = number of subjects
n = number of randomly selected annotations
k = number of possible labels

The output value is a float value between -1 and +1. A fleis kappa value of -1 indicates that there is no agreement observed at all, 0 indicates that there is no better than chance, and +1 indicates that there is a perfect agreement. A fleiss kappa value between 0.2 and 0.4 is considered as fair, and between 0.4 and 0.6 as moderate. A kappa above 0.6 is considered as good.

For the annotation of the label 'Limitation', a total of 50 subjects (sentences) were used (N=50). For each of these fifty sentences, a random sample of three annotations were selected and compared (n=3) with a number of possible labels of four (k=4). The output showed

an agreement of 0.515 for the label ‘Limitation’, which is a moderate agreement.

The label ‘Limitation’ did not achieve a good agreement (an agreement above 0.6), because of three main factors. One of the main factors was the misinterpretation of the label ‘Limitation’. In the daily processes of Elsevier, the label ‘Limitation’ is used as a limitation for all kinds of Individual Eligibility rather than as a limitation for the specific Individual Elgibility of characteristic ‘Individual Applicant Type’. Although this change in meaning of the term was stressed in the annotation protocol, it was sometimes not interpreted correctly as found by manually evaluating the annotations. For example, in the sentence: *“Individuals living or working within a 50 mile radius of the conference location are eligible to receive the conference registration fee only”.* Half of the annotators that annotated this sentence, marked this sentence as ‘Limited’. Although it contains information on individual eligibility, it does not contain information on the different career stages (‘Individual Applicant Type’) an individual applicant is required to be in.

The second factor is the role of natural language data that can be interpreted in multiple ways. For example, the sentence: “*Applicants undertaking a masters or a PhD must also include a separate statement which provides information on how the project applied for differs from their masters or PhD work.”*, was interpreted by two out of the seven annotators as ‘Not Specified’, while the remaining annotators thought it contained information on ‘Individual Applicant Type’ with a specific limitation (value ‘Limited’).

The third factor has to do with the reading comprehension of the annotators. For example, the sentence: *“Upper second class honours (2:1) degree from a uk institution (or overseas award deemed equivalent via uk naric.”*, has the word institution in it and the sentence is assigned the value ‘Organization’, to indicate that only organizations can apply. Most of the annotators read it correctly, but a a few annotators did not, which decreased the agreement.

Besides for ‘Limitation’, the inter-annotator agreement was also calculated for the levels of ‘Individual Applicant Type’ (career stages) using the Fleiss Kappa formula. However, instead of the four possible labels, two possible labels could be assigned for these annotations (k=2). The results of the calculations are shown in Table 6.5. The inter-annotator agreement of ‘Undergraduate’ and ‘New Faculty’ show a value close to chance, while the agreement values of ‘Graduate’ and ‘Degree’ are fair and moderate respectively.

Level of ‘Individual Applicant Type’	**Fleiss Kappa**
Undergraduate	0.045
Graduate	0.352
New Faculty	-0.036
Degree	0.402

Table 6.5: Fleiss Kappa values of the levels of ‘Individual Applicant Type’

What was observed, and had a large impact on the low values of the inter-annotator agreements of all levels, is that for almost each subject (sentence) one annotator did not agree. Since there are only a few annotators and sentences per level of 'Individual Applicant Type' (3-7 annotators and 10-20 sentences), the Fleiss Kappa statistic is heavily impacted by one annotator that does not agree on a subject. The disagreements were not caused by one annotator particularly, but different annotators did not agree with the other annotators for each subject. The disagreement among annotators was caused by various factors. One of the factors is the fact that disagreement of label 'Limitation' affects the annotation of the levels of 'Individual Applicant Type'. Only the sentences that are annotated with the value 'Limited' could be annotated with the levels of 'Individual Applicant Type'.

The second and third factor that contributed to the low agreement values for the annotation of 'Limitation', also contributed to the low agreement values of the annotation of the levels of 'Individual Applicant Type'. For example, the sentence: *"Senior scientists, young investigators, and postdoctoral fellows may serve as principal investigator."* was interpreted in different ways by different annotators. One annotator assigned the 'New Faculty' level, while three other annotators did not. A possible cause for disagreement is the lack of context. The annotator who assigned the 'New Faculty' label may have interpret the 'principal investigator' as the applicant, while the three other annotators did not interpret this. Additional sentences surrounding the sentence could provide more information on the role of the principal investigator. Also, the factor on reading comprehension impacts the agreement of the levels of 'Individual Applicant Type' negatively. Take for example the following sentence: *"Graduate students and junior postdoctoral fellows (i.e., fellows with less than 3 years postdoctoral training by the application submission deadline) are not eligible for this award."*. One third of the annotators overlooked the word 'not' and assigned the value 'New Faculty', while the sentence clearly states that the applicants at the level of 'New Faculty' may not apply.

These factors party explain the low agreement values. However, they do not explain why the levels 'Undergraduate' and 'New Faculty' have such low values compared to the levels 'Graduate' and 'Degree'. The main factor that explains this is the number of subjects (sentences) that are used in the agreement calculation. The number of subjects is low for the levels 'Undergraduate' and 'New Faculty' (around 10 sentences) when compared to the other levels 'Graduate' and 'Degree' (around 20 sentences). This results in a higher impact on the inter-annotator agreement when annotators do not agree on a subject.

Furthermore, for the level 'Undergraduate', the interpretation of the definition of 'Undergraduate' impacts the agreement values negatively. The interpretation of the definition for this level seems to be different for different annotators. This is found in, for example, the sentence: *"Please also note the additional eligibility criteria: students studying on courses with RIBA candidate status may also be considered."*. The definition of 'Undergraduate' states that only the label 'Undergraduate' can be assigned if the sentence explicitly states that bachelor or undergraduate students may apply. However, multiple annotators assigned the value

'Undergraduate'. This annotation can be explained, with some additional knowledge on the term RIBA candidate status. The RIBA candidate status is a status for programs that prepare students for professional practice [100]. The program status can also be given to programs for which bachelor students can also apply. Given this knowledge, it is debatable whether the sentence explicitly states that bachelor students may apply. This sentence also shows that a difference in domain knowledge can affect the annotations. These considerations make it difficult to agree on whether to assign the level 'Undergraduate' or not.

Similarly, annotators have differences in the interpretation of the level 'New Faculty'. For example, in the following sentence: *"Eligibility requirements citizenship: u.s. citizen only degree: doctoral degree received within the last 60 months or currently pursuing."*. From the definitions, three levels could be assigned: 'Graduate', 'New Faculty', and 'Degree', because it could be that the applicant is currently pursuing a doctoral degree ('Graduate'), recently obtained doctoral degree ('New Faculty') or has obtained a degree ('Degree'). Annotators clearly do not agree on this sentence, half of the annotators assigned both 'Graduate' and 'New Faculty', while the other half assigned 'Graduate' and 'Degree'. Possible the annotators did not have the full defintions at ahnd or did not have the time to check them, as the defintions were very clear in this case.

All the factors discussed above have a negative impact on the inter-annotator agreement and result in low Fleiss Kappa values for the different annotations.

6.2 Evaluation Dataset at Funding Opportunity Level

Besides the dataset for the three classification tasks, an evaluation dataset to test the system at the funding opportunity level had to be created and annotated. The dataset consists of 300 funding opportunities from which the ids of these funding opportunities were sampled from the ids of the 4.983 sentences in the balanced dataset. This to ensure a fair distribution of each label of 'Limitation'. Naturally, it was ensured that the evaluation dataset did not overlap with the training data.

6.2.1 Annotation of dataset at funding level

A funding opportunity is represented by all its text parts available in the database. These text parts were collected in one file and presented to an annotator, who assigned labels. The same labels and annotation protocol were used as in the annotation for the sentences. The full annotation protocol can be found in Appendix C.1. The funding opportunities were annotated by approximately the same experts who annotated the sentences. Seven experts annotated 12 funding opportunities each. The amount of funding opportunities that were annotated was kept small to ensure minimized data annotation. Similar to the dataset of the three classification tasks, each funding opportunity was annotated by one annotator, except

for 12 funding opportunities which were annotated by multiple annotators (n>3). These 12 funding opportunities were used to verify the quality of the annotated dataset by calculating their inter-annotator agreement, see section “Inter-Annotator Agreement”. The annotation of 12 funding opportunities took an annotator approximately 35 to 50 minutes, depending on the length of the funding opportunity. The annotation was considered more difficult compared to the sentence annotation task, due to the large amount of text in a funding opportunity.

The funding opportunities with an ‘Organization’ label were discarded from the evaluation dataset, as also done in the annotated dataset on sentence level. This results in an evaluation dataset of 42 funding opportunities. The distribution of the classes in the evaluation dataset is shown in Table 6.6. The distribution of the limitation values is expected, because there are less funding opportunities in the dataset that have a ‘Not Limited’ value than there are ‘Limited’ and ‘Not Specified’ funding opportunities as shown in section 4.3 of Chapter 4.

Class	**Number of funding opportunities**
Limited	27
Not Limited	2
Not Specified	13 (+28 Organization FOs)

Table 6.6: Distribution of the ‘Limitation’ values in the evaluation dataset at funding opportunity level

The distribution of the levels of ‘Individual Applicant Type’ among the 27 funding opportunities with a limitation value ‘Limited’ is visualised in Table 6.7. The table shows that all classes are represented in the evaluation dataset, which ensures that all classes were evaluated.

Level of ‘Individual Applicant Type’	**Number of funding opportunities**
Undergraduate	4
Graduate	12
New Faculty	6
Degree	8

Table 6.7: The distribution of levels of ‘Individual Applicant Type’ in the evaluation dataset at funding opportunity level

6.2.2 Quality assessment annotations at funding opportunity level

The inter-annotator agreement is calculated using the same statistic as the datasets at sentence level, namely the Fleiss Kappa which outputs a value between -1 and +1. A Fleiss kappa value of -1 indicates that there is no agreement observed at all, 0 indicates that there

is agreement no better than chance, and +1 indicates that there is a perfect agreement.

The Fleiss Kappa for label 'Limitation' is 0.666, which is a good agreement. For the levels of 'Individual Applicant Type' there is a moderate agreement for 'Graduate', and a fair agreement for 'Degree'. The 'Undergraduate' and 'New Faculty' levels have an agreement close to chance. The exact values of the Fleiss Kappa can be found in Table 6.8. The inter-annotator agreement values for the levels of 'Individual Applicant Type' at funding opportunity level are similar to the inter-annotator agreements on sentence level. Although annotators remarked that the annotation of funding opportunities was more difficult than the annotation of sentences due to the length of the text, the inter-annotator agreements are comparable or even slightly higher for the label 'Limitation'. A possible explanation is that due to the length of the text, the context of the sentences is a bit more clear, which results in more agreement among annotators.

Other than the lack of context, the same factors that were identified as factors that contributed to the low inter-annotator agreement values for the dataset at sentence level, were also found in the annotations at funding opportunity level: the influence of the disagreements in the annotation of 'Limitation', the interpretation of natural language data, reading comprehension, and the definition of 'Undergraduate'. The wrongly interpreted definition of 'New Faculty' was not found at funding opportunity level. However, due to evaluating a small portion of funding opportunities, it might still occur in the other thousands of funding opportunities that were not evaluated in this small portion. Examples at funding opportunity level for these factors can not be shared due to the confidentiality of the data.

Level of 'Individual Applicant Type'	**Fleiss Kappa**
Undergraduate	-0.029
Graduate	0.400
New Faculty	-0.029
Degree	0.273

Table 6.8: Fleiss Kappa values of the levels of 'Individual Applicant Type' at funding level

6.3 Suggestions on improving the data quality

Different possible causes (factors) for the low quality datasets were identified in this Chapter. To mitigate these causes, several improvements are formulated in this section and discussed below.

First of all, the inter-annotator agreement values were found to be very sensitive, because they were claculated on only a small set of items. However, they do provide some indication

of the data annotation quality. To get more insight into the quality of the data annotations, more data points need to be annotated by multiple annotators.

The sample size partly explains the low Fleiss Kappa scores, but does not give insight in the actual disagreement between annotators. In this Chapter four patterns in the data annotation process were identified that have a possible negative impact on the disagreement between annotators.

1. Misinterpretation of the label 'Limitation', due to the different meaning of the label in daily processes within Elsevier.

2. Misinterpretation of the definitions of 'Undergraduate' and 'New Faculty'. A possible cause would be that the definitions of the different labels are not used during the annotation process. Another possible cause is a difference in domain knowledge among annotators.

3. Variations in the interpretation of the segments by annotators. A possible cause is the lack of context in annotating sentences. Also the difference in domain knowledge could facilitate differences in interpretations.

4. Flaws in reading comprehension. Some words might have been missed or interpreted in isolation. A possible cause of these reading comprehension mistakes may be the speed at which was annotated.

The four patterns found in the disagreements have one or more possible causes that arose from design choices or its effect could be reduced by design choices in the annotation process (annotation protocol). Figure 6.1 shows the 'BEFORE' situation, the current situation of how the data points were annotated ,and the 'AFTER' situation, a hypothetical design which can be used for future annotations.

One of the changes in the protocol is to present each sentence or funding opportunity separately to an annotator. This is shown at number 5 in Figure 6.1. This may let the annotators feel less rushed by the large amount of sentences that are visible that still require annotation. It is expected that this change results in annotators taking more time for a sentence or funding opportunity and therefore will read more carefully. It is expected that this will reduce errors made by the lack of reading comprehension.

Another major change is that, instead of the annotation protocol including definitions of labels being presented on a separate screen, the definitions are presented right next to the sentence or funding opportunity to be annotated. This is expected to reduce disagreements created by misinterpretations of definitions of different classes including 'Undergraduate' and 'New Faculty' (2). Similarly, differences in domain knowledge can be counteracted by also presenting a domain specific list of definitions on screen (3).

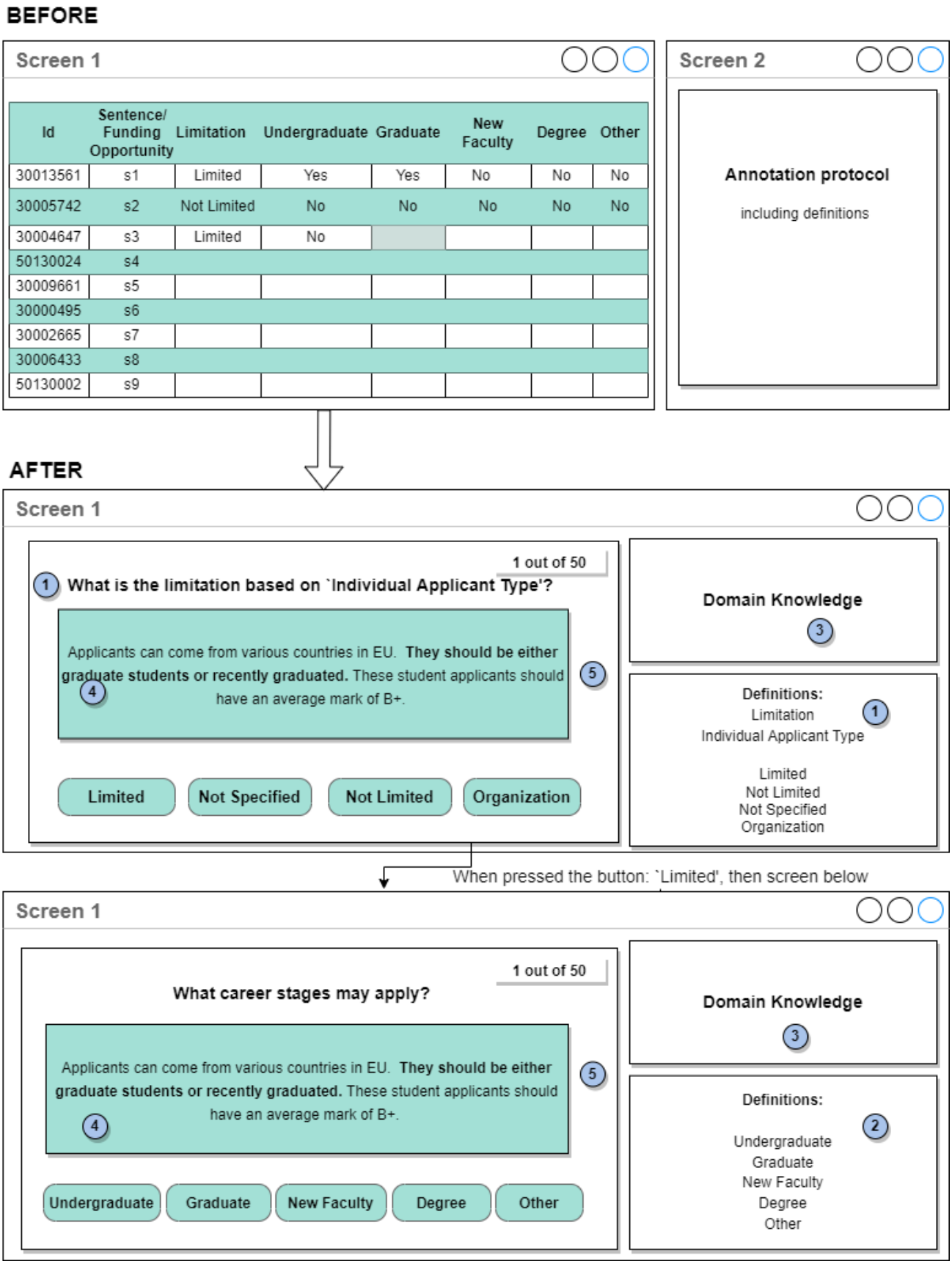

Figure 6.1: Annotation design improvements

The misinterpretation of the label 'Limitation' can be mitigated by including a question above each annotation which specifies that the limitation should be based on 'Individual Applicant Type'. Furthermore, the definition and its deviation from the daily processes is also highlighted in the definitions section on the screen, see (1) in Figure 6.1.

Although natural language data is dependent on interpretation at its core and can be interpreted by different people in different ways, the level of agreement among annotators can be increased by the use of context. The more context, the more unambiguously the text is expected to be interpreted. The annotations show disagreements, which could be mitigated by adding context. For example, by adding the sentences before and after the sentence that needs to be annotated, as visualised in Figure 6.1 with (4). Note that this can only be applied to sentences that are part of a coherent segment (sentences before and after belong together and are semantically connected).

6.4 Conclusion

This Chapter presented steps to acquire a balanced labelled dataset with minimized data annotation to answer the research question: *How to acquire balanced labelled datasets of sufficient quantity and quality with minimized annotation?* First, a new approach was presented to balance a highly unbalanced dataset prior to annotation. In this approach, two groups of sentences are selected and combined: sentences which are likely to contain information on the underrepresented class and segments that are not. Together these form a more balanced dataset. The selection of segments required three steps: 1) converting sentences into embeddings, 2) clustering these embeddings, and 3) determining the keyword ratio in the sentences of the clusters to estimate how likely they are to contain information of the underrepresented class. Balancing the dataset before data annotation, resulted in less data annotation to acquire the same dataset size compared to after data annotation using undersampling techniques. The oversampling techniques were not suited, because of the high imbalance which put lots of weight on just a few sentences.

Besides balancing the dataset, simply using a smaller dataset also reduces the number of data points to be annoated. Literature showed that a small dataset ($<$1500 data points per class) could outperform a larger dataset, as longs as the data quality of the smaller dataset is high compared to the larger dataset. The datasets discussed in this Chapter were therefore kept small, approximately 1000 data points per class for the first and second classification task. For the third classification task, the items per class were significantly lower (ranging between 120 and 400 data points per class), because this dataset is build from one class ('Limited') of the second classification task. These datasets are small, therefore the quality of the datasets was all the more important.

The quality of the datasets was determined by calculating the inter-annotator agreement using the Fleiss Kappa for each label. The datasets of the three classification tasks were

created as one dataset, so also the quality of the these datasets was evaluated as one. The dataset for the three classification tasks showed moderate Fleiss Kappa value for the label 'Limitation'.. For the labels of the different levels of 'Individual Applicant Type;, smaller values were found, some even close to chance. A similar pattern was found for the quality of the dataset at funding opportunity level. Here the 'Limitation' label showed a good agreement and the levels of 'Individual Applicant Type' showed lower Fleiss Kappa values with values close to chance. The low Fleiss Kappa showed that the quality of the datasets was suboptimal. Different possible causes (factors) for the low quality datasets were identified in this Chapter, such as misinterpretation of labels 'Limitation', 'Undergraduate' and 'New Faculty', but also variations in the interpretation of segments by annotators and the flaws in reading comprehension. Improvements in the data annotation process (annotation protocol) to mitigate these causes were presented alongside. Due to resource limitations the datasets could not be annotated again using the suggestions on improving the data annotation process (see section 6.3). However, these improvements are important lessons to ensure high quality datasets with minimized data annotation.

Chapter 7

Experiments

The balanced and annotated dataset at sentence level created in the previous Chapter is used as a training and evaluation dataset for the three classification tasks. These three classification tasks form three experiments which correspond to the first two sub-questions of the research question: *How well can BERT extract information from funding opportunities compared to BiLSTM and Conventional Machine Learning models?*. The fourth experiment presented in this Chapter evaluates the system at funding opportunity level that corresponds to the third sub-question. Each of these experiments is discussed in detail in section 7.1. The experiments all implement a BERT, BiLSTM, and Conventional ML models to compare BERT to BiLSTM and Conventional ML approaches. The implementations for the experiments are similar and discussed per model in sections 7.2, 7.3, and 7.4. The results of the four experiments are presented in Chapter 8.

7.1 Four Experiments

The first experiment corresponds to the first classification task to obtain segments that contain information on 'Individual Applicant Type' with the label 'Specified' (combination of labels 'Not Limited' and 'Limited'). The annotated dataset with the labels 'Not Specified' vs. 'Specified' has a slight imbalance towards the 'Not Specified' class ('Specified': 818 vs. 'Not Specified': 879). Therefore, a random subsample of 818 sentences that belong to the class 'Not Specified' was combined with all the sentences in class 'Specified.' This results in a dataset of 1632 sentences, which is used as training and evaluation data for the first experiment. The second experiment corresponds to the second classification task to obtain segments that contain information on a limitation with the label 'Limited.' The data used for training and evaluating this classification task consists of 99 'Not Limited' sentences and a random sample of 99 'Limited' sentences. The first and second classification task together provide an answer to the question: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform in determining whether segments contain information of interest?*, where segments with the label 'Limited' are seen as segments of interest.

The third experiment corresponds to the third classification to obtain career stages that may apply to a segment. It provides an answer to the question: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform in determining the type of information in a segment of interest?*. The data used for this classification task consists of 779 sentences. Each sentence can belong to multiple classes: 'Undergraduate,' 'Graduate,' 'New Faculty,' or 'Degree.' Each class is represented in at least 120 sentences.

Besides the three experiments, there is a fourth experiment that evaluates the system at funding opportunity level to answer the question: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform at funding opportunity level?*. The system at funding opportunity level combines the labels at sentence level from all sentences of a funding opportunity and predicts the labels at funding opportunity level. The data used for this classification task consists of 42 funding opportunities. The results can be found in Chapter 8 of this report.

The four experiments ran on a Slurm cluster with one GPU and eight supporting CPUs. Different sets of GPUs and CPUs were allocated per run. Slurm assigned the GP100 Pascal GPUs to run the BERT models. The BiLSTM was given the more advanced Turing GPUs, NVIDIA GeForce RTX 2080 or NVIDIA GeForce GTX 1080 Ti. The GPUs were shared among users, so only part of the GPU rsources was used at a time. The GPU assignment shows that the BERT required fewer resources to run compared to the BiLSTM models. The Conventional ML implementation required even fewer resources because it ran on a single computer with 8GB RAM using Pycharm.

7.2 BERT

The experiments were implemented in BERT using the Trainer module of Transformers Python Package based on Python 3.8 [120]. The Trainer module uses the pre-trained bert-base-uncased model, which was one of the first BERT models introduced by Devlin et al. [34]. Although the bert-large model shows a better performance than the base model as presented in [34], the performance difference is slight and does not outweigh the extra computational power required to run the large model. Specifically, the uncased version has been chosen as it does not differentiate between uppercase and lowercase versions of the same word. For example, the words 'Model' and 'model' have the same meaning and should therefore be treated as the same. The BERT base uncased model is used for the two main steps in fine-tuning BERT: 1) Tokenization of the data and 2) Feature extraction and classification of the sentences. The default loss function used by the bert-base-uncased model for binary classification is the CrossEntropyLoss() and was also used in the main approach for BERT in the experiments. The third experiment is a multilabel classification task. For this classification task, a multilabel version of the bert-base-uncased model was used, which uses BCEWithLogitsLoss() instead of the CrossEntropyLoss() for binary classification. A sigmoid activation

function is applied to BCEWithLogitsLoss(), which results in a probability for each class equal to likelihood that the sentence belongs to that class. The final outputs are determined by assigning the class if the probability is above 50%. If the probability is below 50%, the class is not assigned to the sentence. This process is repeated for each of the four classes in the classification task.

BERT has a few parameters that highly affect its learning process. Obtaining the optimal set of parameter values for these parameters is crucial, as found in [74]. These parameters are summarized below:

- *Learning Rate* determines how much the current weights are impacted by an update step. In other words, the learning rate affects how quickly the neural network learns.
- *Weight Decay* adds a penalty term to the cost function, which results in a reduction of the weights during backpropagation. This helps to prevent overfitting and prevents exploding gradient.
- *Batch size* is the number of data points considered by the algorithm before the weights are updated. This impacts the error gradient on which the weights are updated, which directly impacts learning.
- *Number of training epochs* is the number of times the training data is used. In one epoch, the training data is used exactly once. The benefit of more epochs is that the model has more data to learn from. However, too many epochs result in overfitting.
- *Seed* is an integer that determines with which random values the weights are initialized.

Finding the optimal set of parameters is performed by hyperparameter tuning. The Trainer module has an integration with the hyperparameter optimization framework Optuna [87], which automates the hyperparameter tuning. Optuna allows the search to happen in parallel and prunes unpromising combinations of parameters (trials). These characteristics enable Optuna to search a large space in a limited time. The default TPE sampler was used in combination with Hyperband pruner. For each variable, the search space in which the optimization operates was bound. Learning rate and weight decay was bound between 1e-5 and 1e-2, number of epochs between 2 and 5 and the seed between 1 and 2000. The batch size was set as a categorical vaariable with either value 8 or 16. Batch sizes larger than 8 and 16 were also considered, but were not feasible due to memory limitations. In total, 100 different combinations of parameters (trials) were explored.

Besides the parameters that were tuned, three parameters were set upfront with a default value. The first parameter is the evaluation strategy, whether the model was evaluated after each step or after a whole epoch. The model was evaluated after each epoch to ensure that the pruner does not kill unpromising trials to early. The second parameter is the evaluation steps, which is the number of update steps in an epoch. A default value of 1000 evaluation steps was set. The third parameter is the warmup ratio, which determines the ratio of the

training examples which is ran first with a smaller learning ratio. A smaller learning ratio for the first examples ensures that they have a smaller impact on the overall training process. The warmup ratio is set to a default of 0.1.

Furthermore, similar to the evaluation process, the hyperparameter tuning process was optimized on the average f1-score of 5-fold cross-validated models to thoroughly test the model's generalizability. The model with the largest average f1-score was considered to have the optimal set of parameters. The model with the optimal set of parameters was evaluated using the complete set of evaluation metrics. The results of the hyperparameter tuning and the results of an evaluation of the models with the optimal parameters can be found in Chapter 8.

7.3 BiLSTM

The BiLSTM models were implemented using the Keras Python package. The BiLSTM implementation uses a Keras Sequential model with plain stacked layers that can be customized. The format of the sequential model in this case study was based on work by Gupta [44]. The model starts with an Embedding Layer at which the embeddings are added, followed by *n* hidden BiLSTM layers which are each followed by a Dropout layer. The model ends with two Dense layers, one RELU layer and one Sigmoid layer. The Sigmoid layer is used for binary classification and outputs a probability of a segment belonging to class 1. If the probability is above 50%, class 1 is assigned. Otherwise, class 2 is set. For the third experiment, which is a multilabel classification task, the number of classes that the sigmoid function with binary cross entropy with logits creates outputs for was altered to fit the four label classes. The sigmoid function outputs similar to the binary classification, a probability of how likely the sentence belongs to a class where a threshold of 50% was used. This process was repeated for each of the four classes.

As discussed in Chapter 5, no BERT embeddings were used in the baseline implementations to thoroughly test BERT in isolation. Instead, the BiLSTM uses GloVe, which were commonly used embeddings before BERT was introduced [92]. The GloVe word embeddings were used to obtain features from a sentence. The model used these features (word vectors) to classify the sentence. In this study, the 'glove 6B 50d' embeddings were used, which have a dimensionality of 50.

The BiLSTM has a different set of parameters to be set upfront compared to the BERT parameters. The parameters in this set which are specific to BiLSTM are explained below:

- *Number of hidden layers* specifies the number of BiLSTM layers including their corresponding Dropout layers. Values between 2 and 10 were explored.
- *Dropout* specifies the percentage of neurons that are dropped out for each weight update cycle. This is a regularization technique which avoids overfitting. Four different

dropouts were used for the hidden layers, dropout layers, last bidirectional layer, and last dense layer respectively. A dropout was set between 0.2 and 0.8.

- *Units* specifies the number of nodes in a BiLSTM layer. Units is a categorical variable with as values: [16,32,64,128,256]

All these parameters are used for building the BiLSTM model and, therefore, highly affect how the model learns. Therefore, hyperparameter tuning was performed on these parameters. In the hyperparameter tuning also, the standard neural network parameters, such as batch size ([8,16]), learning rate (between 1e-5 and 1e-2), and the number of epochs (max 25) were tuned. A larger amount of epochs was explored when compared to the BERT implementation because the BiLSTM was trained from scratch. Similar to BERT, Otpuna was used as framework, a TPE sampler and Hyperband pruner were used to prune unpromising trials and hyperparameter tuning was optimized on the average f1-score of 5-fold cross-validated models. The results of the BiLSTM hyperparameter tuning and the evaluation of the models with the optimal set of parameters can be found in Chapter 8.

7.4 Conventional Machine Learning

Besides the BERT and BiLSTM model, conventional machine learning classifiers were used as baseline models. A Logistic Regression model was utilized for the binary classification tasks and a Random Forest model was deployed for the multilabel classification task. The models were implemented using the Python package Sklearn [110]. These models were not hyperparameter tuned. Instead the default parameter values were used. Information on the parameters and their default values for Logistic Regression and Random Forest can be found in [111] and [112] respectively. The embeddings used as input for the classifiers are the same embeddings used for the BiLSTM implementation, namely ‘Glove.6B.50d’ embeddings. The results of the evaluation of these implementations can be found in Chapter 8.

Chapter 8

Results and Discussion

In this Chapter, the results of the four experiments are shared and discussed. Furthermore, this Chapter provides an answer to the sub-research question: *How well can BERT extract information from funding opportunities compared to BiLSTM and Conventional Machine Learning models?* This question is answered in section 8.5. First, however, its three sub-questions are answered in sections 8.1, 8.2, and 8.3. Section 8.4 shows an additional experiment on the use of BERT embeddings as knowledge part in combination with a simpler model part.

8.1 Selecting segments of interest

The first sub-question: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform in determining whether segments contain information of interest?* is answered in section 8.1.3. The first sub-question consists of two experiments. The results of these two experiments are discussed in the sections 8.1.1 and 8.1.2.

8.1.1 Selecting segments that contain information on 'Individual Applicant Type'

The goal of the first experiment (classification task) is to distinguish between sentences that contain information on 'Individual Applicant Type' and sentences that do not. Three classifiers, BERT, BiLSTM, and Logistic Regression, were deployed. The optimal parameters for the neural network implementations (BERT and BiLSTM) were found using hyperparameter tuning.

The hyperparameter search of 100 trials took 17.8 hours for the BERT implementation and 22.1 hours for the BiLSTM implementation. The models achieved a maximum f1-score of 0.877 and 0.859, respectively. The optimal set of parameters that achieved these f1-scores can be found in Table 8.1. The importance of parameters in obtaining a high objective score (f1-score) for both hyperparameter searches was determined by fANOVA [54] based on the completed trials. The most important parameters for BERT were seed and weight decay which account for 76% of the influence of parameters on the objective value. In contrast, the

most important parameters for BiLSTM were the number of layers and learning rate, which together accounted for 58%. The full results can be found in Appendix D.1. These include parallel plots which visualize the 100 trials of the hyperparameter searches and plots on the importance of parameters.

Besides BERT and BiLSTM, the Conventional ML approach: Logistic Regression was deployed. The three implementations were trained and evaluated using 5-fold cross-validation and took 373.5, 559.2, and 7.2 seconds to complete for BERT, BiLSTM, and Logistic Regression respectively. The metric values averaged over the 5-folds together with their standard deviations are presented in Table 8.2.

Parameter	**BERT**	**BiLSTM**
Learning Rate	2.287e-05	3.687e-03
Batch Size	8	16
Number of epochs	3	4
Weight Decay	3.379e-05	N/A
Seed	1495	N/A
Number of Layers	N/A	5
Units	N/A	32
Dropout 1	N/A	0.4
Dropout 2	N/A	0.2
Dropout 3	N/A	0.6
Dropout 4	N/A	0.8

Table 8.1: The optimal parameters for BERT and BiLSTM of classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

The results presented in Table 8.2 show that the recall values are higher than the precision for the neural network implementations (BERT and BiLSTM). This shows a tendency toward assigning the positive class. This in contrast to the Logistic Regression model, which achieved higher precision than recall. As discussed in Chapter 5, precision and recall are evenly important for this case study and thus their harmonic mean, the f1-score, is the most important metric.

	BERT		**BiLSTM**		**Logistic Regression**	
Precision	0.857	± 0.021	0.811	± 0.050	0.645	± 0.019
Recall	0.899	± 0.021	0.875	± 0.034	0.235	± 0.011
F1-score	0.877	± 0.016	0.840	± 0.019	0.344	± 0.014

Table 8.2: The average metric values of 5-fold cross-validation: precision, recall, and f1-score, and their standard deviation for classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

The BERT model achieved the highest f1-score of 0.877 (marked in the table), followed by BiLSTM with an f1-score of 0.840 and Logistic Regression with an f1-score of 0.344. The low f1-score of Logistic Regression is low shows that a combination of GloVe with such a simple (conventional) machine learning algorithm is insufficient to determine whether the segments contain information on 'Individual Applicant Type'. The low standard deviations values (around 0.02) indicate that the training datasets across the five folds have similar data quality. In other words, both classes are sufficiently represented such that the models can achieve comparable results over five folds.

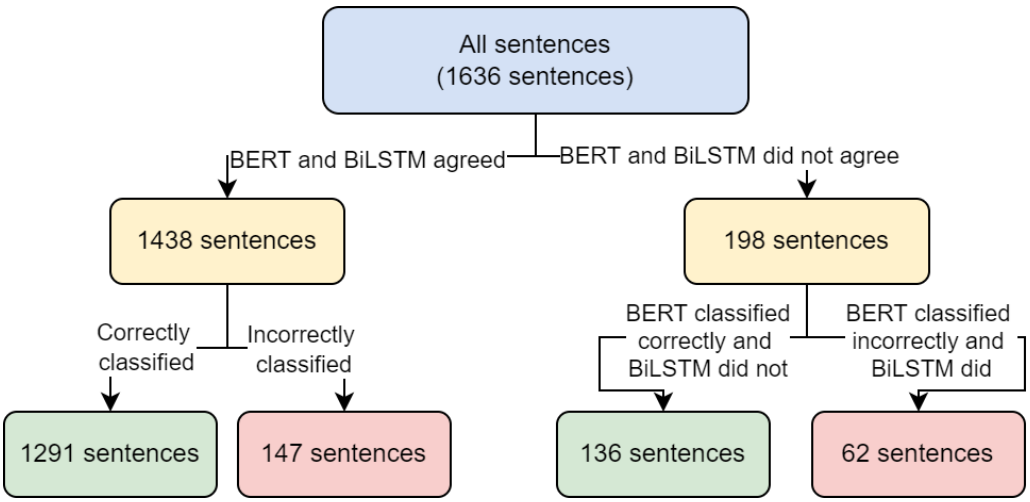

Figure 8.1: Comparison of BERT and BiLSTM classification of sentences for classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

The performances measured by the f1-scores of BERT and BiLSTM are close. Therefore, it is interesting to have a closer look at BERT and BiLSTM. Do these models make the same mistakes in classifying sentences or are there differences? What made BERT outperform BiLSTM? As visualised in Figure 8.1, BERT and BiLSTM often agree with one another (1438 vs. 198 sentences). However, the sentences on which BERT and BiLSTM did not agree (198 sentences), reveal a clear difference between the two. Specifically, this becomes clear when investigating the sentences which were classified correctly by BERT and incorrectly by BiLSTM (136 sentences). For example, take the sentences: "Applications must include: a letter of support from an aats member or a faculty member at a candidates institution." and "Junior research fellows establish formal affiliation with Indian universities and Indian research supervisors.". Although we can not be certain why the BiLSTM model classified these sentences as containing information on 'Individual Applicant Type,' we do know that BiLSTM uses non-contextualized word embeddings. Thus it is highly likely that the BiLSTM flagged the words 'faculty' and 'junior' and largely based its decision on them. BERT, on the other hand, uses its own contextualized embeddings which are more based on surrounding words (context).

8.1.2 Selecting segments that contain information on a limitation

The results for the second classification task are presented in this section with the main goal of classifying sentences into two classes: sentences that have a limitation in who can apply based on 'Individual Applicant Type' or sentences that indicate that anyone can apply for the funding opportunity. A BERT classifier and BiLSTM classifier, and a Logistic Regression classifier were deployed. Similar to the first classification task, a hyperparameter search was conducted for the neural network implementations (BERT and BiLSTM to obtain the optimal set of parameters.

The hyperparameter search of 100 trials (sets of parameters) for BERT took 3.2 hours to complete and achieved a maximum objective score (f1-score) of 0.907. The BiLSTM hyperparameter search of 100 trials (sets of parameters) took 9.5 hours to complete and achieved an f1-score of 0.879. The optimal set of parameters that achieved these f1-scores for both BERT and BiLSTM can be found in Table 8.3. Similar to the first experiment, the importance of parameters in obtaining a high objective score (f1-score) for both hyperparameter searches was calculated using fANOVA [54]. The most important parameters for the second classification task for BERT were seed, the number of training epochs, and weight decay which together accounted for 89% of the influence of parameters on the objective score (f1-score). For BiLSTM, the most important parameter is the number of layers which accounts for 62% of the influence of parameters on the objective score (f1-score). The full results can be found in Appendix D.2, which include plots on the importance of parameters and parallel plots which visualize the 100 trials of the hyperparameter searches.

Three implementations, BERT, BiLSTM, and Logistic Regression, were evaluated using 5-fold cross-validation. The 5-fold cross validation process took BERT 181.6 seconds to complete, BiLSTM 392.8 seconds, and Logistic Regression 6.6 seconds.

The evaluation metrics of the three implementations are presented in Table 8.4. The table presents the average evaluation metric values over the 5-folds alongside their standard deviations. The table shows all three implementations' precision, recall, and f1-score. Similar to the first classification task, the recall is higher than the precision for the BERT and BiLSTM implementation, indicating a progressive model that is eager to assign the positive class. This is again in contrast to the Logistic Regression, which shows a higher precision than recall.

The most important metric, however, is the f1-score. An average f1-score of 0.907 is achieved for BERT, followed by the BiLSTM classifier with an average f1-score of 0.827 and the Logistic Regression classifier with an average f1-score of 0.456. Similar to the first classification task, the Logistic Regression metric values are significantly lower than the values of BERT and BiLSTM and show that GloVe with a simple model is insufficient to determine whether segments contain information on a limitation or are not limited. The metric values' standard deviations are higher than in the first classification task. Standard deviations were around 0.04-0.100 for this classification task compared to the first classification task with

standard deviations around 0.02. A possible explanation is the smaller training and evaluation datasets which result in larger difference in performances across folds.

Parameter	BERT	BiLSTM
Learning Rate	2.023e-05	3.778e-03
Batch Size	8	16
Number of epochs	5	14
Weight Decay	5.265e-03	N/A
Seed	1056	N/A
Number of Layers	N/A	3
Units	N/A	16
Dropout 1	N/A	0.3
Dropout 2	N/A	0.2
Dropout 3	N/A	0.2
Dropout 4	N/A	0.5

Table 8.3: The optimal parameters for the BERT and BiLSTM of classification task 2: Obtain segments that contain information on a limitation

	BERT		BiLSTM		Logistic Regression	
Precision	0.867	± 0.054	0.821	± 0.072	0.480	± 0.048
Recall	0.951	± 0.034	0.846	± 0.110	0.463	± 0.126
F1-score	0.907	± 0.041	0.827	± 0.038	0.456	± 0.075

Table 8.4: The average metric values of 5-fold cross-validation: precision, recall, and f1-score, and their standard deviation for classification task 2: Obtain segments that contain information on a limitation

The difference in the f1-score between BERT and BiLSTM is significantly larger than in the first experiment. A possible explanation is the smaller dataset in which one wrongly classified sentence has a larger impact on the f1-score than a larger dataset. Another possible explanation has to do with the oversampling technique: data augmentation that has been applied to the dataset for this classification task. The oversampling results in more data points that are similar. These similarities might have been more beneficial for BERT than BiLSTM as the oversampling method partly uses BERT embeddings.

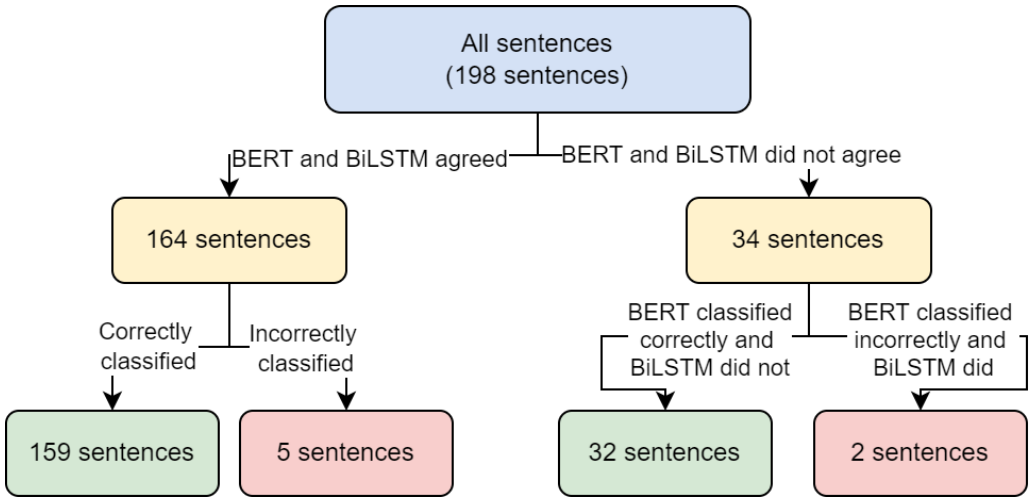

Figure 8.2: Comparison of BERT and BiLSTM classification of sentences for classification task 2: Obtain segments that contain information on a limitation

Similar to the first classification task, a closer look is taken at the differences between BERT and BiLSTM. Figure 8.2 shows that in general, BERT and BiLSTM agree with each other (164 vs. 34 sentences). Again, context seems to be the distinguishing factor between BERT and BiLSTM. Take for example, the sentence: "Scientists of any nationality holding a Ph.D., SCD, M.D., DVM, or DrPH degree or equivalent are eligible to apply; they can reside in the U.S. or elsewhere". The word 'any' signals that anyone can apply. However, this is restricted to the nationality rather than the eligibility on 'Individual Applicant Type'. Most likely, BiLSTM could not catch this subtlety due to its non-contextualized word embeddings, but BERT could due to its semantic knowledge. This difference might have had even more impact on the f1-scores as the data for this experiment contains more context. Especially the data from the class 'Not Limited' has more context. For example, in the sentence: "Category open to all levels of scholars, except early career Degree Requirements Ph.D. (or other terminal degrees) required application.". The part of the sentence with 'except' changes the class from 'Not Limited' to 'Limited'. Everyone can apply except for early career academics ('New Faculty'). Furthermore, the closer look at the data confirmed that the oversampling process favors BERT more than BiLSTM. Most of the new sentences created by BERT were correctly classified by BERT and not by BiLSTM.

8.1.3 Conclusion: Selecting segments of interest

The results of the first two classification tasks discussed above together provide an answer to the first sub-question: *How well does BERT determine whether segments contain information of interest or not of interest?* Both experiments show that BERT outperforms the BiLSTM and Logistic Regression implementation on all performance metrics (precision, recall, and f1-score). The GloVe + Logistic Regression showed to be inadequate for these experiments

in their low performances (f1-scores of 0.344 and 0.456) for both classification tasks. The BERT and BiLSTM implementations showed decent performances, where BERT slightly outperformed BiLSTM. The performance difference between BERT and BiLSTM is suspected to be caused by the contextualized embeddings used by BERT (BERT embeddings) and the non-contextualized embeddings (GloVe embeddings) used by BiLSTM.

Besides the performance, a resource metric, the time taken to complete a 5-fold cross-validation, was used to determine the feasibility of the models. The conventional machine learning model (Logistic Regression) was the quickest. For each of the experiments it took approximately 7 seconds on a local computer with 8GB RAM. The low resources required come at the cost of low performance on both classification tasks (f1-scores of 0.344 and 0.456, respectively). The BERT model shows the highest performance (f1-scores of 0.877 and 0.907 for BERT) but took a bit longer to complete, 3 to 6 minutes depending on the task. It also used a more advanced deploy environment, namely a Slurm Cluster with Pascal GPU. However, compared to the BiLSTM implementation, the resources required for the BERT implementation are reasonable. The BiLSTM implementation took 6 to 9 min. to complete on an even faster Turing GPU, while the performance is slightly lower than the BERT implementation (f1-scores of 0.840 and 0.827). This shows that the BERT implementation is also performing well from a resource point of view, compared to the other two implementations.

The results of the first two classification tasks in this case study indicate that BERT is performing well in determining whether segments contain information of interest by outperforming the two other implementations tested.

8.2 Determining the type of information in a segment

The second sub-question: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform in determining the type of information in a segment of interest?*, is answered in this section. The conclusion can be found in section 8.2.2 and the results of the third experiment that provide an answer to this sub-questions are shared and discussed in section 8.2.1.

8.2.1 Determining career stages that may apply

The main goal for the third experiment was to classify sentences with a limitation into the classes of individual applicant types. Multiple labels may apply per sentence in this experiment, a so-called multiclass multilabel classification task. Similar to the other two classification tasks, a hyperparameter search was performed to obtain the optimal set of parameters for the neural network implementations (BERT and BiLSTM). The results of the hyperparameter tuning are presented below.

The hyperparameter search of 100 trials (sets of parameters) for BERT took 8.7 hours to complete and achieved a maximum f1-score of 0.653. For BiLSTM, the hyperparameter search of 100 trials took 8.5 hours to complete and achieved a maximum objective score (f1-score) of 0.635. The optimal set of parameters that achieved these f1-scores for BERT and BiLSTM can be found in Table 8.5 for both the BERT and BiLSTM. The most important parameters to obtain high objective scores in this classification task were seed and number of training epochs for BERT, which together accounted for 61% of the influence of parameters. For BiLSTM, the most important parameter was the number of layers with a percentage of 79%. The full results, including parallel plots, which visualize the 100 trials of the hyperparameter searches, and the plots on the importance of parameters, can be found in Appendix D.3.

Parameter	**BERT**	**BiLSTM**
Learning Rate	2.228e-04	3.430e-03
Batch Size	16	8
Number of epochs	5	15
Weight Decay	2.811e-04	N/A
Seed	111	N/A
Number of Layers	N/A	2
Units	N/A	32
Dropout 1	N/A	0.6
Dropout 2	N/A	0.4
Dropout 3	N/A	0.5
Dropout 4	N/A	0.5

Table 8.5: The optimal parameters for the BERT and BiLSTM of classification task 3: Obtain career stages that may apply

Besides the BERT and BiLSTM implementation, a Random Forest model was deployed and evaluated as conventional machine learning algorithm. Each of these three implementations was evaluated using 5-fold cross-validation. The evaluation process took BERT 383.5 seconds to complete, BiLSTM 196.5 seconds, and the Random Forest model 7.5 seconds.

The evaluation results are shared in multiple tables. Tables 8.6, 8.7, and 8.8 show the averaged values of precision, recall, and f1-score over the 5-folds together with their standard deviations per level of 'Individual Applicant Type' (career stage) for the BERT, BiLSTM and Random Forest model respectively. For example, in Table 8.6 the precision achieved for the class 'Undergraduate' is 0.527 with a standard deviation of 0.109 using the BERT model. For the BiLSTM model, a precision of class 'Undergraduate' of 0.280 is achieved with a standard deviation of 0.438 as presented in Table 8.7. Similarly, the results of the Random Forest model class can be found in Table 8.8.

The Random Forest model obtained a precision, recall, and f1-score of 0.000 for 'Un-

dergraduate'. For this label, the model predicted no true positives. As this would result in a division by zero in the calculation of precision and recall, their values are instead set to 0.000. In turn, this results in an f1-score of 0.000. Two possible causes have been identified to explain the lack of true positives. First, low data quality with respect to the 'Undergraduate' class (as indicated by a close to chance inter-annotator agreement) makes it difficult for the model to observe patterns in the training data and learn from them. Second, the imbalance in the dataset might have biased the model. The dataset consists of more negative examples (sentences without the label 'Undergraduate':659) than positive examples (sentences labeled with 'Undergraduate': 120). This might have biased the model towards the negative class resulting in less true positives. Combined, these factors could explain the drastically low metric values for class 'Undergraduate' in both the BiLSTM and Logistic Regression implementations. The BERT model achieves a bit higher score than the other two models, but the values are suboptimal compared to the other classes.

The data quality and the class imbalance also play a role in the classification of the three remaining classes. The 'Graduate' class has the highest f1-score, the leas class imbalance (406 positive examples out of 779) and relatively high quality (an inter-annotator agreement value of 0.352). The 'Degree' class came second based on f1-score with data of higher quality (an inter-annotator agreement value of 0.402), but a greater class imbalance (319 positive examples out of 779). This seems to indicate that class imbalance impacts the metric values of this multilabel classification task the most.

The class imbalance seems to also have an impact on the standard deviations of the evaluation metrics. The greater the class imbalance, the more likely a class is underrepresented in the training or evaluation set. This results in varying metric values over the 5-folds in 5-fold cross-validation and larger standard deviations compared to the first two classification tasks.

The class imbalance could also explain that, for this classification task, the precision is higher than the recall for most classes of the BERT and BiLSTM implementation. In contrast to the first two classification tasks, the training dataset contains fewer positive than negative examples. This leads to conservative models which predict fewer positive labels. This results in fewer false positives and in turn leads to higher precision values. Also, the Random Forest model shows higher precision. However, the most important metric is the f1-score. The highest f1 scores for each class are highlighted in the Tables. The BERT model achieves the highest performance (f1-score) for the classes 'Undergraduate' and 'Degree,' while the BiLSTM implementation achieves the highest f1 scores for the 'Graduate' and 'New Faculty' classes. The differences in f1-score between BERT and BiLSTM for the classes 'Graduate' and 'New Faculty' are small (0.792 vs. 0.819 and 0.434 vs. 0.475, respectively) compared to the differences in f1-score between BERT and BiLSTM for the classes 'Undergraduate' (0.468 vs. 0.111 respectively) and 'Degree' (0.675 vs. 0.515 respectively). This results in BERT achieving the highest weighted average f1-score (0.653) over all classes as visualized in Table 8.9. The BiLSTM implementation shows a weighted average f1-score of 0.586 and the Random Forest model a weighted average f1-score of 0.312.

	Undergraduate		Graduate		New Faculty		Degree	
Precision	0.527	±0.109	0.763	±0.138	0.637	±0.308	0.675	±0.108
Recall	0.428	±0.174	0.835	±0.051	0.454	±0.265	0.705	±0.138
F1-score	0.468	±0.140	0.792	±0.089	0.434	±0.087	0.675	±0.050

Table 8.6: The average of metric values for BERT of 5-fold cross-validation with the metrics: precision, recall, and f1-score for classification task 3: Obtain career stages that may apply

	Undergraduate		Graduate		New Faculty		Degree	
Precision	0.280	±0.438	0.789	±0.125	0.658	±0.261	0.577	±0.162
Recall	0.082	±0.126	0.872	±0.077	0.478	±0.230	0.475	±0.134
F1-score	0.111	±0.157	0.819	±0.065	0.475	±0.121	0.515	±0.131

Table 8.7: The average of metric values for BiLSTM of 5-fold cross-validation with the metrics: precision, recall, and f1-score for classification task 3: Obtain career stages that may apply

	Undergraduate		Graduate		New Faculty		Degree	
Precision	0.000	±0.000	0.559	±0.109	0.100	±0.224	0.554	±0.116
Recall	0.000	±0.000	0.644	±0.117	0.010	±0.021	0.243	±0.135
F1-score	0.000	±0.000	0.582	±0.044	0.017	±0.039	0.308	±0.095

Table 8.8: The average of metric values for Random Forest of 5-fold cross-validation with the metrics: precision, recall, and f1-score for classification task 3: Obtain career stages that may apply

	BERT		BiLSTM		Random Forest	
Precision	0.726	±0.067	0.668	±0.157	0.413	±0.078
Recall	0.646	±0.049	0.592	±0.056	0.307	±0.023
F1-score	0.653	±0.040	0.586	±0.060	0.312	±0.039

Table 8.9: The weighted averages of the metric values for BERT, BiLSTM, and Random Forest with the metrics precision, recall, and f1-score, and their standard deviation for classification task 3: Obtain career stages that may apply

Similar to the first two experiments, the implementation with GloVe and the conventional machine learning model, in this case, the Random Forest model, shows to be inadequate for the classification task because of low f1 scores across the four classes. The BERT and BiLSTM implementation shows slightly better performances but compared to the first two classification tasks, the f1-scores are low. The data quality and class imbalance were already

suggested to explain the lower performances. However, there is another aspect that may play a role in the lower performances. The classification task is more complex because it is a multilabel multiclass classification task rather than a relatively simple binary classification task. The performances are expected to be lower for a more complex classification task. This, combined with the fact that it is a relatively small dataset for the complexity of the classification task, can result in models that are under-trained, which results in low performances.

The performances of BERT and BiLSTM show to be close to one another. Similar to the first two experiments, a closer look is taken into the differences between these models. Why did BERT outperform the BiLSTM implementation on the classes 'Undergraduate' and 'Degree'? Why did BiLSTM implementation outperform on the classes 'Graduate' and 'New Faculty'? The manual evaluation of the classified sentences showed that key difference between BERT and BiLSTM implementation was the difference in the use of context, as also found in the first two experiments. The BiLSTM implementation was found to correctly classify more sentences that contain words that often occur in the training data and are linked to a specific class. In contrast BERT better captured information embedded in the semantics, rather than single words. This explains why for the classes 'Graduate' and 'New Faculty', BiLSTM outperforms the BERT implementation. These classes require less contextual knowledge because, often the words 'graduate', 'early career,' and 'young' are used explicitly. For the classes 'Degree' and 'Undergraduate' more diverse sets of words are used to express that there is a limitation on these classes, such as 'senior developers' or 'people with experience' or 'students currently pursuing bachelor's degree'.

8.2.2 Conclusion: Determining the type of information in segments

The results of the third experiment answer the second sub-question: *How well does BERT determine the type of information in a segment of interest?*. The segments of interest were the sentences with a limitation (value 'Limited') and were classified in a multilabel multiclass classification task into four career stages (levels of 'Individual Applicant Type'). The results show that both the BERT and BiLSTM implementation have two classes for which they achieved the highest f1-score over all three implementations. The weighted average of the four classes shows that BERT (f1-score of 0.653) outperforms the other implementations (BiLSTM with an f1-score of 0.586 and Random Forest with an f1-score of 0.312). Also, in this experiment, the implementation of GloVe with a conventional machine learning model (Random Forest model) is insufficient for this classification task, as shown by low performances across all four career stages (classes). The performances of BERT and BiLSTM are considered suboptimal. Multiple factors influenced these suboptimal performances: the complexity of the classification task, the relatively small dataset used, the quality of the dataset and the class imbalance it contains.

The differences in performance between BERT and BiLSTM are small for most of the career stages (classes). BiLSTM was found to correctly classify more sentences that contain

words that often occur in the training data and are linked to a specific class, while BERT was more able to capture when the information was embedded in the semantics rather than single words. It is suspected that this difference can be explained by the difference in knowledge parts (contextualized vs. non-contextualized embeddings) used by the two implementations. Especially the career stages 'Undergraduate' and 'Degree' seem to benefit from the additional context of the BERT knowledge part.

Although BERT achieved the highest performance of the three implementations, both the BiLSTM and Random Forest model are quicker to complete a 5-fold cross-validation on a Turing GPU and an 8GB RAM local computer, respectively. The fewer resources required for the Random Forest model do not make it preferable, however, due to its very low performance. The BiLSTM took only 3 minutes to complete on an advanced Turing GPU, while BERT took 6 minutes on a normal Pascal GPU. The 5-fold cross-validation time can not be directly compared due to the different GPUs used. Therefore, it is assumed that the difference in time and GPU used cancel each other out, resulting in similar resources used by BiLSTM and BERT. Under this assumption, BERT outperforms BiLSTM on all performance metrics and thus outperforms the two other implementations tested.

Although the performances are suboptimal across all models, BERT is performing well in determining the type of information in segments of interest compared to the two other implementations on both the performance and resource metrics.

8.3 From segment level to funding opportunity level

This section focuses on the third sub-question: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform at funding opportunity level*. The conclusion for this sub-question can be found in section 8.3.2. The results on the evaluation of system at funding opportunity leading to this conclusion are shared and discussed in section 8.3.1.

8.3.1 System at Funding Opportunity Level

The results of the evaluation of the system at funding opportunity level for each of the three implementations (BERT, BiLSTM, and Conventional ML) are shared in this section. The labels predicted by the implementations were evaluated against the annotated true labels. Each label ('Limitation' and the four levels of 'Individual Applicant Type') was evaluated, and the precision, recall, and f1-score are presented per label in a Table. The precision, recall and f1-score of the labels for the three implementations are summarized in the Tables 8.10 8.11, 8.12, 8.13, 8.14 for the labels 'Limitation', 'Undergraduate', 'Graduate', 'New Faculty', and 'Degree' respectively. For example, the precision of the class 'Graduate' of the BERT implementation is 0.500, while the precision of the conventional machine learning approach of that

class is 0.333. Some metric values are indicated by 0.000 (see 'Undergraduate') because the label achieved zero true positives, which results in zero precision and zero recall and thus an f1-score of 0.000.

The results of the labels at funding opportunity level are lower compared to each of the classification tasks at sentence level. The difference in performance can be attributed to the conversion of the results at the sentence level to the funding opportunity level. The errors that originated in the first three experiments for all sentences of a funding opportunity carry over and add up at funding opportunity level, which results in larger errors and thus lower performances.

However, more factors have a negative impact on the performances at funding opportunity level. One of them is the test set size. As the test set consists of only 42 funding opportunities, getting a single label wrong greatly lowers the evaluation metric scores. Especially the levels of 'Individual Applicant Type' (career stages) suffer from the small size, because they are evaluated using 4, 12, 6, and 8 funding opportunities for 'Undergraduate', 'Graduate', 'New Faculty' and 'Degree' respectively (see Table 6.7 in Chapter 6).

Besides the data size, the class distribution of the evaluation datasets also impacts the evaluation. Preferably, this class distribution is as realistic as possible. As found in Chapter 4, the distribution of 'Limited,' 'Not Limited,' and 'Not Specified' over the dataset of 195.558 funding opportunities is 33.6%, 0.2%, and 66.2%, which translates to a ratio of 168:1:331. The evaluation dataset, however, has a class distribution ratio of 27:2:13. It includes many more 'Not Limited' and 'Limited' values and is therefore not realistic. Similarly, for the levels of 'Individual Applicant Type' (career stages), the realistic ratio is 1:6:2:8 for 'Undergraduate,' 'Graduate,' 'New Faculty,' and 'Degree' respectively (see Table 4.3 for percentages). In contrast, the evaluation dataset has a ratio of 4:12:6:8 and over represents the classes 'Undergraduate' and 'New Faculty'.

Furthermore, the data quality of the evaluation dataset is affecting the evaluation results. Both the inter-annotator agreement of the labels at funding opportunity and sentence level are low. Especially the classes 'Undergraduate' and 'New Faculty' have low inter-annotator agreements (close to chance, see Chapter 6). This is reflected in the results with lower metric values.

Similar to the third classification task, most classes show higher precision than recall. Again this indicates conservative models, which are reluctant to assign the positive class. Furthermore, Tables 8.10 to 8.14 show that the BERT implementation achieves the highest f1-scores for each of the labels.

Limitation

	BERT	BiLSTM	Convential ML
Precision	0.776	0.669	0.628
Recall	0.643	0.571	0.452
F1-score	0.673	0.632	0.470

Table 8.10: The weighted precision, recall, and f1-score of the label 'Limitation' of the evaluation of system

Undergraduate

	BERT	BiLSTM	Convential ML
Precision	0.500	0.000	0.000
Recall	0.250	0.000	0.000
F1-score	0.333	0.000	0.000

Table 8.11: The precision, recall, and f1-score of the label 'Undergraduate' of the evaluation of system

Graduate

	BERT	BiLSTM	Convential ML
Precision	0.500	0.318	0.333
Recall	0.417	0.583	0.333
F1-score	0.455	0.412	0.333

Table 8.12: The precision, recall, and f1-score of the label 'Graduate' of the evaluation of system

New Faculty

	BERT	BiLSTM	Convential ML
Precision	0.600	0.095	0.000
Recall	0.500	0.333	0.000
F1-score	0.545	0.148	0.000

Table 8.13: The precision, recall, and f1-score of the label 'New Faculty' of the evaluation of system

Degree

	BERT	BiLSTM	Convential ML
Precision	0.455	0.263	0.500
Recall	0.625	0.625	0.125
F1-score	0.526	0.370	0.200

Table 8.14: The precision, recall, and f1-score of the label 'Degree' of the evaluation of system

Besides the precision, recall, and f1-score, inference time is used as evaluation metric at funding opportunity. It is the time required to predict the labels for a funding opportunity. As can be seen in Table , the BiLSTM implementation requires the most time, whereas BERT and the conventional machine learning approach require approximately a fourth of the time of BiLSTM. Do note that the conventional machine learning is run on an 8GB RAM local computer, while the BERT and BiLSTM are run on a cluster with GPU and eight supporting CPUs.

	Inference Time For 42 Funding Opportunities	Inference Time Per Funding Opportunity
BERT	16.4 seconds	0.39 seconds
BiLSTM	63.4 seconds	1.51 seconds
Conventional machine learning	18.4 seconds	0.44 seconds

Table 8.15: The inference time for each of the three implementations

8.3.2 Conclusion: Extracting information from a funding opportunity

The presented results provide information for sub-question 3: *Compared to BiLSTM and Conventional Machine Learning approaches, how does BERT perform at funding opportunity level?*. The results show that for each label BERT outperforms the BiLSTM and conventional machine learning approaches on the most important performance metric, the f1-score. F1-scores for the labels 'Limitation', 'Undergraduate', 'Graduate', 'New Faculty', and 'Degree', of 0.673, 0.333, 0.455, 0.545, and 0.526 respectively were achieved by BERT. The results are suboptimal, especially for the 'Undergraduate' class. After all, errors made in the first three experiments (classification tasks) at funding opportunity level. Furthermore, the small test dataset, low data quality of the test dataset, and the unrealistic class distribution of the test dataset also partially explain the suboptimal performances.

Besides the performance metrics, the resource metric inference time was used to evaluate the three implementations. The Conventional machine learning approach ran on a local 8GB RAM computer in under 18.4 seconds and required the least resources. The BERT implementation is the second best and shows an inference time of 16.4 seconds on a Pascal

GPU. The superior performance of the BERT implementation compared to the conventional machine learning approach outweighs its use of more resources for this case study. This shows that also, from a resource perspective, the BERT implementation is performing well.

Although the performances are suboptimal, BERT is performing well in extracting information from a funding opportunity compared to the BiLSTM and Conventional Machine Learning implementation by outperforming these two implementations.

8.4 SBERT + Conventional ML

The previous experiments consistently showed that BERT was better than BiLSTM at classifying sentences which require contextual knowledge. In contrast, BiLSTM was found to be better at classifying sentences that could be classified solely on individual words. The difference in embeddings between BERT and BiLSTM was suggested as a possible cause for this dichotomy. This makes it interesting to see how BiLSTM would perform with BERT embeddings. Possibly this could even outperform BERT with BERT embeddings. Unfortunately, time limitations did not allow for this test. However, time did allow for a smaller test to measure the impact of BERT embeddings.

	SBERT + CV ML	**GloVE + CV ML**	**BERT + BERT**
Classification 1	0.836 ± 0.013	0.344 ± 0.014	0.877 ± 0.016
Classification 2	0.844 ± 0.025	0.456 ± 0.075	0.907 ± 0.041
Classification 3	0.426 ± 0.053	0.312 ± 0.039	0.653 ± 0.040
Limitation at FO level	0.588	0.470	0.673
Career stages at FO level	0.344	0.187	0.476

Table 8.16: The (weighted) average f1-scores of 5-fold cross-validation for the BERT + Conventional ML, GloVe + Conventional ML and BERT + BERT on the three classification tasks and the system at funding opportunity level for limitation and the career stages

Instead of BiLSTM, a conventional machine learning model with BERT embeddings was used and evaluated against the regular BERT and conventional machinine learning implementations. The so-called BERT + Conventional ML model was implemented using the SBERT knowledge part of Reimers Gurevych [99] because it is already fine-tuned to produce sentence embeddings. This off-the-shelf solution is implemented due to time limitations. Another advantage of using SBERT is that it shows state-of-the-art results and outperforms the bert-base-uncased and GloVe knowledge parts on seven natural language tasks as presented in [99]. To allow for easy comparison, the average f1-scores of the SBERT + Conventional ML approach are shared in Table 8.16 along with the average f1-scores of GloVe +

Conventional ML and BERT + BERT. The resources required for these three implementations are shared in Table 8.17.

The f1-scores in Table 8.16 show that, as expected, SBERT + CV ML outperforms the GloVe + CV ML model. Furthermore, for the first two experiments (binary classification tasks) its f1-scores are only slightly below those of BERT + BERT. The difference is greater for classification task 3 but SBERT + CV ML still outperforms GloVe + CV ML. These observations support the assumption that the BERT knowledge part is an important component in achieving high performance.

	SBERT + CV ML	**GloVe + CV ML**	**BERT + BERT**
Classification 1	52.5 seconds	7.2 seconds	373.5 seconds
Classification 2	17.7 seconds	6.6 seconds	181.6 seconds
Classification 3	34.1 seconds	7.5 seconds	383.5 seconds
Inference time at FO level	24.3 seconds	18.4 seconds	16.4 seconds

Table 8.17: The time taken to complete the 5-fold cross-validation for the three classification tasks and the inference time of labels at funding opportunity level for 42 funding opportunities for the models SBERT + Conventional ML, GloVe + Conventional ML and BERT + BERT.

From a resource perspective, the SBERT + CV ML is beneficial to use. Whereas its performance is only a bit lower than BERT + BERT, especially for the first experiments, SBERT + CV ML requires way less resources. SBERT + CV ML was deployed on a local 8GB RAM computer, whereas BERT + BERT was deployed on a Slurm Cluster using a Pascal GPU. Still SBERT + CV ML was approximately seven to eleven times faster on the individual classification tasks. These findings indicate that SBERT + CV ML approach is a suitable option when a research environment for deep neural networks (preferably a GPU) is not available.

The combination of BERT as a knowledge part with simple conventional ML models (Logistic Regression and Random Forest models) showed slightly lower performance than BERT + BERT. Therefore, it would be interesting to see how well BiLSTM with a BERT knowledge part performs. Another option is to combine BERT with other more complex Conventional ML algorithms. A possibility would be Support Vector Machines, which were the best performing supervised machine learning algorithms for information extraction before neural networks were introduced [69].

8.5 Conclusion

The case study in this master thesis answers the sub-research question: *How well can BERT extract information from funding opportunities compared to BiLSTM and Conventional Machine Learning models?* This sub-research question was split into three sub-questions which were each answered in this Chapter. For all three sub-questions, BERT is performing well by outperforming the BiLSTM and conventional machine learning approach. Also, the BERT model was found to perform well from a resource point of view. Although the conventional machine learning approach required fewer resources, the performance difference outweighs using more resources for the BERT approach. This shows that BERT can extract information from funding opportunities well by outperforming the BiLSTM and conventional machine learning. This is in line with the literature that also shows that BERT outperforms BiLSTM approaches and conventional machine learning approaches in various natural language understanding tasks [10, 107, 91, 20, 57, 138].

Although BERT outperforms the two other implementations, the performance is too low to be used in a real situation (for example, within Elsevier). The first part of the approach that focuses on selecting segments of interest (segments that contain information on 'Individual Applicant Type') shows decent performances. However, if this would be applied to a realistic situation, higher performances (f1-scores close to 1.000) are needed because these classification tasks function as pre-selection for the third classification task that classifies the segments into the four career stages of 'Individual Applicant Type .' The main drop in performance is found in sub-question B, which focuses on determining the type of information in a segment by classifying the segments into the four career stages of 'Individual Applicant Type'. This drop in performance carries over to the third sub-question (sub-question C), which focuses on the results at funding opportunity level. It is expected that the relatively low performances can, for a large part, be attributed to the data used in this case study. The data quality of the training and evaluation datasets, both on sentence level and funding opportunity level, is low as shown by low inter-annotator agreement values (see Chapter 6). Furthermore, the dataset used in the third classification task has a high class imbalance, which impacts learning negatively. Moreover, the size of the evaluation datasets is rather small, which impacts the reliability of the results.

BERT only slightly outperforms BiLSTM across all experiments. Example sentences showed that it is suspected that the difference in embeddings used by the two models led to BERT outperforming BiLSTM. The embeddings used by BERT are contextualized, which means that these embeddings are more based on context than the non-contextualized word embeddings used by BiLSTM. The sentences showed that BiLSTM was better at classifying sentences that could be classified based on individual words. In contrast, BERT showed to be better at classifying sentences that required a combination of words. This might indicate that if BERT contextualized embeddings were used in combination with BiLSTM, this might outperform the BERT implementation. Due to time limitations, this could not be tested in this

master thesis. However, to test the importance of the BERT knowledge part in achieving high performances, the BERT embeddings were tested in combination with simple conventional machine learning models. The results showed that, indeed, the BERT knowledge part has a large influence on the high performances. In future work, it would be interesting to test a BiLSTM model with a BERT knowledge part.

Chapter 9

Limitations and Future Work

This study has some limitations that leave room for future work. The limitations were mentioned throughout the report and summarized here with their potential future steps.

The data quality of the training and evaluation dataset was found to be suboptimal. This is suspected to be the most significant contributor to the suboptimal performances achieved in this study. For future work, to verify this, the study needs to be conducted with a higher-quality dataset. Suggestions to create a higher quality dataset can be found in Chapter 6.

The class imbalance of the dataset for classification task 3 creates a bias towards the negative class. It is expected that the class imbalance is one of the main factors that contributed to lower performances. To verify this, the classification task needs to be re-trained using a balanced dataset.

The size of the training dataset of classification task 3 seemed to be too small for the complexity of the classification task. For future work, a larger training dataset is required.

The size of the evaluation dataset that was used for evaluation of the system at funding opportunity level is too limited. Only 42 funding opportunities remained after annotation that could be used to evaluate the system at funding opportunity level. Especially for the different levels of 'Individual Applicant Type' (career stages), the size of the evaluation dataset is small, with 4 to a max of 12 funding opportunities per class. These small evaluation sets, give a distorted view of the performance of the system at funding opportunity level. For future work, a larger evaluation dataset should be used.

The class distribution in the evaluation dataset that is used for evaluation of the system at funding opportunity level does not reflect the class distribution of the 195.558 funding opportunity database. The evaluation set and, therefore, the evaluation does not fully reflect a real situation where a random set of funding opportunities are predicted.

The Organization Funding Opportunities were disregarded from the training and evaluation datasets. However, in a realistic situation in which the systems would be applied to all

funding opportunities in the database, the Organization funding opportunities can not be left out. As discussed in Chapter 4, the Organization funding opportunities are indistinguishable in the database from individual funding opportunities that do not have the characteristic 'Individual Applicant Type' available. A way to separate these funding opportunities is necessary in order to deploy the system at funding opportunity level to the full database. A possible solution is to use a classifier, for example, BERT + BERT, to separate between organization and individual funding opportunities.

BERT + BiLSTM - The BERT implementation (BERT + BERT) only slightly outperformed the BiLSTM implementation (GloVe + BiLSTM). Furthermore, this master thesis showed by an additional experiment that the knowledge part of BERT had a major impact in achieving high performances. Therefore a combination of BERT and BiLSTM might be interesting to explore in future work. Similarly, work by Dai et al. [30] combined BERT with BiLSTM and CRF to extract medical information from electronic health records. It might be interesting to see if the impact of BERT embeddings generalizes to other contextualized embeddings. In future work experiments can be done with, for example, BiLSTM embeddings (BiLSTM + BiLSTM). Furthermore, the SBERT + conventional ML models (Logistic Regression and Random Forest model) showed performances that were slightly lower than BERT + BERT. Therefore, it might also be interesting to explore SBERT with a bit more advanced conventional machine learning methods, such as Support Vector Machines. These were the best performing supervised machine learning algorithms for information extraction before neural networks were introduced [69].

Sentences vs. Paragraphs - As discussed in Chapter 5, the data needed to be split into sentences because the information on splitting into paragraphs was not available in the text. However, this has a drawback; information that is shared between sentences is lost. For example, the two sentences: "Students currently pursuing a master's study into neuroscience are eligible to apply. This also holds for Ph.D. in that direction.", clearly share information on eligibility between them. The embeddings used by BERT are built specifically with this context in mind. heavily based on this context. Therefore, it would be interesting to research whether the use of paragraphs improves the performance of these implementations. If so, it would be especially interesting to check if it benefits the BERT model most.

Adding context to sentence annotation was suggested under the assumption that splitting into sentences (less context) has a negative effect on the agreement among annotators. For future work, it is interesting to research whether this assumption holds.

Chapter 10

Conclusion and Recommendations

10.1 Conclusion

This work functions as a first step towards automating eligibility criteria extraction from funding data (text) within Elsevier. This thesis researched BERT, as it has shown state-of-the-art results in various natural language understanding tasks, including information extraction. Furthermore, BERT is a pre-trained neural network, which requires less labeled training data to perform well compared to other neural networks. Combined with the fact that BERT was not yet used on funding data, made it interesting to research BERT in information extraction from funding data.

Literature on information extraction showed that BERT is often used as a method for the classification step(s) in the commonly used two-step or three-step approach of information extraction. To test this approach on funding data, a three-step approach with a segmentation step and two classification steps was implemented with funding data from Elsevier, where eligibility criteria 'Individual Applicant Type' was used as a case study. The first classification step aimed to obtain segments of interest (segments that contain information on a limitation of 'Individual Applicant Type'). This selection step was especially found to be useful for this case study, as many sentences in the dataset were not of interest (high imbalance). The second classification step aimed to determine the type of information in a segment (the kind(s) of limitation on 'Individual Applicant Type' a segment contains). Both classification steps were implemented using BERT with BERT both as knowledge and model part. This approach was evaluated against a GloVe + BiLSTM and a GloVe + Conventional ML implementation. The data used in this case study was relatively small ($<$1000 data points per class) and balanced before data annotation using a novel approach based on clustering and keyword ratio to minimize data annotation.

The results showed that BERT outperforms the BiLSTM and Conventional ML approach in the two classification steps and the whole system at funding opportunity level. This is in line with the literature that shows that BERT outperforms BiLSTM and conventional machine learning approaches in various natural language understanding tasks [10, 107, 91, 20, 57, 138]. A manual evaluation of the classified sentences of BERT and BiLSTM suggests that

the BERT knowledge part is key in BERT outperforming BiLSTM. This finding was further explored by an additional experiment that combined a knowledge part of BERT with a simple conventional machine learning approach. This approach was compared to the GloVe + Conventional ML implementation and the BERT + BERT implementation. The results showed indeed that the knowledge part BERT had a major impact on achieving high performance by the fact that the BERT + Conventional ML notably outperformed the GloVe + Conventional ML approach and even achieved performances close to the BERT + BERT implementation.

Although BERT outperforms the two other implementations, its performances (f1-scores) are suboptimal (not close to 1.000). It is expected that the relatively low performances can, for a large part, be attributed to the data used in this case study. The data quality of the training and evaluation datasets, both on sentence level and funding opportunity level, are found to be of low quality, as shown by low inter-annotator agreement values (see Chapter 6). Furthermore, the dataset used in the third classification task has a high class imbalance, negatively impacting the learning process. Moreover, the size of the evaluation dataset is relatively small. These factors made it difficult for the models to achieve high performances. Future steps to mitigate these factors are presented in Chapter 9. Future work would need to be done to confirm that the data limited the performance in this study. If so, a three-step approach with BERT as method for the classification tasks a viable approach to extract information from funding opportunities with minimized data annotation, as long as the data is of high quality and the resources to run a deep neural network, preferably a GPU, are available.

10.2 Recommendations

The results of the additional experiment showed that the performances of the BERT + Conventional ML implementation are only slightly lower than performances of the BERT + BERt implementation. However, the resources it requires are way lower. Therefore, if the resources are unavailable to run a BERT + BERT implementation, a simpler model combined with BERT as a knowledge part can be a good alternative.

In general it is advisable to use paragraphs as segments. In this case study, the text is split into sentences due to the lack of syntactic information in paragraphs, such as headers and white lines. As discussed in Chapter 5, the drawback of splitting into sentences is that information that is shared between sentences is lost. When this approach is applied to different structured funding data that can be split into paragraphs, this is advised. The preservation of information that shared between sentences (context) will likely have a positive effect on learning as BERT makes extensive use of context [34].

The three-step approach with BERT in the two classification steps studied in this master thesis is not necessarily tied to the funding domain. It can be used in any other situation when information needs to be extracted from natural language data, provided that high-quality train-

ing data is available and the resources are available to run a deep neural network (preferably a GPU). Furthermore, the implementation is not limited to the English language. The multilingual BERT model can transfer the knowledge learned in one language (English) to any of the 104 languages it was pre-trained on. However, as found in work by Pires et al. [93], although the model provides good accuracy across many languages, some language pairs do not transfer well because these languages are fundamentally different. Therefore, currently, many BERT models exist which are trained in specific languages, including, Spanish [16], Arabic [104], and Finnish [124] and many more.

There are situations in which other approaches to extract information from text are preferred above BERT. As presented in the paper in which BERT was introduced [34], BERT performs well in capturing the semantics of natural language data due to the size of its training dataset and the attention mechanism that was used [123]. Suppose the information of interest can be easily extracted based on syntactic or lexical properties. For example, extracting e-mail addresses is much easier to obtain with a gazetteer using the '@' symbol than with machine learning approaches. In that case, more straightforward methods, such as rule-based approaches, gazetteers, and DOM, are advised because they do not require a training dataset and a resource-intensive environment to deploy.

References

[1] Charu C. Aggarwal and Cheng Xiang Zhai. *Mining text data*. Vol. 9781461432. 2013, pp. 1–522. ISBN: 9781461432234. DOI: 10.1007/978-1-4614-3223-4.

[2] Eugene Agichtein and Luis Gravano. “Snowball: Extracting Relations from Large Plain-Text Collections”. In: *Proceedings of the fifth ACM conference on Digital Libraries* (2000), pp. 85–94. DOI: 10.1145/336597.336644.

[3] Mebarka Allaoui, Mohammed Lamine Kherfi, and Abdelhakim Cheriet. “Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study”. In: *International Conference on Image and Signal Processing*. Springer. 2020, pp. 317–325.

[4] Jason Anggakusuma, Viny Christanti Mawardi, and Manatap Dolok Lauro. “Resume extraction with conditional random field method”. In: *IOP Conference Series: Materials Science and Engineering* 1007.1 (2020). ISSN: 1757899X. DOI: 10.1088/1757-899X/1007/1/012154.

[5] Nguyen Bach and Sameer Badaskar. “A Review of Relation Extraction”. In: *Literature review for Language and Statistics II 2* May 2011 (2007), p. 15.

[6] Michele Banko and Oren Etzioni. “The tradeoffs between open and traditional relation extraction”. In: *ACL-08: HLT - 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference* June (2008), pp. 28–36.

[7] Michele Banko et al. “Open information extraction from the web”. In: *IJCAI International Joint Conference on Artificial Intelligence* (2007), pp. 2670–2676. ISSN: 10450823.

[8] Nicholas Barr. “Higher education funding”. In: *Oxford Review of Economic Policy* 20.2 (2004), pp. 264–283. ISSN: 0266903X. DOI: 10.1093/oxrep/grh015.

[9] Adam L. Berger et al. “A Maximum Entropy Approach to Natural Language Processing”. In: *Computer Linguistics* 22.1 (1996), pp. 39–71. DOI: 10.3115/1075812.1075844.

[10] Vedant Bhatia et al. “End-to-End Resume Parsing and Finding Candidates for a Job Description using BERT”. In: *CoRR* abs/1910.03089 (2019). arXiv: 1910.03089.

[11] Kalina Bontcheva et al. “TwitIE: An open-source information extraction pipeline for microblog text”. In: *International Conference Recent Advances in Natural Language Processing, RANLP* September (2013), pp. 83–90. ISSN: 13138502.

[12] Sergey Brin. “Extracting Patterns and Relations from the World Wide Web”. In: *The World Wide Web and Databases*. Ed. by Paolo Atzeni, Alberto Mendelzon, and Giansalvatore Mecca. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 172–183. ISBN: 978-3-540-48909-2. DOI: https://doi.org/10.1007/10704656_11.

[13] Markus Bundschus et al. “Extraction of semantic biomedical relations from text using conditional random fields”. In: *BMC Bioinformatics* 9 (2008), pp. 1–14. ISSN: 14712105. DOI: 10.1186/1471-2105-9-207.

[14] Mary Elaine Califf and J Raymond. “Relational Learning of Pattern-Match Rules for Information Extraction”. In: *Proceedings of the sixteenth national conference on artificial intelligence* 328 (1999), p. 334. ISSN: 0003-6951. DOI: 10.1162/153244304322972702.

[15] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. “Density-Based Clustering Based on Hierarchical Density Estimates”. In: *Advances in Knowledge Discovery and Data Mining*. Ed. by Jian Pei et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–172. ISBN: 978-3-642-37456-2. DOI: 10.1007/978-3-642-37456-2_14.

[16] José Cañete et al. “Spanish Pre-Trained BERT Model and Evaluation Data”. In: *PML4DC at ICLR 2020*. 2020.

[17] C Cardie and K Wagstaff. “Noun Phrase Coreference as Clustering”. In: *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Very Large Corpora* 1995 (1999), pp. 82–89.

[18] Raghavendra Chalapathy, Ehsan Zare Borzeshi, and Massimo Piccardi. “Bidirectional LSTM-CRF for Clinical Concept Extraction”. In: *Proceedings of the Clinical Natural Language Processing Workshop (ClinicalNLP)*. 2016, pp. 7–12.

[19] Yuan Chang et al. “Chinese named entity recognition method based on BERT”. In: *2021 IEEE International Conference on Data Science and Computer Application (ICDSCA)*. IEEE. 2021, pp. 294–299. DOI: 10.1109/ICDSCA53499.2021.9650256.

[20] Chantana Chantrapornchai and Aphisit Tunsakul. “Information extraction on tourism domain using SpaCy and BERT”. In: *ECTI Transactions on Computer and Information Technology* 15.1 (2021), pp. 108–122. ISSN: 22869131. DOI: 10.37936/ecti-cit.2021151.228621.

[21] Jie Chen, Chunxia Zhang, and Zhendong Niu. “A Two-Step Resume Information Extraction Algorithm”. In: *Mathematical Problems in Engineering* 2018 (2018). ISSN: 15635147. DOI: 10.1155/2018/5761287.

[22] Diego Chialva and Alexis-Michel Mugabushaka. “DINGO: An Ontology for Projects and Grants Linked Data”. In: *ADBIS, TPDL and EDA 2020 Common Workshops and Doctoral Consortium*. Ed. by Ladjel Bellatreche et al. Cham: Springer International Publishing, 2020, pp. 183–194. ISBN: 978-3-030-55814-7. DOI: 10.1007/978-3-030-55814-7.

[23] Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. “Rule-based information extraction is dead! Long live rule-based information extraction systems!” In: *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference* October (2013), pp. 827–832.

[24] Key-Sun Choi et al. “Word segmentation standard in Chinese, Japanese and Korean”. In: September 2009 (2009), pp. 179–186. DOI: 10.3115/1690299.1690325.

[25] Jacob Cohen. “A Coefficient of Agreement for Nominal Scales”. In: *Educational and Psychological Measurement* 20.1 (1960), pp. 37–46. DOI: 10.1177/001316446002000104.

[26] European Commission. *European Commission: Funding Tenders*. Available at https://ec.europa.eu/info/funding-tenders_en. Accessed: 2022-05-26.

[27] J Contreras et al. “A Semantic Portal for the International Affairs Sector”. In: *Engineering Knowledge in the Age of the Semantic Web*. Ed. by Enrico Motta et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 203–215. ISBN: 978-3-540-30202-5. DOI: 10.1007/978-3-540-30202-5_14.

[28] Jim Cowie and Wendy Lehnert. “Information Extraction”. In: *Communications of the ACM* 39.1 (1996), pp. 80–91. DOI: 10.1145/234173.234209.

[29] Zeyu Dai, Hongliang Fei, and Ping Li. “Coreference Aware Representation Learning for Neural Named Entity Recognition”. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, July 2019, pp. 4946–4953. DOI: 10.24963/ijcai.2019/687.

[30] Zhenjin Dai et al. “Named Entity Recognition Using BERT BiLSTM CRF for Chinese Electronic Health Records”. In: *2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. 2019, pp. 1–5. DOI: 10.1109/CISP-BMEI48845.2019.8965823.

[31] Gerald DeJong. “Prediction and substantiation: A new approach to natural language processing”. In: *Cognitive Science* 3.3 (1979), pp. 251–273. ISSN: 03640213. DOI: 10.1016/S0364-0213(79)80009-9.

[32] Weidong Deng and Yun Liu. “Chinese Triple Extraction Based on BERT Model”. In: *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)*. 2021, pp. 1–5. DOI: 10.1109/IMCOM51814.2021.9377404.

[33] Google Developers. *Clustering Algorithms*. Available at https://developers.google.com/machine-learning/clustering/clustering-algorithms. Accessed: 2022-03-14.

[34] Jacob Devlin et al. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* 1.Mlm (2019), pp. 4171–4186.

[35] Ketan Doshi. *Transformers Explained Visually (Part 2): How it works, step-by-step*. Available at https://towardsdatascience.com/transformers-explained-visually-part-2-how-it-works-step-by-step-b49fa4a64f34. Accessed: 2021-13-12.

[36] Elsevier. *About Elsevier*. Available at https://www.elsevier.com/about. Accessed: 2021-15-11.

[37] Elsevier. *Funding Institutional*. Available at https://www.elsevier.com/solutions/funding-institutional. Accessed: 2021-05-12.

[38] Joseph L Fleiss. "Measuring nominal scale agreement among many raters." In: *Psychological bulletin* 76.5 (1971), p. 378. DOI: 10.1037/h0031619.

[39] Shang Gao et al. "Hierarchical attention networks for information extraction from cancer pathology reports". In: *Journal of the American Medical Informatics Association* 25.3 (Nov. 2017), pp. 321–330. ISSN: 1527-974X. DOI: 10.1093/jamia/ocx131.

[40] Yoav Goldberg. "Assessing BERT's syntactic abilities". In: *arXiv preprint arXiv:1901.05287* (2019).

[41] Santiago González-Carvajal and Eduardo C Garrido-Merchán. "Comparing BERT against traditional machine learning text classification". In: *arXiv preprint arXiv:2005.13012* (2020).

[42] Magnus Gulbrandsen and Jens Christian Smeby. "Industry funding and university professors' research performance". In: *Research Policy* 34.6 (Aug. 2005), pp. 932–950. ISSN: 00487333. DOI: 10.1016/j.respol.2005.05.004.

[43] Zhijiang Guo, Yan Zhang, and Wei Lu. "Attention guided graph convolutional networks for relation extraction". In: *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference* (2020), pp. 241–251. DOI: 10.18653/v1/p19-1024.

[44] Pashupati Gupta. *Building a Text Classification model using BiLSTM*. Available at https://medium.com/analytics-vidhya/building-a-text-classification-model-using-bilstm-c0548ace26f2. Accessed: 2022-03-17.

[45] Aria Haghighi and Dan Klein. "Simple coreference resolution with rich syntactic and semantic features". In: *EMNLP 2009 - Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: A Meeting of SIGDAT, a Special Interest Group of ACL, Held in Conjunction with ACL-IJCNLP 2009* August (2009), pp. 1152–1161. DOI: 10.3115/1699648.1699661.

[46] *HDBSCAN Documentation: Frequently Asked Questions*. Available at https://hdbscan.readthedocs.io/en/latest/faq.html. Accessed: 2022-03-23.

[47] Diana Hicks. "Performance-based university research funding systems". In: *Research Policy* 41.2 (Mar. 2012), pp. 251–261. ISSN: 00487333. DOI: 10.1016/j.respol.2011.09.007.

[48] Jerry R Hobbs et al. "FASTUS: A finite-state processor for information extraction from real-world text". In: (1993), pp. 1172–1178.

[49] Gumwon Hong. “Relation extraction using support vector machine”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 3651 LNAI (2005), pp. 366–377. ISSN: 03029743. DOI: 10.1007/11562214_33.

[50] Jiaqi Hou et al. “Bert-based chinese relation extraction for public security”. In: *IEEE Access* 8 (2020), pp. 132367–132375.

[51] *How HDBSCAN Works*. Available at https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html. Accessed: 2022-03-23.

[52] HuggingFace. *BERT base uncased*. Available at https://huggingface.co/bert-base-uncased. Accessed: 2022-03-14.

[53] Purva Huilgol. *Accuracy vs. F1-score*. Available at https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2. Accessed: 2022-05-15.

[54] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. “An Efficient Approach for Assessing Hyperparameter Importance”. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 1. Bejing, China: PMLR, 22–24 Jun 2014, pp. 754–762. URL: https://proceedings.mlr.press/v32/hutter14.html.

[55] Hideki Isozaki and Hideto Kazawa. “Efficient support vector classifiers for named entity recognition”. In: *COLING 2002: The 19th International Conference on Computational Linguistics*. 2002. DOI: 10.3115/1072228.1072282.

[56] Yaozong Jia and Xiaobin Xu. “Chinese Named Entity Recognition Based on CNN-BiLSTM-CRF”. In: *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS* 2018-Novem (2019), pp. 831–834. ISSN: 23270594. DOI: 10.1109/ICSESS.2018.8663820.

[57] Amit Jindal et al. “Leveraging BERT with Mixup for Sentence Classification (Student Abstract)”. In: *arXiv* (2019), pp. 1–2. ISSN: 23318422.

[58] *John Snow Label Transformers*. Available at https://nlp.johnsnowlabs.com/docs/en/transformers. Accessed: 2022-03-14.

[59] Mandar Joshi et al. “BERT for Coreference Resolution: Baselines and Analysis”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 5803–5808. DOI: 10.18653/v1/D19-1588.

[60] Frans Kaiser, Hans Vossensteyn, and Jos Koelman. “Public funding of higher education : a comparative study of funding mechanisms in ten countries”. In: *IEEE Transactions on Electron Devices - IEEE TRANS ELECTRON DEVICES* (Jan. 2002).

[61] Nanda Kambhatla. “Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction”. In: *Proceedings of the ACL interactive poster and demonstration sessions*. 2004, pp. 178–181.

[62] Subhradeep Kayal et al. “A Framework to Automatically Extract Funding Information from Text”. In: *Machine Learning, Optimization, and Data Science*. Ed. by Giuseppe Nicosia et al. Cham: Springer International Publishing, 2019, pp. 317–328. ISBN: 978-3-030-13709-0. DOI: 10.1007/978-3-030-13709-0_27.

[63] Subhradeep Kayal et al. “Tagging Funding Agencies and Grants in Scientific Articles using Sequential Learning Models”. In: *Association for Computational Linguistics* Proceeding (2017), pp. 216–221. DOI: 10.18653/v1/w17-2327.

[64] Chih Hao Ku, Alicia Iriberri, and Gondy Leroy. “Crime Information Extraction from Police and Witness Narrative Reports”. In: *2008 IEEE Conference on Technologies for Homeland Security*. 2008, pp. 193–198. DOI: 10.1109/THS.2008.4534448.

[65] Anurag Kulshrestha, Venkataraghavan Krishnaswamy, and Mayank Sharma. “Bayesian BILSTM approach for tourism demand forecasting”. In: *Annals of Tourism Research* 83 (2020), p. 102925. ISSN: 0160-7383. DOI: https://doi.org/10.1016/j.annals.2020.102925.

[66] Guillaume Lample et al. “Neural Architectures for Named Entity Recognition”. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 260–270. DOI: 10.18653/v1/N16-1030.

[67] W Lehnert et al. “Description of the CIRCUS system as used for MUC-4”. In: *Proceedings of the Fourth Message Understanding Conference (MUC-4)* (1992), pp. 282–288.

[68] Haizhou Li and Baosheng yuan Yuan. “Chinese word segmentation”. In: *Language, Information and Computation* (1998), pp. 212–217.

[69] Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. “SVM based learning system for information extraction”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 3635 LNAI (2005), pp. 319–339. ISSN: 16113349. DOI: 10.1007/11559887_19.

[70] Liefa Liao and Changpei Yang. “Enterprise risk information extraction based on BERT”. In: *2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP)*. 2022, pp. 1454–1457. DOI: 10.1109/ICSP54964.2022.9778504.

[71] Chen Lin et al. “A BERT-based Universal Model for Both Within- and Cross-sentence Clinical Temporal Relation Extraction”. In: *Proceedings of the 2nd Clinical Natural Language Processing Workshop*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, June 2019, pp. 65–71. DOI: 10.18653/v1/W19-1908.

[72] Chunlei Liu et al. “Assessing the public image of the NIH Clinical and Translational Science Awards (CTSA) program through analysis of publications”. In: *Proceedings - 2016 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2016* (2017), pp. 1207–1213. DOI: 10.1109/BIBM.2016.7822691.

[73] Qi Liu, Matt J Kusner, and Phil Blunsom. “A survey on contextual embeddings”. In: *arXiv preprint arXiv:2003.07278* (2020).

[74] Yinhan Liu et al. “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692* (2019).

[75] Berty Chrismartin Lumban, Immanuel Rhesa Suhendra, and Christian Halim. “Catapa resume parser: End to end Indonesian resume extraction”. In: *ACM International Conference Proceeding Series* (2019), pp. 68–74. DOI: 10.1145/3342827.3342832.

[76] Chengyao Lv et al. “A Novel Chinese Entity Relationship Extraction Method Based on the Bidirectional Maximum Entropy Markov Model”. In: *Complexity* 2021 (2021), p. 8. DOI: 10.1155/2021/6610965.

[77] Christopher D. Manning. *Rethinking Text Segmentation Models: An Information Extraction Case Study*. Tech. rep. University of Sydney, 1998.

[78] James Mayfield, Paul McNamee, and Christine Piatko. “Named entity recognition using hundreds of thousands of features”. In: *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*. 2003, pp. 184–187.

[79] Andrew McCallum. “Information extraction: Distilling structured data from unstructured text”. In: *Queue* 3.9 (2005), pp. 48–57. DOI: 10.1145/1105664.1105679.

[80] Leland McInnes et al. “UMAP: Uniform Manifold Approximation and Projection”. In: *Journal of Open Source Software* 3.29 (2018). DOI: 10.21105/joss.00861.

[81] George A Miller. “WordNet: a lexical database for English”. In: *Communications of the ACM* 38.11 (1995), pp. 39–41. DOI: 10.1145/219717.219748.

[82] Minh Tien Nguyen, Dung Tien Le, and Linh Le. “Transformers-based information extraction with limited data for domain-specific business documents”. In: *Engineering Applications of Artificial Intelligence* 97 (Jan. 2021), p. 104100. ISSN: 09521976. DOI: 10.1016/j.engappai.2020.104100.

[83] *NLP Augmentation*. Available at https://github.com/makcedward/nlpaug. Accessed: 2022-04-09.

[84] *NLTK Documentation*. Available at https://www.nltk.org/api/nltk.tokenize.html. Accessed: 2022-03-14.

[85] Niall O’Mahony et al. “Deep learning vs. traditional computer vision”. In: *Science and information conference*. Springer. 2019, pp. 128–144. DOI: 10.1007/978-3-030-17795-9_10.

[86] Christopher Olah. *Understanding LSTM Networks*. Available at https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2021-13-12.

[87] *Optuna: Optimize Your Optimization*. Available at https://optuna.org/. Accessed: 2022-05-16.

[88] World Health Organization. *How WHO is funded?* Available at https://www.who.int/about/funding. Accessed: 2021-05-12.

[89] Fuchun Peng and Andrew McCallum. “Information extraction from research papers using conditional random fields”. English. In: *Information Processing and Management* 42.4 (2006), pp. 963–979. DOI: 10.1016/j.ipm.2005.09.002.

[90] Nanyun Peng et al. “Cross-sentence N-ary relation extraction with graph LSTMs”. In: *arXiv* 5 (2017), pp. 101–115. ISSN: 23318422. DOI: 10.1162/tacl_a_00049.

[91] Yifan Peng, Qingyu Chen, and Zhiyong Lu. “An Empirical Study of Multi-Task Learning on BERT for Biomedical Text Mining”. In: *Proceedings of the BioNLP 2020 workshop* 1 (2020), pp. 205–214. ISSN: 23318422. DOI: 10.18653/v1/2020.bionlp-1.22.

[92] Jeffrey Pennington, Richard Socher, and Christopher Manning. “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162.

[93] Telmo Pires, Eva Schlinger, and Dan Garrette. “How Multilingual is Multilingual BERT?” In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 4996–5001. DOI: 10.18653/v1/P19-1493.

[94] Cambridge University Press. *Funding in Cambridge Dictionary*. Available at https://dictionary.cambridge.org/dictionary/english/funding. Accessed: 2021-05-12.

[95] Surya Priy. *Clustering in Machine Learning*. Available at https://www.geeksforgeeks.org/clustering-in-machine-learning/. Accessed: 2022-03-14.

[96] A. Maiwald R. Evtimov M. Falli. *Comparing Bidirectional Encoder Representations from Transformers (BERT) with DistilBERT and Bidirectional Gated Recurrent Unit (BGRU)*. Available at https://humboldt-wi.github.io/blog/research/information_systems_1920/bert_blog_post/. Accessed: 2021-13-12.

[97] Red Bull Racing. *Partners*. Available at https://www.redbullracing.com/int-en/partners. Accessed: 2021-05-12.

[98] Payam Refaeilzadeh, Lei Tang, and Huan Liu. “Cross-Validation”. In: *Encyclopedia of Database Systems*. Ed. by Ling Liu and M. Tamer Özsu. New York, NY: Springer New York, 2016, pp. 1–7. ISBN: 978-1-4899-7993-3. DOI: 10.1007/978-1-4899-7993-3_565-2.

[99] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence embeddings using siamese BERT-networks”. In: *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference* (2020), pp. 3982–3992. DOI: 10.18653/v1/d19-1410.

[100] RIBA. *RIBA Validated schools in the UK*. Available at https://www.architecture.com/education-cpd-and-careers/riba-validation/riba-validated-schools-uk. Accessed: 2022-05-26.

[101] Daphne Rietvelt. *Research Topics: Information Extraction from Text*. Tech. rep. University of Twente, 2021.

[102] Ellen Riloff. “Automatically Constructing a Dictionary for Information Extraction Tasks”. In: *Proceedings of the Eleventh National Conference on Artificial Intelligence*. AAAI’93. Washington, D.C.: AAAI Press, 1993, pp. 811–816. ISBN: 0262510715.

[103] Jesús Barrasa Rodríguez, Oscar Corcho, and Asunción Gómez-Pérez. “A semantic portal for fund finding in the EU: Semantic upgrade, integration and publication of heterogeneous legacy data”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4253 LNAI.December 2013 (2006), pp. 588–597. ISSN: 16113349. DOI: 10.1007/11893011_75.

[104] Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. “KUISAIL at SemEval-2020 Task 12: BERT-CNN for Offensive Speech Identification in Social Media”. In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 2054–2059. DOI: 10.18653/v1/2020.semeval-1.271.

[105] Sunita Sarawagi and William W. Cohen. “Semi-markov conditional random fields for information extraction”. In: *Advances in Neural Information Processing Systems* (2005). ISSN: 10495258.

[106] Kristie Seymore, Andrew McCallum, Roni Rosenfeld, et al. “Learning hidden Markov model structure for information extraction”. In: *AAAI-99 workshop on machine learning for information extraction*. 1999, pp. 37–42.

[107] Hyeong Jin Shin et al. “BERT-based Spatial Information Extraction”. In: *Proceedings of the Third International Workshop on Spatial Language Understanding* 8 (2020), pp. 10–17. DOI: 10.18653/v1/2020.splu-1.2.

[108] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. *The Performance of LSTM and BiLSTM in Forecasting Time Series*. 2019. DOI: 10.1109/BigData47090.2019.9005997.

[109] Gonçalo Simoes, Helena Galhardas, and Luısa Coheur. “Information Extraction tasks: a survey”. In: *Simpósio de Informática 2009* (2009), p. 540.

[110] Sklearn. *Scikit-learn: Machine Learning in Python*. Available at https://scikit-learn.org/stable/. Accessed: 2022-05-26.

[111] Sklearn. *Sklearn: Logistic Regression*. Available at https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html. Accessed: 2022-05-26.

[112] Sklearn. *Sklearn: Random Forest*. Available at https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed: 2022-05-26.

[113] Andrew Smith and Miles Osborne. “Using gazetteers in discriminative information extraction”. In: *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*. 2006, pp. 133–140.

[114] Stephen Soderland et al. “CRYSTAL inducing a conceptual dictionary”. In: *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 2*. 1995, pp. 1314–1319.

[115] Oswaldo Solarte-Pabón et al. “Information extraction from Spanish radiology reports using multilingual BERT”. In: *CLEF eHealth* (2021).

[116] Standford. *GloVe: Global Vectors for Word Representation*. Available at https://nlp.stanford.edu/projects/glove/. Accessed: 2022-05-23.

[117] Yanyuan Su, Jian Zhang, and Jianhao Lu. “The Resume Corpus: A Large Dataset for Research in Information Extraction Systems”. In: *2019 15th International Conference on Computational Intelligence and Security (CIS)*. 2019, pp. 375–378. DOI: 10.1109/CIS.2019.00087.

[118] Chi Sun et al. “How to Fine-Tune BERT for Text Classification?” In: *Chinese Computational Linguistics*. Ed. by Maosong Sun et al. Cham: Springer International Publishing, 2019, pp. 194–206. ISBN: 978-3-030-32381-3. DOI: 10.1007/978-3-030-32381-3_16.

[119] Jie Tang et al. “Information extraction: Methodologies and applications”. In: *Emerging Technologies of Text Mining: Techniques and Applications* (2008), pp. 1–33. DOI: 10.4018/978-1-59904-373-9.ch001.

[120] *Trainer module HuggingFace*. Available at https://huggingface.co/docs/transformers/main_classes/trainer. Accessed: 2022-05-16.

[121] Henry Tsai et al. “Small and Practical BERT Models for Sequence Labeling”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3632–3636. DOI: 10.18653/v1/D19-1374.

[122] *UMAP Documentation: Basic UMAP Parameters*. Available at https://umap-learn.readthedocs.io/en/latest/parameters.html. Accessed: 2022-03-23.

[123] Ashish Vaswani et al. “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[124] Antti Virtanen et al. “Multilingual is not enough: BERT for Finnish”. In: *arXiv e-prints* (2019), arXiv–1912.

[125] Bernhard Waltl, Georg Bonczek, and Florian Matthes. “Rule-based information extraction: Advantages, limitations, and perspectives”. In: *Jusletter IT* February (2018). ISSN: 1664848X.

[126] Tianyu Wan, Wenhui Wang, and Hui Zhou. “Research on Information Extraction of Municipal Solid Waste Crisis Using BERT-LSTM-CRF”. In: *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 205–209. ISBN: 9781450377607. DOI: 10.1145/3443279.3443314. URL: https://doi.org/10.1145/3443279.3443314.

[127] Yan Wang, Yacine Allouache, and Christian Joubert. “Analysing CV Corpus for Finding Suitable Candidates using Knowledge Graph and BERT”. In: *DBKDA 2021, The Thirteenth International Conference on Advances in Databases, Knowledge, and Data Applications*. 2021.

[128] Daya C. Wimalasuriya and Dejing Dou. “Ontology-based information extraction: An introduction and a survey of current approaches”. In: *Journal of Information Science* 36.3 (2010), pp. 306–323. DOI: 10.1177/0165551509360123.

[129] Dongkuan Xu and Yingjie Tian. “A Comprehensive Survey of Clustering Algorithms”. In: *Annals of Data Science* 2.2 (2015), pp. 165–193. ISSN: 2198-5804. DOI: 10.1007/s40745-015-0040-1.

[130] Kui Xue et al. “Fine-tuning BERT for joint entity and relation extraction in Chinese medical text”. In: *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2019, pp. 892–897. DOI: 10.1109/BIBM47256.2019.8983370.

[131] Haitao Yu et al. “Relation extraction with bert-based pre-trained model”. In: *2020 International Wireless Communications and Mobile Computing (IWCMC)*. 2020, pp. 1382–1387. DOI: 10.1109/IWCMC48107.2020.9148384.

[132] Kun Yu, Gang Guan, and Ming Zhou. “Resume information extraction with cascaded hybrid model”. In: *ACL-05 - 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference* June (2005), pp. 499–506. DOI: 10.3115/1219840.1219902.

[133] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. “Kernel Methods for Relation Extraction”. In: *Journal of Machine Learning Research* 3.6 (2003), pp. 1083–1106.

[134] Dmitry Zelenko, Chinatsu Aone, and Jason Tibbetts. “Coreference Resolution for Information Extraction”. In: *Proceedings of the Conference on Reference Resolution and Its Applications*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 24–31.

[135] Ying Zeng et al. “A Convolution BiLSTM Neural Network Model for Chinese Event Extraction”. In: *Natural Language Understanding and Intelligent Applications*. Ed. by Chin-Yew Lin et al. Cham: Springer International Publishing, 2016, pp. 275–287. ISBN: 978-3-319-50496-4. DOI: 10.1007/978-3-319-50496-4_23.

[136] Nancy R Zhang. “Hidden markov models for information extraction”. In: *Technical Report. Stanford Natural Language Processing Group* (2001).

[137] Peng Zhang et al. “TRIE: end-to-end text reading and information extraction for document understanding”. In: *Proceedings of the 28th ACM International Conference on Multimedia*. 2020, pp. 1413–1422. DOI: 10.1145/3394171.3413900.

[138] Ruixue Zhang et al. “Rapid adaptation of BERT for information extraction on domain-specific business documents”. In: *arXiv* July (2020). ISSN: 23318422. DOI: 10.48550/arXiv.2002.01861.

[139] Yukun Zhu et al. “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 19–27. DOI: 10.1109/ICCV.2015.11.

[140] Shicheng Zu and Xiulai Wang. “Resume Information Extraction with A Novel Text Block Segmentation Algorithm”. In: *International Journal on Natural Language Computing* 8.5 (2019), pp. 29–48. ISSN: 23194111. DOI: 10.5121/ijnlc.2019.8503.

Appendix A

Background in BERT

RNN and (Bi)LSTM

Before transformer-based machine learning algorithms were introduced, such as the BERT model [34], LSTM models and, more specifically, BiLSTM models were the best performing neural network models for information extraction. LSTM models are based on recurrent neural networks (RNNs). As visualized in Figure A.1, recurrent neural networks have a loop. This loop allows information captured in one step to be used in future steps. How long and what kind of information is kept depends on the weights of the layer and the input data. Recurrent neural networks are often used for understanding and predicting natural language. For example, predicting the next word in the sentence: "The grass has the color ...". The available information is forwarded between steps, as visualized in Figure A.2, to predict the next word, which in our example is the word "green". The "available information" (grass) and its relation to the "to be predicted word" (green) is called a dependency. In this example, the distance between the available information and the prediction is relatively small. However, this is not always the case. These dependencies can become very large, also called long-term dependencies. In such situations, RNNs are not able to connect the past information with the prediction and, therefore not able to detect the dependency.

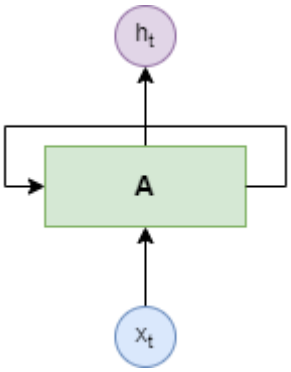

Figure A.1: Recurrent Neural Network (RNN) adapted from [86]

Long Short Term Memory neural networks, for short LSTMs, also use the recurrent neural network but are capable of handling these long-term dependencies. The long-term dependencies can be better managed by LSTMs, because available information (context) is kept for longer in a cell state (C_t). A visualization of a LSTM model can be found in Figure A.3. At every step the LSTM runs three processes: 1) Based on the previous prediction (H_{t-1}) and the new input information (X_t), it is determined what information from the cell state (C_{t-1}) can be removed. 2) It is determined what information from the new input data (X_t) to add to the

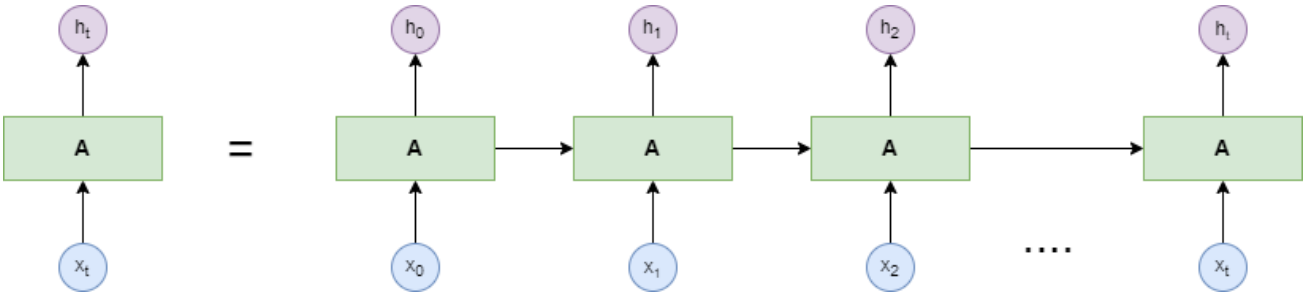

Figure A.2: Rolled out RNN adapted from [86]

cell state. 3) The cell state is updated, old information is removed, and new information is added as decided in the previous steps. 4) The output (prediction) is determined. In our example, this was the next word in a sentence. In processes 1 and 2, the relevance of keeping the information is determined. This is calculated using a sigmoid function, which outputs a float between 0 (do not keep this at all) and 1 (entirely sure to keep). The information below a certain threshold is removed, and above a certain threshold, information is added to the cell state.

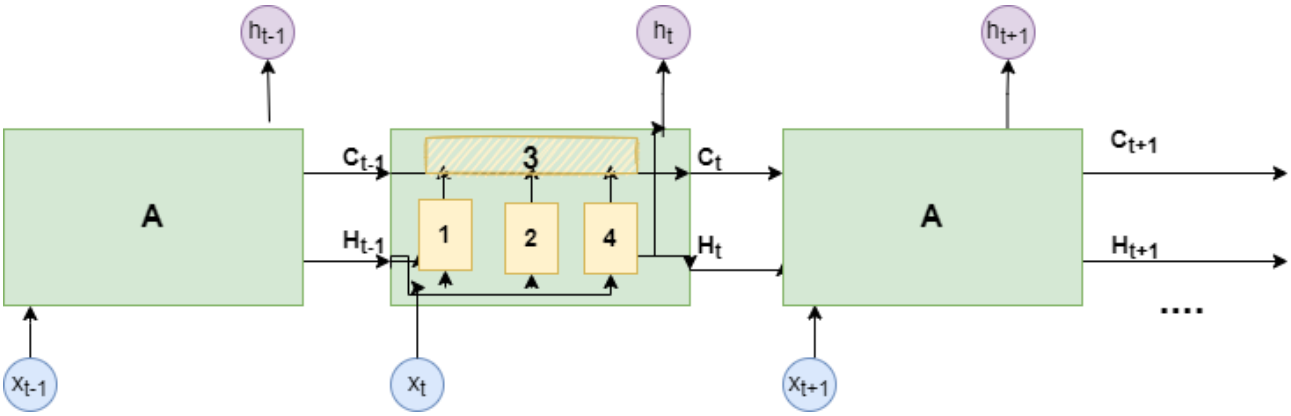

Figure A.3: LSTM adapted from [86]

LSTMs can understand and predict natural language with these processes. However, the drawback of LSTMs is that these models can only handle the available (past) context (information on the left). If the information on the right (future) could also be used, the model's context increases, resulting in a more accurate understanding and predictions. This suggestion was implemented in BiLSTM neural networks, which combined two LSTMs: 1) an LSTM that takes the context on the right (forward) and 2) an LSTM that takes the context on the left (backward). These combined allow for more context as input and increase the model's performance. A visualization of the BiLSTM can be found in Figure A.4.

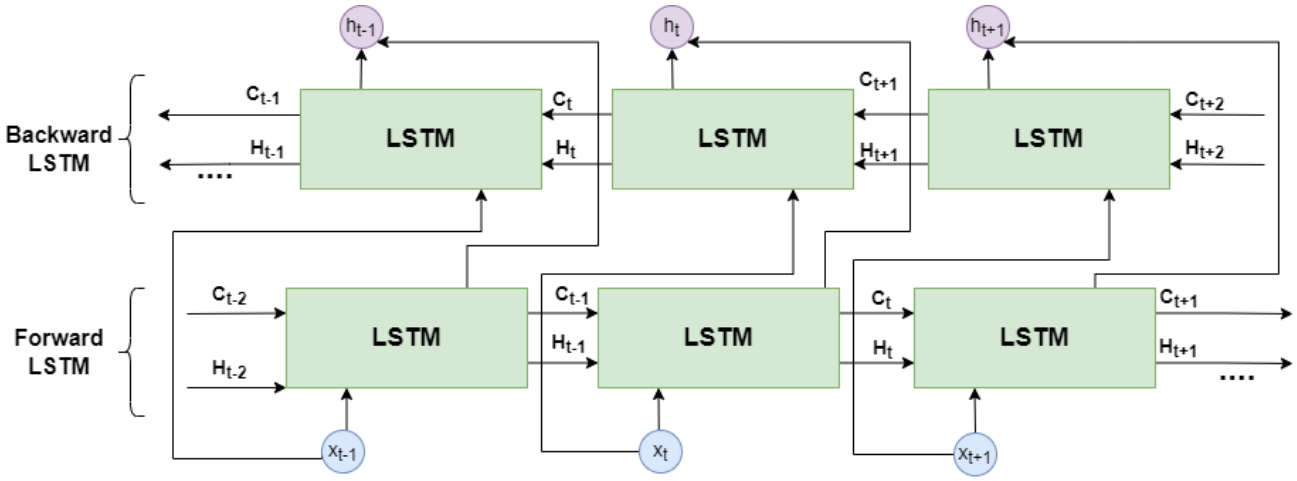

Figure A.4: BiLSTM adapted from [65, 86]

Transformers and BERT

Combining two neural networks to manage the bi-directionality of natural language is also used in an encoder-decoder structure. An encoder-decoder structure uses a recurrent neural network to convert natural language text into a context vector and a second recurrent neural network to put the context vector back into natural language, but in a different format. This encoder-decoder structure is often used for translating text (machine translation). An example encoder-decoder structure for machine translation is shown in Figure A.5.

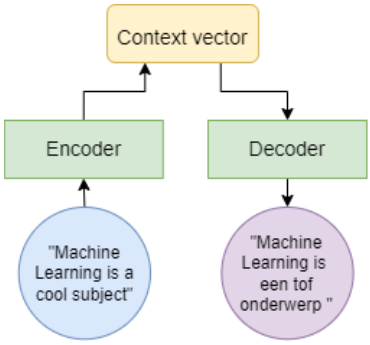

Figure A.5: Encoder-Decoder structure

Vaswani et al. [123] proposed the first transformers model in 2017. The transformer model uses the encoder-decoder structure. However, the encoder-decoder structure of a transformer model does not consist of a recurrent neural network. Instead, it uses an attention mechanism combined with a simple feed-forward neural network. This combination showed such good results in the research by Vaswani et al. [123] that the computationally complex recurrent neural networks could be replaced by the attention mechanism [123]. What is this so-called “attention mechanism”? In a recurrent neural network, it is only determined what kind of information (context) should be saved, but not the importance of the information. The attention mechanism calculates the importance of all words to all other words. Words which are part of a dependency, such as the words “It” and “drinking tea” in the sentences: “I love drinking tea. It brings me joy”, are linked together, and their importance to one another is therefore high. By adding the importance between words to the context vector, a better understanding of the meaning of the text can be formed.

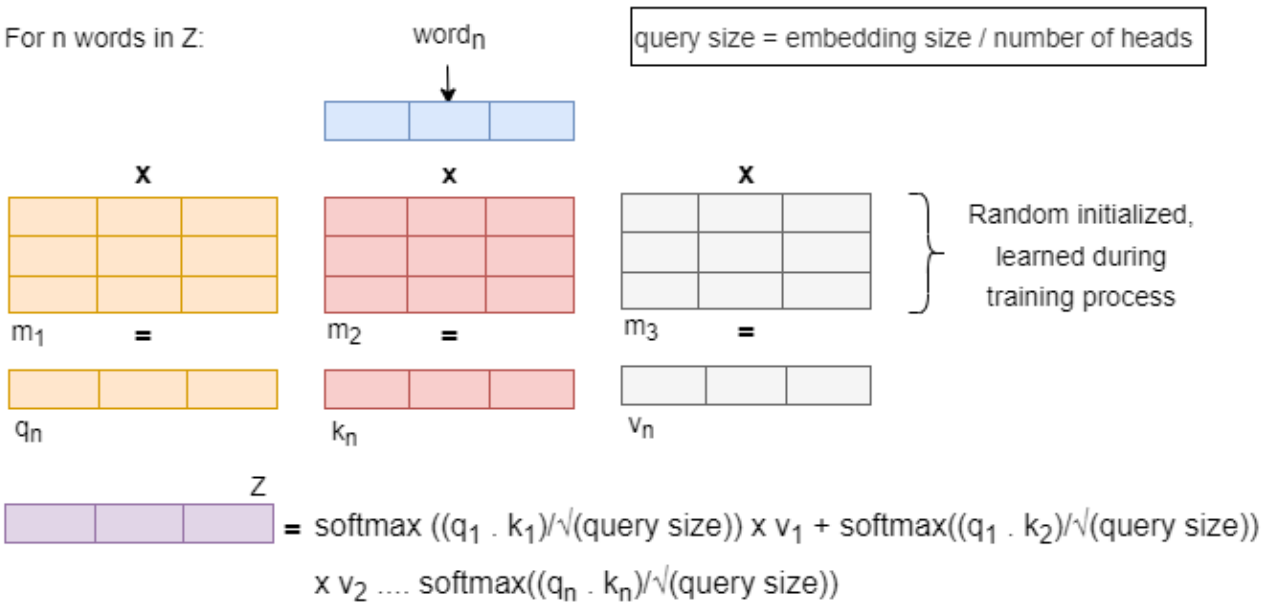

Figure A.6: Attention calculation adapted from [96]

How is this attention calculated? As visualized in Figure A.6, for each word in a sentence (Z), the context vector of that word in that sentence is multiplied with three matrices, which results in three new vectors (query, key, value). The matrices are randomly initialized and learned during the training process. The dot products of q and the k of all other words (k_1 ... k_n) are calculated and divided by the square root of the query size. A softmax is applied to each of these dot products and multiplied with the corresponding value vector (v). These values are summed for a sentence and result in a new context vector for sentence Z, which includes the importance of words.

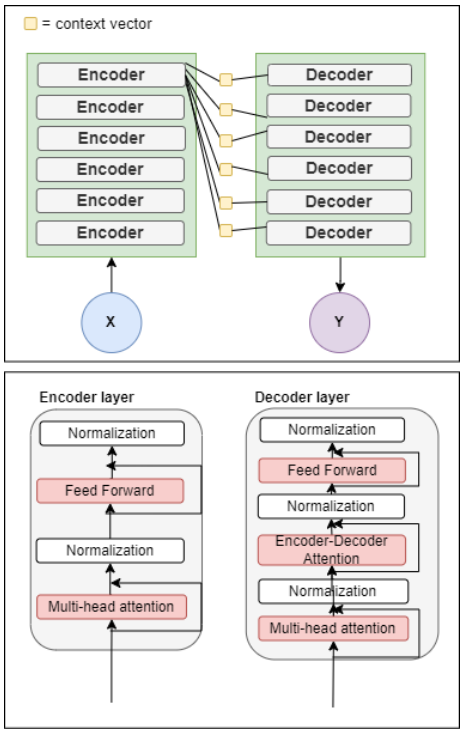

Figure A.7: Transformer model used in [123]

As explained by Vaswani et al. [123], the attention mechanism is multi-headed. Multi-head attention means that the attention calculation is done multiple times with different randomly initiated matrices. As a result, multiple new importance-informed context vectors are outputted. To get to one vector, the model learns a weight matrix, similar to the matrices in the attention calculation, which determines what information is included from which context vector (atten-

tion heads). This attention mechanism is explained to be part of the encoder. However, the decoder uses a similar kind of multi-head attention mechanism as well as the "Encoder-Decoder attention" to decode the context vector into natural language. The details on the difference between these different kinds of attention are lengthily explained in the series of posts by Ketan Doshi [35]. Like any other neural network, a transformer model requires multiple layers to effectively learn the weights. Therefore, multiple layers are linked. The transformer model used in Vaswani et al. [123] is visualised in Figure A.7.

BERT

BERT was published in 2019 by Devlin et al. [34] and shows state-of-the-art results in numerous natural language understanding tasks. BERT is based on the work by Vaswani et al. [123], but instead of an encoder-decoder structure, only the encoder side of the model is used. The encoder in the BERT base model consists of 12 encoder layers, while most transformer models have 6 encoder layers. The encoder layers output a context vector often combined with a feed-forward layer to perform classification or any other form of output based on the context vector. BERT performs so well in numerous natural language understanding tasks because of the amount of training done on BERT. BERT is trained on a large dataset consisting of BooksCorpus [139] (800 million words) and English Wikipedia (2500 million words). The training consisted of two tasks: masked language modeling and next sentence prediction.

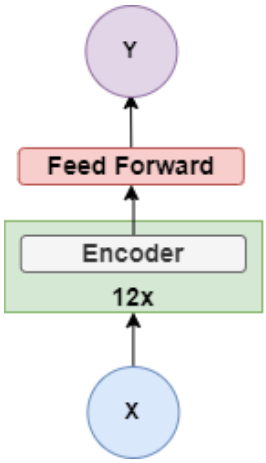

Figure A.8: BERT model published in [34]

1) In masked language modeling, a part of the word tokens in the input text are randomly chosen and masked. The task for the model is to predict the masked word tokens. The prediction is based on the surrounding words in the form of a context vector, which contains the word's meaning. To ensure that the model also includes the word itself, 80% of the word tokens to be masked are replaced with the [MASK], 10% with a random token, and the other 10% are kept original.

2) In the next sentence prediction, the model is given two sentences (A and B) and is asked whether sentence B follows sentence A directly or not. In 50% of the cases, this is the case, and in 50% of the cases, it is not. Such a simple task helps the BERT model to understand the relation between pieces of text.

To use this pre-trained transformer model (BERT) for your specific task, it requires some additional training, also called fine-tuning. The fine-tuning of the model to fit your task requires some data of your task with their corresponding outputs from which the BERT model can learn. The self-attention mechanism in the encoder layers easily captures the features

of interest for your specific task based on the expected output. The fine-tuning task is inexpensive, and limited training data is required compared to other neural networks because of the great natural language understanding knowledge obtained during the large pre-training process. After fine-tuning, the BERT model is usable for your specific task.

Appendix B

Results HDBSCAN clustering

UMAP Parameters			HDBSCAN Parameters			Score
Number of Neighbors	**Number of Components**	**Metric**	**Minimum Samples**	**Minimum Cluster Size**	**Cluster Selection Method**	**Relative Validity**
15	2	canberra	2	100	eom	0.562
15	2	cosine	4	200	eom	0.823
15	2	euclidean	100	300	eom	0.516
15	10	canberra	15	200	eom	0.791
15	10	cosine	100	100	eom	0.789
15	10	euclidean	15	200	eom	0.784
15	50	canberra	4	200	eom	0.695
15	50	cosine	4	15	eom	0.296
15	50	euclidean	2	5	eom	0.297
50	2	canberra	100	15	eom	0.542
50	2	cosine	2	5	eom	0.549
50	2	euclidean	4	200	eom	0.608
50	10	canberra	50	100	eom	0.285
50	10	cosine	2	5	eom	0.606
50	10	euclidean	2	100	eom	0.259
50	50	canberra	4	200	eom	0.265
50	50	cosine	4	5	eom	0.587
50	50	euclidean	2	200	eom	0.399
100	2	canberra	100	5	eom	0.393
100	2	cosine	2	5	eom	0.532
100	2	euclidean	100	200	eom	0.788
100	10	canberra	50	200	eom	0.474
100	10	cosine	4	5	eom	0.590
100	10	euclidean	15	50	eom	0.222
100	50	canberra	2	100	eom	0.222

100	50	cosine	2	5	eom	0.591
100	50	euclidean	4	50	eom	0.232
20	2	canberra	100	15	eom	0.612
200	2	cosine	2	5	eom	0.535
200	2	euclidean	50	5	eom	0.519
200	10	canberra	50	100	eom	0.373
200	10	cosine	4	5	eom	0.601
200	10	euclidean	2	15	eom	0.295
200	50	canberra	50	200	eom	0.459
200	50	cosine	4	5	eom	0.583
200	50	euclidean	2	5	eom	0.720

Table B.1

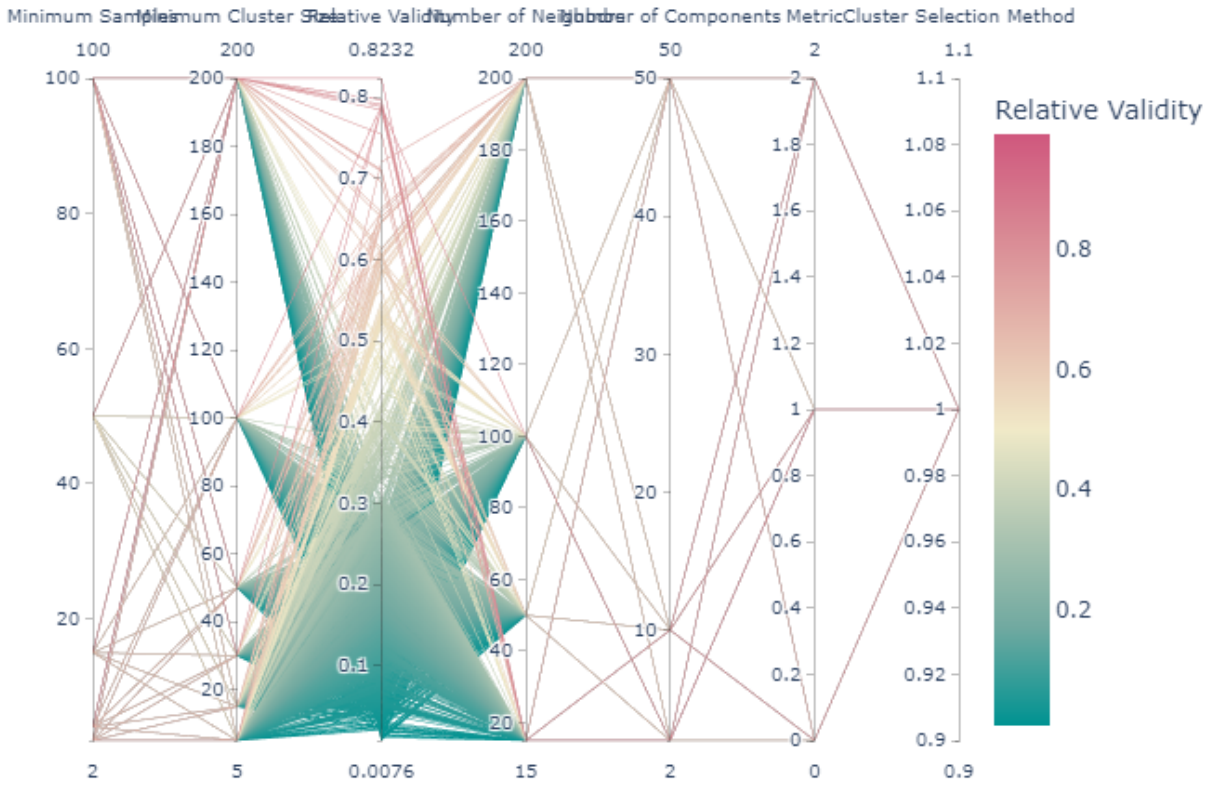

Figure B.1: Parallel plot of clustering

Appendix C

Annotation Process

The annotation process was done in excel, where the annotation protocol (Screen 2, see Figure C.1) was displayed in another excel tab than the to be annotated funding opportunities or sentences. Details on the Annotation Protocol can be found below in Section C.1. The funding opportunities or sentences were displayed row-wise, and the to-be-inserted labels were inserted column-wise. A visualization can be found in Figure C.1.

C.1 Annotation Protocol

Text as introduction for annotating sentences

The data presented in sentences come from the funding data that is used for recommending funding opportunities to our customers. The goal of this annotation effort is to improve the quality of the data by information extraction (classification) using sentences. Therefore, it would be great if this data was annotated so that we could improve our classifiers. We would like to know information on a specific part of a funding opportunity, namely the individual applicant type. Individual applicant type defines what education levels are required to apply for the funding opportunity, and at what stage in the career life the applicant needs to be.

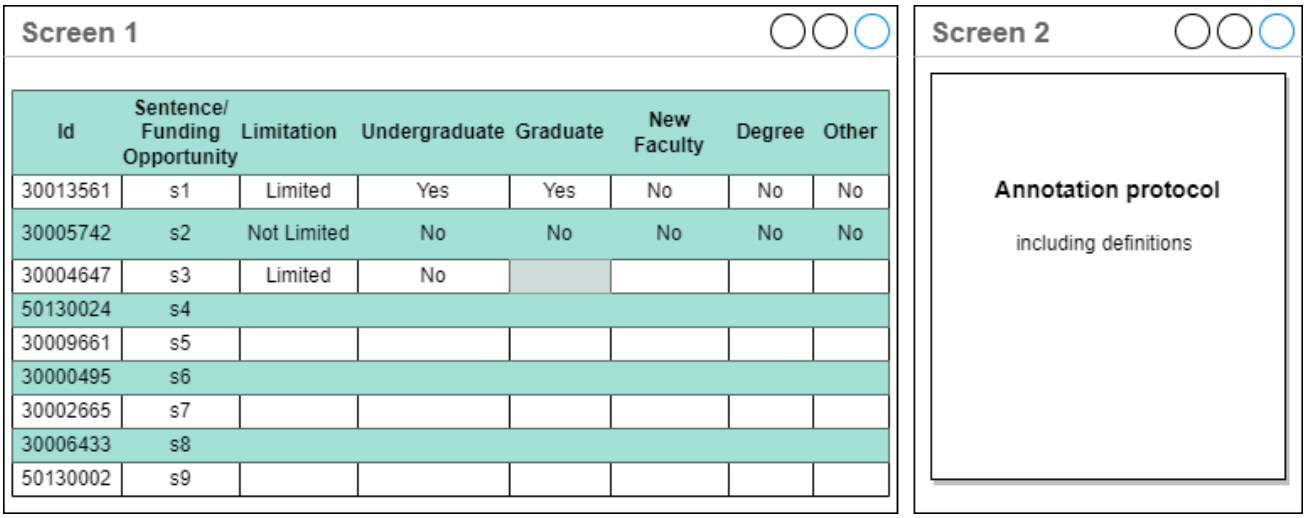

Figure C.1: Visualization of Annotation

Text as introduction for annotation funding opportunities

The data presented in funding opportunities is used for recommending funding opportunities to our customers. The goal of this annotation effort is to evaluate a system that improves the quality of the data by information extraction (classification). It would be great if data is annotated so that we can evaluate the system created. The system focuses on improving the Individual Applicant Type, so that is the type of information we are interested in. Individual applicant type defines at what stage in the career life the applicant needs to be.

Part of labeling guide that is shared between annotating funding opportunities and sentences

As the name 'Individual Applicant Type' already suggests, this information applies only to funding opportunities that allow individuals to apply. Such individuals can be independent researchers or standalone artists. This is in contrast to funding opportunities for organizations, such as research groups or businesses that can apply.

This project focuses on *individual funding opportunities*. Therefore, if you find a funding opportunity or sentence that implies that it is an organizational funding opportunity, please specify it as *Organization*. We would like to know some information on applicant type:

1. whether the funding opportunity or sentence contains information on individual eligibility based on individual applicant type. This is different from the 'Limitation' value that is used in most daily processes within Elsevier. Possible values that are given to the sentence are:

    - **Limited** - there is evidence in the funding opportunity or sentence that there is a limitation to who can apply based on individual applicant type
    - **Not Limited** - there is evidence in the funding opportunity or sentence that there is no limitation in who can apply based on individual applicant type
    - **Not Specified** - the funding opportunity or sentence does not specify anything on individual applicant type.
    - **Organization** - the funding opportunity or sentence indicates that the Opportunity is only open for Organizations to apply.

    A funding opportunity or sentence can only have one of these values and can be filled out using a drop-down list in the column Limitation.

2. **In cases where there is a limitation (value limited)**, the last five columns of the dataset are also required to be filled out, which specify what kind of limitation there

is. Fill out for which stages in a career the funding opportunity is open to apply: undergraduate, graduate, new faculty, degree, or other. Fill the value 'Yes' (via drop-down menu) in when the funding opportunity is open to that stage in the career and 'No' (see drop-down list) when the funding opportunity is not open to that stage in the career. Note that multiple columns can be checked yes and/or No at the same time. The definitions of the different values are explained below.

Below the values are explained:

- **Undergraduate** - Opportunities that specifically mention that bachelor or undergraduate students may apply
- **Graduate** - Opportunities that specifically mention that graduate students may apply. This is also used for undergraduate students who are completing a degree and seeking funding for continuing graduate studies. This can include Master's students and Doctoral students.
- **New Faculty** - Opportunities for which new faculty who are considered inexperienced or emerging may apply. This can include postdoctoral applicants, new, young, emerging researchers, early career investigators, or junior faculty applicants. This classification is used across all disciplines, such as emerging artists or young investigators in cardo medicine.
- **Degree** - Opportunities for which an applicant has obtained a degree or is experienced (researcher) (if stated in the Opportunity)
- **Other** - Opportunities for which individuals can apply, but none of the above categories apply

Appendix D

Hyperparameter tuning results for BERT and BiLSTM

D.1 Hyperparameter tuning results: Obtain segments that contain information on 'Individual Applicant Type'

The results of the hyperparameter tuning for both the BERT implementation and the BiLSTM implementation of classification task 1 can be found in parallel plots in Figure D.1 and Figure D.2. The parallel plots show the different trials (set of parameters) as lines. The objective value (f1-score) is the parameter on the left, on which the hyperparameter search is optimized. The parameters are presented to the right of the objective value on the x-axis. A line (trial) travels through each parameter, and the height at which the line crosses a parameter indicates the parameter value. This results in a set of parameters represented by the line. The height at which the line crosses the objective parameter indicates the value of the objective (f1-score) of that set of parameters. The value of the objective parameter is also visually represented by the darkness of the line. The darker the color of the line, the higher the objective value.

Figure D.1 shows the parallel plot of the BERT implementation. What stands out is the many dark lines around a learning rate between 1e-3 and 1e-5. A similar pattern can be found for the seed parameter with values between 1000 and 2000 and for the weight decay parameter between 1e-4 and 1e-5. The pattern of ranges for parameters is also found in the hyperparameter search of the BiLSTM implementation in Figure D.2. For example, a range of the learning rate between 1e-3 and 1e-4, and number of layers between 2 and 6. These ranges of many dark lines indicate that the optimal value of that parameter is likely to be in that range. This statement is confirmed with the optimal set of parameters which can be found in Table 8.1 for both BERT and BiLSTM in Chapter 8, section 8.1.

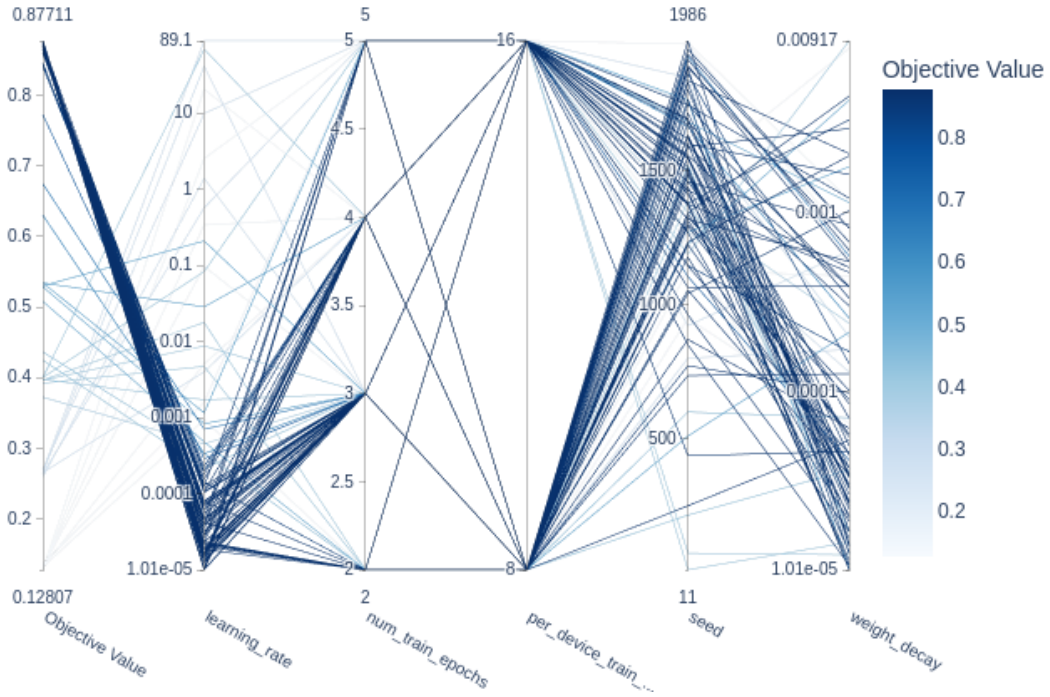

Figure D.1: Hyperparameter results for the BERT implementation of classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

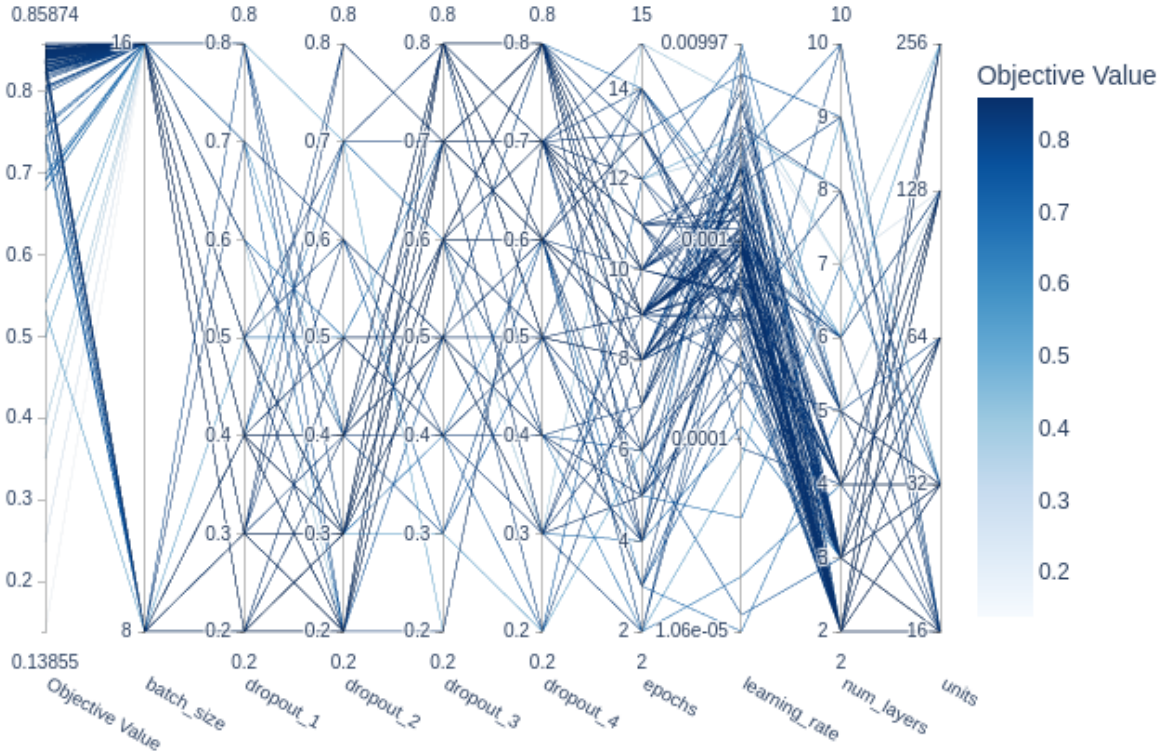

Figure D.2: Hyperparameter results for the BiLSTM implementation of classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

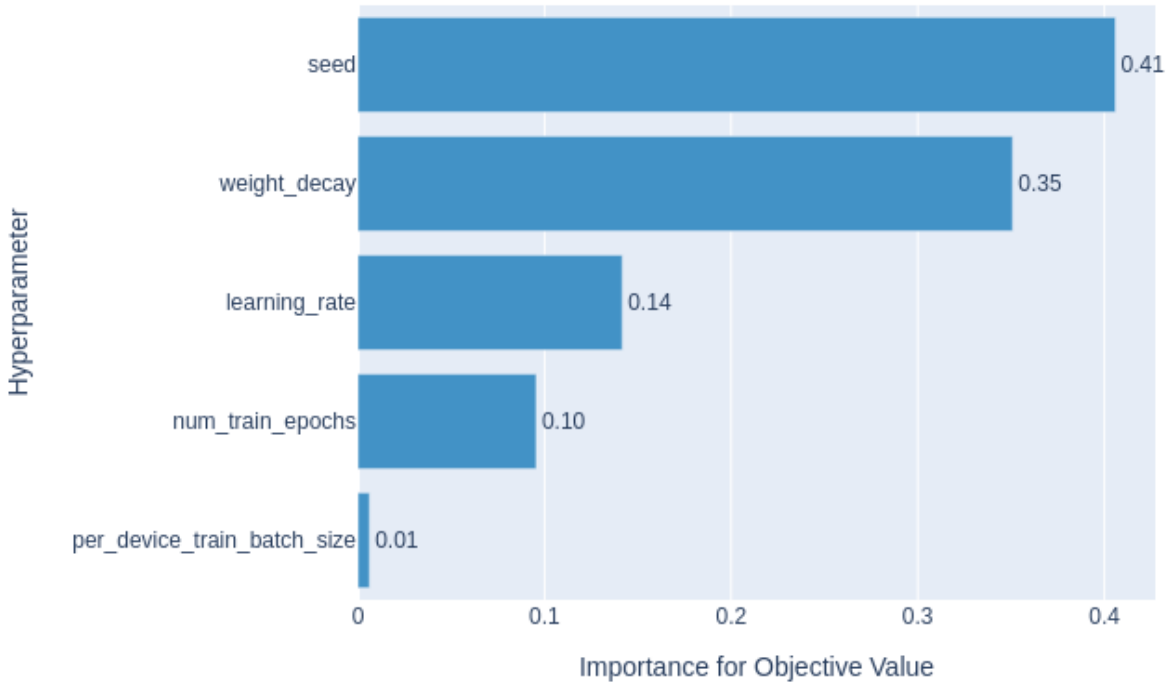

Figure D.3: Hyperparameter importance for the BERT implementation of classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

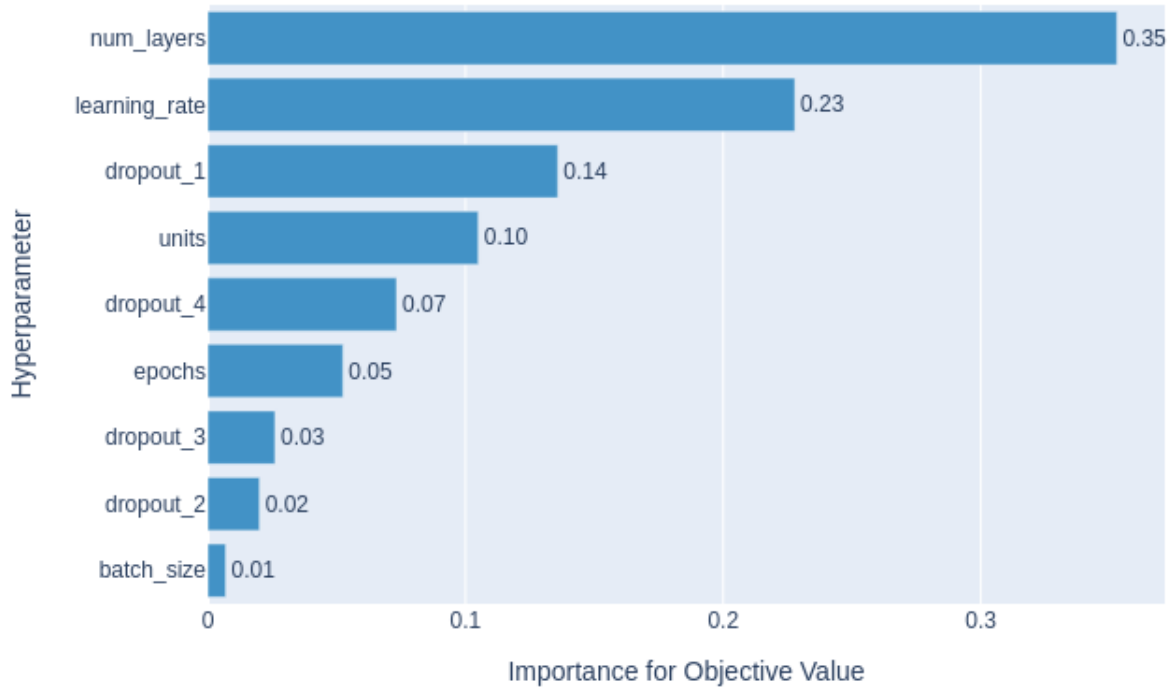

Figure D.4: Hyperparameter importance for the BiLSTM implementation of classification task 1: Obtain segments that contain information on 'Individual Applicant Type'

For the hyperparameter searches, the importance of parameters to obtaining a high objective value (f1-score) is calculated using the fANOVA evaluation algorithm [54] based on the completed trials. Figure D.3 and Figure D.4 show the importance of the parameters of the hyperparameter search of the BERT implementation and BiLSTM implementation respectively. For the BERT implementation, the parameters seed and weight decay are the most important parameters and together account for 76% of the influence of the parameters on the objective value. In the BiLSTM implementation, the number of layers and the learning rate are the most important parameters. They together account for 58% of the influence of the parameters on the objective value.

What is interesting to observe is that parameters that have clear ranges in which optimal values are likely to fall, such as the seed parameter for the BERT implementation or the number of layers for the BiLSTM implementation classification 1, play an important role (high importance value) in obtaining a high objective value. This observation can be logically explained. The effect of important parameters on obtaining a high objective value is large, resulting in specific value ranges of the parameter that result in high objective values and specific value ranges of the parameter that result in low objective values. These values of high and low objective values are visualized in the parallel plots as the ranges.

D.2 Hyperparameter tuning results: Obtain segments that contain information on a limitation

The full results on the 100 trials of the hyperparameter searches are visualised in parallel plots in Figure D.5 and Figure D.6 for BERT and BiLSTM, respectively. Figure D.5 shows many dark color lines around the learning rate parameter with values between 1e-3 and 1e-5. The same pattern is observed with the parameter number of training epochs with values 3 to 5, a batch size of 8, and the seed parameter between 1000 and 2000. Also, in the BiLSTM implementation, parameters show many dark-colored lines around specific values, namely the number of layers with values between 2 and 6, and number of epochs with values above 9, see Figure D.6. The dark-colored lines indicate high objective scores (f1-scores), showing ranges of values that are likely to contain optimal parameter values. The assumption of the ranges is confirmed by the optimal set of parameters that fall into the specified ranges above, as visualized in Table **??** and **??** for BERT and BiLSTM, respectively.

Similar to the hyperparameter searches of the first classification task, the importance of parameters for obtaining a high objective score (f1-score) is calculated. The important results of the BERT implementation are shown in Figure D.7. The most important parameters for the BERT implementation are the seed parameter, the number of training epochs, and weight decay, which together account for 89% of the variability of objective value (f1-score). The most important parameter for the BiLSTM implementation is the number of layers, which accounts for 62% of the variability of objective value (f1-score), as visualized in Figure D.8.

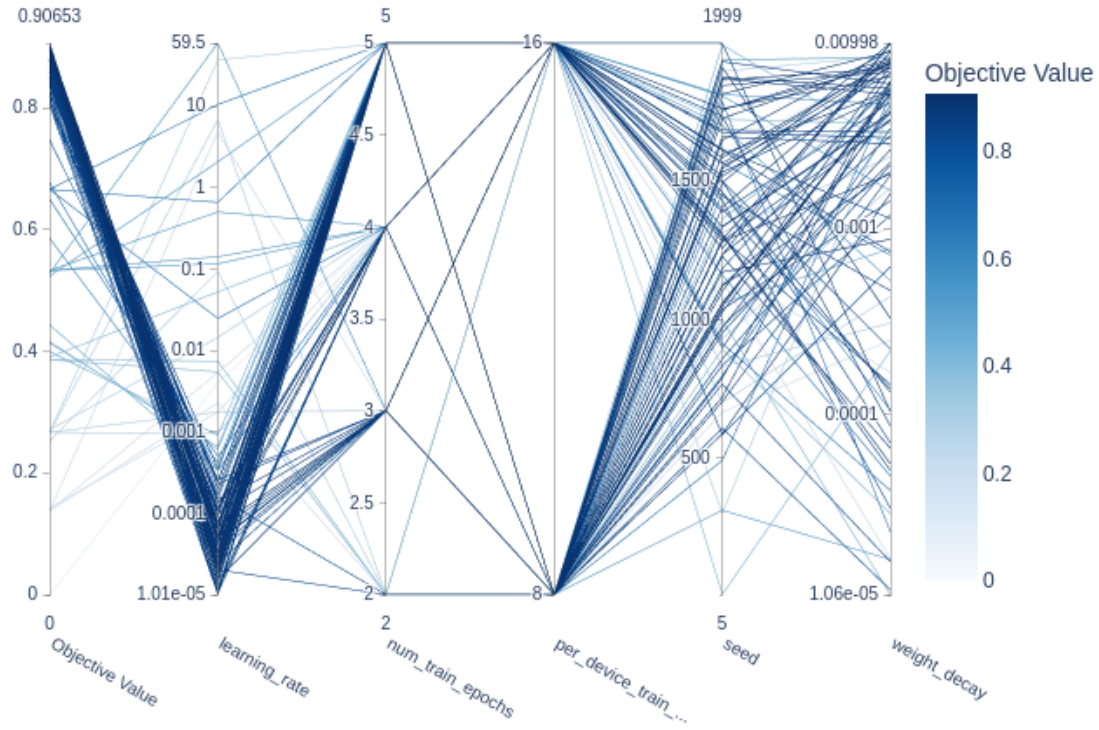

Figure D.5: Hyperparameter results for the BERT implementation of classification task 2: Obtain segments that contain information on a limitation

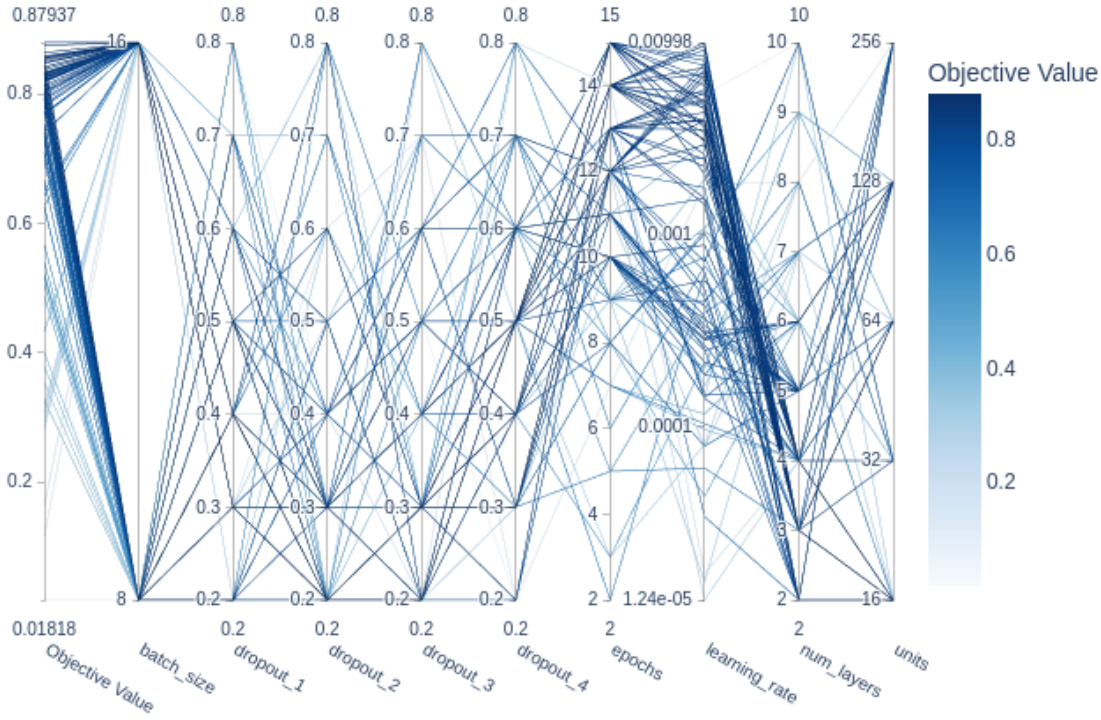

Figure D.6: Hyperparameter results for the BiLSTM implementation of classification task 2: Obtain segments that contain information on a limitation

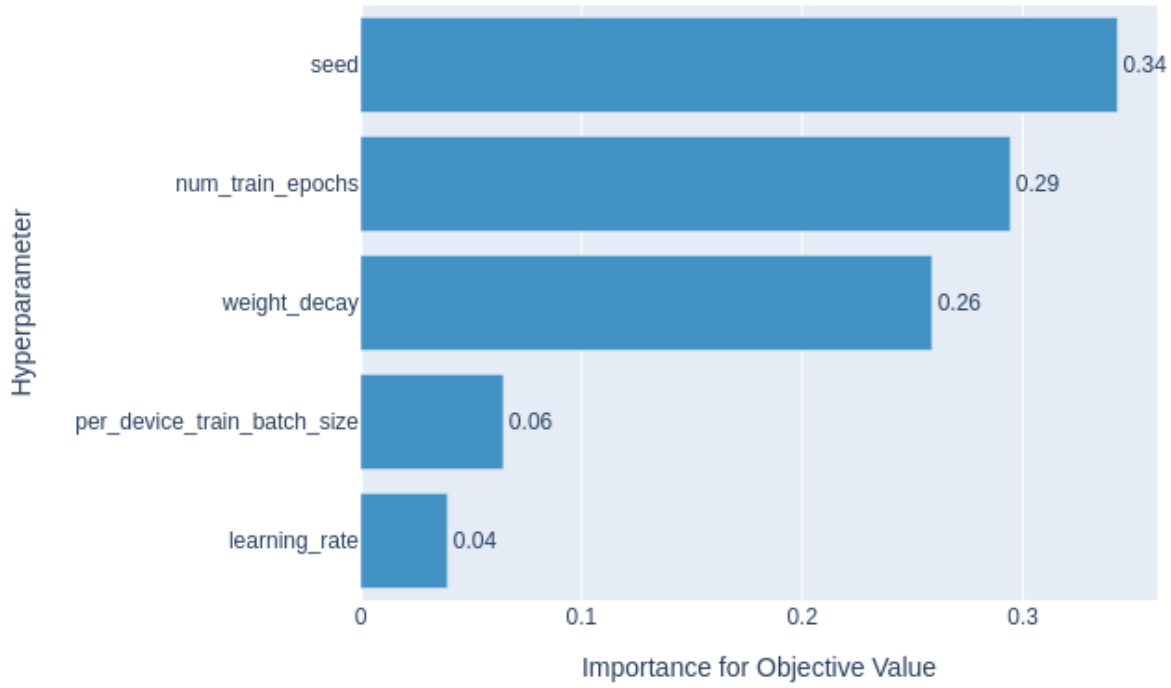

Figure D.7: Hyperparameter importance for the BERT implementation of classification task 2: Obtain segments that contain information on a limitation

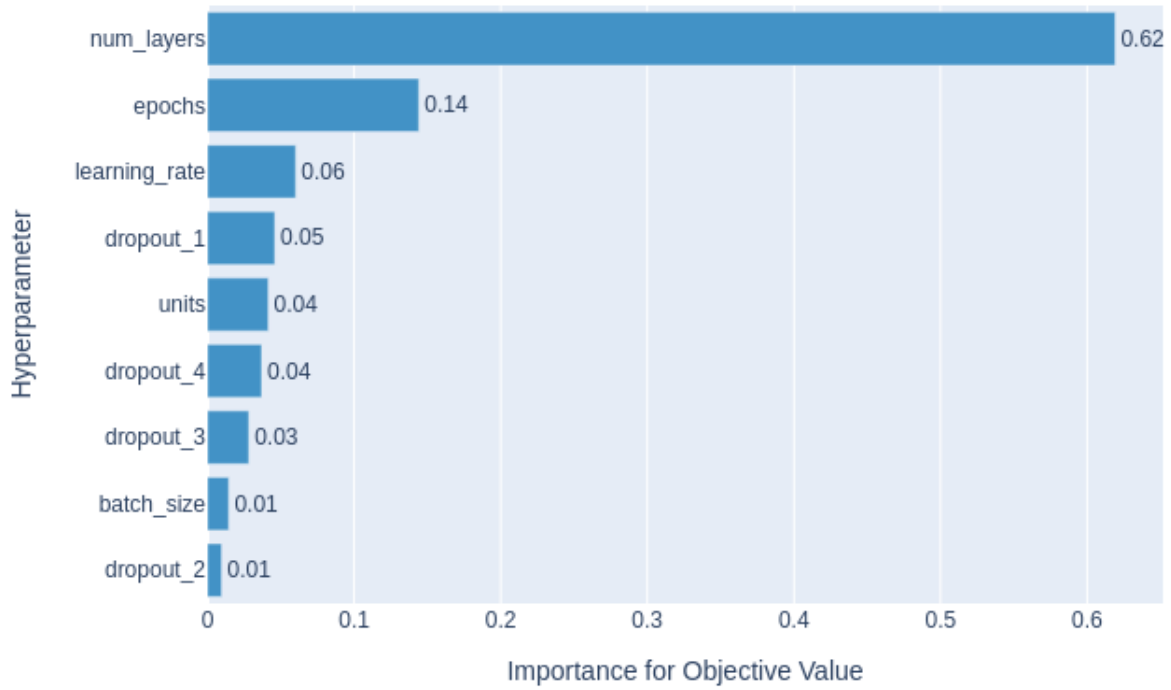

Figure D.8: Hyperparameter importance for the BiLSTM implementation of classification task 2: Obtain segments that contain information on a limitation

A similar pattern as found in the first classification task can be found in the hyperparameter results. Most parameters with clear ranges in which optimal values are likely to fall also have high importance to the objective value. This is true for, for example, the parameters seed, and number of training epochs for the BERT implementation, and the number of layers for the BiLSTM implementation.

D.3 Hyperparameter tuning results: Obtain career stages that may apply

The full results of all trials are visualised in two parallel plots: Figure D.9 for BERT and Figure D.10 for BiLSTM. Figure on BERT D.9 shows many dark-colored lines (high objective trials) in a learning rate range of values between 1e-3 and 1e-5, and the number of training epochs of 3 or 5. For the BiLSTM hyperparameter, search the high objective lines in the number of layers ranging from 2 to 5 and epochs above 10. The number of units shows a slight preference between 16 and 64. The optimal values for the parameters are likely to be in these ranges. This statement is confirmed with the optimal set of parameters visualised in Table 8.5 for BERT and Table D.10 for BiLSTM.

Similar to the two classification tasks, the importance of the parameters to the objective value (f1-score) was calculated. For BERT, the parameters seed, number of training epochs, and the learning rate show the largest contributions to a high objective value (38%, 23%, and 19% respectively), while batch size contributes the least (5%) as visualized in Figure D.11. The BiLSTM's most important parameter is the number of layers with an influence of 79%. The parameters with the second and third highest importance are epochs and units with the importance of 10% and 2%, respectively. The batch size, learning rate, and the dropout values contribute the least (either 1% or 2%) as visualized in Figure D.12. Similar to the two other classification tasks, a pattern is observed that patterns that show ranges of optimal values also show high importance in obtaining high objective values. This holds, for example, for the parameters learning rate in the BERT implementation and the number of layers in the BiLSTM implementation.

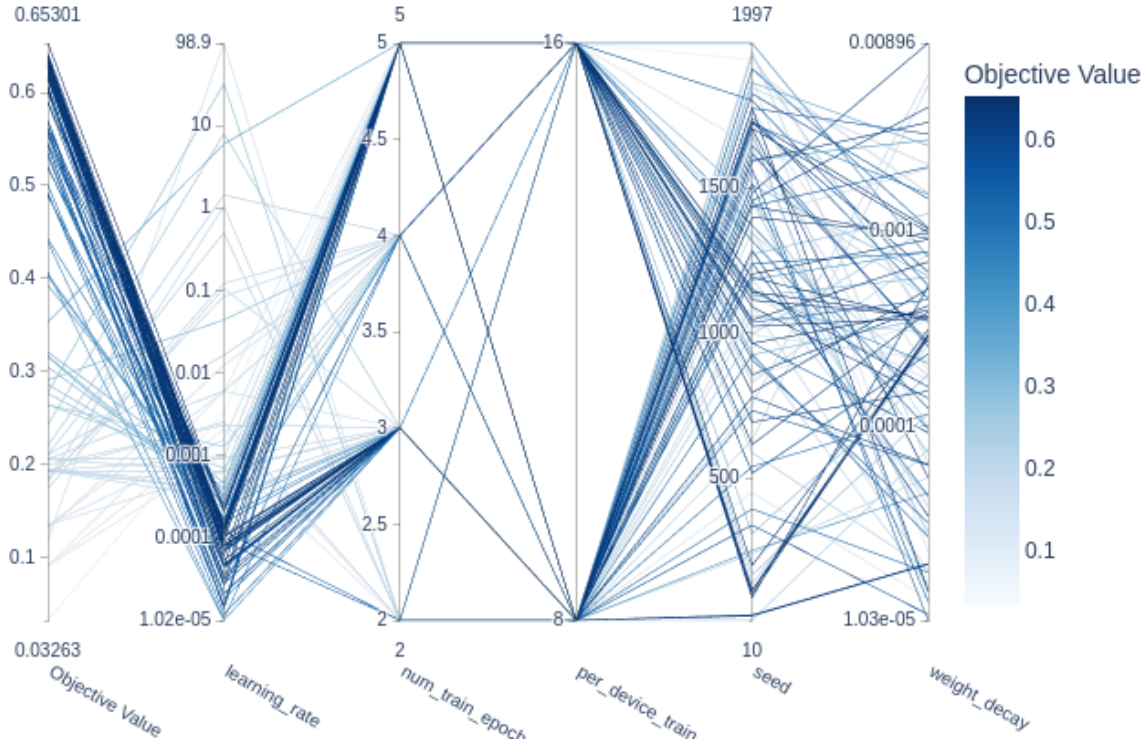

Figure D.9: Hyperparameter results for the BERT implementation of classification task 3: Obtain career stages that may apply

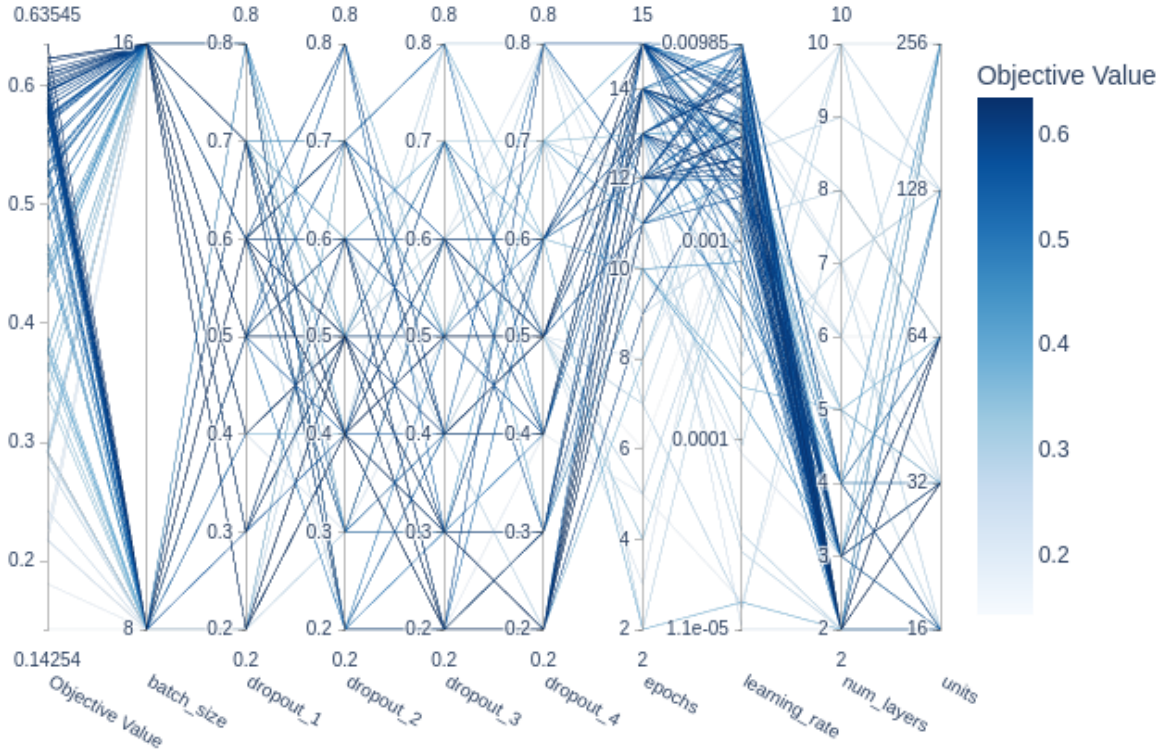

Figure D.10: Hyperparameter results for the BiLSTM implementation of classification task 3: Obtain career stages that may apply

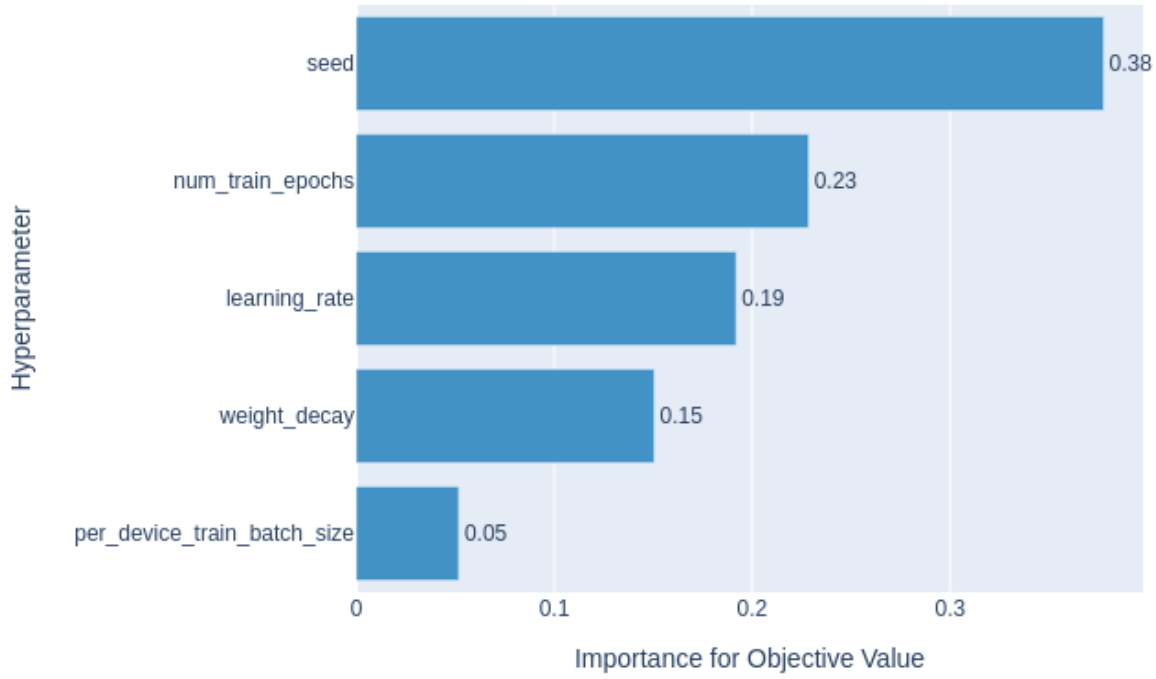

Figure D.11: Hyperparameter importance for the BERT implementation of classification task 3: Obtain career stages that may apply

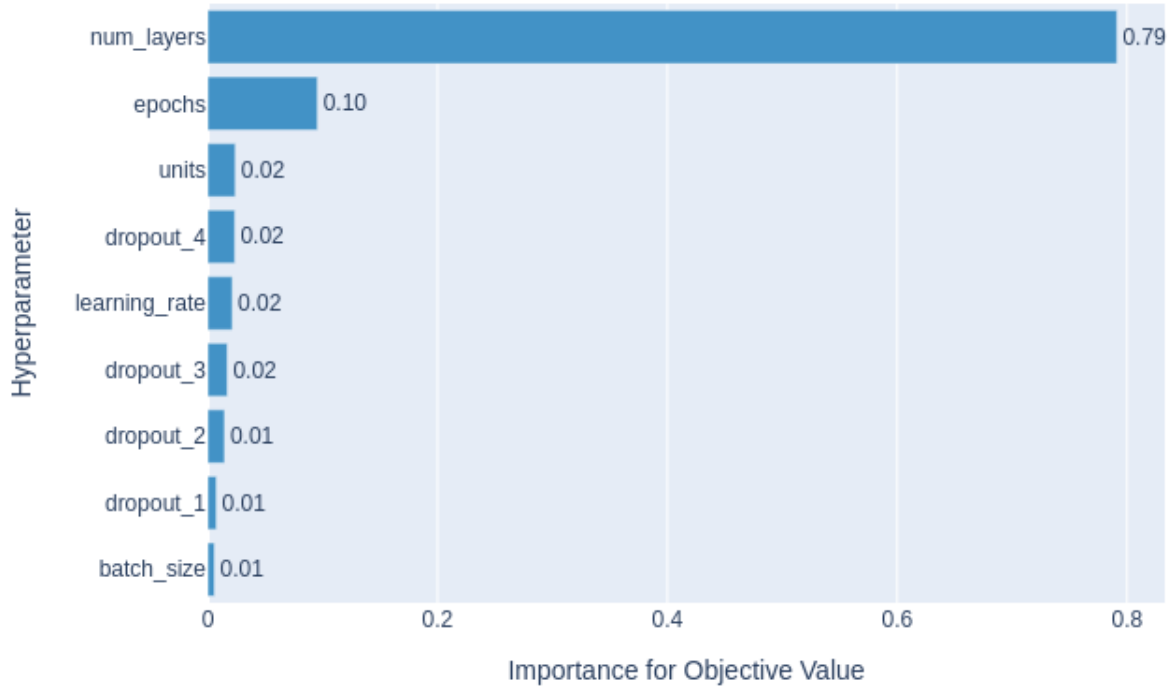

Figure D.12: Hyperparameter importance for the BiLSTM implementation of classification task 3: Obtain career stages that may apply