CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Automated Metadata Extraction for Job Advertisements

Master's thesis in Computer science and engineering

EVELINA STRAUSS & USAMA SAFDAR

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

Automated Metadata Extraction for Job Advertisements

EVELINA STRAUSS & USAMA SAFDAR

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

EVELINA STRAUSS & USAMA SAFDAR

Supervisor: Moa Johansson, Departement of Computer Science and Engineering
Advisor: Jonas Södergren and Maria Dalhage, Arbetsförmedlingen
Examiner: Richard Johansson, Department of Computer Science and Engineering

Master's Thesis 2022
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Gothenburg, Sweden 2022

EVELINA STRAUSS & USAMA SAFDAR
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This thesis is written in collaboration with the Swedish Public Employment Service and aims to investigate methods and techniques to automatically extract metadata from unstructured texts. The Swedish Public Employment Service collect job ads from different private job boards and these ads consist of a title and description and are thus of an unstructured format. Adding metadata to such job advertisements will allow individuals to search and filter ads posted on Platsbanken, the Swedish Public Employment Service's website that advertises jobs.

This is phrased as a classification problem where a job advertisement is classified into one of the following classes capturing different requirements: Education/No education, Experience/No experience, Driving license/No driving license and Full-time/Part-time. Three different classification models are implemented and tested: a baseline dictionary lookup, Support Vector Machine, and BERT. BERT achieves the highest accuracy for sub-problems Education (0.90) and Experience (0.81), while SVM achieves the highest accuracy for Driving license (0.89) and Work type (0.87).

Keywords: Machine learning, NLP, text classification, computer, science, computer science, project, thesis.

Acknowledgements

We would like to start by thanking our advisors from the Swedish Public Employment Service, Jonas Södergren and Maria Dalhage, for their valuable insights into the Swedish labor market, and for providing us with resources to carry out this thesis. We would also like to thank our supervisor, Moa Johansson, for her support through each step of the process. Her valuable feedback has guided us through the work of the thesis, for which we are very grateful.

Evelina Strauss & Usama Safdar, Gothenburg, June 2022

Contents

List of Figures

[1]https://commons.wikimedia.org/wiki/File:Nonlinear_SVM_example_illustration.svg
[2]https://commons.wikimedia.org/wiki/File:Perceptron_moj.png
[3]https://commons.wikimedia.org/wiki/File:Colored_neural_network.svg

List of Tables

1 Introduction

In February 2022, the unemployment rate was 7.2 percent in Sweden, which corresponds to approximately 400,000 inhabitants [1]. Several websites advertise available jobs to bring together those who are searching for a job with those who are searching for employees to reduce the unemployment rate. Platsbanken is one of these sites and is run by the Swedish Public Employment Service, a government agency whose mission is to contribute to increased employment in Sweden [2]. Recently, the Swedish Public Employment Service has identified a reduction in the number of job advertisements in Platsbanken, despite continued high demand from both applicants and the market[1]. This is problematic as the Swedish Public Employment Service works closely with Statistics Sweden (SCB) to create statistics about Sweden's labor market, and base such statistics on data from Platsbanken. This in turn leads to inaccurate statistics about Sweden's labor market since not all advertisements are posted there. The Swedish Public Employment Service, therefore, initiated a project together with Statistics Sweden (SCB) with the aim of improving job search by inviting collaboration with private job boards, such as LinkedIn and Monster, to increase the number of ads at Platsbanken. The Swedish Public Employment Service gathers job advertisements from private job boards by using tools for scraping advertisements from websites. As a result of the ongoing project, the Swedish Public Employment Service has seen an increase of thirty percent more ads in Platsbanken[1].

An essential part of finding a job is to search and filter jobs based on preferences. This could include wanting to filter by required education level, full-time positions or positions that require a driving license. Job advertisements added through Platsbanken have a structured format that contains metadata, such as previous experience, education, and driving license. This format allows individuals to search and filter such ads which facilitates the process of finding and applying for jobs. However, the data collected from private job boards only consist of a title and description, which are unstructured textual data. This unstructured format does not allow either search or filtering, which makes it difficult to retrieve and display wanted job advertisements. The Swedish Public Employment Service, therefore, plan to solve this problem by transforming the unstructured data into a structured format by adding metadata to every job ad. Manually adding metadata is a time-consuming task, and they are therefore looking for a solution that can handle it automatically. Metadata is often described as *"data about data"* and can be used to label and organize data which can be useful in many applications. By transforming unstructured text data

[1]Personal conversation with Jonas Södergren, employee at the Swedish Public Employment Service.

into a structured format, one can easily retrieve and access specific texts from, for example, a database. This can be applied and used in different ways depending on the user and domain, but for this specific use case (for the Swedish Public Employment Service) adding metadata to unstructured job descriptions will bring valuable information that can allow search and filtering. To perform such a task, textual analysis needs to be applied to the text (job descriptions) in order to understand the context and automatically extract the needed metadata. Metadata can be implicitly or explicitly generated from unstructured texts. Implicit generation involves understanding the context of the texts and generating metadata based on such understanding. An explicit solution is more straightforward where, for example, the presence of a specific word is used to generate metadata. Machine learning has previously been used to automatically extract metadata from textual documents, whether implicitly or explicitly [3, 4].

1.1 Purpose

The purpose of the master thesis is to answer the following research question: *"What methods and techniques can be used to automatically extract metadata from unstructured texts"*. The use case in this thesis will be job advertisements and how to automatically extract metadata from their descriptions. The Swedish Public Employment Service has requirements regarding the metadata they wish to extract and add to every job advertisement. Therefore, the following metadata will be extracted from each job advertisement:

- Whether a job requires education or not
- Whether a job requires previous experience or not
- Whether a job requires a driving license or not
- Whether a job is full-time or part-time

1.2 Goal

The goal of this thesis is to investigate what methods and techniques can be used to automatically extract metadata from unstructured texts. The focus will lie within supervised learning, as binary classification will be performed. By investigating useful methods and techniques for this task, metadata can be automatically extracted from unstructured texts, which can be applied in different domains for purposes such as classifying, categorizing and labeling unstructured data. Unstructured texts can then more easily be accessed and retrieved from, for example, a database. For this specific use case, the Swedish Public Employment Service can use such techniques to display and offer more job advertisements in a structured format on their website Platsbanken. It enables filtering, and searching for job ads, and contributes to a larger collection of ads in a shared platform, which facilitates the process of finding relevant jobs.

1.3 Limitations

On average, the Swedish Public Employment Service generates about 70,000 job advertisements every day on Platsbanken. Only a subset of these ads, close to 10,000, have been used in this thesis to train and evaluate models. The reason for this is that the data needs to be manually annotated to perform classification tasks, which is a time-consuming process.

Most job advertisements posted at Platsbanken and on private job boards are written in Swedish but there are ads written in English as well. Approximately 3% of all ads in the dataset are written in English, which is a small number and we will therefore not filter out such ads and handle them all the same. We will focus on handling Swedish text when examining models and tools since that is the written language of almost all ads posted at Platsbanken and private job boards.

The initiated project between the Swedish Public Employment Service and Statistics Sweden, called Job-tech, consists of several modules. Except for automatic metadata extraction, they are, among other things, handling occurrences of duplicates after scraping ads from private job boards. This thesis is only limited to automatic metadata extraction from job advertisements.

1.4 Outline

Chapter 2, *Literature Review*, presents previous work within the field of automatic metadata extraction, and document classification. Previous studies have mainly used machine learning and natural language processing methods to automatically extract metadata. Common models are Support Vector Machines, Naive Bayes, and Neural Networks.

Chapter 3, *Background*, introduces necessary theory and information about methods and techniques used in the thesis. Concepts related to text classification, neural networks, evaluation, and optimization are presented.

Chapter 4, *Method*, describes the data used in the thesis and how the annotation process was carried out. It presents three classifiers: the baseline classifier, Support Vector Machine classifier, and BERT classifier, and how the implementation of the different models was achieved.

Chapter 5, *Results*, presents the result of the three classifiers in regards to Accuracy, Precision, Recall, and F1-score. We compare the results of all four tasks with these three classifiers and find that BERT performs best on Education and Experience, while Support Vector Machine performs better on Driving license and Work type. Support Vector Machine and BERT both perform much better than the baseline.

Chapter 6, *Discussion*, addresses the suggestions, experiences and issues that occurred during the thesis. It discusses the performance of the models and what might be realistically applied. A comparison between the sub-problems is performed, where we discuss why sub-problem Experience have lower accuracy compared to the other sub-problems. This section also includes a discussion regarding the dataset, duplicates and the manual annotation process. Finally, the thesis is summarised with the main findings and concludes whether the used models can be used to extract metadata from unstructured texts.

2
Literature Review

2.1 Automatic metadata extraction

Automatic extraction of metadata from documents has been studied for several years by many researchers. This is because metadata is an effective way to access and retrieve digital records stored in a database. However, many digital records are lacking in metadata, which requires enriching such records with metadata. Metadata in our specific use case is, for example, whether a job requires education or not, while it for an article in a digital library can be who the author is. Manually carrying out such a process is extremely time-consuming and costly, which opens the door to handle it automatically. This section will present previous work within the field to show what methods and techniques have been used to automatically extract metadata from digital records to effectively retrieve them from a database.

Several studies have been conducted on the use of machine learning models when automatically extracting metadata. Kovačević et al. [5] studied how to automatically extract metadata from scientific publications in PDF format to retrieve digital records from CRIS UNS. The authors handled the extraction as a classification problem, where different parts of each publication were classified into eight predefined classes: Title, Authors, Affiliation, Address, Email, Abstract, Keywords, and Publication note. Experiments were performed using four different classification models: Support Vector Machine (SVM), Decision Tree, Naive Bayes, and K-nearest Neighbours (KNN). The performance of the models was measured using precision, recall and F1-score, and the result showed that eight separate Support Vector Machines achieved the best result. The results for all eight Support Vector Machines were between 0.81 and 0.99 for precision, recall and F1-score. The SVM model was then integrated and tested in CRIS UNS, a software system, and the result showed equally good performance as the experiment.

Han et al. [3] applied a similar approach in the same domain, using Support Vector Machine to extract metadata from the header of a research paper. The goal of their study was to improve the quality of metadata in digital libraries, which in turn provides usability in digital libraries. To extract metadata, such as email, telephone and affiliation, the model first classified each line of the header as either single- or multi-class, where the classes represented the metadata elements. To increase the performance of such line-based classification, the model inspected the predicted classes from the previous- and following lines in the header. Lines with multiple classes were divided into chunks of words to extract metadata from such lines. For example, one line can include two authors' names which need to be separated to extract correct metadata from the document. By using this method, the authors achieved an overall accuracy of 92.9% when extracting metadata from the header of a research paper.

It is also common to apply natural language processing methods to extract desired metadata. Rule-based systems are such a method, where human-made rules are used to extract metadata based on the content of the document [6, 7, 8]. Such rules can, for example, be whether the input text is bold or not when extracting the title of a document. Lu et al. [9] used a combination of rules and machine learning to automatically generate metadata from volumes of journals used in digital libraries. For users to access desired journals and articles within the digital library, they need to be enriched with metadata. The authors, therefore, proposed a system that could extract metadata on volume-level, issue-level and article-level for digital journals. This would allow users to access functionalities, such as turning pages and identifying specific articles. Volume-level- and issue-level metadata, such as title and issue number, were generated using a rule-based approach. Article-level metadata, such as article title and authors, was generated using Support Vector Machines. The SVM classified whether a line was the start- or the end of an article, or none of them. This in turn allowed the system to recognize the presence of a new article, and could thereafter generate information about the title and authors by inspecting the first- and following lines. The authors tested the proposed method on three different journals, *Proceedings of the Entomological Society of Washington*, *Journal of Natural Philosophy,Chemistry and the Arts*, and *Magazine of Natural History and Journal of Zoology,Botany,Mineralogy, Geology, and Meteorology.* The first journal received a precision score of 0.98 and recall of 0.94. The second journal received a precision score of 0.67 and recall of 0.66. The third journal received a precision score of 0.91 and recall of 0.82. The system was integrated into a digital library called the Internet Archive to be tested in the real world, where the performance corresponded to the experiment.

Similar to Lu et al. [9], Misra et al. [4] created an Automated metadata Extraction system using a combination of machine learning models and rule-based search models to extract metadata. The system consisted of two parts: layout recognition, and a search model. Compared to Lu et al., the authors used a Support Vector Machine and Hidden Markov Model (HMM) to create their recognition model that first classified each line and identified different segments of the document, such as title and body. Based on the output from the layout recognition model, the authors extracted desired metadata using rule-based search patterns on the different segments. They extract metadata such as Title, Issue Date, and Case Number.

2.2 Document classification

The presented methods of automatically extracting metadata using classification mainly involve classification of segments in a document, such as title, and abstract. This approach differs to some extent from what we plan to use, as we plan to classify each document based on the content. We will therefore also examine some previous research in document classification to see what methods and techniques have been used.

Document classification is part of content analysis since the models must understand the content of a document to be able to classify it correctly. It is a simple task for a human, but more difficult for a computer [10]. To carry out such an analysis and classify documents, researchers have used different machine learning models, where Support Vector Machines (SVM) have been recognized to outperform other classification models in this task [11, 12]. Wang et al. [11] proposed a SVM algorithm for document classification, and compared it with a Decision-tree algorithm, Naive Bayes classifier and K-nearest neighbour (KNN). To classify documents, the authors performed document representation using tf-idf as well as feature selection using entropy weighting scheme. To evaluate their proposed algorithm, the authors used two different datasets: Reuter-21578 and TREC-AP (both datasets consist of news stories). Recall, precision and F1-score was calculated to compare the performance of the proposed method and the Decision-tree, Naive Bayes and KNN classifiers. On Reuter-21578, the proposed SVM algorithm achieved a F1-score of 0.84, while the Decision-tree achieved 0.78, Naive Bayes achieved 0.65 and KNN a score of 0.67. On the second dataset, TREC-AP, did the proposed method achieve a F1-score of 0.73, while the Decision-tree achieved 0.60, Naive Bayes 0.56 and KNN a score of 0.60. The results show that the proposed model (SVM) outperforms other text classification algorithms.

BERT is a neural network architecture used for several tasks within natural language processing, including document classification. It is a popular method due to its impressive performance results, but comes with drawbacks regarding being computationally expensive. Adhikari et al. [13] fine-tuned BERT for document classification. Original BERT consists of two variants: BERT-large that contains millions of parameters computation, and BERT-base with fewer parameters. Using both versions of BERT for document classification where the number of parameters are very large is computationally heavy and time-consuming. To overcome this issue, the authors propose an approach of knowledge distillation, called KD-LSTM. This would transfer knowledge from BERT-large into a smaller knowledge distilled Long short-term memory (LSTM) to reduce the inference time. The authors tested different models, including KD-LSTM, BERT-large, and BERT-base, on four different datasets. The datasets were Reuters-21578, arXiv Academin Paper dataset, IMDB reviews, and Yelp 2014 reviews. The results showed that the proposed method, KD-LSTM, received a similar accuracy score as BERT-large and BERT-base while being almost 40% faster than BERT-base. On the Reuters-21578 dataset (consisting of news articles), KD-LSTM achieved an accuracy of 0.91, where BERT-large achieved 0.92, and BERT-base 0.905. Khadhraoui et al. [14] also fine-tuned BERT for text classification where they created a model called CovBERT to classify online literature to identify relevant sources in the research of COVID-19. The model performed text classification based on the abstract of papers and the authors began by creating a dataset consisting of 4,304 scientific papers collected from PubMed. Each paper was classified into either COVID-19, Virology, Public Health or Mental Health. To train and evaluate the model, the dataset was split into 80% training and 20% test and accuracy, recall, precision and F1-score was calculated for evaluation. CovBERT achieved an accuracy of 0.94, a recall score of 0.88, precision score 0.86 and F1-score of 0.86.

Researchers have investigated the usage of content analysis of job advertisements to identify wanted skills in different job domains [15, 16, 17]. Specific keywords are matched to the job descriptions to identify what skills are usually wanted in different fields. Verma et al. [15] performed content analysis of job advertisements to identify skills wanted in analytical jobs. The authors specifically investigated four job categories: Business Intelligence Analyst, Data Scientist, Business Analyst and Data Analyst. They created a classification framework that classified skills to the different job categories based on occurrence of skills to identify desired skills in each job category. They created a skill list consisting of 17 different skill categories where each contained one or multiple skills. 1,235 job ads from the four different job categories were scraped from private job boards, such as Indeed, where each job ad was checked against each skill in each skill category using keyword matching in the description. Each time it was a match, that specific job category was assumed to be associated with the skill category. The authors noted the number of skills related to each job category. The authors found that for all four job categories, decision making was the most wanted skill, and that Business Analysts usually require less technical skills compared to the other.

3
Background

This chapter introduces key concepts and theories behind used methods and techniques in this thesis. It describes machine learning methods, such as Support Vector Machine, and Neural Network, as well as how to implement and evaluate such models. The chapter also describes text classification and the various aspects to consider when working with textual data.

3.1 Support Vector Machine

A Support Vector Machine (SVM) is a supervised machine learning algorithm for classification and regression. We will only consider classification in this section as that is the focus of this thesis. In SVM, classification is performed by finding a line or hyperplane ($w^T \cdot x + b = 0$) that separates data points in an n-dimensional space, where n is the number of features, into the two classes. A data point is predicted as one class depending on what side of the hyperplane it falls on (see Equation 3.1). w^T refers to the weight vector (T is transpose), x is the feature vector and b is bias, meaning that if the value of $w^T \cdot x + b \geq 0$ it is a positive data point, and if the value of $y = w^T \cdot x + b < 0$ it is a negative data point. There are often many possible lines or hyperplanes that can separate the two classes (see Figure 3.1). The objective is therefore to find the optimal hyperplane that separates the two classes, which is done by maximizing the margin as seen in Figure 3.2 [18]. The margin can be explained as the distance between two data points on each side of the hyperplane. A larger margin is preferable since in general, it will lower the generalization error and perform better on unseen data.

$$y = \begin{cases} +1, & \text{if } w^T \cdot x + b \geq 0 \\ -1, & \text{if } w^T \cdot x + b < 0 \end{cases} \tag{3.1}$$

SVMs can have hard and soft margins, where a hard margin will choose a hyperplane where all data points are perfectly separable. However, even if the data is linearly separable, we might not want to choose a hyperplane that is a perfect fit for the data if that results in a narrow margin [19]. When there is a narrow margin, there is a risk of overfitting since the hyperplane is adjusted to the training data and risk performing worse on unseen data. A soft margin allows the SVM to make some errors when maximizing the margin to keep it as wide as possible [20]. To use a soft margin, we modify the objective function by adding a second term (C) that regulates the trade-off between maximizing the margin and minimizing the errors.

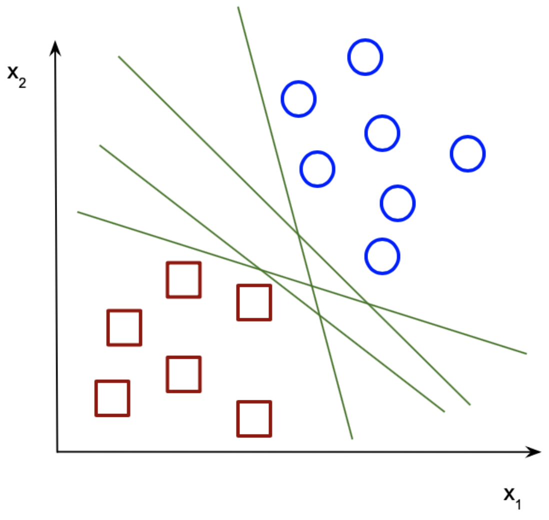


Figure 3.1: Hyperplanes separating two classes in a Support Vector Machine. Several possible hyperplanes to separate the two classes.

Finding an optimal hyperplane can easily be done when data is linearly separable as seen in Figure 3.2. However, most real-world applications use data that is linearly inseparable as seen to the left in Figure 3.3. To handle such cases, the *kernel trick* is used to transform data to being linearly separable by adding more dimensions to the dataset. After the transformation into a higher dimensional space, it is possible to linearly separable the two classes. An example is shown in Figure 3.3, where 2-dimensional data is linearly inseparable but after adding a third dimension the data is now linearly separable.

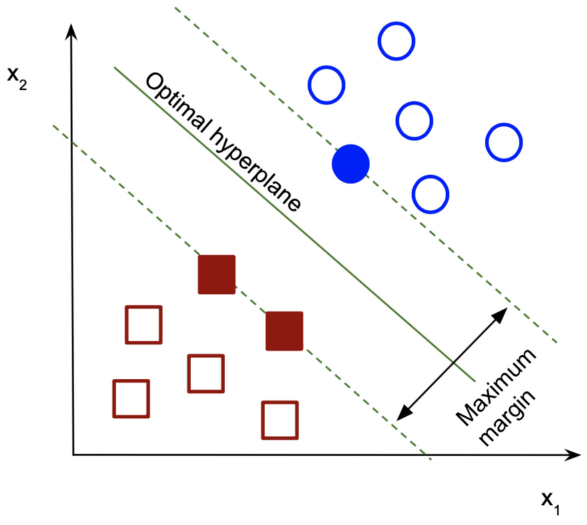


Figure 3.2: An optimal hyperplane that maximizes the margin. The coloured datapoints are called support vectors.

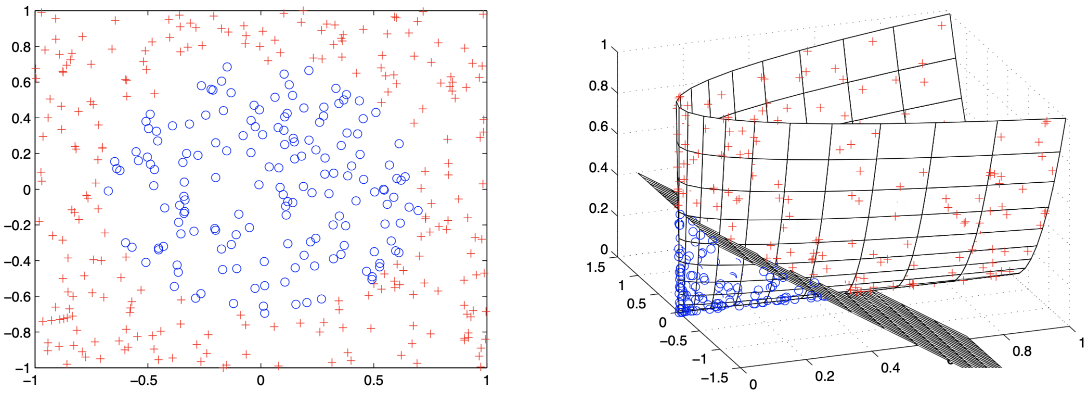


Figure 3.3: Making a dataset linearly separable by adding more dimensions. To the left: 2-dimensional data that is linearly inseparable. To the right: After adding a third dimension the same data is linearly separable [1].

3.1.1 Hyperparameter tuning

Hyperparameters regulate the machine learning model's learning process and thus control the behavior of the model. The values of the hyperparameters need to be initialized before training the model and the choice of such values can directly affect the performance. They, therefore, have an important role in the success of the model [21]. Depending on the machine learning model, there are a different number of hyperparameters available. In SVM, there are mainly two hyperparameters to consider when creating the model: C and the kernel. C, also referred to as regularizer, controls the trade-off between minimizing errors and maximizing the margin [22]. When the value of C is small, the penalty for errors is small which leads to a larger margin which can lead to underfitting. On the other hand, if the value of C is large, there is a larger penalty for errors and the model aims to minimize the number of errors, resulting in a narrower margin that can lead to overfitting.

The kernel function is used to work with the data and form a hyperplane regardless of the dimension; it can thus be helpful when data is linearly inseparable. There are several kernel functions, but the most common ones are: linear, rbf, polynomial, and sigmoid. When choosing either rbf, polynomial or sigmoid, another hyperparameter needs to be considered: Gamma (γ) [21]. γ can simply be described as how far data points from the decision boundary are considered when separating the classes. A low value will consider values further away from the decision boundary. However, there is a risk of not capturing the complexity of the data when γ is too low. A larger value of γ will, on the other hand, only consider values closer to the decision boundary and will therefore create a more complex separation, which can lead to overfitting the data [23].

Tuning hyperparameters is an essential part of the process of creating an effective model, which requires putting time and energy into it to achieve the best possible results. Traditionally, this has been done through manual testing which requires a deep understanding of the hyperparameters and the model structure. However, this can be a time-consuming process and one can therefore instead make use of automatic optimization techniques for hyperparameters. Grid search is a commonly-used optimization technique for hyperparameters that searches through all possible combinations of predefined values of hyperparameters to find the best combination. However, the comparison risk becomes too computationally expensive if too many hyperparameters and unique values are included. For example, if there are k parameters and each has n values, the computational complexity becomes $O(n^k)$. Grid search is therefore preferred if there are not too many hyperparameters or unique values. To deal with this complex problem, one can use random search which is similar but instead testing a subset of the possible combinations [21].

3.2 Neural Network

Neural networks are popular and powerful systems within the field of machine learning that use algorithms to learn and perform tasks by analyzing training data. It can be used, among other things, for clustering and classification [24]. The concept of neural networks is inspired by how a human brain works and processes information, where the system can learn and improve over time. A neural network consists of multiple nodes, which simply is a learning unit that receives information as an input, transforms it by applying a function, and outputs the result. Figure 3.4 shows a single node and how it receives inputs with associated weight w, computes weighted sum of the inputs and applies an activation function [25]. The associated weights of each neuron determine the importance of that input. Larger weights have greater importance to the outcome and vice versa. The purpose of the activation function is to allow the model to learn non-linear functions.

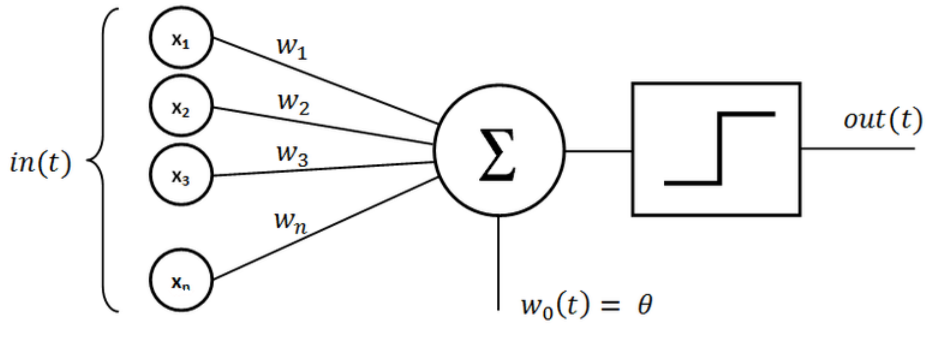


Figure 3.4: Neural network node[2].

In a neural network, nodes are combined into a large network as seen in Figure 3.5. The nodes are divided into different layers: input layer, hidden layer(s), and output layer [26]. The input layer represents the dimension of the input vector and does not perform any calculations. The procedure for each neuron in the layers are as explained previously and shown in Figure 3.4. The input for the first hidden layer comes from the input layer, and the following hidden layers use the outputs from the previous hidden layer as their input. The output from a node determines if it should be passed to the next layer based on a given threshold. If it passes the threshold the data is passed to the next layer. Once all hidden layers have been calculated the output layer uses the output from the last hidden layer as an input to calculate the final output(s) from the neural network. The more hidden layers and neurons the network contains, the deeper and more complex it becomes. This process where nodes are passing data through each layer is called a feedforward neural network [27].

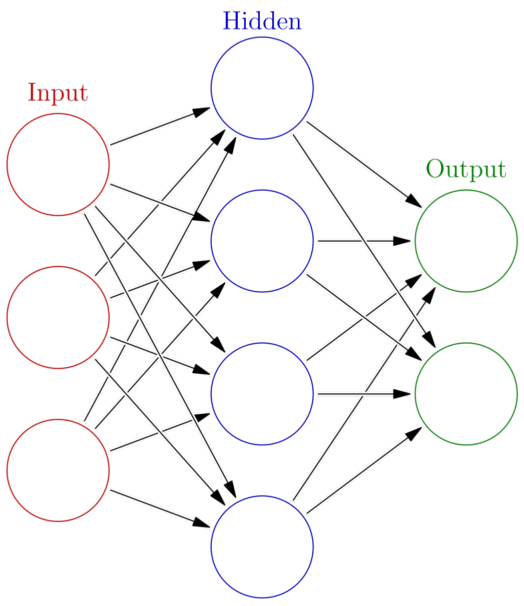


Figure 3.5: Neural network[3]

To make a neural network learn and make use of hidden information it needs to be trained like any other machine learning model. The training process simply consists of adjusting the weights to meet the desired output for the input data. There are two main types of training processes: supervised and unsupervised. The focus of this thesis will be supervised training, which involves the neural network adjusting weights to make the output as close to the desired value by minimizing the loss function. To do so, backpropagation is used to compute gradients with respect to the loss, which are then used by an optimization algorithm to update the model [26]. The learning process begins by randomly assigning weights to the network and compute the output using the random weights. A loss function measures the error rate of the outputs which indicates how close we are to the desired output. Based on this loss function the weights are updated by travelling back from the output layer to the hidden layers where the weights are updated using an optimizer. This process is repeated until the error rate is low and the predicted value is as close to the desired value. However, training too much could overfit the data.

3.2.1 Hyperparameters

Neural networks can have everything from a few hyperparameters to several hundreds that need to be set before the training process begins. This includes hyperparameters of the structure of the network, as well as the training phase. Hyperparameters related to the structure are, among other things, the number of layers and nodes, activation function, and dropout rate. Learning rate, epochs, and batch size are related to the network's training phase.

The number of layers and neurons to choose depends on the task and how complex the data is. If more layers and nodes are added there is a risk of getting a too complex model that takes a long time to train and could lead to overfitting. The activation function allows the model to learn complex patterns in the data by learning neurons non linear representations. The activation function in hidden layer(s) relates to how well the model learns the training data, while the output activation function controls what type of predictions is made by the model. There are several available activation functions, and the choice depends on the task to be accomplished. Common activation functions for the hidden layer(s) are ReLU, Sigmoid and Tanh, and for output layers it is common to use Sigmoid (common for binary classification problems), Softmax (common for multi-class classification problems) and linear (common for regression problems) [28]. Dropout is a technique used to prevent the neural network from overfitting by randomly ignoring neurons when the model is trained [29].

Learning rate determines how much the weights will be adjusted during training. We can thereby control how fast or slow the model finds the optimal weights that minimizes the loss. Lower learning rates equals smaller changes to the weights, meaning that we take smaller steps when moving towards the optimal weights. This will make training take longer and there is a risk of overfitting [30]. If the learning rate is higher it will speed up training with a risk of missing the optimal weights. Number of epochs determines how many times the training data should be run through the network during training. The batch size defines how large samples of training data that should be run before updating the weights. A batch size of 64 means that 64 samples from the training data will be run before updating the weights. A larger batch size requires more memory space.

3.2.2 Optimizer

Optimizers are used to shape our models by modifying the parameters to reduce the overall loss and increase accuracy. They are methods that are used to change the parameters of a neural network, such as weights, to minimize the loss. As discussed in the previous section, neural networks are composed of layers of nodes, and these layers help the model learn patterns. Each node in the network contains weights and biases which are passed onto the next layer to generate an output. The error rate (loss) is calculated based on the output results. The neural network thereafter uses backpropagation to calculate gradients and an optimizer to update the weights in the model to reduce the error rate. This process continues until the loss is minimized and thereby the weights are right. A popular optimization method is Gradient Descent that iteratively updates the models parameters to reach the local minimum of a function. This is an iterative process, where we move from the starting point iteratively by updating the weights to minimize the loss function [31].

In any given model, it is important to carefully choose an optimizer as it can have a great impact on the performance of the model [32]. There are several available optimizers based on gradient descent, and we will briefly explain AdamW that is used in this thesis in BERT. AdamW is a variant of Adam, which is an optimizer that adjusts the size of the steps towards the minimum based on how the gradients change [33]. AdamW makes use of weight decay in an improved version to reduce the chance of overfitting [34]. AdamW gives the ability to optimize the learning rate and weight decay parameters separately which means a change in one parameter would not affect the change in other parameters. As a result, better generalization performance can be achieved. Equation 3.2 shows the update rule for AdamW, where w_t is the new weight and w_{t-1} is the old weight, η_t refers to the step size. $\hat{m}_t$ is the estimate calculated from first moment (averages of the gradients) and $\hat{v}_t$ is the estimate calculated from the second moment (square of gradient). λw_{t-1} refers to weight decay and is performed in the last step when updating weights.

$$w_t = w_{t-1} - \eta_t(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda w_{t-1}) \tag{3.2}$$

3.2.3 BERT

Bidirectional Encoder Representations from Transformers, simply called BERT, is a neural network architecture used for natural language processing. It is an open source framework designed to include a pre-trained deep learning model that can be fine-tuned to solve a wide range of NLP problems. As mentioned before, human language is difficult for computer to understand, and BERT aims to help computers capture context by using the surrounding words [35].

BERT architecture is based on Transformers, which is an architecture that consists of encoders and decoders that are used to read input data (encoder) and predict the output (decoder). BERT only consists of encoders where each encoder consists of two layers: multi-head self-attention, and a feed-forward neural network. Multi-head self-attention learns contextual information about the input sequence by looking at the surrounding words [36]. This allows BERT to capture the contextual meaning of words and sentences in a text by looking at a given word in relation to other words in a sequence.

BERT framework consists of two parts: *pre-training*, and *fine-tuning*. A well-known problem in NLP and when creating language models is the amount of training data required. Incredible amounts of specialized and labelled training data are required for the models to perform well. To address this problem, researchers have developed pre-trained models that have been trained on a large amount of general text data. BERT is a pre-trained model that has been trained using English text from Wikipedia, and can be fine-tuned to a specific domain and problem. In this thesis, BERT will be fine-tuned to fit our problem of understanding the context of job advertisements. The goal of pre-training is to train BERT to understand language and capture the contextual meaning of a text, which is done using two unsupervised tasks: Masked Language Model (MLM), and Next Sentence Prediction (NSP). In the MLM task, a percentage of words in each input sentence are masked, and the model is thereafter asked to predict the masked words based on the context. To predict a masked word, BERT makes use of surrounding words (to the left and right), which makes training bidirectional. The objective of NSP is to predict if sentence B is followed after sentence A, which helps BERT learn the context between sentences [35].

The second part of the framework (fine-tuning) is used to fine-tune BERT for specific tasks, such as text classification, sentiment analysis, and question answering. In this thesis, we will fine-tune BERT for text classification to classify every job advertisement into the predefined classes. Fine-tuning BERT for text classification is simply done by adding a classification layer on top of the model. An example of this is presented in Figure 3.6, where the output from BERT is used as input to a feed-forward neural network that outputs the probabilities of the classes. As seen in Figure 3.6, the first token of an input sequence is a [CLS] token, which is a classification token used to represent the entire sequence of words. The input is passed through layers of encoders inside BERT, and outputs embeddings. The output embedding of [CLS] contains information about the entire input sequence, and is used as input to the classification layer. Another special token is [SEP] which is added in the input sequence to specify the end of a sentence in the input text. This informs the model where the next sentence starts when performing Next Sentence Prediction.

The original BERT model is available in two different sizes: BERT-Base, and BERT-Large. BERT-Base is built on around 800 million trained words, while BERT-Large is made of approximately 2,500 million words. BERT-Base consists of 12 encoders, while BERT-Large has 24 encoders.

3.2.3.1 KB-BERT

To use a language model like BERT in any other language than English, which the original BERT model was aimed for, it needs to be trained on a large amount of data in that specific language. This is the main challenge when creating language models for other languages than English. KB-BERT is a Swedish language model developed by the Royal Library of Sweden, and is built using the same architecture as BERT-Base. They have collected training data from books, digital publications, social media, newspapers, web forums and public inquiries, which corresponds to a corpus of approximately 3,500 million words. KB-BERT outperforms existing Swedish language models, and the reason is mainly due to its large training corpus [37].

3.2.3.2 MBERT

MBERT is a multilingual model that can deal with multilingual datasets. The model has been pre-trained on Wikipedia content from 102 different languages which makes it powerful when dealing with multi-language tasks [38]. Available training data

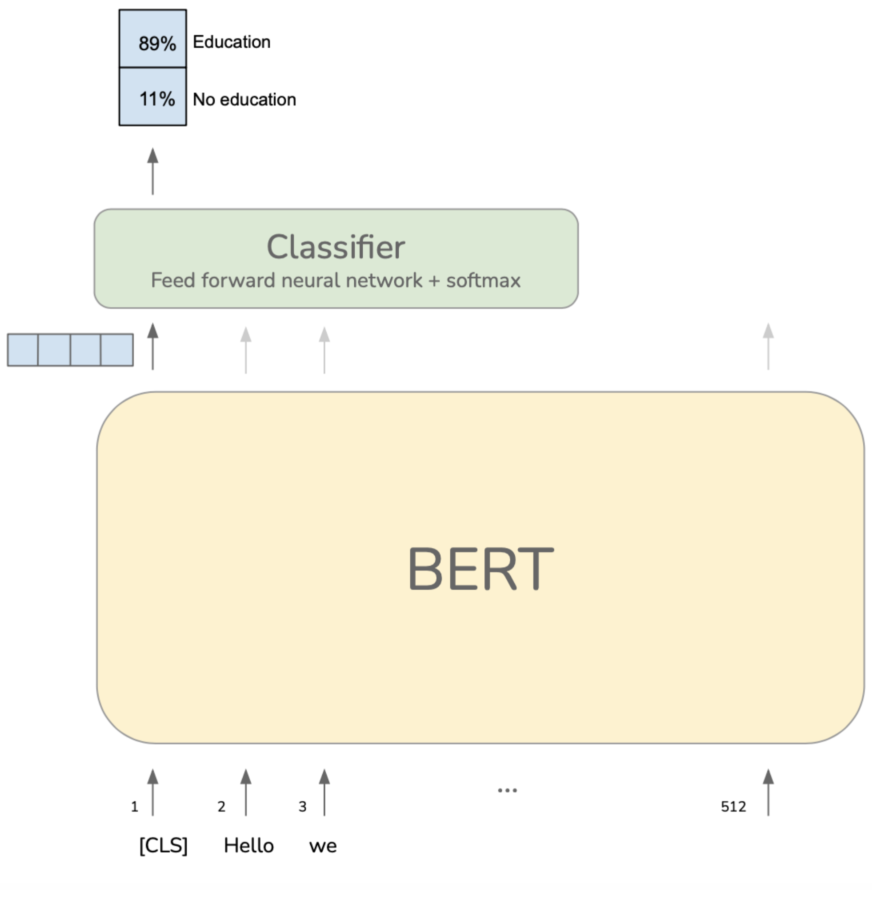


Figure 3.6: Fine-tuned BERT for text classification.

varies between languages, resulting in some languages, such as English and Chinese, having lots of resources, while other languages have less resource content available.

3.3 Text Classification

Text classification lies within the fields of machine learning and natural language processing. Natural language processing (NLP) helps computers manipulate and process speech and text of human language [39]. When used together with statistical models and machine learning, the computer can capture the context and semantic meaning of human language as seen in studies explained in Section 2.1. In this thesis, we will perform text classification using Machine Learning and NLP to classify job advertisements. Most text classification systems follow the same process of starting with text preprocessing and feature extraction, followed by selection of classifier, and finally evaluation of classifier.

3.3.1 Text preprocessing

Preprocessing is an essential component in the text classification pipeline, and involves converting raw text into a form that is analyzable for the classification task

[40]. There are many available preprocessing techniques that can be used, and we will describe some relevant ones for this thesis.

3.3.1.1 Tokenization and lowercase conversion

Tokenization is defined as the process of splitting text into tokens, where tokens can be words, phrases, numbers or other parts [40]. It is common to split text into tokens by identifying white space between two words, for example, the sentence *"I drive a car"* is split into ['I', 'drive', 'a', 'car']. Another preprocessing technique is lowercase conversion which is the process of lowercasing every token in the text [40]. It is a useful technique as it helps to eliminate sparsity issues. For example, the tokens *"Car"*, and *"car"* will both be treated as the same token after lowercase conversion, which helps reduce variation in the text.

3.3.1.2 Stop-words

Stop-words are a collection of common words within a text that are assumed to be irrelevant to the classification task, and are therefore removed [41]. Such words are, for example, *"and"*, *"the"*, and *"it"*. By removing stop-words, the system can focus on more important and valuable words for the classification. Depending on the system and domain, there are different stop-words lists that can be used. For example, we need to use a stop-word list specified for Swedish text since the majority of the job advertisements are written in Swedish. Likewise, the list can be adapted to the domain, for example, a common word is *"good"* which can be considered as a stop-words in some lists. But if the purpose of the system is to conduct a sentiment analysis, it is extremely important that a word like *"good"* is not removed.

3.3.1.3 Normalization

Stemming and lemmatization are techniques used to normalize words in a text. Stemming involves bringing a word into its root form by removing the last part of the word [41]. For example, *"eating"*, and *"eats"* are both converted into *"eat"*. Lemmatization serves the same purpose as stemming, but by using a morphological analysis of the word. Stemming can produce an invalid word, while lemmatization reduces each word to its root that can be found in a dictionary [42]. For example, the lemmatization of *"studies"* would be *"study"*, while the stemming would be *"studi"* which is an invalid word that can not be found in the dictionary. Similarly to lowercase conversion, normalising words reduce the variation in the text since multiple words can have the same root form.

3.3.2 Feature extraction

When working with natural language, raw text cannot be used as an input into the models. This is solved by feature extraction which involves representing the text as a feature vector. A fundamental feature extraction method is bag-of-words, which represents the text as a feature vector containing key-value pairs, where the key is every unique token, and the value is the frequency of that word. For example, the sentences in *"I like dogs"*, and *"I like cats"* are converted into the following list: ['I', 'like', 'dogs', 'cats'] and represented in vector as shown in Table 3.1.

Table 3.1: Vector representation.

Words	Frequency	Vector
I like dogs	I:1, like:1, dogs:1, cats:0	[1, 1, 1, 0]
I like cats	I:1, like:1, dogs:0, cats:1	[1, 1, 0, 1]

The vectors can thereafter be used as an input into the models. However, bag-of-words only captures the occurrence of words and not the context, nor semantics. Meaning that the word *"bank"* have two different meanings: a financial institute, and a part of a river, but are still treated the same. Another limitation relates to the occurrence of common words. Words such as *"the"*, and *"a"* often have a high frequency in documents and will thus be considered as important for the classification even though that might not be the case.

To deal with the latter drawback, one can use Tf-idf which takes into account how relevant a given word is in a document. Tf-idf stands for term frequency-inverse document frequency, and can be broken down in two parts: term frequency (tf), and inverse document frequency (idf). Term frequency refers to the number of times a word occurs in a document, and is thereby similar to bag-of-words. Inverse document frequency considers the frequency of a word across a collection of documents, which thereby identifies words that are common across the collection of documents. Common words across a collection of documents, such as *"the"*, will be assigned a lower weight compared to words that are common in a single document. If we, for example, compare the words *"and"* and *"cow"* we will see different weights assigned to the words. *"And"* is very common across a collection of documents, compared to *"cow"* that might only be present in a single document. *"Cow"* will thereby be assigned a higher weight and thus be more important to the document compared to *"and"* [43].

3.4 Evaluation of classification models

When building and developing a model, an essential part of the process is to get feedback and improve until the model can carry out its desired task well enough. This is done by using evaluation metrics, which measures the performance of a model, and thus determines how effective it is. There are several available metrics, and which one to choose depends, among other things, on the type of model [44]. This thesis concerns binary classification, and we will therefore describe some relevant evaluation metrics for binary classification.

When calculating evaluation metrics, it is common to create a confusion matrix which summarizes the predictions of a classification problem. Figure 3.7 shows how a confusion matrix is constructed for a binary classification problem. TP refers to True Positives and includes all instances where the classifier correctly predicts the positive class. TN refers to True Negatives and includes all instances where the classifier correctly predicts the negative class. FN refers to False Negatives and includes all instances where the classifier incorrectly predicts the negative class. FP refers to False Positives and includes all instances where the classifier incorrectly predicts the positive class.

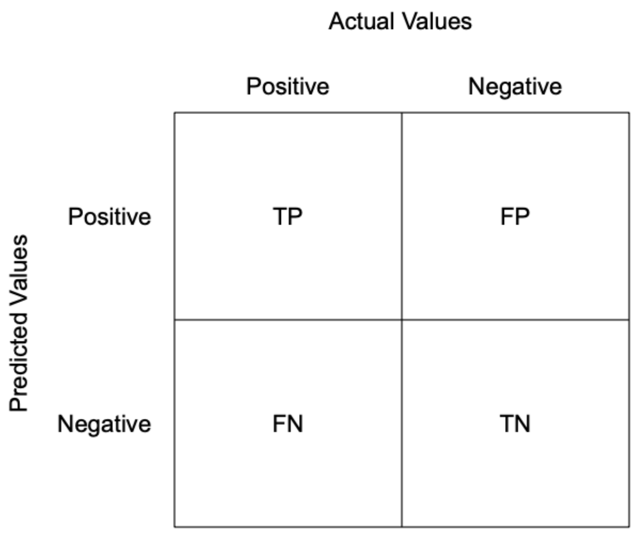


Figure 3.7: Confusion matrix.

Accuracy is the most straightforward metric and measures the correctness of the model (see equation 3.3). However, the simplicity of the metric can be misleading and less suitable for imbalanced datasets [44]. For example, a dataset of 100 instances, where class A is dominant and constitutes 90% of the dataset and class B is the remaining 10%. If our model classifies all instances as class A, our accuracy will be 90% which is considered as very good, but the model has not succeeded in correctly predicting any instances of class B. Therefore, accuracy is recommended to be used together with other metrics, such as precision and recall [45]. Precision attempts to answer the question *"How often is the classifier correct when predicting a positive example?"* (3.4). In our thesis, an example would be the measure of ads that we correctly predicted requiring education, out of all ads requiring education. Recall on the other hand answers the question *"How many actual positives did the classifier correctly identify?"* (3.5). An example would be for all ads requiring education, how many did we correctly identify as requiring education. In a perfect setting, both precision and recall would be 100%. However, that is often not the case as if we receive a high precision, often recall is lower and vice versa. Depending on the use case of the model, some problems might find it more important to achieve a high recall than high precision, for example, if we want to detect as many cancer patients as possible. Although, precision and recall could be equally important for some problems and in such cases, another metric called F1-score can be used. F1-score aims to create a harmonic mean between precision and recall (3.6) [45].

$$Accuracy = \frac{(TP + TN)}{(TN + TP + FN + FP)} \tag{3.3}$$

$$Precision = \frac{(TP)}{(TP + FP)} \tag{3.4}$$

$$Recall = \frac{(TP)}{(TP + FN)} \tag{3.5}$$

$$F1\text{-}score = 2 * \frac{Precision * Recall}{(Precision + Recall)} \tag{3.6}$$

4 Method

This chapter describes the data and methods used in this thesis. Section 4.1 describes the data collection followed by the annotation process in Section 4.2 which elaborates the process and tool to manually carry out annotation. Section 4.3 states libraries and packages that have been used to create and evaluate the models. Section 4.4 describes the baseline classifier, a dictionary lookup used as a baseline to evaluate the other two methods. Section 4.5 presents the Support Vector Machine classifier, which consists of four different classifiers to handle each sub-problem as a binary classification task. The section explains the essential parts of training the model, including preprocessing, feature extraction, and hyperparameter tuning. BERT classifier is described in Section 4.6 and consists of four different classifiers as the SVM to handle the sub-problems. The two phases: Tokenization, and Training are explained in further detail.

4.1 Data Collection

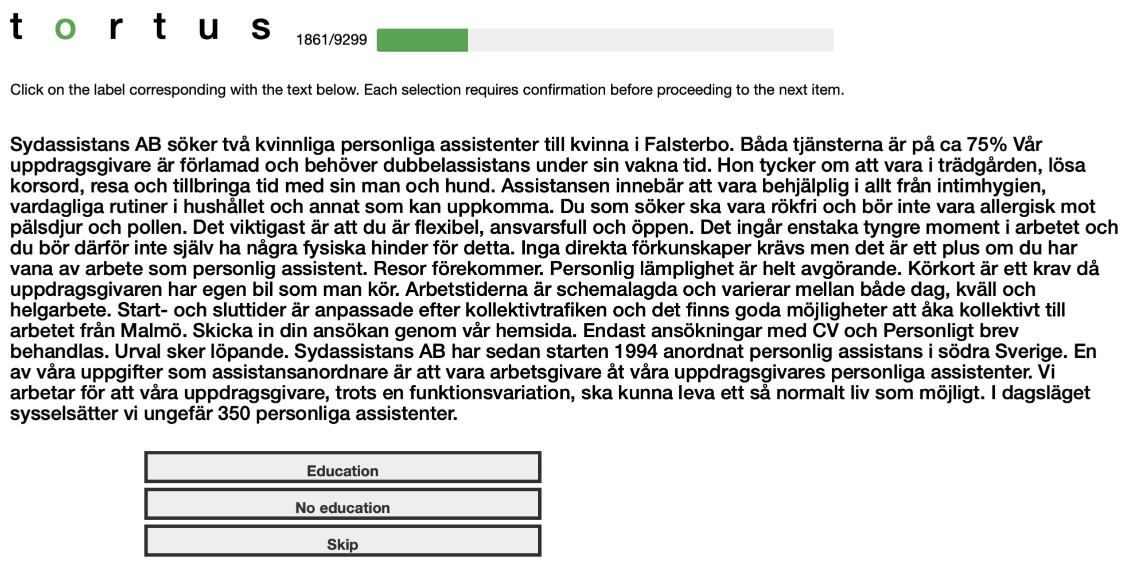
The Swedish Public Employment Service generates a dataset each night, with approximately 70,000 unlabeled job advertisements. A subset of these 70,000 ads, consisting of approximately 10,000 ads, was extracted based on a number of criteria to create a balanced dataset where all classes (Education/No education, Experience/No experience, Driving license/No driving license, and Full-time/Part-time) were equally represented. Ads that are published through Platsbanken[1] have, as previously mentioned, a structured format and are enriched with metadata such as experience, driving license, and full-time/part-time. Such information is given by the employer that publishes an ad at Platsbanken. We have therefore used such previous information to create a balanced dataset where all classes are approximately equally represented. However, *Education* is not added by the employer, but generated automatically by the Swedish Public Employment Service based on type of occupation. The Swedish Public Employment Service has previously annotated occupations (close to 3,000 different occupations) as requiring and not requiring education, where the annotation has been chosen based on what usually requires and does not require education. For example, the occupation "Waiter/waitress" is classified as not requiring education since it usually does not require it. Once an add is added to Platsbanken, the employer chooses an occupation that fits the job and the education classification is therefore added automatically based on the chosen occupation. The employer does not manually choose whether the job requires education

[1]https://arbetsformedlingen.se/platsbanken/

or not and therefore, it is possible that jobs are wrongly classified as requiring and not requiring education. Therefore, manual annotation is done for education and will be explained further in the following section.

4.2 Annotation

The annotation process consisted of manually annotating 10,000 job advertisements as requiring or not requiring education based on the description. There could, as previously mentioned, occur false negatives, and false positives for Education which could negatively affect the quality of the data. To mitigate this risk, the data was annotated manually by the authors regarding whether an ad requires education or not. Annotation was done using an annotation tool called Tortus, which is a Python library built by Langeni [46] with the aim of helping annotators labelling textual data in Jupyter Notebook. Figure 4.1 shows the interface of Tortus in Jupyter Notebook. The user can see how many ads have been annotated and how many are left, as well as reading the ad and choosing a label.



Click on the label corresponding with the text below. Each selection requires confirmation before proceeding to the next item.

Sydassistans AB söker två kvinnliga personliga assistenter till kvinna i Falsterbo. Båda tjänsterna är på ca 75% Vår uppdragsgivare är förlamad och behöver dubbelassistans under sin vakna tid. Hon tycker om att vara i trädgården, lösa korsord, resa och tillbringa tid med sin man och hund. Assistansen innebär att vara behjälplig i allt från intimhygien, vardagliga rutiner i hushållet och annat som kan uppkomma. Du som söker ska vara rökfri och bör inte vara allergisk mot pälsdjur och pollen. Det viktigast är att du är flexibel, ansvarsfull och öppen. Det ingår enstaka tyngre moment i arbetet och du bör därför inte själv ha några fysiska hinder för detta. Inga direkta förkunskaper krävs men det är ett plus om du har vana av arbete som personlig assistent. Resor förekommer. Personlig lämplighet är helt avgörande. Körkort är ett krav då uppdragsgivaren har egen bil som man kör. Arbetstiderna är schemalagda och varierar mellan både dag, kväll och helgarbete. Start- och sluttider är anpassade efter kollektivtrafiken och det finns goda möjligheter att åka kollektivt till arbetet från Malmö. Skicka in din ansökan genom vår hemsida. Endast ansökningar med CV och Personligt brev behandlas. Urval sker löpande. Sydassistans AB har sedan starten 1994 anordnat personlig assistans i södra Sverige. En av våra uppgifter som assistansanordnare är att vara arbetsgivare åt våra uppdragsgivares personliga assistenter. Vi arbetar för att våra uppdragsgivare, trots en funktionsvariation, ska kunna leva ett så normalt liv som möjligt. I dagsläget sysselsätter vi ungefär 350 personliga assistenter.

Figure 4.1: Tortus interface when manually annotating data.

4.2.1 Annotated dataset

The annotated dataset consists of 10,000 ads, and during annotation we discovered that it contained several ads with the exact same description. Technically, these ads are not considered as duplicates since they have different ID:s with the same description. The reason for this could be due to the fact that the same description is used in several ads published in different cities in Sweden. Another reason could be that the same job has been published on several occasions, which results in the same description being present in several different ads. We will therefore treat them as duplicates in this thesis. There were approximately 4,000 duplicated ads in the annotated dataset. By keeping such duplicates, the performance could be negatively impacted since the distribution of ads would be imbalanced when training and testing the models. If, for example, the models were trained on most of the duplicates it will not learn from as many unique descriptions and thereby risk performing worse on unseen data. We therefore chose to remove the duplicates in the annotated dataset, which resulted in approximately 6,000 ads.

To handle this reduction in the number of ads, we performed a second round of manual annotation where we received 4,000 new ads from the Swedish Public Employment Service to annotate. The annotations of the second round were performed the same way as the first round, and possible duplicates were removed before merging into the first annotated dataset. The final dataset used for experiments is presented in Figure 4.2, and consists of 9,131 ads. As seen, all classes except *Education* are approximately balanced, and the final dataset thereby consists of more ads not requiring education than requiring education.

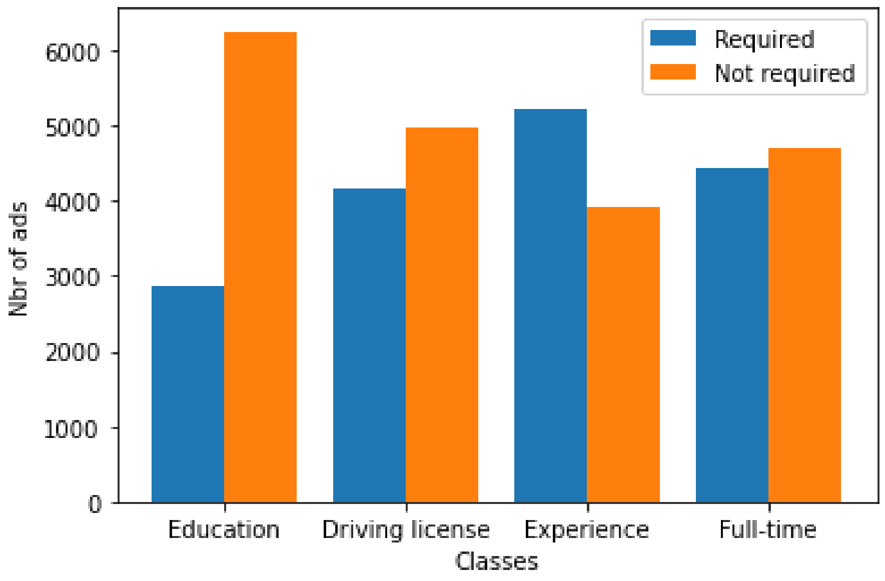


Figure 4.2: The distribution of classes in the final dataset.

4.2.2 Training and test split

The dataset was split into 80% training and 20% test data for each sub-problem with stratification to ensure the same proportion of the two classes in the training and test set as in the full dataset. The same split have been used in all models.

4.3 Libraries

All models and systems that have been implemented in this thesis have been written in Python.

4.3.1 Scikit-learn

Scikit-learn is a machine learning library for Python that can be used to perform predictive data analysis[2]. Scikit-learn offers various algorithms for classification, clustering, regression, and dimensionality reduction. In this thesis, we have only used Scikit-learn classification algorithms, for example, our SVM model was designed through Scikit-learn. The library has also been used to randomly divide the dataset into training and testing for all models. It also provides features for evaluating the models' performance, and we have used it for calculating accuracy and generating a classification report that contains precision, recall and f1-score. For feature extraction, we have used Tf-idf through Scikit-learn, as well as GridSearchCV when tuning hyperparameters.

4.3.2 NLTK

NLTK is a library and toolkit for natural language processing in Python, and is suitable to use when working with human language[3]. NLTK has been used in our preprocessing phase where we have, among other things, tokenized the job descriptions, removed stop-words, and performed stemming. NLTK has built-in word-lists for stop-words, including Swedish stop-words which have been used in this thesis since the job advertisements are mainly in Swedish.

4.3.3 Pytorch and HuggingFace

Pytorch is a widely used deep learning framework, especially when working with neural networks. Pytorch primarily includes tensors which are the data structure that is used to train and build neural networks[4]. In this thesis, we have built our BERT model by using the Pytorch-Transformers library from HuggingFace that contains implementation of the BERT model. To make use of the pre-trained Swedish BERT model from the Royal Library of Sweden, HuggingFace and Pytorch had to be used[5].

4.4 Baseline classifier

A baseline classifier was created to evaluate the other classification systems. The pseudocode for the baseline can be found in Algorithm 1 where the classifier consists of two functions: creating dictionaries (training) and prediction. The pseudocode exemplifies the sub-problem Work type, and the other sub-problems are handled in the same way. The baseline uses a simple approach of dictionary lookup where a dictionary consists of key-value pairs, where the key is a word and the value is the frequency of such words. The baseline handles each sub-problem separately and creates two dictionaries per sub-problem during training. The training phase is where

[2]https://scikit-learn.org/
[3]https://www.nltk.org
[4]https://pytorch.org/
[5]https://huggingface.co/KB/bert-base-swedish-cased

the dictionaries are created, which requires splitting the dataset into training and testing.

Each description was separated into tokens, and each token was thereafter added to a dictionary depending on the label. The value field was updated each time a word appeared. At the end of the training phase, the top 80 pairs were removed from the dictionary as a technique to remove stop-words. The dictionaries for each sub-problem can be found in Appendix A.2. However, only a subset, first 300 key-value pairs, is presented as it would be too much to include full dictionaries. The full dictionaries can be accessed through the following link: Baseline dictionaries. The length of the dictionaries are:

- Education
  - Education: 46,499
  - No education: 53,756
- Experience
  - Experience: 63,301
  - No experience: 37,424
- Driving license
  - Driving license: 40,191
  - No driving license: 58,338
- Work type
  - Full-time: 51,973
  - Part-time: 48,349

To classify a job advertisement, four different prediction functions were created, one for each sub-problem. The function predicts one of the two classes based on the text from the job description as seen in the second function in Algorithm 1. The procedure is similar to creating a dictionary, where each job description is split into tokens, and the function checks if every word occurs in each dictionary. If the word occurs in a dictionary, the score for that given class will increase by one. When the function has iterated over all words it checks which score is the highest and predicts the job advertisement with the class of the highest score. When each word has been checked, the function classifies the job advertisement with the class of the highest score. If the scores from both classes are equal, it will be classified as *"confused"*.

4.5 Support Vector Machine classifier

The Support Vector Machine classifiers were built using Scikit-learn SVM, along with pre-processing tools from NLTK. Four different classifiers were implemented to handle the four different sub-problems. The architecture for the final classification model is the same for each classifier and is presented in Figure 4.3. The input to each classifier is a single job ad that goes through pre-processing, feature extraction, and finally is run through the trained SVM-model. The output is a labelled job ad, which is used when adding metadata to the job ad. We will go into further detail and explain the training phase of the SVM-models in the following section.

Algorithm 1 Baseline classifier

```
FUNCTION Dictionary Creator(text):
Full-time ←[ ]
Part-time ← [ ]
for word in text-split do
    if full-time = $TRUE$ then
        if word not in Full-time then
            Full-time[word]=1
        else
            Full-time[word]+=1
        end if
    end if

    if part-time = $TRUE$ then
        if word not in Part-time then
            Part-time[word]=1
        else
            Part-time[word]+=1
        end if
    end if
end for

FUNCTION Prediction(text):
fulltime-score = 0
parttime-score = 0
for word in text-split do
    if word in Full-time then
        fulltime-score+=1
    end if
    if word in Part-time then
        parttime-score+=1
    end if
end for
if fulltime-score > parttime-score then
    prediction = True
else if fulltime-score < parttime-score then
    prediction = False
else
    prediction = "confused"
end if
```

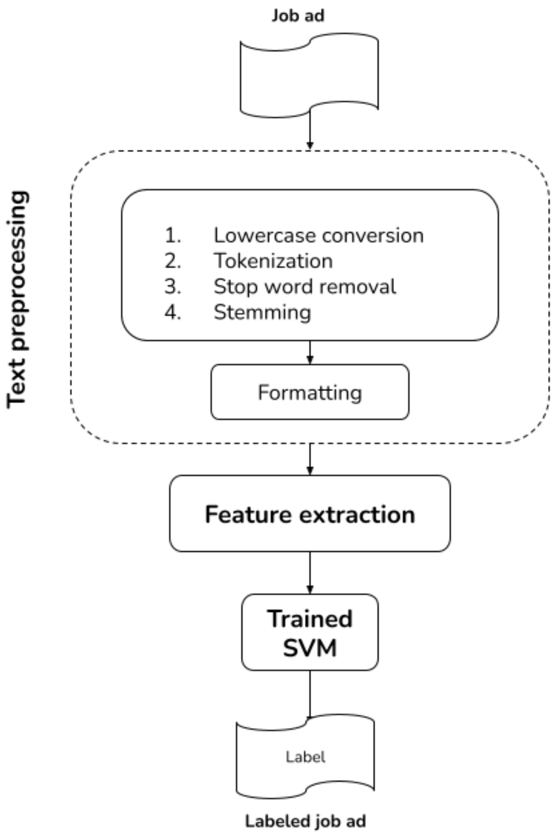


Figure 4.3: Architecture of the final SVM-model where a job ad is classified according to predefined classes.

4.5.1 Training

Training machine learning models is a must before the model can be applied to its domain, and in this case classify job advertisements. Since there are four SVM classifiers, there are also four different training phases, one for each classifier. The architecture for each training phase is the same for each classifier and is presented in Figure 4.4. Training consists of preprocessing the training documents, followed by feature extraction, hyperparameter tuning and finally training the model. We will go further into detail and explain preprocessing, feature extraction, and hyperparameter tuning.

4.5.1.1 Text preprocessing

Our choice of preprocessing tools are lowercase conversion, tokenization, stop-word removal, and stemming. The first step of the module is to convert all words in a document to lowercase, followed by tokenization. By performing lowercase conversion and tokenization on each document, we will have the same format for every document before processing further. The next step is to remove stop-words, which

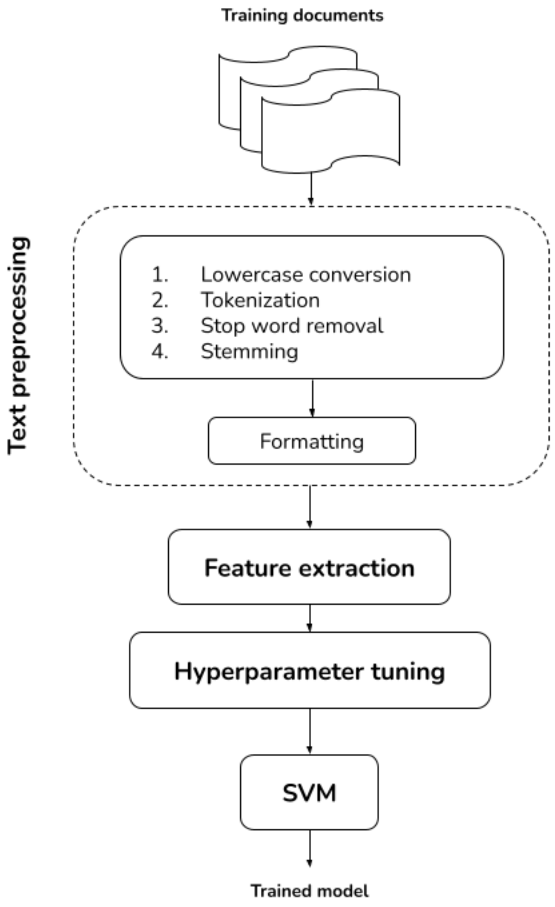


Figure 4.4: Training a SVM classifier.

is done using a list with predefined Swedish stop-words from NLTK. Some Swedish stop-words in the list are, for example, *och*, *att*, *det*, and *i*. The full list of Swedish stop-words can be found in Appendix A.3. The given list is used and checked against every word in each document, and if there is a match such word will be removed from the corpus of the document. Finally, each token is normalized through stemming. NLTK SnowballStemmer can be used for Swedish text and have been used to normalize each word in the document to its root form.

Another component of the preprocessing module is the transformation of non-numerical labels to numerical labels. The labels were converted from *True* or *False*, to numerical labels using Scikit-learn LabelEncoder. This pre-processing module has the same components for both training the SVM-model, and the final SVM classifier.

4.5.1.2 Feature extraction

When working with textual data, we need to perform some feature extraction to transform the raw text to a format that can be used as an input to machine learning models. When creating our classification models, Tf-idf has been used to extract features from the text. Tf-idf takes the text of a document as its input and transforms it to a matrix of tf-idf features that can be used as an input to a model. For the parameter *analyzer*, which controls whether single words or character n-grams should be selected for a feature, the default value "word" was selected which creates unigrams in the representation. What constitutes a token when creating the representation is two or more alphanumeric characters. This module has the same components for both training the SVM classifier, and the final SVM classifier.

4.5.1.3 Hyperparameter tuning

As described in Section 3.1.1, it can be necessary to tune hyperparameters to create an effective model and increase accuracy. There are several available techniques to perform hyperparameter tuning, and in this thesis we have used grid search. When using grid search, we need to decide what hyperparameters to tune, and their range. Table 4.1 presents the chosen hyperparameters and their range for all SVM-models. Once the parameters have been chosen, 5-fold cross-validation is used on the training data to find the best combination of these values. To decide which combination of hyperparameters were the best, accuracy is calculated and the combination with the highest accuracy will be chosen. Once that combination has been found it will run once again to build a new SVM classifier using those hyperparameters.

Table 4.1: SVM hyperparameters and their range for tuning process.

C	Gamma	Kernel
0.1	0.1	rbf
1	0.01	poly
10	0.001	sigmoid

Grid search was performed on each SVM-model and the chosen parameters for the sub-problems are presented in Table 4.2. As we can see, all four SVM classifiers use *C = 10*, and *gamma = 0.1*, where the only difference is choice of kernel. Education and driving license use *kernel = sigmoid*, where experience and work type use *kernel = rbf*.

Table 4.2: Selected hyperparameters of SVM-models.

	Education	Experience	Driving license	Work type
C	10	10	10	10
Gamma	0.1	0.1	0.1	0.1
Kernel	sigmoid	rbf	sigmoid	rbf

4.6 BERT classifier

Two different types of BERT classifiers were created, one monolingual model (Swedish) and one multilingual model. For both types, four different BERT classifiers were implemented to handle the four different sub-problems. The architecture for the monolingual- and multilingual models is the same and the difference is only the data they have been pre-trained on. In this thesis, we will refer to the monolingual model as KB-BERT and the multilingual model as MBERT. BERT classifiers were built using the HuggingFace Library, a Transformer library that contains PyTorch implementation of BERT. To perform classification, a classification layer has to be added on top of the BERT model and in this thesis we have used a single linear layer.

The architecture of the classification system is divided into two parts: Tokenization, and Training, which will be explained further.

4.6.1 Tokenization

The process of tokenizing and transforming text has to be performed by a special tokenizer provided by the HuggingFace library. In this thesis, we use two different tokenizers: one for KB-BERT (provided by the Royal Library of Sweden) and one for MBERT. The tokenizer provides us with a method called *encode_plus* that helps us format the text in a required format for BERT. It will perform the following functions:

- Split text into tokens
- Add [CLS] and [SEP] tokens
- Pad or truncate text into maximum length
- Create attention mask

BERT has two constraints related to length of input text, where each input must have the same length, and the maximum length is 512 tokens. To deal with this constraint, each input text has to be either padded or truncated to match the set maximum length. Since our data consists of descriptions from job advertisements, it is possible that the description is longer than 512 tokens, leading to us having to truncate the description. Sun et al. [47] discussed this issue, and tried different truncation methods. They compared three approaches: keeping first 510 tokens, keeping last 510 tokens, and selecting first 128 and last 382 tokens. The last method was most successful in their study. We therefore performed similar experiments in this thesis where we investigated which of the three approaches had the highest accuracy in each sub-problem. The experiments were conducted using a subset of the dataset, consisting of 1,000 ads. The results are presented in Table 4.3. As we can see, the first approach including first 510 tokens had the highest accuracy for all sub-problems, and will therefore be used in this thesis.

Table 4.3: Result from maximum length experiment.

	Education	Experience	Driving license	Work type
Head	**0.90**	**0.76**	**0.92**	**0.90**
Head+Tail	0.89	0.74	0.88	0.88
Tail	0.82	0.75	0.83	0.87

4.6.2 Training

Once we have formatted our input data, we can train and fine-tune our model. The procedure for training is the same for KB-BERT and MBERT with the difference of the pre-trained model that is being loaded. For KB-BERT, we load with a pre-trained Swedish model provided from the Royal Library of Sweden, and for MBERT we load it with the multilingual model. Before beginning training the model, we will set the parameters and create an optimizer. In this thesis, we have used AdamW as the optimizer, and the following parameters for fine-tuning:

- Batch size: 8
- Learning rate: 2e-5
- Epochs: 4

Devlin et al. [35] recommend using a batch size of 16 or 32, a learning rate of 5e-5, 3e-5 or 2e-5, and epochs of 2, 3 or 4. To be able to run the model without computationally failing, we had to choose a batch size of 8. Training is done for 4 epochs, where the model is trained and validated in each epoch. In each training loop the model will calculate the loss by feeding the network with our input data, compute gradients through a backward pass, update parameters using the optimizer, and update the learning rate. To validate, the model just needs to feed the network with input data and compute the loss.

5 Results

This chapter presents quality results from the annotation process, as well as results of the baseline classifier, Support Vector Machine classifier, and BERT classifier. The four sub-problems are compared against each other, as well as the different classifiers. Finally, learning curves for Support Vector Machine classifier and BERT classifier are presented.

5.1 Annotation

The dataset used in this thesis was, as previously explained, manually annotated by the authors. To ensure high quality annotations, we created an annotation specification, as well as double annotating a subset of the dataset followed by an inter-annotator agreement. A specification is used to state the purpose of the annotation as well as explaining the process and how the annotation is carried out. The purpose is to create a mutual understanding for the annotators, such as what defines the different classes. The annotation specification can be found in Appendix A.1. An inter-annotator agreement measures reliability and to what extent annotators agree. Several metrics can be used to measure this, and in this thesis, Cohen's Kappa (see equation 5.1) has been used. p_o refers to the observed relative agreement among annotators, and p_e refers to the hypothetical probability of annotators agreeing by chance. p_o is calculated by counting the number of occurrences where annotator 1 and 2 agree, divided by the total number of annotated ads. To calculate p_e, we first have to calculate the probability that both annotators randomly annotate education, and the probability that both annotators randomly annotate no education. p_e is then calculated by adding both results [48].

$$K = \frac{p_o - p_e}{1 - p_e} \tag{5.1}$$

Cohen's Kappa result is based on a scale from 0 to 1, where 0 represents random chance of agreement, and 1 is perfect agreement. Our result came back at 0.905, which means that our annotation was consistent.

5.2 Baseline classifier

The results from the baseline classifier is presented in Table 5.1, which includes accuracy, recall, precision and F1-score for all four baseline classifiers. We will examine each sub-problem separately in greater detail.

Table 5.1: Results from baseline classifier.

	Education	Experience	Driving license	Work type
Accuracy	0.65	0.74	0.74	0.75
Recall	0.78	0.81	0.49	0.65
Precision	0.48	0.75	0.87	0.80
F1-score	0.59	0.78	0.62	0.72

5.2.1 Education

As seen in Table 5.1, the baseline achieves an accuracy of 0.65, indicating that it correctly predicts a class 65 times out of 100. Recall answers the question *"How many actual positives did the classifier correctly identify?"*, and the baseline achieved a score of 0.78. This means that the baseline correctly identifies 78% of all job advertisements requiring education. Precision answers the question *How often is the classifier correct when predicting a positive example?"*, and the baseline had a score of 0.48. A precision score of 0.48 indicates that when the baseline predicts a job advertisement as requiring education, it is correct 48% of the time. F1-score is the harmonic mean between precision and recall, and the score for the baseline is 0.59.

5.2.2 Experience

The baseline achieves an accuracy of 0.74, meaning that it correctly classifies 74% of all ads. This is a higher score compared to education which only had 0.65, and thereby tells us that our baseline is performing better when classifying ads as requiring previous experience or not. The baseline got a recall score of 0.81, indicating that the baseline correctly identifies 81% of all job ads requiring previous experience. Similarly to education, the baseline received a lower score for precision compared to recall, 0.75. This means that when the baseline predicts a job advertisement as requiring previous experience, it is correct 75% of the time. F1-score for the baseline is 0.78. Compared to education, there are more ads requiring previous experience than not requiring previous experience in the final dataset in Figure 4.2. We would thereby expect the model to perform equally well on both classes, or even better at predicting the positive class (requiring previous experience).

5.2.3 Driving license

The accuracy for driving license is 0.74 and the baseline thereby correctly classifies 74% of all ads. The score for recall is 0.49, which tells us that the baseline correctly identifies 49% of all job ads requiring a driving license. This is a lower score compared to education and experience, and the baseline thereby has more difficulties in correctly identifying ads requiring a driving license compared to requiring education or previous experience. The precision score is 0.87, meaning that when the baseline predicts a job advertisement as requiring a driving license, it is correct 87% of the time. This score is higher compared to both education, and experience, and thus tells us that the baseline more often is correct when predicting that a job ad is requiring a driving license compared to the others. The baseline achieves a F1-score of 0.62.

5.2.4 Work type

The final sub-problem, work type, achieves an accuracy of 0.75, meaning that the baseline correctly classifies 75% of all ads. The score for recall is 0.65, and is higher compared to driving license, but lower than education and experience. A recall score of 0.65 tells us that our model correctly identifies 65% of all job ads with full-time positions. The precision score is 0.80 which means that when the baseline predicts a job advertisement as a full-time position, it is correct 80% of the time. This is again a fairly high score, indicating that our model often is correct when predicting that a job ad is a full-time position. The baseline achieves a F1-score of 0.72.

The baseline performed reasonably well on Experience, Driving license and Work type (Accuracy > 0.70), but not so well on Education (Accuracy < 0.70).

5.3 Support Vector Machine classifier

The results from the SVM classifier can be found in Table 5.2. We have included the score for accuracy, recall, precision and F1-score for all four classifiers. The results for each sub-problem will be examined further.

Table 5.2: Results from SVM classifier.

	Education	Experience	Driving license	Work type
Accuracy	0.89	0.78	0.89	0.87
Recall	0.78	0.84	0.88	0.87
Precision	0.87	0.79	0.87	0.87
F1-score	0.82	0.81	0.88	0.87

5.3.1 Education

Education receives an accuracy of 0.89, meaning that SVM correctly classifies 89% of all ads. This score is higher compared to the baseline, which achieved a score of 0.65 which indicates that the SVM classifier is better at predicting whether a job advertisement requires education or not. Recall is 0.78 and thereby tells us that our model correctly identifies 78% of all job ads requiring education. The precision score is 0.87 which means that when the model predicts a job advertisement as requiring education, it is correct 87% of the time. The SVM classifier receives a F1-score of 0.82.

5.3.2 Experience

The SVM classifier achieves an accuracy of 0.78 which means that it correctly classifies 78% of all ads. This score is lower compared to education which can indicate that the model is having more difficulties in predicting previous experience compared to education. This result is the opposite to the baseline which found it more difficult to predict whether a job ad requires education or not. The recall score is 0.84 and tells us that the SVM classifier correctly identifies 84% of all job ads requiring previous experience. A precision score of 0.79 means that when the model predicts a job ad as requiring previous experience, it is correct 79% of the time. The F1-score received for experience is 0.81.

5.3.3 Driving license

The SVM classifier for driving license achieves an accuracy 0.89, indicating that it correctly classifies 89% of all ads. It is an accuracy equally as good as education, and better than experience. The score for recall is 0.88 and tells us that the SVM classifier correctly identifies 88% of all job ads requiring a driving licence. A precision score of 0.87 means that when the model predicts a job ad as requiring a driving licence, it is correct 87% of the time. The model achieved a F1-score of 0.88.

5.3.4 Work type

The task where the model is asked to classify whether a job requires full-time or part-time position resulted in an accuracy of 0.87. This score is far better than the baseline (0.75). This indicates that the SVM model can easily predict whether a job ad is requiring full-time or part-time. Precision, recall and F1-score also receive a score of 0.87 which means that our classifier has classified the same amount of false positives as false negatives. Meaning that the classifier incorrectly classified as many ads as full-time positions, as it incorrectly classifies ads as part-time positions.

The Support Vector Machine classifier performed better than the baseline on all categories, especially on Education, Driving license and Work type. The accuracy for Experience was 0.78 for SVM and 0.74 for the baseline, which is a small improvement.

5.4 BERT classifier

The results from KB-BERT classifier can be found in Table 5.3, and the results from MBERT classifier can be found in Table 5.4. The results from all four sub-problems are presented together to get a quick overview. As seen in both tables, KB-BERT (Table 5.3) performs better compared to MBERT. For future comparisons we will only focus on KB-BERT classifier (Table 5.3).

Table 5.3: Results from KB-BERT classifier.

	Education	Experience	Driving license	Work type
Accuracy	0.90	0.81	0.88	0.86
Recall	0.85	0.83	0.89	0.86
Precision	0.84	0.84	0.86	0.86
F1-score	0.84	0.83	0.88	0.86

Table 5.4: Results from MBERT classifier.

	Education	Experience	Driving license	Work type
Accuracy	0.89	0.77	0.87	0.86
Recall	0.82	0.76	0.87	0.84
Precision	0.82	0.81	0.85	0.86
F1-score	0.82	0.79	0.86	0.85

5.4.1 Education

When predicting whether a job ad requires education or not, the BERT model achieved an accuracy of 0.90. BERT thereby manages to correctly classify 90% of the ads. The recall score is slightly lower, with a value of 0.85. This score tells us that BERT correctly identifies 85% of all job ads requiring education. BERT achieves a precision score of 0.84 which means that when it predicts a job ad as requiring education, it is correct 84% of the time. The F1-score is the harmonic mean between precision and recall and is 0.84 for education.

5.4.2 Experience

BERT achieves an accuracy of 0.81 for the sub-problem experience, which means it classifies 81% of the ads correctly. Similarly to the baseline and SVM, BERT is having more difficulties in predicting whether an ad requires previous experience or not, compared to education or not. The recall score is 0.83 which indicates that 83% of all job ads requiring experience are correctly identified. BERT achieves a precision score of 0.84, which demonstrates that when BERT predicts a job ad as requiring previous experience, it is correct 84% of the time. The F1-score for the experience is 0.83.

5.4.3 Driving license

When predicting whether an ad requires a driving license or not, BERT achieves an accuracy of 0.88, which tells us that it classifies 88% of the ads correctly. This score is higher compared to experience, but lower than education. Meaning that BERT finds it more difficult to classify job ads as requiring previous experience or not, compared to education and driving license. The recall score is 0.89 which demonstrates that 89% of all job ads requiring driving licence are correctly identified.

The precision score is 0.86, and indicates that when it predicts a job ad as requiring a driving licence, it is correct 86% of the time. F1-score for the driving licence is 0.88.

5.4.4 Work type

The accuracy for work type is 0.86, meaning that it classifies 86% of the ads correctly. Recall, precision, and F1-score have the same value as accuracy, which indicates that our classifier has classified the same amount of false positives and false negatives. In other words, the classifier incorrectly classifies as many ads as full-time positions, as it incorrectly classified ads as part-time positions. This result is the same as SVM, which also has the same value for accuracy, recall, precision, and F1-score.

BERT classifier also performs better compared to the baseline on all categories, and better compared to the SVM on Education and Experience. It performs slightly worse compared to SVM on Driving license and Work type.

5.5 Comparison between baseline, SVM and BERT

In this section, all three models will be compared for each sub-problem. The results will be presented and compared against each other for the different models.

5.5.1 Education

The results for the first sub-problem, Education vs No education, is presented in Table 5.5, where all three methods are included. If we examine the accuracy of each model we see that SVM and BERT are both essentially better at correctly predicting job advertisements compared to the baseline. This indicates that using a machine learning method that is able to capture the context of job descriptions is useful in this task. SVM and BERT are almost equally good at correctly predicting ads, with an accuracy of 0.89 and 0.90 respectively.

As seen in the Table 5.5, the score of recall is the same for the baseline and SVM (0.78), while it is higher for BERT (0.85), meaning that BERT is better at returning most of the positive data points (ads requiring education). The baseline's precision score is the lowest (0.48) compared to SVM (0.87) and BERT (0.84) which means that SVM is more accurate when predicting a positive data point (ads requiring education). If we compare precision and recall, we notice that recall (0.78) is essentially higher than precision (0.48) for the baseline. This means that the baseline will return many ads as requiring education, where some will be correct but a lot of them are incorrect when evaluated. It is the opposite for SVM where precision (0.87) is higher than recall (0.78) as seen in Table 5.5, meaning that SVM will return fewer ads requiring education, where most of them are correct. BERT has both high recall (0.85) and precision (0.84), which shows that the model is returning accurate results and is able to find the positive cases in our dataset.

Education is slightly imbalanced, where there are more ads not requiring education compared to requiring education. To compare and evaluate models with imbalanced data, F1-score is commonly used. As seen in table 5.5, BERT achieves the highest F1-score of all three models.

Table 5.5: Combined results from all methods for Education/No education.

	Accuracy	Recall	Precision	F1-score
Baseline	0.65	0.78	0.48	0.59
SVM	0.89	0.78	**0.87**	0.82
BERT	**0.90**	**0.85**	0.84	**0.84**

5.5.2 Experience

The results for the second sub-problem, Experience vs No experience, is presented in Table 5.6, where all three methods are included. BERT is again the model with the highest accuracy, with a score of 0.81. SVM and BERT both achieved lower accuracy scores for experience compared to education. This indicates that it is more difficult for the two classifiers to capture and understand whether a job ad requires previous experience or not. The reason for this is unknown, but it could be that job advertisements more clearly state when education is required or not, compared to when previous experience is required or not.

All three models receive similar recall scores, but SVM and BERT have a slightly higher score. It tells us that both machine learning models are better at correctly identifying all ads requiring previous experience, compared to the baseline. The precision score is again higher for SVM, and BERT compared to the baseline. A higher precision score relates to a lower false positive rate, meaning that less ads are classified as positive when they actually belong to the negative class. It means that when a job advertisement is predicted as requiring previous experience, SVM and BERT are more often correct compared to the baseline. BERT achieves a higher score for both recall and precision compared to SVM and the baseline, which result in a higher F1-score. This information is useful when choosing the appropriate model for the sub-problem since we aim for both high precision and recall when choosing a model.

Table 5.6: Combined results from all methods for Experience/No experience.

	Accuracy	Recall	Precision	F1-score
Baseline	0.74	0.81	0.75	0.78
SVM	0.78	**0.84**	0.79	0.81
BERT	**0.81**	0.83	**0.84**	**0.83**

5.5.3 Driving license

The results for the third sub-problem, Driving license vs No driving license, is presented in Table 5.7, where all three methods are included. For this sub-problem, SVM achieves the highest accuracy of 0.89, where BERT is close by with a score of 0.88. They both outperform the baseline as in the previous sub-problems. The recall score is low for the baseline, which demonstrates that some of the ads requiring driving license are never predicted. A higher recall score achieved in SVM and BERT is more desirable as the classifier will identify more ads requiring driving license compared to the baseline. Precision on the other hand is high for all three models, around 0.87, and shows that the models are equally good at correctly predicting ads requiring driving license. Compared to the baseline, both SVM and BERT achieve high scores for recall and precision, and thereby have a high F1-score of 0.88.

Table 5.7: Combined results from all methods for Driving license/No driving license.

	Accuracy	Recall	Precision	F1-score
Baseline	0.74	0.49	**0.87**	0.62
SVM	**0.89**	0.88	**0.87**	**0.88**
BERT	0.88	**0.89**	0.86	**0.88**

5.5.4 Work type

The results for the last sub-problem, Full-time vs Part-time, is presented in Table 5.8, where all three methods are included. For full-time vs part-time, SVM achieves the highest accuracy (0.87). However, BERT is once again close by, with an accuracy of 0.86. The baseline achieves a lower accuracy, as for the other sub-problems which demonstrates that by using machine learning models, one can capture the meaning of job advertisements better.

Precision and recall has the same score (0.87) for SVM and BERT (0.86), which tells us that the false positive rate is equal to the false negative rate. This means that there are as many ads that are predicted as full-time positions, when actually being part-time positions, as ads being predicted as part-time positions when actually being full-time positions. For that matter, SVM is slightly better compared to BERT since it scores 0.87 on accuracy, precision, recall and F1-score compared to 0.86 for BERT on the same metrics. The baseline on the other hand has lower scores for precision, recall and F1-score. Recall is lower compared to precision, which demonstrates that the baseline gives us fewer results with higher accuracy.

SVM and BERT outperforms the baseline on all sub-problems. SVM performs slightly better than BERT on Driving license and Work type, whereas BERT performs slightly better than SVM on Education and Experience.

Table 5.8: Combined results from all methods for Full-time/Part-time.

	Accuracy	Recall	Precision	F1-score
Baseline	0.75	0.65	0.80	0.72
SVM	**0.87**	**0.87**	**0.87**	**0.87**
BERT	0.86	0.86	0.86	0.86

5.6 Learning curve

In this thesis, a dataset of 9,131 ads have been used when training the models. This is due to the fact that annotation had to be done manually which is a time-consuming task. However, it is interesting to investigate how the models develop and learn as more data is added. This could give an indication to the Swedish Public Employment Service of whether it should invest in annotating more ads in the future to increase the models' performance. We have therefore plotted a learning curve for the sub-problem Education/No education in SVM and BERT to see how the training- and test score change as more data is added. Figure 5.1 presents the result for the SVM model, and we can see that by adding more data, the test score increases and the training score slowly decreases. This indicates that the model is still learning and the test score could increase further if more data was added.

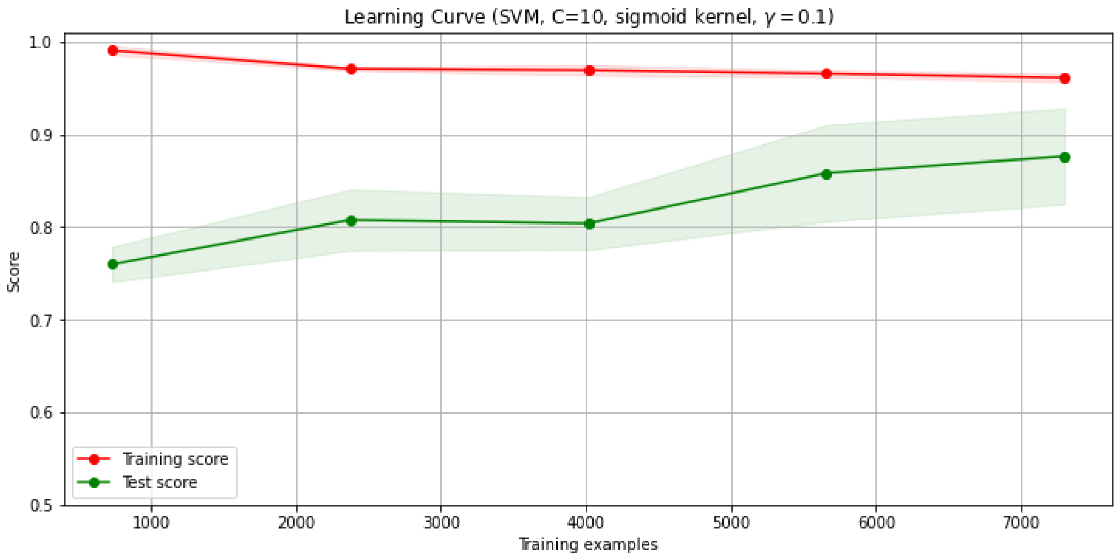


Figure 5.1: Learning curve for SVM model for sub-problem Education vs No education.

Figure 5.2 presents the learning curve for BERT on Education/No education. We can see that by adding more data, the model does not continue to learn, and stays around the same test score. The training score initially starts to decrease, but increases again after 6,000 ads. The test score stays around 0.90 even if more data is added to the training phase. These results differ from Figure 5.1 where SVM was still learning when more data was added.

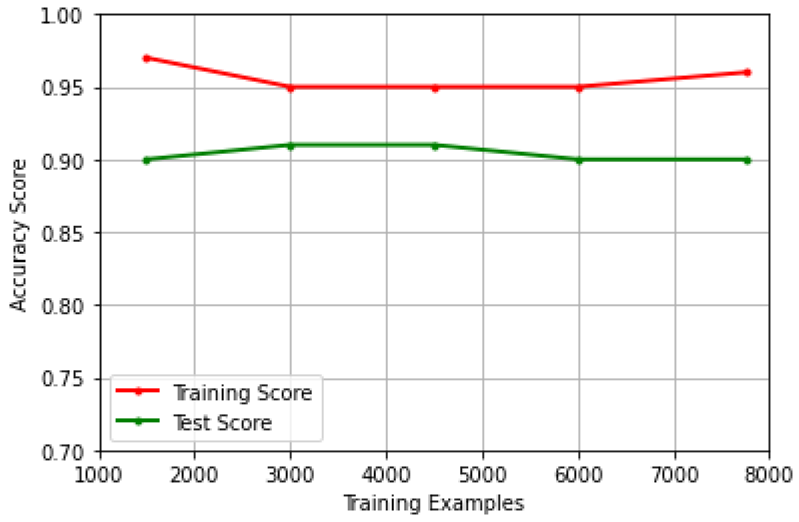


Figure 5.2: Learning curve for BERT model for sub-problem Education vs No education.

6 Discussion

When discussing the results of models, it is often a question of whether the models perform well or not. Results influence whether models can be applied in reality or not, and since this thesis is written in collaboration with the Swedish Public Employment Service, the question of how good is good enough needs to be discussed with them. When they are developing and implementing models, they normally aim for an accuracy of 98% or higher depending on the task. As mentioned before, they work closely with Statistics Sweden (SCB) to create accurate statistics about Sweden's labor market with the result of models developed by the Swedish Public Employment Service. Therefore, it is important that models are accurate and reflect reality when their result forms the basis of statistics about the labor market. However, not all tasks and models from the Swedish Public Employment Service are used for statistics. Our task of extracting metadata from job advertisements will not be used for any statistics. The main purpose of automatically extracting metadata is to create higher value for the labor market and create a better user experience. Therefore, the results from such models will only be used to facilitate the process of job seekers finding suitable jobs. The results, therefore, do not have to be 98% or higher, an accuracy around 90% and higher is acceptable. Higher accuracy is preferable since it will correctly classify more ads, but in this case, the main purpose is to offer individuals the ability to filter ads. It is therefore more about creating a user-friendly website where individuals easily can find suitable jobs, rather than how accurate the models are. Another notable aspect when discussing accuracy of this kind of tool is how time-consuming it would be to manually carry out a classification to extract metadata from job advertisements. Even if accuracy is not as high as one would initially have wanted, it saves a lot of time by using this kind of tool that handles it automatically.

BERT is the only model that succeeds to achieve an accuracy of 90% for the sub-problem Education. However, if we examine the learning curve for SVM in Figure 5.1, we can see that the model has not yet converged and is still learning for the sub-problem Education. It is still learning as the test score is increasing when more training examples are added. This could indicate that it is worth for the Swedish Public Employment Service to invest in manually annotating more data to create a larger dataset to train the models on.

If we compare the learning curves for SVM and BERT we can see that they differ. The test score for SVM is still increasing, whereas the test score for BERT stays the same when more data is added. For training sample size less than 5,500 there is a greater difference between training and testing which points to overfitting. Although, for a training size larger than 5,500 the model is better and the gap between training and testing is getting smaller. This indicates that the model is getting better and is learning from the data. However, if we examine the learning curve for BERT, we see that the gap between training and testing is consistent even when the training size increases. This is a sign of overfitting and high variance in the model.

Another notable aspect to address when discussing the results of the models is the computational aspect. SVM and BERT are very close in their performance on all sub-problems and it can therefore be interesting to compare the time it took to train and test the two models. BERT took longer to train compared to SVM, in general it took almost twice as long to train BERT compared to SVM. SVM could approximately handle 2 documents/second, while BERT could handle 1 document/second. And despite this, BERT did not lead to significantly better results compared to SVM.

6.1 Sub-problems

When we examine the results from the models we notice some similarities between the achieved accuracy from SVM and BERT for the different sub-problems. Education achieves an accuracy of 89% and 90% for SVM and BERT respectively. Experience achieves an accuracy of 78% and 81% for SVM and BERT respectively. Driving license achieves an accuracy of 89% and 88% for SVM and BERT respectively. Work type achieves an accuracy of 87% and 86% for SVM and BERT respectively. The two models are close in performance, where BERT is slightly better for the first two sub-problems and SVM for the last two. However, if we compare the sub-problems against each other we see that Experience is notably lower compared to the other, for both SVM and BERT. This indicates that predicting whether a job requires previous experience or not is a more difficult task compared to the other sub-problems for both classifiers. To understand why, we need to examine the nature of the sub-problems. Education, Driving license, and Work type are all relatively straightforward with little room for interpretation. Education itself is understandable where everyone shares the same understanding of what is meant by education. Both employers and employees know how and where to get an education. There is also a mutual understanding and knowledge internationally of what is meant by education, which creates the same understanding regardless of the origin of employers or employees. The same applies to Driving license and Work type. A driving license is given at a special institution, as a type of education where one must pass a test to get a driving license. There are different types of driver's licenses with different qualifications that may be required for different jobs. It is clear what is meant and what is required to meet such requirements. Regarding Work type, it is known that a full-time position equals 40h/week according to Swedish standards. Anything less than that counts as part-time, which creates a mutual understanding of the concepts.

Previous experience is, on the other hand, a more diffuse area, where employers and employees may have different perceptions of what is meant by previous experience. It can differ between employers, which can make it confusing for job seekers to actually know what is required of them to apply for the job. There is no mutual understanding of what previous experience means, if it is experience that is documented from previous jobs or education, or if it can be gained somewhere else. Such diffusion can affect the performance of the models and therefore explain why Experience does not reach as high accuracy as the other sub-problems. With this in mind, it is difficult to tweak the models to increase their performance since it lies within the nature of the sub-problems. If there is no mutual understanding by us humans, it will be difficult for algorithms to find patterns among the ads.

6.2 Dataset

The dataset generated from the Swedish Public Employment Service consists of previous job advertisements posted at Platsbanken and contains unique job ads, as well as duplicates which are job ads with the same description. From the point of view of the Swedish Public Employment Service, this is not duplicates as they all have different IDs and could be posted in different cities and at different times. When models are trained on data that are exactly the same (text in this case), it will not generalize as well and make accurate predictions on new unseen ads. To cope with this issue we have chosen to consider it as a problem of duplication. Such ads were therefore removed to create a dataset of unique texts that was used when training the models. This is an example of how reality and raw data can differ and how one can not always use real life data as an input to a model without any modifications. In reality, these ads are not duplicates, but when they are handled and processed they are considered as duplicates due to their nature which can not be changed.

Another notable thing regarding the dataset is the language of the written job advertisements. Platsbanken is a Swedish website that posts jobs in Sweden, resulting in most job advertisements being written in Swedish. However, during annotation, we noticed that some ads were written in English. It was a small number of ads but it resulted in the final dataset containing English ads. We chose to keep such ads to create a representative picture of reality. When developing the models, we mostly chose tools and libraries helpful for Swedish text since almost all job advertisements are written in Swedish. We chose Swedish stop-words and a Swedish stemmer when developing our SVM model, and a Swedish pre-trained model in BERT. That we have chosen tools for Swedish text might have affected the performance of the models. Especially BERT which makes use of a pre-trained Swedish model, which means that the model has not been pre-trained on any English text at all. Even though there are so few ads written in English now, there may be more advertisements written in English in the future. On such occasions, it is important to implement models that can handle and make accurate predictions in both languages, and train the models on more English ads. To explore solutions to the problem, we explored a multilingual BERT model through HuggingFace. However, as seen in Table 5.4 it performed worse compared to the Swedish model. This could be due to the fact that the multilingual model has not been pre-trained on such a large corpus as the Swedish model. However, it is important to address this issue so the Swedish Public

Employment Service can investigate other possible solutions to developing models that are accurate in more languages than Swedish.

6.3 Annotation process

Performing manual annotation of data is a very time-consuming task where individuals go through thousands of data points by hand. The job advertisements' description is often of different lengths and many advertisements have a long description that takes time to read to understand which class to choose. Our annotation took a total of about 3 weeks to complete, where we both annotated data eight hours a day which corresponds to a total of 240 hours. It is thus a time-consuming task that requires full concentration when performed to maintain a good and consistent quality throughout the annotation process.

Manually annotating whether a job advertisement requires education or not may seem like a simple task, but in reality, the answer was not always that obvious. Many job advertisements were very obvious regarding whether they require education or not by clearly stating it, while others were more difficult to understand based on the description. For example, a certain job in an industry from one company could require education, which other companies did not require. This meant that as an annotator you needed to read each job advertisement clearly to know whether education was required or not, rather than what usually applies to such jobs which was challenging as some descriptions were quite long and difficult to understand what actually was required. Another example relates to whether a certain education is meritorious. In several cases, it was difficult to understand whether the job really required education, or whether it was rather something that was meritorious. This was especially challenging as one of the annotators did not speak Swedish, which makes it difficult to understand the text since most ads are written in Swedish. The ads were then translated into Swedish, which made the process take even longer and the translation is not always accurate which could affect the chosen class.

As mentioned in Section 5.1, we carried out a quality check, where 1,000 advertisements were double-annotated and Cohen's Kappa was used to measure the quality. The result came back at 0.905, which shows consistent annotation between the two annotators, but not to 100%. It can be seen as a simple task to understand whether a job advertisement requires education or not, but based on the results, there were ads where we had different views on the matter. This can of course affect the quality of the dataset and the performance of the models as the data forms the basis of learning in the models. However, we were only two annotators during this process, which made it easier for us to discuss certain cases during the process which made us more comfortable in our role. It might have been more difficult if there were more annotators involved with different views on the matter. We therefore believe that it is important to study the domain you annotate and exemplify more in the annotation specification to create a better common understanding of the annotation for all annotators. Otherwise, the dataset risks poorer quality, which can affect the outcomes of the models.

6.4 Truncating text in BERT

As previously mentioned, BERT is a language model that makes use of pre-trained models to perform accurate tasks on textual data. When using BERT, there is a limit to the number of tokens that can be included when training the model with our data. A maximum of 512 tokens can be included, which results in some job advertisements being truncated, and important information for the classification may be lost. This is in contrast to SVM which has no restriction but includes all words in a text, regardless of length. The fact that there are job ads that will be truncated may lead to valuable information for the classification being excluded, which can affect the model's performance. We performed an experiment as shown in Table 4.3, which shows that the model achieved the highest accuracy when the first 510 tokens were used. However, we do not know how much better or worse the model's performance would have been if all tokens were included during the training process. When you truncate text, it can also result in sentences being interrupted and cut off, which could lead to the whole sentence losing comprehension. This can make it more difficult for models to understand the context. For example, if the last word of a truncated job advertisement is *"driving license"*, the following word could, among other things, be either *"driving license is required"* or *"driving license is not required"*. There is a great difference between the two options since they belong to different classes. Such information is therefore valuable for the model's learning, and it will be more difficult to distinguish between the two classes for such advertisements when words are missing.

Even though BERT truncates text and uses the first 510 tokens of a job advertisement, the model still achieves good performance. BERT is better at classifying Education and Experience compared to SVM, and SVM is better at classifying Driving license and Work type compared to BERT. It would therefore be interesting to see how a model like BERT would perform if all words of a job advertisement are included when being trained. This, of course, comes with drawbacks as it takes much longer to train and run a model that will handle that much more data.

6.5 Summary and conclusion

In this thesis, we aimed to investigate different methods and techniques that can be used to automatically extract metadata from unstructured texts. By extracting metadata from unstructured texts, one can transform the texts into a structured format that can be used for organising and accessing unstructured data in a structured manner. This specific use case of extracting metadata from job advertisements can facilitate the process of finding a suitable job as the user has the ability to search and filter ads based on preferences on the Swedish Public Employment Service's website Platsbanken. Three different models have been evaluated on the subject: a dictionary lookup, Support Vector Machine (SVM) and BERT.

SVM and BERT successfully manage to classify job advertisements into predefined classes with a relatively high accuracy (78-90%) where the predicted class can be used as metadata for the specific job advertisement. The sub-problem Experience obtained the lowest accuracy for SVM and BERT (78-81%). This could be due to the fact that there is no clear understanding of what is meant by previous experience, compared to the other sub-problems. These results indicate that models such as SVM and BERT can be used to successfully extract metadata automatically from unstructured texts. Such a tool can be applied in several domains to enable search and retrieval of unstructured texts. The Swedish Public Employment Service can implement such a tool to display and offer more job advertisements on their website, and create a better user experience when individuals can search and filter ads based on their preferences. Working with such an automated tool will also save an incredible amount of time compared to if it were to be performed manually.

Bibliography

[1] Arbetsförmedlingen. (2021). Statistik om arbetslöshet och arbetssökande. Available: https://arbetsformedlingen.se/statistik/arbetsloshet (accessed 2022-03-25)

[2] Arbetsförmedlingen. (n.d.). Vårt uppdrag. Available: https://arbetsformedlingen.se/om-oss/var-verksamhet/vart-uppdrag (accessed 2021-12-10)

[3] Han, H., Giles, C. L., Manavoglu, E., Zha, H., Zhang, Z., Fox, E. A. (2003, May). Automatic document metadata extraction using support vector machines. In *2003 Joint Conference on Digital Libraries, 2003. Proceedings.* (pp. 37-48). IEEE.

[4] Misra, D., Chen, S., Thoma, G. R. (2009, January). A system for automated extraction of metadata from scanned documents using layout recognition and string pattern search models. In *Archiving Conference* (Vol. 2009, No. 1, pp. 107-112). Society for Imaging Science and Technology.

[5] Aleksandar Kovačević, D., Ivanović, B., Milosavljević, Z., Konjović and D., Surla. Automatic extraction of metadata from scientific publications for CRIS systems. IN *September 2011,Program Electronic Library and Information Systems 45(4):376-396.*

[6] Yilmazel, O., Finneran, C. M., Liddy, E. D. (2004, June). Metaextract: an NLP system to automatically assign metadata. In *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries, 2004.* (pp. 241-242). IEEE.

[7] Azimjonov, J., Alikhanov, J. (2018). Rule based metadata extraction framework from academic articles. *arXiv preprint arXiv:1807.09009.*

[8] Huynh, T., Hoang, K. (2010, November). GATE framework based metadata extraction from scientific papers. In *2010 International Conference on Education and Management Technology* (pp. 188-191). IEEE.

[9] Lu, X., Kahle, B., Wang, J. Z., Giles, C. L. (2008, June). A metadata generation system for scanned scientific volumes. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries* (pp. 167-176).

[10] Borko, H., Bernick, M. (1963). Automatic document classification. *Journal of the ACM (JACM), 10(2),* 151-162.

[11] Wang, Z. Q., Sun, X., Zhang, D. X., Li, X. (2006, August). An optimal SVM-based text classification algorithm. In *2006 International Conference on Machine Learning and Cybernetics* (pp. 1378-1381). IEEE.

[12] Yang, Y., Liu, X. (1999, August). A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 42-49).

[13] Adhikari, A., Ram, A., Tang, R., Lin, J. (2019). Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398.*

[14] Khadhraoui, M., Bellaaj, H., Ammar, M. B., Hamam, H., Jmaiel, M. (2022). Survey of BERT-Base Models for Scientific Text Classification: COVID-19 Case Study. *Applied Sciences, 12*(6), 2891.

[15] Verma, A., Yurov, K. M., Lane, P. L., Yurova, Y. V. (2019). An investigation of skill requirements for business and data analytics positions: A content analysis of job advertisements. *Journal of Education for Business, 94*(4), 243-250.

[16] Brooks, N. G., Greer, T. H., Morris, S. A. (2018). Information systems security job advertisement analysis: Skills review and implications for information systems curriculum. *Journal of Education for Business, 93*(5), 213-221.

[17] Verma, A., Lamsal, K., Verma, P. (2021). An investigation of skill requirements in artificial intelligence and machine learning job advertisements. *Industry and Higher Education*, 0950422221990990.

[18] Noble, W. S. (2006). What is a support vector machine?. Nature biotechnology, 24(12), 1565-1567.

[19] Gandhi, R. (2018). Support Vector Machine - Introduction to Machine Learning Algorithms. Towards Data Science. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (accessed 2022-02-10)

[20] Cortes, C., Vapnik, V. (1995). Support-vector networks. *Machine learning, 20*(3), 273-297.

[21] Yang, L., Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing, 415*, 295-316.

[22] Friedrichs, F., Igel, C. (2005). Evolutionary tuning of multiple SVM parameters. *Neurocomputing, 64*, 107-117.

[23] Scikit Learn. (n.d). RBF SVM parameters. Available: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html (Accessed 2022-02-28)

[24] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon, 4*(11), e00938.

[25] Shanmuganathan, S. (2016). Artificial neural network modelling: An introduction. In *Artificial neural network modelling* (pp. 1-14). Springer, Cham.

[26] Svozil, D., Kvasnicka, V., Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems, 39*(1), 43-62.

[27] IBM. Neural Networks. (2020). Available: https://www.ibm.com/cloud/learn/neural-networks (accessed 2022-03-03)

[28] Sharma, S., Sharma, S., Athaiya, A. (2017). Activation functions in neural networks. *towards data science, 6*(12), 310-316.

[29] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research, 15*(1), 1929-1958.

[30] Smith, L. N. (2018). A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820.*

[31] Han, S. H., Kim, K. W., Kim, S., Youn, Y. C. (2018). Artificial neural network: understanding the basic concepts without mathematics. *Dementia and Neurocognitive Disorders, 17*(3), 83-89.

[32] Choi, D., Shallue, C. J., Nado, Z., Lee, J., Maddison, C. J., Dahl, G. E. (2019). On empirical comparisons of optimizers for deep learning. *arXiv preprint arXiv:1910.05446.*

[33] Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

[34] Loshchilov, I., Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101.*

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

[36] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems, 30.*

[37] Malmsten, M., Börjeson, L., Haffenden, C. (2020). Playing with Words at the National Library of Sweden–Making a Swedish BERT. *arXiv preprint arXiv:2007.01658.*

[38] Xu, H., Van Durme, B., Murray, K. (2021). BERT, mBERT, or BiBERT? A Study on Contextualized Embeddings for Neural Machine Translation. *arXiv preprint arXiv:2109.04588.*

[39] Chowdhary, K. (2020). Natural language processing. *Fundamentals of artificial intelligence*, 603-649.

[40] Uysal, A. K., Gunal, S. (2014). The impact of preprocessing on text classification. *Information processing management, 50*(1), 104-112.

[41] Denny, M. J., Spirling, A. (2018). Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it. *Political Analysis, 26*(2), 168-189.

[42] Hickman, L., Thapa, S., Tay, L., Cao, M., Srinivasan, P. (2022). Text preprocessing for text mining in organizational research: Review and recommendations. *Organizational Research Methods, 25*(1), 114-146.

[43] Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. *Information Processing Management, 39*(1), 45-65.

[44] Hossin, M., Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining knowledge management process, 5*(2), 1.

[45] Leonard, L. C. (2017). Web-based behavioral modeling for continuous user authentication (CUA). In *Advances in Computers* (Vol. 105, pp. 1-44). Elsevier.

[46] Siphu Langeni, Python package from Pypi.org. Available: https://github.com/SiphuLangeni/tortus (Accessed 2022-01-15)

[47] Sun, C., Qiu, X., Xu, Y., Huang, X. (2019, October). How to fine-tune bert for text classification?. In *China national conference on Chinese computational linguistics* (pp. 194-206). Springer, Cham.

[48] McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochemia medica, 22*(3), 276-282.

A Appendix 1

A.1 Annotation Specification

The purpose of the annotation process is to create a dataset that can be used to classify job advertisements based on predefined classes, to enable searching and filtering of job ads. Supervised learning models will classify each job advertisement according to predefined classes, and in order to correctly classify job ads, they need to be trained on labelled data. At present, there is no such dataset and we (the authors of this thesis) will therefore create one ourselves by annotating whether a job advertisement require education or not. The data is given by the Swedish Public Employment Service.

Annotators can choose from two predefined classes where each job advertisement is labelled with one of the two classes. A job ad can, for example, not be labelled as both “Experience” and “No experience”. We will describe Education further to create a mutual understanding of what defines the given classes for all annotators.

The predefined classes are:

- **Education**: A job advertisement should be labelled as “Education” if the job requires previous education to apply for the job. For example, “bachelor degree in economics”. This also includes jobs where education is not clearly stated, but understood that it is needed, for example, a job ad for a doctor. A job advertisement can be either “Education” or “No education”, not both.
- **No education**: A job advertisement should be labelled as “No education” if the job does not require previous education. A job advertisement can be either “Education” or “No education”, not both. If education is not specified in the description, the default value will be “No education”, except if it is understood that this job requires education, for example, a doctor.

A final note is if an ad states that something is meritorious, then we will not classify it as a requirement.

A.2 Baseline dictionaries

The dictionaries created in each sub-problem are presented, only including a subset of 300 key-value pairs.

A.2.1 Education

A.2.1.1 Requiring education

{'möjlighet': 836, 'for': 836, 'efter': 831, ' · ': 831, 'inte': 825, 'under': 818, 'varje': 810, 'utbildning': 795, 'barn': 763, 'sker': 758, 'uppdrag': 758, 'nära': 749, 'our': 748, 'team': 746, 'andra': 744, 'arbetet': 738, 'få': 730, ' • ': 727, 'finns': 714, 'genom': 699, 'ta': 698, 'skapa': 691, 'del': 683, 'are': 679, 'person': 677, 'bra': 674, 'arbetsuppgifter': 674, 'as': 667, 'sedan': 662, 'välkommen': 660, 'nya': 656, 'utveckla': 651, 'jobba': 637, 'innebär': 635, 'också': 627, 'behöver': 623, 'kunna': 610, 'is': 607, 'enligt': 580, 'kunder': 580, 'företag': 574, '–': 573, 'skolan': 571, 'löpande': 562, 'tal': 556, 'hela': 552, 'ligger': 548, 'förmåga': 547, 'lärare': 545, 'kunskaper': 542, 'minst': 540, 'många': 539, 'göra': 531, 'rollen': 527, 'hjälpa': 513, 'sig': 512, 'viktigt': 511, 'idag': 510, 'passar': 506, 'detta': 502, 'kollegor': 499, 'år': 494, 'utvecklas': 491, 'ca': 491, 'experience': 489, 'medarbetare': 485, 'will': 483, 'tar': 482, 'verksamhet': 480, 'när': 476, 'via': 475, 'över': 473, 'per': 463, 'personlig': 449, 'därför': 448, 'positiv': 445, 'stort': 442, 'egna': 442, 'ge': 441, 'vikt': 437, 'upp': 436, 'meriterande': 427, 'bästa': 427, 'gör': 426, 'ingår': 425, 'ansvar': 424, 'alltid': 424, 'bli': 424, 'intresse': 417, 'be': 412, 'mot': 411, 'består': 411, 'skola': 408, 'hög': 407, 'sina': 405, 'utifrån': 403, 'sin': 403, 'roll': 401, 'just': 400, 'nå': 396, 'hjälper': 395, 'utveckling': 392, 'sveriges': 389, 'man': 388, 'krav': 386, 'års': 385, 'inför': 384, 'kunskap': 384, 'tidigare': 383, 'personal': 383, 'undervisning': 382, 'se': 382, 'verksamheten': 377, 'mellan': 376, 'ansök': 374, 'fram': 372, 'stockholm': 372, 'trivs': 371, 'människor': 371, 'kompetens': 369, 'blir': 369, 'hur': 369, 'här': 363, 'allt': 363, 'möjligheter': 362, 'samarbete': 360, 'ansökan!': 358, 'that': 357, 'lära': 355, 'your': 354, 'ab': 353, 'två': 351, 'kontakt': 348, 'övriga': 348, 'viktig': 347, 'behov': 347, 'fler': 346, 'elever.': 345, 'have': 345, 'intervjuer': 343, 'mål': 341, 'skrift': 341, 'jobbar': 341, 'eleverna': 338, 'emot': 338, 'bland': 338, 'work': 338, 'engagemang': 335, 'rätt': 333, 'on': 333, 'an': 332, 'pedagogiska': 330, 'sätt': 329, 'all': 325, '1': 324, 'redan': 322, 'erbjuda': 322, 'denna': 317, 'service': 317, 'vilket': 316, 'elev': 316, 'lätt': 314, 'dig.': 313, 'trygg': 313, 'engelska': 310, 'skicka': 310, 'tycker': 308, 'skolans': 307, 'självständigt': 304, 'går': 303, 'or': 303, 'arbetat': 302, 'undervisa': 299, 'vad': 299, 'privatlärare': 299, 'personliga': 296, 'information': 293, 'flexibel': 293, 'väl': 291, 'några': 291, 'flera': 291, 'behov.': 291, 'såväl': 290, 'komma': 290, 'annan': 289, 'relevant': 287, 'tjänst': 286, 'at': 285, 'samtidigt': 284, 'bättre': 283, 'sista': 279, 'fokus': 279, 'tillträde': 277, 'förutsättningar': 276, 'miljö': 276, 'växande': 275, 'ansvarar': 275, 'digitala': 275, 'business': 272, 'kvalifikationer': 272, 'lägger': 270, 'stöd': 266, 'skillnad': 266, 'hemsida': 266, 'innan': 265, 'start': 264, 'deras': 262, 'meriterande.': 262, 'stora': 261, 'läs': 261, 'än': 260, 'skapar': 258, 'jobb': 256, '6': 256, 'annat': 255, 'minuter': 254, 'oavsett': 252, 'spännande': 252, 'uppdraget': 249, 'samarbeta': 249, 'brinner': 246, 'this': 246, 'personer': 245, 'år.': 245, 'pedagogisk': 245, 'from': 245, 'intresserad': 244, 'tror': 244, 'hemma': 243, 'öppen': 243, 'delar': 242, 'ger': 240, 'vision': 239, 'working': 238, 'gillar': 236, 'känner': 236, 'eget': 234, 'skall': 233, 'vidare': 231, 'urval': 229, 'undervisningen': 229, 'it': 227, 'företaget': 226, 'förskola': 225, 'aktivt': 225, 'sjuksköterska': 224, 'ut': 224, 'development': 224, 'egen': 222, 'sverige': 220, 'anpassa': 220, '#jobbjustnu': 220, 'social': 220,

’jobbet’: 219, ’perfekt’: 219, ’5’: 219, ’skolor’: 218, ’legitimerad’: 218, ’högt’: 216, ’engagerad’: 215, ’överenskommelse.’: 213, ’barnen’: 213, ’bidra’: 212, ’tro’: 212, ’https://www.allakando.se/laxhjalp-jobb/’: 211, ’relationer’: 211, ’lärande’: 210, ’*’: 210, ’snabbast’: 209, ’läxhjälpare’: 206, ’leda’: 205, ’skrift.’: 205, ’genomför’: 205, ’timmar’: 204, ’söka’: 204, ’helt’: 204, ’föräldrar’: 203, ’tillgång’: 202, ’cv’: 201, ’frågor’: 201, ’antal’: 200, ’kanske’: 200, ’digital’: 200, ’oss.’: 199, ’driva’: 198, ’arbete.’: 198, ’personligt’: 198, ’tre’: 197, ’bör’: 196, ’hantera’: 196, ’lön’: 196, ’allakando?’: 196, ’som:’: 196, ’bidrar’: 194, ’deltid’: 193, ’krav.’: 192, ’000’: 192, ’utbildning.’: 192, ’egenskaper’: 192, ’dig:’: 191, ... }

A.2.1.2 Not requiring education

{’detta’: 1244, ’erbjuder’: 1237, ’personlig’: 1232, ’stor’: 1225, ’tidigare’: 1212, ’viktigt’: 1192, ’service’: 1172, ’ · ’: 1155, ’mer’: 1154, ’andra’: 1135, ’man’: 1130, ’medarbetare’: 1126, ’tillsammans’: 1118, ’sker’: 1117, ’företag’: 1071, ’krav’: 1049, ’enligt’: 1034, ’allt’: 1030, ’goda’: 1029, ’jobbet’: 1008, ’’: 1004, ’a’: 1004, ’möjlighet’: 995, ’också’: 994, ’via’: 985, ’gillar’: 980, ’över’: 976, ’jobbar’: 971, ’körkort’: 962, ’rollen’: 955, ’ab’: 946, ’team’: 943, ’välkommen’: 941, ’få’: 939, ’ • ’: 930, ’hela’: 928, ’löpande’: 924, ’alltid’: 920, ’därför’: 890, ’ca’: 887, ’tal’: 886, ’tycker’: 880, ’rätt’: 876, ’the’: 869, ’försäljning’: 854, ’genom’: 850, ’när’: 846, ’sig’: 840, ’jag’: 821, ’ge’: 812, ’bli’: 810, ’meriterande’: 805, ’positiv’: 783, ’hög’: 783, ’stort’: 778, ’högt’: 768, ’barn’: 765, ’krav.’: 756, ’utbildning’: 753, ’to’: 748, ’upp’: 745, ’bästa’: 743, ’sedan’: 740, ’tar’: 729, ’ingår’: 723, ’idag’: 722, ’göra’: 720, ’vad’: 714, ’gör’: 711, ’redan’: 711, ’ansvar’: 711, ’personliga’: 710, ’mellan’: 707, ’utvecklas’: 697, ’emot’: 690, ’här’: 689, ’egen’: 685, ’många’: 681, ’skicka’: 676, ’hjälpa’: 672, ’varje’: 672, ’måste’: 671, ’skapa’: 664, ’#jobbjustnu’: 663, ’lära’: 653, ’trivs’: 653, ’innan’: 651, ’intresse’: 650, ’mot’: 645, ’lätt’: 644, ’kunden’: 643, ’*’: 643, ’–’: 642, ’övriga’: 639, ’behov.’: 638, ’assistent’: 629, ’år’: 628, ’blir’: 615, ’självständigt’: 614, ’personal’: 614, ’jobb’: 605, ’you’: 602, ’social’: 601, ’människor’: 598, ’of’: 587, ’några’: 586, ’flexibel’: 586, ’we’: 584, ’arbetsplatsen’: 583, ’annat’: 581, ’vikt’: 578, ’förmåga’: 569, ’kunder.’: 567, ’eget’: 566, ’vilket’: 565, ’egna’: 563, ’noggrann’: 560, ’öppen’: 554, ’ut’: 544, ’fokus’: 543, ’kontakt’: 538, ’bör’: 535, ’b-körkort’: 534, ’går’: 533, ’företaget’: 531, ’kollegor’: 529, ’lön’: 529, ’just’: 525, ’kunskaper’: 525, ’något’: 524, ’inget’: 521, ’behov’: 519, ’meriterande.’: 515, ’stora’: 514, ’mat’: 514, ’läs’: 513, ’flera’: 512, ’skall’: 508, ’utan’: 507, ’anpassa’: 506, ’hjälp’: 505, ’års’: 502, ’minst’: 502, ’varierande’: 501, ’sverige’: 497, ’denna’: 496, ’bil’: 494, ’säljare’: 493, ’be’: 491, ’for’: 491, ’bland’: 489, ’sin’: 487, ’per’: 484, ’består’: 482, ’oss.’: 478, ’själv’: 476, ’se’: 475, ’två’: 473, ’personer’: 472, ’fokuserar’: 466, ’arbetsuppgifterna’: 462, ’krävs’: 461, ’komma’: 460, ’dag’: 460, ’öppna’: 460, ’are’: 457, ’hantera’: 455, ’roll’: 454, ’start’: 454, ’älskar’: 451, ’kompetens,’: 450, ’erbjuda’: 447, ’vidare’: 446, ’stockholm’: 446, ’kund’: 445, ’skrift’: 444, ’utveckla’: 444, ’runt’: 443, ’ansökan!’: 442, ’is’: 441, ’ligger’: 440, ’hur’: 440, ’fler’: 439, ’förutsättningar.’: 438, ’verksamhet’: 430, ’ute’: 426, ’tror’: 425, ’arbetsplats’: 423, ’fram’: 422, ’någon’: 420, ’engelska’: 420, ’känner’: 413, ’helt’: 409, ’fast’: 408, ’produkter’: 408, ’såväl’: 405, ’ansökningar’: 404, ’mål’: 400, ’tillgång’: 397, ’intervjuer’: 396, ’samtidigt’: 395, ’sveriges’: 393, ’uppdrag’: 392, ’möjligheter’: 391, ’största’: 391, ’tjänst’: 390, ’leverera’: 388, ’cv’: 386, ’assistans’:

386, ’nära’: 386, ’extra’: 383, ’sista’: 381, ’ger’: 380, ’timmar’: 378, ’främst’: 377, ’hålla’: 376, ’hjälper’: 376, ’gäster’: 375, ’spännande’: 375, ’dig.’: 374, ’work’: 373, ’söka’: 371, ’brinner’: 370, ’heltid’: 369, ’egenskaper’: 365, ’skrift.’: 364, ’kräver’: 363, ’kunskap’: 361, ’hand’: 360, ’personligt’: 358, ’ny’: 358, ’leda’: 357, ’lägger’: 355, ’ansök’: 352, ’deras’: 345, ’sätt’: 344, ’anställning’: 343, ’dagliga’: 343, ’2’: 341, ’tjänster’: 341, ’viktig’: 341, ’urval’: 338, ’with’: 333, ’utifrån’: 329, ’van’: 329, ’gå’: 329, ’intresserad’: 329, ’restaurang’: 328, ’arbetat’: 327, ’liknande’: 327, ’koncept’: 326, ’väldigt’: 325, ’arbete.’: 324, ’.’: 324, ’städning’: 324, ’än’: 323, ’rekrytering’: 322, ’kundens’: 321, ’tillträde’: 321, ’följa’: 320, ’trygg’: 318, ’kanske’: 318, ’ansvarar’: 318, ’löpande.’: 316, ’eftersom’: 316, ’utöver’: 315, ’arbetstid’: 313, ’1’: 313, ’kompetens’: 312, ’annan’: 312, ’första’: 311, ’kontor’: 309, ’omgående’: 309, ’utföra’: 308, ’vilja’: 307, ’passar’: 306, ’tillsättas’: 306, ’anställda’: 306, ’tempo’: 306, ’ingen’: 305, ’medarbetare.’: 302, ’snart’: 302, ’engagemang’: 300, ’

295, ’team.’: 294, ’bor’: 294, ’utveckling’: 293, ’varmt’: 292, ’b’: 291, ’flytande’: 291, ’språket’: 289, ’tid’: 288, ’stöd’: 287, ’information’: 287, ’will’: 287, ’arbetstider’: 285, ’6’: 285, ’lyckas’: 285, ’arbetstiderna’: 285, ’miljö’: 284, ... }

A.2.2 Experience

A.2.2.1 Requiring previous experience

’goda’: 1268, ’you’: 1232, ’arbetsuppgifter’: 1230, ’bra’: 1230, ’mycket’: 1227, ’behöver’: 1223, ’ • ’: 1210, ’del’: 1195, ’team’: 1166, ’nya’: 1150, ’finns’: 1142, ’andra’: 1132, ’mer’: 1103, ’’: 1074, ’with’: 1057, ’we’: 1050, ’också’: 1020, ’sker’: 1001, ’medarbetare’: 998, ’viktigt’: 997, ’välkommen’: 975, ’rollen’: 974, ’genom’: 958, ’möjlighet’: 949, ’innebär’: 949, ’enligt’: 947, ’tidigare’: 946, ’detta’: 934, ’service’: 920, ’skapa’: 917, ’for’: 916, ’företag’: 889, ’löpande’: 886, ’utbildning’: 884, ’få’: 876, ’tal’: 846, ’hela’: 840, ’över’: 813, ’därför’: 811, ’krav’: 807, ’barn’: 804, ’via’: 801, ’utveckla’: 800, ’meriterande’: 789, ’are’: 788, ’hög’: 786, ’stort’: 785, ’kunskaper’: 782, ’ansvar’: 782, ’elever’: 775, ’our’: 770, ’varje’: 767, ’personlig’: 746, ’upp’: 745, ’alltid’: 741, ’–’: 740, ’allt’: 736, ’sig’: 735, ’ingår’: 732, ’ab’: 731, ’minst’: 729, ’gillar’: 727, ’förmåga’: 722, ’as’: 712, ’jobbar’: 709, ’positiv’: 707, ’rätt’: 703, ’ca’: 701, ’intresse’: 694, ’här’: 690, ’bli’: 686, ’års’: 685, ’tar’: 681, ’sedan’: 673, ’nära’: 671, ’vikt’: 668, ’när’: 667, ’utvecklas’: 662, ’ge’: 661, ’is’: 660, ’göra’: 657, ’många’: 652, ’mot’: 649, ’bästa’: 637, ’uppdrag’: 633, ’år’: 633, ’gör’: 632, ’man’: 629, ’tycker’: 623, ’idag’: 620, ’roll’: 618, ’personal’: 618, ’övriga’: 615, ’trivs’: 613, ’högt’: 604, ’vad’: 600, ’lätt’: 600, ’skicka’: 597, ’självständigt’: 591, ’kollegor’: 585, ’hjälpa’: 584, ’ligger’: 568, ’kunskap’: 564, ’blir’: 555, ’emot’: 553, ’behov’: 549, ’personliga’: 544, ’verksamhet’: 543, ’krav.’: 542, ’kontakt’: 539, ’behov.’: 538, ’egna’: 528, ’will’: 526, ’mellan’: 525, ’försäljning’: 522, ’består’: 522, ’meriterande.’: 516, ’se’: 516, ’två’: 515, ’denna’: 515, ’flera’: 511, ’fokus’: 511, ’experience’: 506, ’stora’: 504, ’vilket’: 501, ’körkort’: 499, ’be’: 493, ’engelska’: 490, ’utveckling’: 489, ’just’: 486, ’kompetens’: 485, ’eget’: 484, ’flexibel’: 482, ’komma’: 476, ’människor’: 474, ’fram’: 470, ’ansökan!’: 469, ’innan’: 465, ’annat’: 461, ’utifrån’: 460, ’skolan’: 460, ’sin’: 460, ’verksamheten’: 459, ’ansvarar’: 458, ’social’: 458, ’tror’: 457, ’öppen’: 457, ’skrift’: 455, ’såväl’: 447, ’bland’: 447, ’kunder.’: 447, ’sista’: 445, ’erbjuda’: 445, ’redan’: 444, ’mål’: 442, ’allakando’: 439, ’jobb’: 438, ’skall’: 437, ’intervjuer’: 435, ’all’: 434, ’lära’: 428, ’arbetsplatsen’: 425, ’egen’: 425, ’information’: 424, ’hantera’: 423, ’viktig’: 421, ’möjligheter’: 420, ’jobbet’: 420, ’anpassa’: 420, ’tillträde’: 418, ’hur’: 418, ’samarbete’: 417, ’lärare’: 415, ’stockholm’: 415, ’spännande’: 413, ’företaget’: 412, ’sätt’: 411, ’några’: 410, ’arbetat’: 408, ’känner’: 408, ’samtidigt’: 406, ’sverige’: 404, ’your’: 403, ’noggrann’: 400, ’vidare’: 400, ’hjälper’: 398, ’lägger’: 398, ’kunden’: 397, ’ut’: 397, ’lön’: 394, ’oss.’: 394, ’have’: 394, ’bör’: 392, ’b-körkort’: 392, ’work’: 391, ’väl’: 388, ’läxhjälp’: 388, ’that’: 388, ’trygg’: 385, ’#jobbjustnu’: 384, ’tjänst’: 383, ’start’: 380, ’an’: 378, ’cv’: 375, ’brinner’: 375, ’miljö’: 374, ’fokuserar’: 372, ’personligt’: 371, ’per’: 370, ’varierande’: 368, ’kvalifikationer’: 368, ’sina’: 367, ’sveriges’: 366, ’öppna’: 366, ’engagemang’: 365, ’något’: 364, ’måste’: 364, ’mat’: 363, ’urval’: 362, ’arbetsuppgifterna’: 361, ’6’: 358, ’går’: 354, ’leda’: 353, ’on’: 353, ’egenskaper’: 352, ’förutsättningar.’: 351, ’1’: 350, ’ger’: 349, ’arbete.’: 348, ’heltid’: 346, ’intresserad’: 346, ’främst’: 345, ’deras’: 343, ’van’: 340, ’kompetens,’: 340, ’passar’: 339, ’skrift.’: 339, ’delar’: 338, ’personer’: 338, ’inför’: 335, ’överenskommelse.’: 334, ’stöd’: 333, ’nå’: 330, ’dag’: 329, ’hand’: 329, ’arbetsplats’: 327, ’läs’: 327, ’hålla’: 325, ’kund’: 325, ’fler’: 325, ’runt’: 321, ’dagliga’: 320, ’ny’: 319, ’dig.’: 317, ’extra’: 316, ’2’: 316, ’än’: 315, ’själv’: 315, ’annan’: 314, ’liknande’: 314, ’största’: 314, ’krävs’: 309, ’at’: 309, ’skola’: 307, ’fast’: 307, ’anställning’: 307, ’anställda’: 307, ’någon’: 305, ’löpande.’: 302, ’bidra’: 302, ’älskar’: 301, ’tjänster’: 301, ’samarbeta’: 301, ’produkter’: 300, ’tillsättas’: 300, ’frågor’: 299, ’or’: 298, ’utan’: 297, ’varmt’: 296, ’engagerad’: 296, ’*’: 296, ’ute’: 296, ’aktivt’: 294, ’ansvara’: 293, ’kvalitet’: 292, ’snart’: 288, ’utöver’: 287, ’första’: 284, ’gott’: 283, ’ansökningar’: 279, ’business’: 278, ’inget’: 278, ’drivs’: 278, ’driva’: 277, ’självgående’: 277, ’rekrytering’: 277, ’vilja’: 277, ’leverera’: 276, ’söka’: 272, ’ansök’: 272

A.2.2.2 Not requiring previous experience

{’finns’: 897, ’man’: 889, ’mycket’: 886, ’möjlighet’: 882, ’sker’: 874, ’tillsammans’: 861, ’stor’: 814, ’detta’: 812, ’jobbet’: 807, ’nya’: 800, ’del’: 793, ’få’: 793, ’erbjuder’: 765, ’a’: 759, ’företag’: 756, ’andra’: 747, ’sedan’: 729, ’barn’: 724, ’varje’: 715, ’viktigt’: 706, ’goda’: 692, ’arbetsuppgifter’: 689, ’ca’: 677, ’the’: 671, ’enligt’: 667, ’jag’: 667, ’utbildning’: 664, ’via’: 659, ’allt’: 657, ’när’: 655, ’tidigare’: 649, ’hela’: 640, ’över’: 636, ’krav’: 628, ’välkommen’: 626, ’sig’: 617, ’medarbetare’: 613, ’idag’: 612, ’jobbar’: 603, ’alltid’: 603, ’också’: 601, ’hjälpa’: 601, ’löpande’: 600, ’to’: 599, ’tal’: 596, ’göra’: 594, ’ge’: 592, ’genom’: 591, ’redan’: 589, ’lära’: 580, ’per’: 577, ’körkort’: 571, ’service’: 569, ’ab’: 568, ’många’: 568, ’tycker’: 565, ’mellan’: 558, ’*’: 557, ’bli’: 548, ’ · ’: 543, ’bästa’: 533, ’tar’: 530, ’därför’: 527, ’utvecklas’: 526, ’team’: 523, ’positiv’: 521, ’uppdrag’: 517, ’assistent’: 516, ’we’: 509, ’rollen’: 508, ’rätt’: 506, ’gör’: 505, ’#jobbjustnu’: 499, ’människor’: 495, ’gillar’: 489, ’år’: 489, ’går’: 482, ’egen’: 482, ’of’: 481, ’egna’: 477, ’emot’: 475, ’elever’: 475, ’–’: 475, ’passar’: 473, ’försäljning’: 471, ’några’: 467, ’nära’: 464, ’personliga’: 462, ’fler’: 460, ’måste’: 458, ’ansök’: 454, ’you’: 454, ’innan’: 451, ’läs’: 447, ’ • ’: 447, ’meriterande’: 443, ’kollegor’: 443, ’just’: 439, ’skapa’: 438, ’upp’: 436, ’stort’: 435, ’sin’: 430, ’blir’: 429, ’jobb’: 423, ’ligger’: 420, ’ingår’: 416, ’sveriges’: 416, ’vad’: 413, ’trivs’: 411, ’for’: 411, ’be’: 410, ’mot’: 407, ’krav.’: 406,

’hög’: 404, ’stockholm’: 403, ’flexibel’: 397, ’förmåga’: 394, ’behov.’: 391, ’hur’: 391, ’skicka’: 389, ’is’: 388, ’helt’: 382, ’bland’: 380, ’vilket’: 380, ’högt’: 380, ’personal’: 379, ’personer’: 379, ’hjälp’: 377, ’annat’: 375, ’intresse’: 373, ’hjälper’: 373, ’övriga’: 372, ’består’: 371, ’tillgång’: 371, ’ut’: 371, ’dig.’: 370, ’kunden’: 369, ’verksamhet’: 367, ’timmar’: 367, ’inget’: 365, ’social’: 363, ’här’: 362, ’lätt’: 358, ’utan’: 356, ’ansvar’: 353, ’are’: 348, ’kontakt’: 347, ’vikt’: 347, ’arbetsplatsen’: 347, ’företaget’: 345, ’något’: 342, ’se’: 341, ’öppen’: 340, ’bör’: 339, ’start’: 338, ’själv’: 335, ’möjligheter’: 333, ’ansökan!’: 331, ’inför’: 331, ’bil’: 331, ’lön’: 331, ’skrift’: 330, ’självständigt’: 327, ’fram’: 324, ’erbjuda’: 324, ’work’: 320, ’behov’: 317, ’eget’: 316, ’sverige’: 313, ’minst’: 313, ’fokus’: 311, ’säljare’: 310, ’två’: 309, ’anpassa’: 306, ’assistans’: 304, ’intervjuer’: 304, ’skall’: 304, ’söka’: 303, ’bättre’: 303, ’dag’: 299, ’mål’: 299, ’denna’: 298, ’utveckla’: 295, ’krävs’: 293, ’tjänst’: 293, ’kunder.’: 292, ’veckan’: 292, ’flera’: 292, ’annan’: 287, ’1’: 287, ’kunskaper’: 285, ’bor’: 285, ’sina’: 284, ’with’: 284, ’oss.’: 283, ’varierande’: 282, ’kanske’: 278, ’vidare’: 277, ’noggrann’: 276, ’fokuserar’: 276, ’någon’: 275, ’komma’: 274, ’samtidigt’: 273, ’nå’: 273, ’utifrån’: 272, ’fast’: 272, ’stora’: 271, ’ger’: 271, ’arbetsuppgifterna’: 270, ’engagemang’: 270, ’än’: 268, ’viktig’: 267, ’växande’: 267, ’hemsida’: 266, ’arbetsplats’: 266, ’förutsättningar.’: 265, ’deras’: 264, ’kompetens,’: 263, ’sätt’: 262, ’meriterande.’: 261, ’öppna’: 259, ’utbildningen’: 258, ’hemma’: 258, ’runt’: 258, ’

256, ’år.’: 255, ’at’: 254, ’our’: 252, ’såväl’: 248, ’trygg’: 246, ’ansökningar’: 245, ’endast’: 244, ’ute’: 244, ’största’: 244, ’will’: 244, ’ingen’: 242, ’brinner’: 241, ’känner’: 241, ’koncept’: 241, ’b-körkort’: 240, ’engelska’: 240, ’älskar’: 240, ’oavsett’: 240, ’kräver’: 237, ’roll’: 237, ’idag!’: 236, ’kund’: 232, ’skrift.’: 230, ’arbetstid’: 229, ’sök’: 229, ’extra’: 228, ’.’: 228, ’produkter’: 228, ’fördel’: 228, ’hantera’: 228, ’lägger’: 227, ’intresserad’: 227, ’observera’: 225, ’elev’: 224, ’arbetat’: 221, ’hand’: 221, ’följa’: 220, ’stöd’: 220, ’hålla’: 219, ’ibland’: 217, ’anställning’: 216, ’väldigt’: 215, ’sista’: 215, ’spännande’: 214, ’2’: 213, ’förutsättningar’: 213, ’tror’: 212, ’cv’: 212, ’privatlärare’: 211, ’ni’: 211, ’första’: 210, ’coachning’: 210, ’lite’: 209, ’leda’: 209, ’rekrytering’: 209, ’mat’: 209, ’främst’: 206, ’eftersom’: 206, ’urval’: 205, ’kontor’: 205, ’egenskaper’: 205, ’gå’: 204, ’års’: 202, ’18’: 202, ’minuter’: 201, ’skillnad’: 200, ’leverera’: 200, ’dagar’: 199, ’tjänster’: 199, ’hitta’: 199, ’genomför’: 199, ’roligt’: 198, ’som:’: 198, ’hemsida.’: 197, ’flytande’: 197, ... }

A.2.3 Driving license

A.2.3.1 Requiring driving license

{’stor’: 911, ’ · ’: 890, ’nya’: 879, ’mycket’: 879, ’olika’: 874, ’man’: 874, ’krav’: 845, ’medarbetare’: 845, ’andra’: 839, ’service’: 838, ’företag’: 817, ’innebär’: 805, ’ab’: 800, ’tidigare’: 772, ’sker’: 758, ’erbjuder’: 743, ’tal’: 732, ’hela’: 718, ’rollen’: 717, ’också’: 709, ’’: 705, ’tillsammans’: 701, ’goda’: 692, ’jobbar’: 689, ’ • ’: 688, ’välkommen’: 676, ’över’: 671, ’jag’: 661, ’rätt’: 652, ’enligt’: 652, ’meriterande’: 638, ’a’: 627, ’därför’: 623, ’möjlighet’: 618, ’löpande’: 608, ’gillar’: 605, ’alltid’: 601, ’team’: 601, ’krav.’: 594, ’när’: 592, ’allt’: 588, ’mer’: 584, ’ingår’: 580, ’sig’: 580, ’få’: 562, ’tycker’: 562, ’b-körkort’: 558, ’personliga’: 556, ’egen’: 556, ’hög’: 548, ’måste’: 548, ’bli’: 545, ’the’: 543, ’stort’: 534, ’sedan’: 532, ’ca’: 525,

’kunden’: 524, ’assistent’: 524, ’genom’: 520, ’positiv’: 513, ’försäljning’: 509, ’upp’: 503, ’självständigt’: 501, ’ge’: 501, ’ansvar’: 493, ’högt’: 490, ’utbildning’: 487, ’personal’: 486, ’#jobbjustnu’: 483, ’gör’: 479, ’noggrann’: 478, ’via’: 478, ’många’: 477, ’bil’: 473, ’bästa’: 471, ’flexibel’: 468, ’göra’: 467, ’trivs’: 466, ’mellan’: 466, ’övriga’: 463, ’kunder.’: 460, ’skicka’: 458, ’mot’: 458, ’förmåga’: 452, ’behov.’: 449, ’jobbet’: 447, ’idag’: 446, ’*’: 445, ’lätt’: 445, ’hjälpa’: 443, ’annat’: 441, ’vad’: 439, ’egna’: 433, ’of’: 427, ’vikt’: 424, ’eget’: 423, ’företaget’: 422, ’meriterande.’: 421, ’denna’: 416, ’to’: 413, ’skapa’: 412, ’vilket’: 412, ’år’: 412, ’varje’: 409, ’skall’: 408, ’arbetsplatsen’: 405, ’intresse’: 403, ’sverige’: 400, ’bör’: 398, ’we’: 398, ’blir’: 397, ’varierande’: 396, ’jobb’: 393, ’här’: 392, ’kunskaper’: 389, ’ut’: 389, ’lära’: 386, ’–’: 386, ’social’: 385, ’människor’: 384, ’roll’: 383, ’öppen’: 383, ’tar’: 382, ’utvecklas’: 382, ’skrift’: 382, ’arbetsuppgifterna’: 373, ’anpassa’: 370, ’kund’: 363, ’sin’: 350, ’minst’: 346, ’hantera’: 345, ’just’: 343, ’hjälp’: 341, ’kollegor’: 340, ’behov’: 338, ’emot’: 337, ’ute’: 337, ’personer’: 337, ’leverera’: 336, ’bland’: 334, ’dag’: 334, ’krävs’: 330, ’you’: 329, ’runt’: 328, ’assistans’: 327, ’redan’: 325, ’hur’: 324, ’for’: 322, ’tillgång’: 321, ’fokuserar’: 321, ’heltid’: 320, ’is’: 320, ’består’: 319, ’engelska’: 319, ’stora’: 318, ’kompetens,’: 318, ’se’: 317, ’kräver’: 313, ’ansökan!’: 311, ’lön’: 310, ’öppna’: 310, ’verksamhet’: 308, ’komma’: 307, ’innan’: 307, ’förutsättningar.’: 305, ’stockholm’: 305, ’b’: 304, ’två’: 303, ’något’: 300, ’köra’: 300, ’främst’: 300, ’utan’: 299, ’flera’: 297, ’tjänst’: 296, ’ligger’: 296, ’egenskaper’: 296, ’sista’: 294, ’uppdrag’: 294, ’urval’: 291, ’cv’: 290, ’inget’: 289, ’kontakt’: 288, ’skrift.’: 287, ’års’: 285, ’fast’: 285, ’produkter’: 285, ’intervjuer’: 284, ’work’: 284, ’arbete.’: 282, ’utföra’: 281, ’are’: 280, ’van’: 277, ’eftersom’: 277, ’såväl’: 276, ’någon’: 275, ’tror’: 274, ’oss.’: 274, ’fokus’: 273, ’kunskap’: 273, ’säljare’: 272, ’fram’: 268, ’mål’: 268, ’känner’: 267, ’.’: 267, ’städning’: 267, ’nära’: 266, ’viktig’: 265, ’liknande’: 264, ’personligt’: 263, ’största’: 262, ’hålla’: 261, ’hand’: 259, ’dig.’: 259, ’själv’: 259, ’samtidigt’: 257, ’kompetens’: 257, ’anställda’: 257, ’ger’: 256, ’går’: 256, ’tjänster’: 255, ’några’: 254, ’hjälper’: 253, ’lägger’: 251, ’extra’: 250, ’anställning’: 249, ’erbjuda’: 248, ’älskar’: 247, ’utveckla’: 247, ’möjligheter’: 246, ’tillträde’: 246, ’ny’: 245, ’dagliga’: 242, ’2’: 242, ’gå’: 241, ’än’: 240, ’vidare’: 238, ’medarbetare.’: 237, ’kundens’: 236, ’löpande.’: 236, ’bor’: 233, ’tillsättas’: 231, ’sätt’: 231, ’utför’: 229, ’spännande’: 229, ’väldigt’: 228, ’arbetat’: 227, ’språket’: 227, ’omgående’: 226, ’ansvarar’: 225, ’per’: 225, ’väl’: 224, ’fysiskt’: 223, ’arbetsplats’: 221, ’at’: 221, ’kvalitet’: 220, ’uppgifter’: 219, ’hem’: 218, ’stockholm.’: 217, ’tunga’: 217, ’deras’: 217, ’tid’: 216, ’form’: 215, ’vilja’: 214, ’serviceinriktad’: 214, ’varmt’: 213, ’butiker’: 213, ’innefattar’: 212, ’lyckas’: 212, ’6’: 209, ’kanske’: 209, ’with’: 209, ’kunderna’: 209, ’be’: 207, ’självgående’: 207, ’bilar’: 205, ’kollektivavtal’: 205, ’intresserad’: 204, ’information’: 203, ’utgår’: 202, ’viss’: 202, ’kvalifikationer’: 202, ’stark’: 199, ’företagets’: 199, ’samarbete’: 199, ’kontor’: 198, ’såsom’: 198, ’ansökningar’: 197, ’will’: 195, ’snabbt’: 194, ’ansökan.’: 194, ’första’: 194, ’start’: 193, ’nästa’: 193, ’tempo’: 193, ’sveriges’: 193, ’sätt.’: 192, ’ledande’: 192, ’överenskommelse.’: 192, ’ingen’: 192, ’miljö’: 192, ... }

A.2.3.2 Not requiring driving license

{’svenska’: 1288, ’goda’: 1268, ’mycket’: 1234, ’möjlighet’: 1213, ’elever’: 1209, ’stor’: 1179, ’arbetet’: 1170, ’we’: 1161, ’bra’: 1147, ’with’: 1132, ’sker’: 1117,

’person’: 1108, ’få’: 1107, ’ · ’: 1096, ’finns’: 1094, ’team’: 1088, ’kunna’: 1087, ’innebär’: 1081, ’varje’: 1073, ’nya’: 1071, ’utbildning’: 1061, ’del’: 1051, ’andra’: 1040, ’genom’: 1029, ’for’: 1005, ’arbetsuppgifter’: 1003, ’via’: 982, ’ • ’: 969, ’kunder’: 966, ’enligt’: 962, ’skapa’: 943, ’välkommen’: 925, ’också’: 912, ’löpande’: 878, ’sedan’: 870, ’nära’: 869, ’uppdrag’: 856, ’are’: 856, ’ca’: 853, ’our’: 849, ’utveckla’: 848, ’tar’: 829, ’–’: 829, ’företag’: 828, ’tidigare’: 823, ’detta’: 823, ’utvecklas’: 806, ’allt’: 805, ’idag’: 786, ’göra’: 784, ’jobbet’: 780, ’över’: 778, ’sig’: 772, ’medarbetare’: 766, ’rollen’: 765, ’hela’: 762, ’as’: 759, ’ge’: 752, ’viktigt’: 752, ’alltid’: 743, ’många’: 743, ’hjälpa’: 742, ’personlig’: 733, ’när’: 730, ’is’: 728, ’per’: 722, ’därför’: 715, ’positiv’: 715, ’tal’: 710, ’år’: 710, ’redan’: 708, ’bästa’: 699, ’be’: 696, ’minst’: 696, ’ligger’: 692, ’emot’: 691, ’bli’: 689, ’kollegor’: 688, ’stort’: 686, ’kunskaper’: 678, ’upp’: 678, ’förmåga’: 664, ’intresse’: 664, ’här’: 660, ’gör’: 658, ’service’: 651, ’läs’: 645, ’man’: 644, ’hög’: 642, ’ansvar’: 642, ’passar’: 626, ’tycker’: 626, ’några’: 623, ’jobbar’: 623, ’lära’: 622, ’mellan’: 617, ’gillar’: 611, ’fler’: 610, ’innan’: 609, ’verksamhet’: 602, ’års’: 602, ’mot’: 598, ’kontakt’: 598, ’skolan’: 598, ’meriterande’: 594, ’vikt’: 591, ’krav’: 590, ’sveriges’: 589, ’blir’: 587, ’ansök’: 587, ’människor’: 585, ’just’: 582, ’går’: 580, ’will’: 575, ’vad’: 574, ’består’: 574, ’egna’: 572, ’ingår’: 568, ’lärare’: 560, ’trivs’: 558, ’rätt’: 557, ’fokus’: 549, ’utifrån’: 547, ’sin’: 540, ’se’: 540, ’skicka’: 528, ’behov’: 528, ’utveckling’: 527, ’fram’: 526, ’start’: 525, ’övriga’: 524, ’två’: 521, ’erbjuda’: 521, ’hjälper’: 518, ’stockholm’: 513, ’lätt’: 513, ’personal’: 511, ’möjligheter’: 507, ’flera’: 506, ’experience’: 503, ’sina’: 500, ’ab’: 499, ’inför’: 498, ’högt’: 494, ’bland’: 493, ’ansökan!’: 489, ’hur’: 485, ’försäljning’: 484, ’leda’: 482, ’behov.’: 480, ’nå’: 478, ’mål’: 473, ’roll’: 472, ’kunskap’: 472, ’verksamheten’: 471, ’vilket’: 469, ’jobb’: 468, ’stora’: 457, ’intervjuer’: 455, ’trygg’: 454, ’personliga’: 450, ’engagemang’: 448, ’1’: 447, ’komma’: 443, ’sätt’: 442, ’vidare’: 439, ’your’: 438, ’social’: 436, ’helt’: 434, ’timmar’: 433, ’annan’: 430, ’have’: 430, ’skola’: 429, ’dig.’: 428, ’work’: 427, ’söka’: 426, ’kompetens’: 424, ’brinner’: 424, ’viktig’: 423, ’samtidigt’: 422, ’såväl’: 419, ’självständigt’: 417, ’lön’: 415, ’that’: 415, ’öppen’: 414, ’all’: 412, ’samarbete’: 412, ’engelska’: 411, ’flexibel’: 411, ’*’: 408, ’undervisning’: 407, ’något’: 406, ’oss.’: 403, ’skrift’: 403, ’arbetat’: 402, ’veckan’: 402, ’#jobbjustnu’: 400, ’spännande’: 398, ’denna’: 397, ’annat’: 395, ’tror’: 395, ’an’: 394, ’själv’: 391, ’deras’: 390, ’bättre’: 386, ’år.’: 383, ’känner’: 382, ’personer’: 380, ’tjänst’: 380, ’mat’: 380, ’ut’: 379, ’on’: 378, ’information’: 377, ’eget’: 377, ’lägger’: 374, ’arbetsplats’: 372, ’intresserad’: 369, ’miljö’: 368, ’ansvarar’: 368, ’arbetsplatsen’: 367, ’sista’: 366, ’pedagogiska’: 365, ’ger’: 364, ’stöd’: 362, ’elever.’: 362, ’eleverna’: 356, ’meriterande.’: 356, ’anpassa’: 356, ’inget’: 354, ’krav.’: 354, ’utan’: 354, ’tillträde’: 352, ’egen’: 351, ’än’: 343, ’väl’: 342, ’at’: 342, ’skapar’: 339, ’växande’: 336, ’företaget’: 335, ’or’: 334, ’skall’: 333, ’bör’: 333, ’6’: 332, ’förutsättningar’: 328, ’utbildningen’: 328, ’ansökningar’: 327, ’fokuserar’: 327, ’undervisa’: 325, ’hemsida’: 325, ’elev’: 324, ’rekrytering’: 321, ’skolans’: 320, ’sverige’: 317, ’relevant’: 315, ’öppna’: 315, ’gäster’: 311, ’förutsättningar.’: 311, ’ur’: 310, ’kanske’: 309, ’kvalifikationer’: 308, ’privatlärare’: 308, ’ledare’: 308, ’hjälp’: 307, ’

306, ’digitala’: 306, ’hantera’: 306, ’någon’: 305, ’skillnad’: 304, ’coachning’: 304, ’som:’: 303, ’oavsett’: 302, ’delar’: 302, ’deltid’: 302, ’överenskommelse.’: 301, 'första': 300, 'endast': 300, 'samarbeta': 300, 'hemma': 298, 'följa': 297, 'cv': 297, 'barnen': 297, 'from': 296, 'snart': 296, 'största': 296, 'dag': 294, 'extra': 294, 'utöver': 294, 'koncept': 294, 'fast': 294, 'älskar': 294, 'personligt': 293, 'minuter': 292, 'hand': 291, '5': 289, 'bidra': 288, 'engagerad': 288, '2': 287, 'aktivt': 287, 'frågor': 285, 'kompetens,': 285, ... }

A.2.4 Work type

A.2.4.1 Full-time

{'team': 1039, 'mycket': 1034, 'del': 1031, 'erbjuder': 983, 'arbetsuppgifter': 979, 'finns': 977, 'we': 976, 'olika': 975, 'detta': 969, 'stor': 964, 'medarbetare': 949, 'företag': 926, 'tillsammans': 922, 'rollen': 915, 'with': 912, 'goda': 907, 'andra': 903, 'viktigt': 893, 'service': 888, '': 871, '•': 863, 'krav': 851, 'tidigare': 851, 'for': 834, 'innebär': 828, 'tal': 828, 'man': 807, 'ab': 792, 'hela': 790, 'sker': 787, 'över': 753, 'körkort': 738, 'mer': 734, 'jobbar': 729, 'också': 722, 'välkommen': 720, 'meriterande': 709, 'möjlighet': 706, 'rätt': 705, 'our': 704, 'få': 701, 'are': 697, 'därför': 689, 'bli': 688, 'genom': 679, 'alltid': 677, 'löpande': 667, 'försäljning': 667, 'enligt': 652, 'hög': 649, 'utbildning': 641, 'gillar': 639, 'personlig': 636, 'via': 629, 'as': 613, 'när': 609, 'upp': 606, 'allt': 605, 'stort': 604, 'sig': 604, '–': 596, 'ansvar': 590, 'högt': 586, 'utvecklas': 579, 'is': 579, 'krav.': 559, 'bästa': 559, 'mot': 555, 'positiv': 550, 'skapa': 543, 'kunskaper': 539, 'b-körkort': 539, 'göra': 535, 'skicka': 535, 'ingår': 531, 'sedan': 531, 'trivs': 528, 'tycker': 526, 'vad': 523, 'ge': 521, 'jobbet': 519, 'roll': 518, 'företaget': 512, 'många': 511, 'intresse': 504, 'gör': 500, 'idag': 497, 'will': 495, 'personliga': 494, 'övriga': 494, 'kunder.': 487, 'självständigt': 483, 'här': 482, 'tar': 482, 'kunden': 481, 'denna': 473, 'social': 472, 'eget': 471, 'personal': 471, 'blir': 469, 'varje': 468, 'kollegor': 467, 'lätt': 465, 'egen': 459, 'vikt': 458, 'behov.': 456, 'stora': 454, 'jobb': 449, 'noggrann': 447, 'vilket': 447, 'förmåga': 444, 'lära': 443, 'ca': 442, 'hjälpa': 441, 'egna': 440, 'sverige': 436, 'minst': 434, 'år': 432, 'emot': 430, 'skrift': 430, 'experience': 430, 'meriterande.': 428, '#jobbjustnu': 428, 'redan': 423, 'människor': 417, 'säljare': 416, 'arbetsplatsen': 415, 'måste': 414, 'just': 414, 'års': 410, 'be': 408, 'annat': 407, 'mellan': 403, 'heltid': 399, 'öppen': 399, 'utveckla': 398, 'ut': 394, 'hur': 389, 'anpassa': 389, 'flexibel': 386, 'produkter': 384, 'lön': 383, 'uppdrag': 382, 'hantera': 379, 'engelska': 379, 'består': 372, 'varierande': 369, 'ansökan!': 367, 'an': 367, 'that': 367, 'kund': 365, 'flera': 363, 'work': 363, 'have': 362, 'mål': 360, 'all': 359, 'your': 359, 'komma': 358, 'behov': 357, 'kunskap': 357, 'dag': 357, 'skall': 356, 'utan': 356, 'bör': 355, 'spännande': 354, 'kontakt': 353, 'tror': 353, 'fokus': 353, 'fokuserar': 353, 'något': 350, '*': 349, 'såväl': 348, 'fram': 346, 'sin': 346, 'känner': 345, 'stockholm': 345, 'bland': 341, 'krävs': 341, 'sista': 340, 'egenskaper': 338, 'runt': 338, 'se': 337, 'fast': 337, 'nära': 335, 'cv': 335, 'erbjuda': 334, 'kompetens,': 334, 'öppna': 332, 'möjligheter': 331, 'innan': 331, 'arbetsuppgifterna': 330, 'samtidigt': 329, 'intervjuer': 329, 'skrift.': 328, 'information': 327, 'ansvarar': 326, 'at': 326, 'kompetens': 325, 'bil': 325, 'ute': 323, 'verksamhet': 319, 'största': 319, 'förutsättningar.': 319, 'inget': 317, 'två': 317, 'urval': 315, 'viktig': 312, 'främst': 312, 'leverera': 310, 'ligger': 307, 'anställda': 307, 'tjänster': 306, 'van': 305, 'tjänst': 304, 'personer': 303, 'vidare': 302, 'oss.': 301, 'arbete.': 301, 'utveckling': 299, 'on': 299, 'arbetsplats':

298, ’hjälper’: 294, ’några’: 292, ’personligt’: 292, ’än’: 292, ’arbetat’: 291, ’kontor’: 290, ’hand’: 290, ’liknande’: 289, ’sveriges’: 288, ’lägger’: 287, ’älskar’: 285, ’nästa’: 285, ’jag’: 284, ’lyckas’: 283, ’dig.’: 283, ’ger’: 283, ’ny’: 282, ’hjälp’: 279, ’b’: 277, ’sätt’: 272, ’or’: 271, ’själv’: 270, ’någon’: 269, ’går’: 269, ’väl’: 265, ’2’: 264, ’utföra’: 263, ’samarbete’: 262, ’kräver’: 261, ’miljö’: 261, ’idag!’: 257, ’medarbetare.’: 257, ’driven’: 256, ’vilja’: 255, ’löpande.’: 254, ’deras’: 254, ’köra’: 254, ’flytande’: 252, ’gå’: 251, ’.’: 251, ’eftersom’: 250, ’kundens’: 249, ’brinner’: 249, ’väldigt’: 248, ’kvalitet’: 246, ’dagliga’: 244, ’form’: 244, ’första’: 243, ’this’: 242, ’ledande’: 239, ’extra’: 239, ’business’: 239, ’6’: 239, ’hålla’: 238, ’team.’: 238, ’engagemang’: 238, ’tillsättas’: 237, ’tid’: 237, ’start’: 234, ’kanske’: 232, ’tillträde’: 231, ’assistent’: 228, ’företagets’: 227, ’utifrån’: 226, ’sätt.’: 226, ’vem’: 226, ’varmt’: 225, ’from’: 225, ’anställning’: 225, ’självgående’: 224, ’delar’: 224, ’tillgång’: 224, ’snabbt’: 222, ’utgår’: 222, ’frågor’: 221, ’intresserad’: 221, ’kvalifikationer’: 221, ’working’: 221, ’såsom’: 218, ... }

A.2.4.2 Part-time

{’sker’: 1088, ’kunna’: 1079, ’mycket’: 1079, ’finns’: 1062, ’innebär’: 1058, ’goda’: 1053, ’bra’: 1051, ’personlig’: 1045, ’person’: 1030, ’varje’: 1014, ’a’: 985, ’andra’: 976, ’få’: 968, ’the’: 964, ’enligt’: 962, ’del’: 957, ’arbetsuppgifter’: 940, ’ca’: 936, ’utbildning’: 907, ’också’: 899, ’välkommen’: 881, ’ · ’: 875, ’sedan’: 871, ’genom’: 870, ’via’: 831, ’kunder’: 829, ’to’: 828, ’löpande’: 819, ’skapa’: 812, ’viktigt’: 810, ’nya’: 807, ’nära’: 800, ’ • ’: 794, ’allt’: 788, ’detta’: 777, ’uppdrag’: 768, ’per’: 758, ’sig’: 748, ’tidigare’: 744, ’hjälpa’: 744, ’idag’: 735, ’ge’: 732, ’tar’: 729, ’företag’: 719, ’göra’: 716, ’när’: 713, ’man’: 711, ’många’: 709, ’jobbet’: 708, ’utveckla’: 697, ’över’: 696, ’hela’: 690, ’år’: 690, ’ligger’: 681, ’mellan’: 680, ’positiv’: 678, ’förmåga’: 672, ’alltid’: 667, ’of’: 666, ’medarbetare’: 662, ’tycker’: 662, ’team’: 650, ’därför’: 649, ’gör’: 637, ’passar’: 620, ’läs’: 619, ’–’: 619, ’ingår’: 617, ’stort’: 616, ’tal’: 614, ’bästa’: 611, ’redan’: 610, ’utvecklas’: 609, ’minst’: 608, ’service’: 601, ’emot’: 598, ’fler’: 597, ’skolan’: 593, ’verksamhet’: 591, ’innan’: 585, ’några’: 585, ’krav’: 584, ’jobbar’: 583, ’we’: 583, ’gillar’: 577, ’jag’: 576, ’upp’: 575, ’här’: 570, ’ansök’: 568, ’rollen’: 567, ’går’: 567, ’egna’: 565, ’lära’: 565, ’intresse’: 563, ’you’: 562, ’kollegor’: 561, ’vikt’: 557, ’människor’: 552, ’bli’: 546, ’ansvar’: 545, ’sin’: 544, ’hög’: 541, ’lärare’: 539, ’kontakt’: 533, ’kunskaper’: 528, ’personal’: 526, ’meriterande’: 523, ’består’: 521, ’se’: 520, ’blir’: 515, ’personliga’: 512, ’just’: 511, ’behov’: 509, ’två’: 507, ’ab’: 507, ’utifrån’: 506, ’rätt’: 504, ’*’: 504, ’mot’: 501, ’timmar’: 500, ’trivs’: 496, ’be’: 495, ’sveriges’: 494, ’lätt’: 493, ’flexibel’: 493, ’övriga’: 493, ’for’: 493, ’vad’: 490, ’bland’: 486, ’start’: 484, ’hjälper’: 477, ’års’: 477, ’stockholm’: 473, ’behov.’: 473, ’fokus’: 469, ’is’: 469, ’inför’: 467, ’leda’: 460, ’#jobbjustnu’: 455, ’skicka’: 451, ’fram’: 448, ’egen’: 448, ’trygg’: 444, ’sina’: 441, ’flera’: 440, ’are’: 439, ’annan’: 435, ’självständigt’: 435, ’erbjuda’: 435, ’vilket’: 434, ’ansökan!’: 433, ’assistent’: 432, ’helt’: 431, ’annat’: 429, ’with’: 429, ’verksamheten’: 425, ’1’: 424, ’skola’: 423, ’möjligheter’: 422, ’veckan’: 421, ’hur’: 420, ’personer’: 414, ’jobb’: 412, ’intervjuer’: 410, ’måste’: 408, ’söka’: 406, ’dig.’: 404, ’nå’: 403, ’sätt’: 401, ’undervisning’: 399, ’öppen’: 398, ’högt’: 398, ’engagemang’: 397, ’komma’: 392, ’krav.’: 389, ’kunskap’: 388, ’utveckling’: 386, ’skall’: 385, ’mål’: 381, ’år.’: 381, ’själv’: 380, ’viktig’: 376, ’oss.’: 376, ’bör’: 376, ’tillgång’: 375, ’vidare’: 375, ’ut’: 374, ’tjänst’: 372, ’hjälp’: 369, ’tillträde’: 367, ’brinner’: 367, ’mat’: 363, ’bättre’: 362, ’elever.’: 359, ’arbetsplatsen’: 357, ’kompetens’: 356, ’något’: 356, ’skrift’: 355, ’deras’: 353, ’intresserad’: 352, ’stöd’: 351, ’engelska’: 351, ’samtidigt’: 350, ’samarbete’: 349, ’meriterande.’: 349, ’social’: 349, ’work’: 348, ’såväl’: 347, ’lön’: 342, ’denna’: 340, ’lägger’: 338, ’arbetat’: 338, ’hemsida’: 338, ’ger’: 337, ’roll’: 337, ’anpassa’: 337, ’deltid’: 335, ’pedagogiska’: 335, ’körkort’: 332, ’eget’: 329, ’eleverna’: 326, ’inget’: 326, ’försäljning’: 326, ’elev’: 322, ’stora’: 321, ’sista’: 320, ’ur’: 320, ’undervisa’: 319, ’utbildningen’: 318, ’our’: 318, ’förutsättningar’: 317, ’endast’: 317, ’bor’: 316, ’tror’: 316, ’hemma’: 314, ’gäster’: 313, ’någon’: 311, ’växande’: 311, ’koncept’: 308, ’skapar’: 308, ’privatlärare’: 308, ’ansökningar’: 307, ’

306, ’hålla’: 306, ’skolans’: 305, ’extra’: 305, ’överenskommelse.’: 304, ’känner’: 304, ’rekrytering’: 302, ’6’: 302, ’väl’: 301, ’arbetsuppgifterna’: 301, ’miljö’: 299, ’anställning’: 298, ’utdrag’: 298, ’följa’: 298, ’utan’: 297, ’förutsättningar.’: 297, ’arbetsplats’: 295, ’fokuserar’: 295, ’öppna’: 293, ’än’: 291, ’minuter’: 290, ’skillnad’: 289, ’kvalifikationer’: 289, ’kanske’: 286, ’kunden’: 285, ’arbetstid’: 285, ’uppdraget’: 284, ’observera’: 284, ’samarbeta’: 282, ’ledare’: 282, ’varierande’: 281, ’sverige’: 281, ’assistans’: 275, ’hemsida.’: 275, ’will’: 275, ’oavsett’: 274, ’utöver’: 274, ’spännande’: 273, ’hantera’: 272, ’barnen’: 272, ’aktivt’: 272, ’dag’: 271, ’kompetens,’: 269, ’coachning’: 267, ’ansvarar’: 267, ’engagerad’: 266, ’fördel’: 266, ’dem’: 265, ’2’: 265, ’perfekt’: 264, ’personligt’: 264, ’snart’: 263, ’krävs’: 261, ’gott’: 260, ’hand’: 260, ’5’: 260, ’dagliga’: 256, ’delar’: 256, ... }

A.3 Swedish stop-words

[’och’, ’det’, ’att’, ’i’, ’en’, ’jag’, ’hon’, ’som’, ’han’, ’på’, ’den’, ’med’, ’var’, ’sig’, ’för’, ’så’, ’till’, ’är’, ’men’, ’ett’, ’om’, ’hade’, ’de’, ’av’, ’icke’, ’mig’, ’du’, ’henne’, ’då’, ’sin’, ’nu’, ’har’, ’inte’, ’hans’, ’honom’, ’skulle’, ’hennes’, ’där’, ’min’, ’man’, ’ej’, ’vid’, ’kunde’, ’något’, ’från’, ’ut’, ’när’, ’efter’, ’upp’, ’vi’, ’dem’, ’vara’, ’vad’, ’över’, ’än’, ’dig’, ’kan’, ’sina’, ’här’, ’ha’, ’mot’, ’alla’, ’under’, ’någon’, ’eller’, ’allt’, ’mycket’, ’sedan’, ’ju’, ’denna’, ’själv’, ’detta’, ’åt’, ’utan’, ’varit’, ’hur’, ’ingen’, ’mitt’, ’ni’, ’bli’, ’blev’, ’oss’, ’din’, ’dessa’, ’några’, ’deras’, ’blir’, ’mina’, ’samma’, ’vilken’, ’er’, ’sådan’, ’vår’, ’blivit’, ’dess’, ’inom’, ’mellan’, ’sådant’, ’varför’, ’varje’, ’vilka’, ’ditt’, ’vem’, ’vilket’, ’sitta’, ’sådana’, ’vart’, ’dina’, ’vars’, ’vårt’, ’våra’, ’ert’, ’era’, ’vilkas’]