Web-Based Economic Activity Classification

Comparing semi-supervised text classification methods to deal with noisy labels

Jos Smalbil

Master Thesis
Computer Science

TUDelft

Web-Based Economic Activity Classification

Comparing semi-supervised text classification methods to deal with noisy labels

by

Jos Smalbil

to obtain the degree of Master of Science in Computer Science
Data Science & Technology Track
to be defended publicly on Friday February 21, 2020 at 14:00 PM.

Delft University of Technology

Faculty of Electrical Engineering, Mathematics & Computer Science
Department of Software Technology
Web Information Systems Group

Project duration: May 15, 2019 – February 21, 2020

Thesis committee:	Prof. dr. ir. G.J.P.M Houben,	Delft University of Technology, chair
	Dr. C. Lofi,	Delft University of Technology, supervisor
	Prof. dr. ir. G. Jongbloed,	Delft University of Technology
	Dr. ir. A. van Delden,	Statistics Netherlands

Preface

Before you lies the thesis 'Web-Based Economic Activity Classification', the basis of which is a comparison of semi-supervised text classification methods to deal with noisy labels. The thesis has been written to fulfil the graduation requirements for the Data Science & Technology Track at Delft University of Technology. I was engaged in researching and writing this thesis from May 2019 to February 2020.

The research project was undertaken at the request of Statistics Netherlands, where I undertook an internship. The research was challenging, but conducting extensive investigation has allowed me to answer the intended research objective. Fortunately, my external supervisors Arnout van Delden and Dick Windmeijer and internal supervisor Christoph Lofi were always available and willing to answer my queries.

I would like to thank my supervisors for their time spent to guide and support the process. Arnout has given extensive feedback and could always answer my queries. Dick provided technical knowledge and helped me in tough times. Christoph gave precise and critical feedback such that I could improve upon my work.

Furthermore, I would wish to thank Sepideh Mesbah for the feedback and evaluation of my work. I am also grateful to Arjan van Loon for the work on constructing the custom classification schemes.

Finally, I would like to thank my family and friends for their support during my whole university career.

Jos Smalbil
Delft, February 2020

Abstract

In order to provide accurate statistics for industries, the classification of enterprises by economic activity is an important task for national statistical institutes. The economic activity codes in the Dutch business register are less accurate for small enterprises since small enterprises are not labelled manually. To increase the quality of the register, automatic classification of enterprises based on their websites has been tried with supervised text mining techniques. The performance of current supervised text mining techniques is limited by the available accurately labelled training data. Since inaccurate labels are available for all enterprises, the current study investigates how to leverage the noisy labelled data to improve the economic activity classification of small enterprises based on their webpage texts. The current study compares the performance of various semi-supervised methods that enlarge the training data by leveraging the abundance of noisy labelled data. The methods are compared against a supervised baseline, which uses all noisy data as is. The proposed proportional weakly self-training method queries noisy labelled instances through high probability sampling and filters mispredicted instances. Results showed that proportional weakly self-training improves upon the supervised baseline while requiring far less training instances. From qualitative analyses, we conclude that the filter of proportional weakly self-training reduces error propagation compared to classic self-training. Additional experimental results showed that large enterprises are less suitable as training data for prediction of small enterprises and that top-k performance scores improve results but are not yet sufficient for semi-automatic classification. Further examination of error detection methods is recommended to improve web-based economic activity classification.

Contents

1 Introduction

Official statistics often rely on international classification standards such as the type of economic activity of businesses. The economic activity classification in the European Union is further referred to as the NACE classification. The hierarchical NACE classification system is used by all national statistical institutes in the European Union and can be used to monitor trends across economic (sub)sectors (e.g., for researchers, policymakers, fund managers, and investment managers), to allow companies to relate to their competitors and to perform competitive intelligence actions (Berardi et al., 2015).

The quality of economic statistics by (sub)sector in the Netherlands is based on the General Business Register (GBR). The GBR contains all 1.8 million enterprises in The Netherlands (CBS, 2020) and their main economic activity, the NACE code. The main economic activity of an enterprise is the activity with the largest added value. The NACE codes are mostly derived from the Chamber of Commerce which identifies the economic activity of new enterprises. The NACE codes in the GBR are prone to errors since an enterprise can change their main economic activity or perform multiple economic activities while little effort is put into the maintenance of the NACE codes. To keep the NACE codes of enterprises updated, Statistics Netherlands holds a yearly Structural Business Statistics survey where a small sample of most of the Dutch enterprises is selected and in which the economic activity of enterprises might be derived. The NACE code is manually edited by experts if the code is not in line with values on product and costs filled in on surveys or when it is an outlier, but only for influential (large) enterprises. For small enterprises, the manual validation of NACE codes is too expensive. An automatic process is necessary to obtain accurate NACE codes for small enterprises.

At Statistics Netherlands, for many of the enterprises in the GBR a website address is available (ten Bosch et al., 2019). Websites can give valuable information about the current economic activity of an enterprise (Berardi et al., 2015). Roelands (2017) and Kühnemann (2019) used the text elements of the enterprise's websites to predict their economic activity with supervised text classification methods (e.g., Naïve Bayes and SVM models). However, results showed that the machine learning models performance is limited by the amount of available labelled data. Other mentioned issues are the within-class heterogeneity and the class imbalance in the data sets (Roelands, Kühnemann).

In the previous studies, accurate NACE codes for the training data were only derived from the yearly survey's, which is a limited sample of all enterprises. The large noisy labelled data set from the GBR was not used at all. This study investigates how to leverage noisy labelled data to enlarge the training set with the aim to improve the economic activity classification of small enterprises based on their webpage texts.

Two problem domains related to the case study that we treat need to be clearly defined. One domain is the noisy labelled or in other words inaccurately labelled data. The problem of handling noisy labelled training data is a type of weak supervision (Zhou, 2017). The other domain concerns a classification task using a large amount of unlabelled data and a limited amount of labelled data, which is known as semi-supervised learning (Zhu, 2008).

A popular semi-supervised learning method is active learning (Settles, 2009). In active learning an oracle, e.g., a human expert, can be queried to obtain the ground-truth of unlabelled instances. In our case, the noisy labels can act as an oracle. Active learning aims to minimise the number of queries needed for good model performance. The most used query strategy is uncertainty sampling. The opposite sampling strategy is to query existing instances of which the learner is most certain how to label and add them to the training set which is referred to as self-training (Mihalcea, 2004). Both active learning and self-training require less labelled training data to start with than traditional supervised techniques. (Stikic et al., 2008)

In short, to automatically classify the text of small enterprise's webpages into their main economic activity the main problem of a limited number of labelled examples needs to be solved. Therefore the research objective of the current study is to evaluate methods that enlarge the training set by leveraging the abundance of noisy labelled data.

Our main contributions are as follows:

- A performance comparison of several semi-supervised text classification methods, e.g., self-training and active learning. The comparison addresses the different type of query strategies.
- A performance comparison for several choices of initial training data. The comparison addresses the usefulness of webpage texts of large enterprises for the prediction of small enterprises.
- A performance comparison of the semi-supervised text classification methods for different groups of classes and some characteristic classes. The comparison examines the results in detail.

Another more theoretical contribution is our derivation of definitions for top-k performance scores.

The remainder of this thesis report is structured as follows. Chapter 2 presents a background on NACE classification and discusses related work on handling noisy data and semi-supervised learning methods. Chapter 3 describes the semi-supervised methods used in the experiments. Chapter 4 describes the data flow of the text mining pipeline, the evaluation metrics and a bootstrap method for uncertainty estimation. Chapter 5 presents results for three experiments; with four semi-supervised classification methods, with different types of initial training data and with top-k performance scores. Furthermore, a quantitative analysis and two qualitative analyses examine the results in more detail. Chapter 6 discusses the key findings of our research and contains suggestions for future work on web-based economic activity classification.

2 Background and Related Work

This chapter first discusses previous work on the specific problem of NACE classification, i.e., predicting the economic sector of enterprises. Some estimates for the misclassification rate are derived from literature and an overview is given of recent supervised approaches to the classification problem. Then our two defined problem domains are introduced: noisy labelled data and semi-supervised learning. We translate our problem to a semi-supervised classification problem with noisy labelled data. Several semi-supervised learning techniques are examined that can deal with the large set of available noisy data. We discuss the primary characteristics of each technique.

2.1. NACE classification

2.1.1. NACE taxonomy

The NACE classification system is a hierarchical taxonomy used to classify enterprises in the European Union. Specifically, the taxonomy describes the type of economic activity of enterprises. The taxonomy defines for each group a unique identifier (i.e., NACE code) together with a description. The NACE code identifies the group of enterprises to which a single enterprise belongs. The NACE taxonomy has four levels of hierarchy. In The Netherlands, an additional fifth level of hierarchy is used. The taxonomy can be considered as an 'is-a hierarchy' where each level is a subset of the level above it. For example, enterprises with a fifth-level NACE code of 47641 are classified as 4764 at the four-digit NACE level (Table 2.1). The current study uses the leaf classes (i.e., the five-digit NACE codes) to construct three new classification schemes (Section 4.1.2).

Table 2.1: Characteristics of the NACE taxonomy. The fifth level of detail is used at Statistics Netherlands

NACE code	$\#Groups$	Group example
A-U	21	G Wholesale and retail trade; repair of motor vehicles and motorcycles
01-99	88	47 Retail trade (not in motor vehicles)
011-990	272	476 Shops selling reading, sports, camping and recreation goods
0111-9900	615	4764 Shops selling bicycles and mopeds, sports and camping goods and boats
01111-99000	952	47641 Shops selling bicycles and mopeds

2.1.2. Misclassification rate

In NACE classification the goal is to obtain for each enterprise their correct main economic activity code. Previous work on NACE classification shows that NACE codes available to national statistical institutes are not always adequately covering the true state of affairs concerning the type of activities in enterprises. For instance, Christensen (2008) estimates that in a binary classification setting 18% of Swedish enterprises were misclassified as services rather than production, i.e., their NACE code from the business register did not match their true type of activity. The misclassification rate varied over different industries with some sectors having a misclassification rate of over one fourth. The findings were robust to taking into account that some enterprises have more than one industry code. Burger et al. (2015) report, in an internal audit at Statistics Netherlands on the quality of the GBR, 97% of the three-digit NACE codes were correct for large enterprises (20 employees or more) in the retail trade sector while 69% of the three-digit NACE codes were correct for small enterprises (up to 19 employees) averaged over industries. The proportion of correct NACE codes is higher for large enterprises than for small enterprises because more resources are invested in classifying large enterprises' economic activity through profiling. Another study by van Delden et al. (2016) estimates that one-third of Dutch enterprises in the car trade sector were misclassified. The original data of five consecutive years used in their study could be obtained to estimate the businesses transition probability to change from one NACE class to another. We verified that approximately 2-3% of businesses in the car trade sector have their five-digit NACE class changed throughout a year. Currently, the error rate of the NACE codes in the GBR is unknown.

2.2. Supervised approaches

Recently there have been several approaches to automatically obtain a classification of industry code through supervised machine learning methods. Berardi et al. (2015) used website texts to predict the industry of Italian enterprises according to a taxonomy of 216 classes designed for market research purposes. The study used and evaluated an extensive list of both endogenous (e.g., URL, title, body) and exogenous features. The most predictive exogenous features were derived from Alexa, which was not available for the current study. Of the endogenous features, the URL and website texts (e.g., title, body, headings) were most predictive. Roelands (2017) and Kühnemann (2019) also used website texts to predict the industry of Dutch enterprises. Roelands used a taxonomy with 9 main and 29 subcategories, while Kühnemann attempted to predict 111 classes. Caterini (2018) did not use websites texts but self-reported texts submitted to the Italian Chamber of Commerce describing the economic activity of the enterprise.

The current study can be seen as a continuation of research on the economic activity classification of enterprises based on their websites by Roelands and Kühnemann. Both studies suggest the poor text classification performance for many of the classes is (partly) due to the limited amount of available labelled data. The main distinction with the previous studies is that the current study adds noisy labels to the training set. Since nowadays website data is plenty available, the abundance of noisy labelled website texts could be leveraged. We assume, by leveraging the abundance of data with noisy labels, both the size and quality of the training data can be increased. In numbers, Roelands retrieved a GBR sample of 5 010 websites and Kühnemann retrieved a total of 50 654 website texts with NACE labels for the yearly survey data.

2.3. Handling noisy data

A large amount of noisy labelled or in other words inaccurately labelled data is available. The class labels are inaccurate since little effort is put into the maintenance of the labels. Noisy or weak training data are a source of weak supervision when the data is used to obtain labelled training data. In general, there are three typical types of weak supervision (Zhou, 2017): incomplete supervision, where only a subset of training data is given with labels; inexact supervision, where the training data are given with only coarse-grained labels; and inaccurate supervision, where the given labels are not always ground-truth. In practice, these three types often occur simultaneously (Zhou, 2017). Our situation can be viewed as incomplete supervision when we consider the noisy data to be unlabelled. This view is realistic when the quality of the labels is very poor. Alternatively, we could assume inaccurate supervision where (part of) the data is considered as noisy labelled.

In inaccurate supervision, a typical scenario is learning with noisy labels where for each example complete class information is provided although the correctness is not guaranteed (Hernández-González et al., 2016). In learning with noisy labels three types of approaches can be distinguished (Frénay and Verleysen, 2013): label noise-robust models, label noise-tolerant learning algorithms and data cleaning methods. Most studies dealing with label noise assume random classification noise, i.e., the class labels are subject to random noise. Label noise-robust models (i.e., models that still perform well despite the presence of label noise (Frénay et al., 2014)) are a straightforward approach to deal with label noise. Another approach is to use label noise-tolerant learning algorithms. Herein prior information on the label noise is leveraged (i.e., the label noise is not completely at random). In practice, data cleaning is a basic approach to deal with label noise by somehow identifying the potentially mislabelled instances (Zhou, 2017, Müller and Markert, 2019). The detected suspicious instances can then be either removed or relabelled (Frénay and Verleysen, 2013).

2.4. Semi-supervised learning

In case we would naïvely assume only incomplete supervision semi-supervised learning is commonly used (Zhou, 2017). Semi-supervised learning attempts to automatically exploit a large amount of unlabelled data in addition to a limited amount of labelled data to improve learning performance. In semi-supervised learning a set of labelled instances $\{X, Y\} = \{x_i, y_i; i = 1, \dots, n\}$, where x_i denotes the vector of features for unit i and y_i denotes the class label of unit i, is given together with a set of unlabelled instances $X' = \{x'_1, \dots, x'_m\}$, i.e., the set of feature vectors for all m units. The goal is now to find the corresponding labels $Y' = \{y'_1, \dots, y'_m\}$ for the unlabelled instances. Historically the oldest semi-supervised learning approach is self-training (Chapelle et al., 2006). Nowadays active learning is becoming more popular (Settles, 2009).

2.4.1. Active learning

Incomplete supervision with (human) intervention is also referred to in the literature as active learning. In active learning, one can query the ground-truth of unlabelled instances using a (human) oracle. Active learning aims to minimise the number of queries needed for good model performance, i.e., it attempts to select the most valuable unlabelled instance to be labelled by the oracle (Olsson, 2009). The primary question in active learning is query formulation: how to choose which instances to try next. The query strategy determines the ordering of instances to be checked manually. The most used strategy is called uncertainty sampling. In uncertainty sampling, an active learner queries the instances it is least certain or least confident how to label and retrieves its label through the oracle.

2.4.2. Self-training

For incomplete supervision, instead of uncertainty sampling, another option is to apply the opposite sampling strategy; query existing instances of which the learner is most certain how to label and add them to the training set, which is called self-training (Mihalcea, 2004, McClosky et al., 2006, Settles, 2009), bootstrapping (Zhu, 2008, Pise and Kulkarni, 2008) or incremental semi-supervised training (Rosenberg et al., 2005) and which requires no human intervention. Self-training is a semi-supervised learning method commonly used in Natural Language Processing (Pise and Kulkarni, 2008, Tanha et al., 2017). Self-training first trains a classifier with the small amount of labelled data. The classifier then iteratively labels the unlabelled data with a certain probability. Typically the most confident unlabelled points, together with their predicted labels, augment the training set and the classifier is then re-trained. The incremental labelling of the most certain instances attempts to minimise the number of incorrect labels since randomly adding all of the weakly labelled data to the training set can lead to more incorrect labels.

Perceptive readers could now note the self-training classifier uses its own predictions to teach itself. The self-teaching is indeed problematic since the error rate is prone to error propagation after each iteration. To solve the potential problem of error propagation many extensions to the self-training procedure have been proposed. Li and Zhou (2005) propose a self-training with editing algorithm which utilises a data editing method to identify and remove the mislabelled instances from the self-labelled data. Only reliable self-labelled instances are used to enlarge the labelled training set. Didaci and Roli (2006) propose the concept of ensemble-driven self-training. Here each classifier is trained with the instances which are labelled by an ensemble of multiple classifiers.

A sometimes neglected setting in self-training is the use of proportional or class-specific sampling (Mihalcea, 2004, Li and Zhou, 2005, Didaci and Roli, 2006, Triguero et al., 2014). Proportional sampling ensures that the class distribution in the labelled data is maintained (i.e., in each self-training iteration the number of selected instances assigned to a class is proportional to the prior probability of that class in the training set). The proportional sampling is necessary to ensure the prior probabilities of the augmented training set during the self-training procedure do not change.

In self-training (and active learning) the choice in the number of iterations and the number of added training instances in each iteration influences the speed and performance of the learning process. The number of iterations can be defined in advance or the algorithm can be run until the unlabelled data set is empty. The size of the batch of instances is dependent on the amount of computing time one has. For smaller sized sets the learning progresses slower than for larger sizes. However, a too-large size could harm the self-training performance and it becomes more difficult to find query strategies for selecting a good batch (Attardi et al., 2012).

A more general technique than self-training is co-training (introduced by Blum and Mitchell, 1998), where instead of a single learner one requires multiple learners, each with a different view (in terms of features) of the data. When one learner is confident of its predictions about the data, the predicted label of the data is applied to the training set of the other learners. Co-training can be refined by using an agreement-based objective function as suggested by Collins (1999) and Mihalcea (2004) and as more theoretically justified by Dasgupta (2001).

2.5. Learning with noisy labels

Classic self-training is prone to error propagation. If the model's predictions on unlabelled data are confident but wrong the erroneous data is nevertheless incorporated into training. The model's errors are then amplified in future iterations. Since we have access to noisy labels for all data we can combine the paradigms of semi-supervised learning and weak supervision. Instead of applying semi-supervised classification in a setting with incomplete supervision, we aim to apply the learning process in a setting with inaccurate supervision in which we can leverage the available noisy labels.

For active learning, we propose to use the noisy labels as if they were coming from a (noisy) oracle. The query strategy can be determined by the learner in the semi-supervised setting where the least likely predicted instances are incorporated into the training set (Algorithm 4).

For self-training, we propose self-training with noisy labels where instances are only added to the training set if the predicted label agrees with the noisy label (Algorithm 5). The selection of instances to be added to the training set is a form of data cleaning. Data cleaning aims to prevent mislabelled instances from entering the training set.

2.6. Summary

The NACE misclassification rate in the GBR in the Netherlands has been studied earlier but is recently not accurately estimated. There have been several approaches to the problem of NACE classification. The performance of supervised approaches using only webpage texts was limited by the amount of available labelled data. The methods of active learning and self-training are similar techniques. The query strategy and the iteration parameters are the vital control elements in both settings. Since we have access to noisy labels for all data we combine the two paradigms of semi-supervised learning and weak supervision. To leverage the abundance of noisy labelled data we introduce in the next chapter weakly semi-supervised variants of active learning (Algorithm 4) and self-training (Algorithm 5).

3 Semi-Supervised Learning Methods

In the previous chapter, we explored the approaches of semi-supervised learning and handling noisy data. This chapter presents the proposed weakly semi-supervised learners to be used in the experiments. In weakly semi-supervised classification, weakly refers to the usage of noisy (inaccurate) labels while semi-supervised refers to the incremental learning phase where some kind of query strategy is applied on a large set to expand a small training set. Which query strategy to apply is an important question. The key factor to be changed in our methods is therefore the applied query strategy in the weakly semi-supervised learning setting.

In general, our weakly semi-supervised classifiers (Algorithm 1) can be described as iterative methods having as input three data sets, a classifier and a batch-size. The selection of the three data sets as well as the complete text-mining pipeline is discussed in the next Chapter 4. For now, we refer to the data sets as the labelled gold training set L, the test set T and the noisy labelled set N. In the learning phase, a probabilistic classifier is first trained on labelled set L and iteratively expands L with more instances derived from noisy set N. In each iteration, the classifier's performance is bench-marked on a fixed test set T.

Algorithm 1 General weakly semi-supervised algorithm

```
Input:
    Set of labelled training instances $L$.
    Set of noisy labelled instances $N$.
    Classifier $C$.
    Batch-size $b$.
Procedure:
 1: Apply supervised method to train a classifier $C$ with $L$.
 2: while $|N|$ decreases do
 3:     Select a subset of instances $P$ from $N$.
 4:     Expand $L$ with (a subset of) $P$
 5:     Remove (a subset of) $P$ from $N$
 6:     Retrain classifier $C$ with enlarged set $L$.
 7: end while
Output:
    Enlarged labelled instances set $L$ and classifier $C$.
```

In the next sections, the exact procedure for four weakly (semi-)supervised classification methods is described. The Figures 3.1-3.4 show the main characteristics and differences for each method.

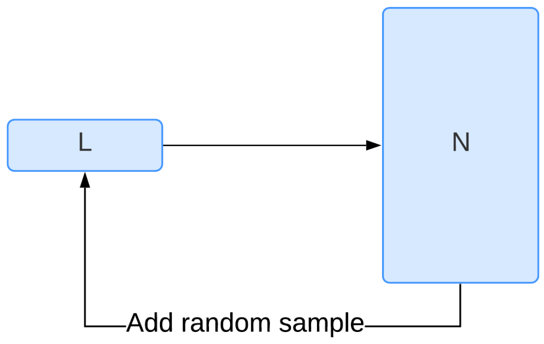

Figure 3.1: Weakly supervised baseline (BL, Section 3.1)

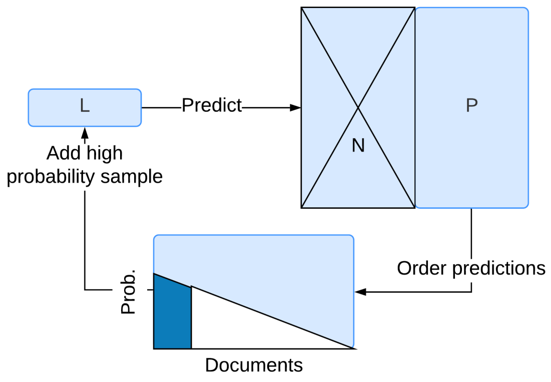

Figure 3.2: Self-training (ST, Section 3.2)

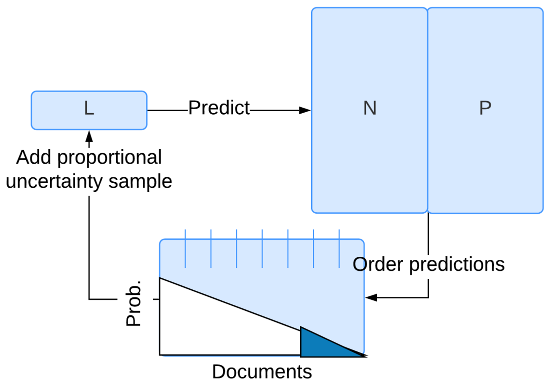

Figure 3.3: Proportional weakly active learning (PWAL, Section 3.3)

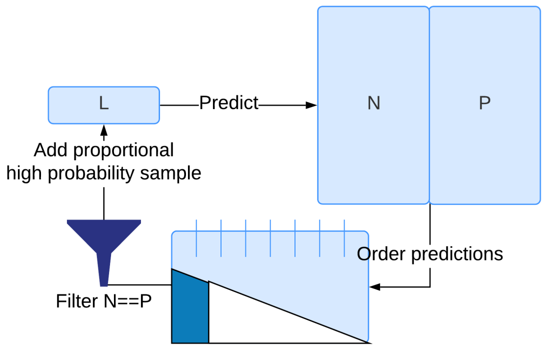

Figure 3.4: Proportional weakly self-training (PWST, Section 3.4)

3.1. Weakly supervised baseline

First, a weakly supervised baseline method is defined (Figure 3.1, Algorithm 2). BL is weakly supervised because it uses the labels in the noisy set N (i.e., the method selects randomly a batch of b instances from N in each iteration). In each iteration, b instances are added to L and removed from N and the classifier C is retrained with the expanded set L. To compare the performance of our methods, classifier C predicts the class labels in each iteration for instances in the fixed test set T.

Algorithm 2 Weakly supervised baseline (random sampling)

```
Procedure:
1: Apply supervised method to train a classifier $C$ with $L$.
2: while $|N| > 0$ do
3:     Select randomly $b$ instances $P$ from $N$.
4:     $L = L \cup P$ with the noisy labels of $P$
5:     $N = N \setminus P$
6:     Retrain classifier $C$ with enlarged set $L$.
7: end while
```

3.2. Self-training

Semi-supervised self-training uses its own predictions on unlabelled data to expand the labelled training set. The query strategy could be denoted as high probability sampling. The ST strategy (Figure 3.2, Algorithm 3) is based on the probabilities of the predictions on all instances in the noisy set N and on the batch-size b. In each iteration, the b instances with the highest predicted probability are added to the labelled set L together with their predicted label.

Algorithm 3 Self-training (high-probability sampling)

```
Procedure:
 1: Apply supervised method to train a classifier $C$ with $L$.
 2: while $|N| > 0$ do
 3:     Predict labels of $N$.
 4:     Select the $b$ most confident predicted instances $P$.
 5:     $L = L \cup P$ with the predicted labels of $P$
 6:     $N = N \setminus P$
 7:     Retrain classifier $C$ with enlarged set $L$.
 8: end while
```

Semi-supervised self-training is known to be prone to error propagation. Additionally, the available noisy labels are not used at all (the ‘cross’ in Figure 3.2). In the next sections, we aim to introduce two semi-supervised learners less prone to error propagation by leveraging the available noisy labels using heuristic query strategies.

3.3. Proportional weakly active learning

In self-training, the query strategy searches for the most confident instances from the noisy set. However, we have access to noisy labels and in the context of active learning these labels can act as a noisy oracle. When having access to an oracle, and in general for active learning, the most used query strategy is uncertainty sampling. Uncertainty sampling is a query strategy in which the least confident instances are selected.

In PWAL (Figure 3.3, Algorithm 4) the least confident predicted instances are added to L in each iteration. Besides the query strategy, we apply the proportional stratified or class-specific sampling technique. In proportional sampling, the number of instances to be added in each iteration is fixed and calculated in advance for each class to ensure fixed probabilities of the class prior. Each sample c_i is sized proportional to the fraction of class i in the population and is calculated such that $c_i \propto \frac{|N_i|}{|N|}$ and $\sum c_i \approx b$, where $|N_i|$ denotes the number of instances of class i in the population.

Algorithm 4 Proportional weakly active learning (uncertainty sampling)

```
Procedure:
 1: Calculate for each class $i$ the proportional number of instances to be added $c_i \propto \frac{|N_i|}{|N|}$ in each iteration
    such that $\sum c_i \approx b$.
 2: Apply supervised method to train a classifier $C$ with $L$.
 3: while $|N| > 0$ do
 4:     Predict labels of $N$.
 5:     for each class $i$ in classes do
 6:         Select the $c_i$ least confident predictions $P$.
 7:         $L = L \cup P$ with the noisy labels of $P$
 8:         $N = N \setminus P$
 9:         Retrain classifier $C$ with enlarged set $L$.
10:     end for
11: end while
```

3.4. Proportional weakly self-training

PWST (Figure 3.4, Algorithm 5) first calculates for each class the proportional number of instances to be added in each iteration. The method feeds labelled set L into a classifier and then repeatedly predicts all labels from N. In each iteration and for each class, a set of c_i instances PN is added to L and removed from N. PN is constructed by first selecting the predictions that match the noisy label and then selecting the c_i most confident predictions. The iterations continue until set N is stable. Set N stabilises when none of its predicted labels match the corresponding noisy labels.

Algorithm 5 Proportional weakly self-training (high-probability sampling)

```
Procedure:
 1: Calculate for each class $i$ the proportional number of instances to be added $c_i \propto \frac{|N_i|}{|N|}$ in each iteration
    such that $\sum c_i \approx b$.
 2: Apply supervised method to train a classifier $C$ with $L$.
 3: while $|N|$ is decreasing do
 4:     Predict labels of $N$.
 5:     for each class $i$ in classes do
 6:         For the most confident predictions $P$, select $c_i$ instances $PN \subset P$ where predicted label
    equals the noisy label.
 7:         $L = L \cup PN$
 8:         $N = N \setminus PN$
 9:         Retrain classifier $C$ with enlarged set $L$.
10:     end for
11: end while
```

3.5. Other methods

While implementing the described methods, several other methods not shown in the current chapter were tested. In PWST, the selection of instances is a form of data cleaning where (presumed) mislabelled instances are prevented to enter the training set. One can either keep (as done in Algorithm 5) the mislabelled instances in the noisy set or delete them. The deletion of the mislabelled instances, i.e., the instances where the predicted label did not match the noisy label, performed similarly to our proposed PWST (Algorithm 5) and is therefore not further evaluated. Furthermore, the variants without proportional or class-specific sampling both for PWAL and PWST were implemented but preliminary results again showed similar learning curves as our proposed methods. Since self-training is often implemented proportional to maintain the class-distribution of the initial data set (Mihalcea, 2004, Li and Zhou, 2005, Didaci and Roli, 2006, Triguero et al., 2014) we used the proportional variant for weakly self-training and weakly active learning.

4

Data and Text Mining Methods

The previous chapter described the proposed machine learning methods (e.g., self-training and active learning) to deal with noisy data. The proposed weakly semi-supervised methods require as input three labelled sets of data and a classifier. This chapter describes the process to obtain the labelled data sets, the text mining pipeline, the classification method, the evaluation metrics used and a method to assess the prediction confidence.

4.1. Data

This section describes how we obtained the labelled data sets. A summary of the data flow of the text mining pipeline is shown in Figure 4.1. The data set started with is the Dutch General Business Register (GBR) containing for each enterprise a website URL, a NACE code and the enterprise size. First, the text of an enterprise's webpage is scraped (Subsection 4.1.1). Secondly, only enterprises with a NACE code in our used classification schemes are selected (Subsection 4.1.2). Thirdly, a filter step is applied based on the obtained webpage texts (Subsection 4.1.3). Fourthly, exploratory analysis is performed on the resulting labelled data (Subsection 4.1.4). Finally, the selection of the test set T, the gold set L and the noisy set N from the labelled documents is described (Subsections 4.1.5 and 4.1.6). The construction and selection of the features are described in Section 4.2.

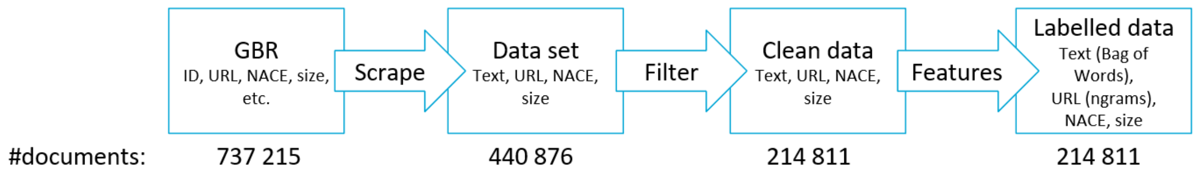

Figure 4.1: Summary of the data flow of the text mining pipeline.

4.1.1. From GBR to scraped data set

In 2018 there were a total of $1\ 665\ 795$ Dutch business units (CBS, 2020). Each business unit or enterprise is listed in the GBR at Statistics Netherlands. From the GBR of 2018, we obtained $737\ 215$ unique webpage addresses belonging to an enterprise. Besides the business unit's unique identifier, the GBR also contains the classification of the number of employees (i.e., the enterprise size) and the NACE code of each enterprise.

With a list of webpage addresses from the Chamber of Commerce, a large set of Dutch enterprise's webpages (± 1.8 million) had been scraped and cached at Statistics Netherlands in May 2019. The text of the main webpage of each website was scraped resulting in a cache size of 1.5 terabyte. For each of the $737\ 215$ unique URL's obtained from the GBR, we requested the cached HTML webpage. The requests were successful for $440\ 876$ of the $737\ 215$ webpages, i.e., a successful HTTP response status code $(200 - 299)$. With a Python library, the clean texts from an HTML webpage are obtained. The library jusText (Pomikálek, 2011) is a boilerplate content removal tool designed to preserve mainly text containing full sentences and is therefore well suited for creating linguistic resources. All texts heuristically qualified by jusText as ‘bad’ were removed from the HTML webpage, resulting in a clean text for each enterprise. Now the data set has 440 876 documents and contains for each business identifier a five-digit NACE code, a business size code, a URL and a clean webpage text (Figure 4.1).

Note the GBR data is from the year 2018 although the webpages were scraped in May 2019. Since approximately 2-3% of businesses change their five-digit NACE class throughout a year we expect to see at least a similar increase in error rate besides the (currently unknown) misclassification rate in the GBR.

4.1.2. Classification schemes

The full NACE classification scheme currently used by all member states in the European Union has 615 unique classes at the fourth level (consisting of four-digit numerical codes). In the Netherlands, a fifth numerical code is added resulting in 952 unique classes at the fifth level. Such a high number of classes is detrimental for the classification performance. Therefore we decided, with the help of a NACE domain expert, to split-up the five-digit NACE classification scheme in three smaller schemes. The schemes are constructed in accordance with the manual annotating process at Statistics Netherlands to determine the NACE code of an enterprise. Herein first the type of business chain and the type of good of an enterprise is determined by an expert. Furthermore, the schemes are constructed with the aims of removing between-class imbalance (i.e., obtaining more similar class frequencies) and removing within-class imbalance (i.e., obtaining more homogeneous classes) by merging or splitting individual classes.

The first scheme has 8 classes and is about the type of business chain of an enterprise (Table 4.1). The second scheme has 120 classes and is about the type of good an enterprise is associated with. The third scheme is constructed as the combination of the type of business chain and the type of good, resulting in a total of 181 unique economic activity classes.

Table 4.1: List of classes in the type of business chain scheme

Class name
Detailhandel in winkel
Detailhandel overig
Detailhandel via internet
Dienst
Groothandel
Industrie
Markthandel
Reparatie

The set of NACE codes used in all the three custom classification schemes are a subset of the set of the five-digit NACE codes in the GBR. Specifically, the included NACE codes are from the economic sectors manufacturing, construction, distributive trades and services. These economic sectors contribute to a large portion of employees and net turnover in trade and industry. The labelled data set is filtered by only selecting the documents where the five-digit NACE codes are in the subset of NACE codes used in our classification schemes. After the selection, 303 331 of 440 876 documents remain in our data set.

4.1.3. Clean data

To prevent low-quality text data from entering the learning system, we delete the following documents from our data set of 303 331 documents:

- Non-Dutch webpages as detected by the library Langdetect (Shuyo, 2010) (64 909)
- Webpage texts that are duplicated (50 562)
- Webpages with uninformative content (18 045)
- Webpage texts with less than 20 characters (8 063)

Webpages with uninformative content were manually selected by using a rule-based text filter. The filter excludes documents containing text that indicate the webpage is uninformative such as ‘is offline’ or ‘is gereserveerd’. The result of the filter for low-quality text data is a clean data set with for each enterprise a website text, a URL, a NACE code and a size code. After the filter step 215 843 of 303 331 labelled documents remain as input for our learning system (Figure 4.1).

4.1.4. Labelled documents

The 215 843 labelled documents are used to construct the test set, the gold set and the noisy set for the semi-supervised learning methods. The distribution of the documents among the classes in the classification schemes are shown in Figure 4.2 for each of the three classification schemes.

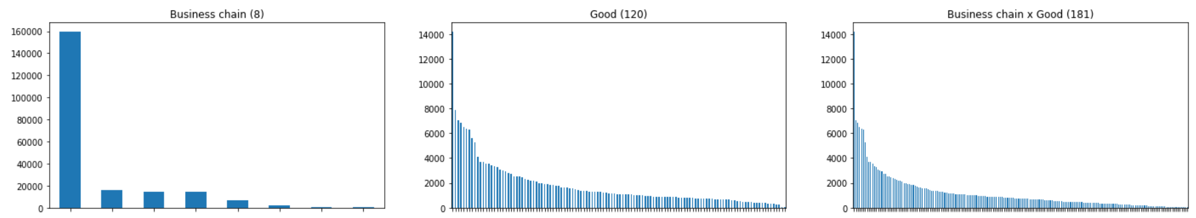

Figure 4.2: Document distribution over classes in the classification schemes: the ‘Type of business chain’ scheme with 8 classes, the ‘Type of good’ scheme with 120 classes and the ‘Type of business chain x Type of good’ scheme with 181 classes.

There is a large class imbalance in all three classification schemes, especially in the ‘Type of business chain’ scheme with 8 classes where the most frequent class (‘Dienst’) amounts to 74% of the 215 843 documents. For the other two classification schemes, the most frequent class (‘Organisatieadviesbureaus’) amounts to 6.6% of the labelled documents.

Another important distribution in the set of labelled documents is that of the size of the enterprises. The size of an enterprise is categorized into 20 categories. The categories are defined by the number of employees of an enterprise and ranges from 0 to 2000+ employees. The distribution of the 215 843 documents by enterprise size is shown in Figure 4.3.

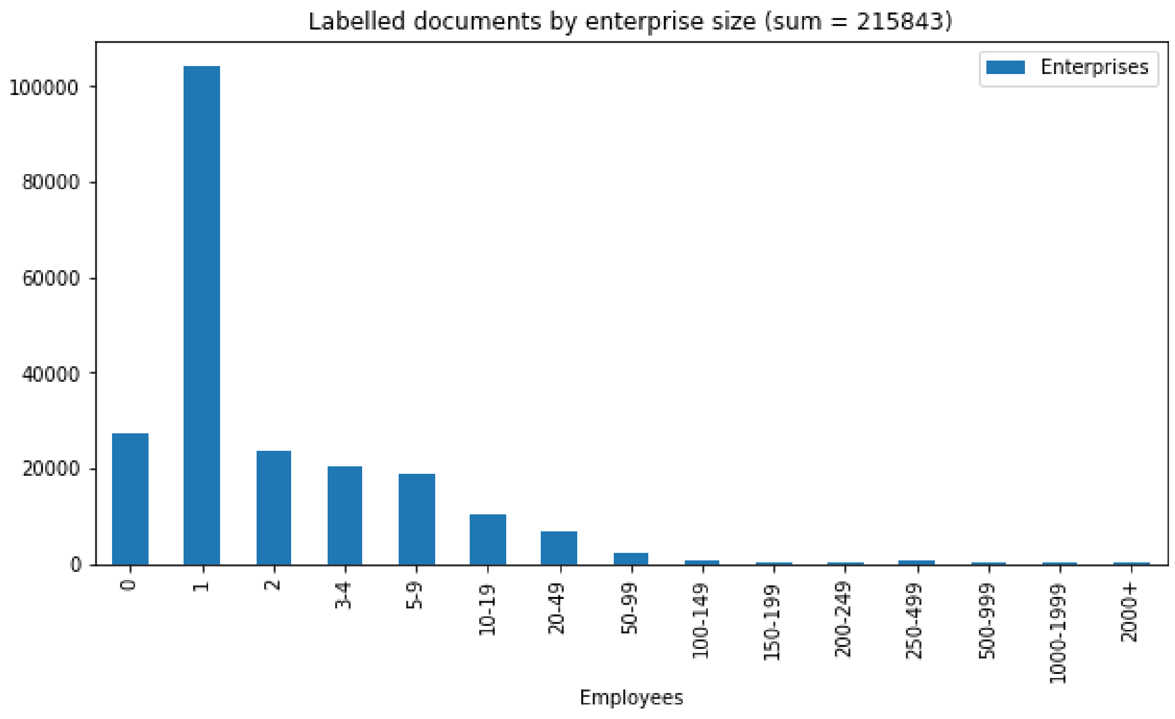

Figure 4.3: Document distribution by number of employees of an enterprise.

Almost half of the enterprises have only one employee. Most one-man enterprises belong to self-employed freelancers. An enterprise has zero employees when it has no registered employees, e.g., start-ups or holdings.

4.1.5. Selection of test set

The goal of Statistics Netherlands is to improve the accuracy of the NACE labels within the GBR for small enterprises (Chapter 1). Therefore from the labelled documents the small enterprises with less than 5 employees are selected. The test set is constructed by a stratified random sample of 20% from the selected set of small enterprises. The stratification is done with respect to the class labels (i.e., the random sample contains approximately the same percentage of samples of each target class as the complete set of small enterprises). The test set T has a total of $35\ 093$ labelled documents and is always left unchanged.

4.1.6. Selection of gold sets

Semi-supervised classification methods need a gold labelled training set to initialize. Gold refers to the assumption that the labels in the training set are correct. Since we do not have access to a perfectly labelled set, we use a set that comes as close as possible to a perfectly labelled set. In the remainder, we refer to this as the gold sets. The gold sets are constructed to compare the effect of different types and sizes of the gold set. Four gold sets are considered. Table 4.2 shows the gold set types and sizes for which a gold set is constructed.

Table 4.2: Characteristics of the four gold sets. Size refers to the number of labelled samples in the gold set, type refers to the number of employees of the enterprise.

Gold set	Size S ($\approx 8\ 000$)	Size L ($40\ 380$)
Type S ($0-4$)	Size S, type S	-
	Size S, type S (no type L)	
Type L ($5+$)	Size S, type L	Size L, type L

- **The size L, type L gold set** is constructed by selecting from the labelled documents the enterprises with 5 employees or more ($40\ 380$ documents). The rationale behind the selection of large enterprises is that large enterprises have more accurate labels than small enterprises.

- **The size S, type S gold set** is constructed by selecting from the labelled documents the enterprises which have less than 5 employees and which are in the yearly survey of 2015, 2016 or 2017. Because the size of the selection is limited, not all classes might be represented in the selection. Therefore for each economic activity class in each of the three used classification schemes, we add random documents from that class until a minimum of 20 selected documents are obtained. Using at least 20 documents per class provides a more stable generalisation performance (Dumais et al., 1998).

- **The size S, type S (no type L) gold set** is constructed by first deleting all large companies from the labelled documents. A stratified sample is taken from the remaining documents such that the sample size equals the size of the 'size S, type S' gold set ($\approx 8\ 000$).

- **The size S, type L gold set** is constructed by first selecting a random sample from the enterprises with 5 employees or more ($40\ 380$ documents), such that the sample size approximately equals the size of the 'size S, type S' gold set ($\approx 8\ 000$). Subsequently, for each economic activity class in each of the three used classification schemes, we add random documents from that class until a minimum of 20 selected documents are obtained.

The remaining documents (i.e., the documents not in the test set nor gold set) define the noisy set N, except in the 'size S, type S (no type L)' gold set where additionally the large enterprises are completely deleted.

4.2. Features

4.2.1. Derivation of features

Two types of features are assembled; one for the webpage texts and one for the webpage domains. For feature representation of text data, common approaches used in literature are the bag of words model and the n-gram model. We apply both approaches: for webpage texts a bag of words feature representation is used while for the webpage domains a character n-gram feature representation with a length for n of 3 to 6 is used.

The content of the webpage texts is cleaned before extracting the features: all special symbols are deleted, all numbers are deleted, duplicated white-spaces are deleted. The text is then transformed into tokens. The webpage domains are obtained by removing any prefixes (i.e., 'http://' and 'www') and suffixes (e.g., '.com', '.nl') from the website URL.

4.2.2. Term weighting

Term weighting is applied to put a stronger emphasis on tokens high frequent in a particular text but low frequent in the other texts. Term weighting is therefore a way of increasing the weight of terms that are assumed to be important for classification. In this study, we apply the state-of-the-art BM25 weighting scheme (also known as Okapi weighting method) to both types of features. $BM25$ is a bag-of-words retrieval function that ranks a set of documents based on the terms appearing in each document, regardless of their proximity within the document. For the $BM25$ method, the frequency of term t in document d (tf), the inverse-document frequency (idf) values for the terms, the document length ($len(d)$) and the average document length ($avgdl$) are calculated using the full data set. The $BM25$ weighting formula is now defined as:

$$BM25(t,d) = idf(t) \cdot \frac{tf(tf,d) \cdot (k_1 + 1)}{tf(t,d) + k_1 \cdot (1 - b + b \cdot \frac{len(d)}{avgdl})} \tag{4.1}$$

where $b = 0.75$ and $k_1 = 1.5$ are constants as in Manning et al. (2008, p.232-233) and where $idf(t)$ is defined as:

$$idf(t) = log \frac{0.5 + N - df(t)}{0.5 + df(t)} \tag{4.2}$$

where N denotes the total number of documents d and $df(t)$ denotes the document frequency (i.e., the number of documents d containing term t).

The $BM25$ function is applied to all features in all documents: irrespective of documents residing in the gold set, the noisy set or the test set. All documents can be used as input for the weighting function since in our semi-supervised setting we have no real 'unseen' data, i.e., all webpages are available beforehand.

4.2.3. Feature selection

Text classification is usually a problem with a high dimensional feature space. For example, when using a bag of words approach the number of unique words in the corpus is often larger than the number of single documents. Feature selection is a dimensionality reduction technique that aims at selecting a subset of features from the input data by removing irrelevant, redundant or noisy features while maintaining the model performance. One way of feature selection is to set a threshold for the minimum and maximum frequencies that a feature occurs in the document collection. The motivation is that highly (in)frequent features add much less information since they are either present in almost every document or present in only a few documents.

For the bag of words features a minimum document frequency of 10 is used. Since in the character n-gram feature type the number of features is even higher, a larger minimum document frequency of 50 is chosen for the n-gram features derived from the webpage domain names.

Additional feature reduction techniques such as stopword removal and lemmatization have been tried but were not implemented in the final models.

4.3. Classification method

After applying feature selection the number of features in our term-frequency document matrix is still reasonably high. To prevent overfitting due to a high number of features relative to the number of documents a linear classifier is a good choice. In multi-class text classification, the most widely used classification methods are the Support Vector Machine (SVM) and the multinomial Naïve Bayes (NB) classifier. Both classifiers are linear but since NB is faster than a SVM and NB outperforms a SVM in similar settings (Roelands, 2017) we decide to use the NB classifier for our experiments.

In a general text classification setting, the predicted class $\hat{c}$ a given document d belongs to is formulated as:

$$\hat{c} = argmax_c P(c|d) = argmax_c P(c)P(d|c) \tag{4.3}$$

The term Naïve Bayes refers to the strong independence assumptions in the Naïve Bayes model rather than the particular distribution of each feature. A NB model assumes each of the features used are conditionally independent of one another given the true class. Formally the Naïve Bayes conditional independence assumption states when given some class c, the probability of observing a document d (or equivalently, the terms t_1 through t_{n_d} of document d) is:

$$P(d|c) = P(t_1, \ldots, t_{n_d}|c) = \Pi_{k=1}^{n_d} P(t_k|c) \tag{4.4}$$

where t_k denotes the term at position k of document d. While the NB assumption is 'naïve' it turns out that in practice NB models work well even if its assumptions are violated (Zhang, 2004). When defining a document space X consisting of all documents we can write the class-conditional probability for document d as:

$$P(d|c) = \Pi_{1 \leq k \leq n_d} P(X_k = t_k|c) \tag{4.5}$$

where X_k is the random variable for position k in the document and takes as values terms from the vocabulary. Thus $P(X_k = t|c)$ is the probability that in a document of class c the term t will occur in position k.

Because in NB classification each term is considered separately the actual position of the terms is not relevant. The positional independence assumption therefore states:

$$P(X_{k_1} = t|c) = P(X_{k_2} = t|c) \tag{4.6}$$

for all positions k_1, k_2, terms t and classes c.

After applying the conditional independence and positional independence assumptions the final decision rule that the multinomial Naïve Bayes classifier maximises (i.e., the class $\hat{c}$ in Equation 4.3 for which the rule reaches a maximum) is:

$$P(c)P(d|c) = P(c)\Pi_{1 \leq k \leq n_d} P(X = t_k|c) \tag{4.7}$$

where the prior probability $P(c)$ is determined as the number of documents labelled as class c divided by the total number of documents d and where the conditional probability $P(X = t_k|c)$ is estimated as the relative frequency of term t_k in documents belonging to class c. Note the relative frequency uses the positional independence assumption and includes multiple occurrences of a term in a document.

A problem with maximising the above decision rule is the occurrence of term-class combinations in the test set not occurring in the training set, resulting in zero probabilities and thus wiping out all other information in the other probabilities when they are multiplied. To eliminate zeros Lidstone smoothing adds a value of $0 < \alpha < 1$ to each count for the relative frequency of term t_k belonging to class c. Similar as in Kühnemann (2019), we set $\alpha = 0.1$. For the implementation of the NB classifier the Python package Scikit-learn (Pedregosa et al., 2011) is used.

4.4. Evaluation metrics

To evaluate the experiment's performance and to compare the results against each-other evaluation metrics are used. For classification tasks the terms true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) compare the predictions of a classifier against the true document label in the test set. The terms positive and negative refer to the classifier's prediction while the terms true and false refer to whether that prediction corresponds to the true class label (Table 4.3).

Table 4.3: Confusion matrix for class c

	Predicted label c	Predicted not c
True label c	True positive (TP_c)	False negative (FN_c)
True not c	False positive (FP_c)	True negative (TN_c)

4.4.1. Accuracy

The most basic evaluation metric for classification tasks is the accuracy measure, defined in the multi-class setting as (van Asch, 2013):

$$\text{accuracy} = \mathcal{A} = \frac{\sum\limits_{c \in C} TP_c}{N} \tag{4.8}$$

where C is the set of classes in the test set, TP_c is the number of true positives for class c and N is the total number of instances in the test set.

4.4.2. F1-scores

The F_1-score is a widely used metric in multi-class text classification. The F_1-score is defined as the harmonic mean of precision and recall. Both precision and recall can be calculated using so-called micro- or macro-averaging. In micro-averaging, the elements of the confusion matrix are aggregated over all classes while in macro-averaging the elements are used to calculate the metric scores per class and then averages these scores. Therefore micro-averaging treats all predictions equally (with a bias towards larger classes) while macro-averaging treats all classes equally. Micro-averaged precision and micro-averaged recall are defined as:

$$\text{micro-averaged precision} = \frac{\sum\limits_{c \in C} TP_c}{\sum\limits_{c \in C} TP_c + \sum\limits_{c \in C} FP_c} \tag{4.9}$$

$$\text{micro-averaged recall} = \frac{\sum\limits_{c \in C} TP_c}{\sum\limits_{c \in C} TP_c + \sum\limits_{c \in C} FN_c} \tag{4.10}$$

Note in the multi-class setting $\sum\limits_{c \in C} FP_c = \sum\limits_{c \in C} FN_c$ since every single false positive for one class is a single false negative for another class. Further note $\sum\limits_{c \in C} TP_c + \sum\limits_{c \in C} FP_c = N$ and $\sum\limits_{c \in C} TP_c + \sum\limits_{c \in C} FN_c = N$. Therefore micro-precision, micro-recall and accuracy are the same. Since the F_1-score is calculated as the harmonic mean of precision and recall, the micro-averaged F_1-score is also the same:

$$\text{micro-averaged } F_1\text{-score} = \text{micro-averaged precision} = \text{micro-averaged recall} = \text{accuracy} \tag{4.11}$$

The equivalence of the micro-averaged scores is a disadvantage if one wants to compare precision and recall or to tune the system such that it gains more precision at the cost of recall or vice versa.

To calculate the macro-equivalent scores, we first define the precision and recall of a single class c:

$$P_c = \frac{TP_c}{TP_c + FP_c} \tag{4.12}$$

$$R_c = \frac{TP_c}{TP_c + FN_c} \tag{4.13}$$

Now the macro-averaged precision and macro-averaged recall are defined as the averages over all C classes:

$$\text{macro-averaged precision} = \frac{\sum_{c \in C} P_c}{|C|} \tag{4.14}$$

$$\text{macro-averaged recall} = \frac{\sum_{c \in C} R_c}{|C|} \tag{4.15}$$

Finally, the macro-averaged F_1 score is defined as the harmonic mean of macro-averaged precision and macro-averaged recall. Because in macro-averaging scores are first calculated per class the metric is insensitive to class imbalance and treats all classes as equal. In contrast, micro-averaging treats each prediction as equal and is therefore sensitive to class imbalance.

4.4.3. Reciprocal Rank

The Naïve Bayes classifier is probabilistic and can return for each class a classification probability, resulting in a ranked list of classes that appear to be the most plausible for the webpage. Instead of 'hard classification' where only the top-predicted class is considered to evaluate the results, in 'soft classification' the ranked list of predicted classes is used. A straight-forward soft classification measure is the reciprocal rank (RR) of a document d, defined as

$$RR(d) = \frac{1}{r(d)} \tag{4.16}$$

where $r(d)$ is the rank of the true class of document d in the ranked list of predicted classes. A variant of RR, introduced by Berardi et al. (2015), only looks at the top$-k$ predicted classes and gives no credit to classes at rank $(k+1)$ or higher and is defined as:

$$RR_k(d) = \begin{cases} \frac{1}{r(d)} & : r(d) \leq k \\ 0 & : r(d) > k \end{cases} \tag{4.17}$$

.

Similar as with the F_1 score both micro-averaged and macro-averaged scores can be calculated for the reciprocal rank. The micro-averaged scores RR_k^{μ} are calculated by averaging $RR_k(d)$ across all the test instances whereas the macro-averaged scores RR_k^{M} are calculated by first computing the class-specific averages of $RR_k(d)$ and then averaging the results across the classes.

4.4.4. Top-k F1-scores

The F_1-score is calculated only by considering the top-predicted classes for each document. However, by considering the top-k predicted classes we will define the extended $F_1(k)$-score. For the extension of the F_1-score, the four categories in the confusion matrix need to be redefined. While for the true class labels no changes occur, for the predicted class labels we now want to look at the top-k predicted classes instead of only the top predicted class. To prevent inconsistencies we only want to look at the top-k predicted classes if the true class label matches the class label for which we are calculating the

Table 4.4: Conditional top-k confusion matrix for class c

	c in top-k predictions	c not in top-k predictions
True label c	True positive (TP_c)	False negative (FN_c)

	c is top prediction	c not top prediction
True not c	False positive (FP_c)	True negative (TN_c)

confusion matrix. Therefore the new confusion matrix is conditional: the decision to look at the top-k predicted classes depends on the true class label (Table 4.4).

To illustrate how the conditional top-k confusion matrix is used Table 4.5 shows the decisions for four documents with their true labels and their first three predicted labels. The decisions are based on the class that we are calculating; if this class matches the true label we look at the first three predictions, else we look only at the top prediction. For example, when calculating the decisions for class U we see that the true labels match for the first two documents. Therefore we use the upper part of the conditional top-k confusion matrix; when U is in the first three predictions the decision category is TP_U, otherwise the decision category is FN_U.

Table 4.5: Examples of top-3 prediction according to the conditional top-k confusion matrix in Table 4.4

		Predictions			Class-based decision		
Document	True label	1	2	3	U	V	W
1	U	W	V	U	TP_U	TN_V	FP_W
2	U	W	V	Z	FN_U	TN_V	FP_W
3	V	U	V	W	TN_U	TP_V	TN_W
4	V	V	U	W	FP_U	TP_V	TN_W

Using Table 4.4, the top-k precision, top-k recall and $F_1(k)$ scores can be calculated with the same formula's as in Subsection 4.4.2, both for micro-averaged and macro-averaged scores. Note the redefinition of the confusion matrix does not affect the counts of the false positives and the true negatives, only the counts of the true positives and the false negatives change. Precisely the number of true positives goes up with the same amount as the number of false negatives goes down. With this observation we can derive the following lemma:

Lemma 1. *The top-k F_1 score, $F_1(k)$, is a monotonically increasing function of k.*

Proof. The $F_1(k)$ score is defined as the harmonic mean of top-k precision and top-k recall and can be directly expressed in terms of the four categories in the top-k confusion matrix for class c:

$$F_{1,c}(k) = \frac{2TP_c}{2TP_c + FN_c + FP_c} \tag{4.18}$$

We use the observation that the number of true positives goes up with the same amount n as the number of false negatives goes down to calculate the $F_{1,c}(k+1)$ score for a top-($k+1$) confusion matrix, given an arbitrary top-k confusion matrix:

$$\begin{aligned}
F_{1,c}(k+1) &= \frac{2(TP_c + n)}{2(TP_c + n) + (FN_c - n) + FP_c} && (4.19)\\
&= \frac{F_{1,c}(k)(2TP_c + FN_c + FP_c) + 2n}{2TP_c + FN_c + FP_c + n} && (4.20)\\
&\geq \frac{F_{1,c}(k)(2TP_c + FN_c + FP_c + 2n)}{2TP_c + FN_c + FP_c + n} && (4.21)\\
&= F_{1,c}(k) + \frac{n}{2TP_c + FN_c + FP_c + n} && (4.22)\\
&\geq F_{1,c}(k) && (4.23)
\end{aligned}$$

for some integer $n \geq 0$. □

4.5. Uncertainty estimation

The evaluation metrics described in the previous section all provide an estimation of the performance of a classifier, given a labelled test set and the predictions. To assess the confidence of the performance estimations uncertainty estimates such as the standard error and the confidence intervals need to be obtained. In this section, we show how to calculate uncertainty estimates for the performance scores by applying the bootstrap method principle to the test set.

4.5.1. Bootstrapping test set

The test set T was constructed as a stratified random sample from the set of small enterprises. When applying the bootstrap method to the labels of T together with the classifier's predicted labels, a sequence of independent estimations for a given metric is obtained through resampling from the units of T. The sequence of independent estimations allows to obtain an estimation of the sampling distribution and uncertainty estimates such as the standard deviation for the given metric.

Given metric M we define the sequence of independent bootstrap estimations as M^* where each single estimation M_i^* is defined for $1 \leq i \leq R$ for some integer R. The corrected sample standard deviation for the metric M is then defined as:

$$s_M = \sqrt{\frac{1}{R-1}\sum_{i=1}^{R}(M_i^* - m(M^*))^2} \tag{4.24}$$

where $m(M^*)$ denotes the mean of the sequence M^* and where R denotes the sequence length (i.e., the number of bootstrap resamples from the units of T). Each M_i^* is calculated by applying the metric function M to one simple random resample from the units of T. One bootstrap resample from the units of T consists of random draws with replacement of the same size as T (i.e., 35 093).

4.5.2. Testing differences

The bootstrap method allows to obtain an estimate of the standard deviation s_M for the performance scores $\hat{\theta}$ of our semi-supervised methods. With the performance scores and standard deviations of different methods, we can give the reader an understanding of the significance of the results. For example, to test whether the performance of the baseline (BL) method is different from the proportional weakly self-training (PWST) method, we test the null hypothesis $H_0 : \hat{\theta}^{BL} = \hat{\theta}^{PWST}$ against the alternative hypothesis $H_1 : \hat{\theta}_i^{BL} \neq \hat{\theta}_i^{PWST}$. The null hypothesis can be rejected at a 0.05 level of significance if shown that $|\hat{\theta}^{BL} - \hat{\theta}^{PWST}| \geq 2S_M(\theta^{BL} - \theta^{PWST})$ where θ^{BL} and θ^{PWST} denote the accuracy scores for the predictions on the bootstrapped test set for respectively BL and PWST. The standard deviation of the difference for metric M in sequence A compared to sequence B is defined as:

$$s_M(A-B) = \sqrt{Var(A-B)} = \sqrt{Var(A) + Var(B) - 2Cov(A,B)} \tag{4.25}$$

where A and B each denote the accuracy scores for the predictions on the bootstrapped test set for one of the semi-supervised methods. Note that in the bootstrap estimations the same random seed should be used for the two resample procedures to be able to calculate the covariance of the sequences.

5 Experiments and Results

The previous chapter described the process to obtain the data and the text mining pipeline. This chapter presents the experiments conducted for our research objective. The main research objective was to evaluate methods that increase the size of the training data by leveraging the abundance of noisy labelled data. The first experiment compares the performance of the semi-supervised methods (Chapter 3) for three classification schemes (Section 4.1.2). The second experiment evaluates the effect of using different types and sizes for the gold set (Section 4.1.6). The third experiment evaluates the top-k performance scores. Table 5.1 shows for each of the three experiments the settings used. Furthermore, one quantitative and two qualitative analyses are performed.

Table 5.1: Overview of used settings for the three experiments.

Exp.	SSL method	Gold set	Scores	Section
I	Supervised baseline (BL) Semi-supervised self-training (ST) Proportional weakly active learning (PWAL) Proportional weakly self-training (PWST)	'Size S, type S (no L)'	top-1	5.1
II	Proportional weakly self-training	'Size L, type L' 'Size S, type S (no L)' 'Size S, type S' 'Size S, type L'	top-1	5.2
III	Proportional weakly self-training	'Size S, type S (no L)'	top-k	5.3

In the experiments in the current chapter, the text mining pipeline and the supervised classifier is left unchanged. The set of labelled documents (Section 4.1.4) is always used to construct the three labelled sets. The labelled test set T, containing $35\ 093$ small enterprises, is left unchanged. The feature representations used are bag of words for the webpage texts and character n-grams for the webpage domains (Section 4.2.1). To both feature representations the $BM25$ weighting function is applied to construct a term-document matrix (Section 4.2.2). Feature selection is applied by only including terms in the term-document matrix if the term exceeds a minimum document frequency threshold (Section 4.2.3). In the weakly semi-supervised methods (Algorithm 2) the used classifier is multinomial Naïve Bayes (Section 4.3) and the batch-size is fixed at $10\ 000$ documents in each iteration of the method.

5.1. Comparison of semi-supervised methods

5.1.1. Research objective

In Chapter 3 we described the four implemented classification methods for dealing with the noisy data. The methods differ in the applied query strategy, i.e., which subset of instances to choose from the noisy set. The current experiment aims at evaluating the effect of the query strategies. The main question

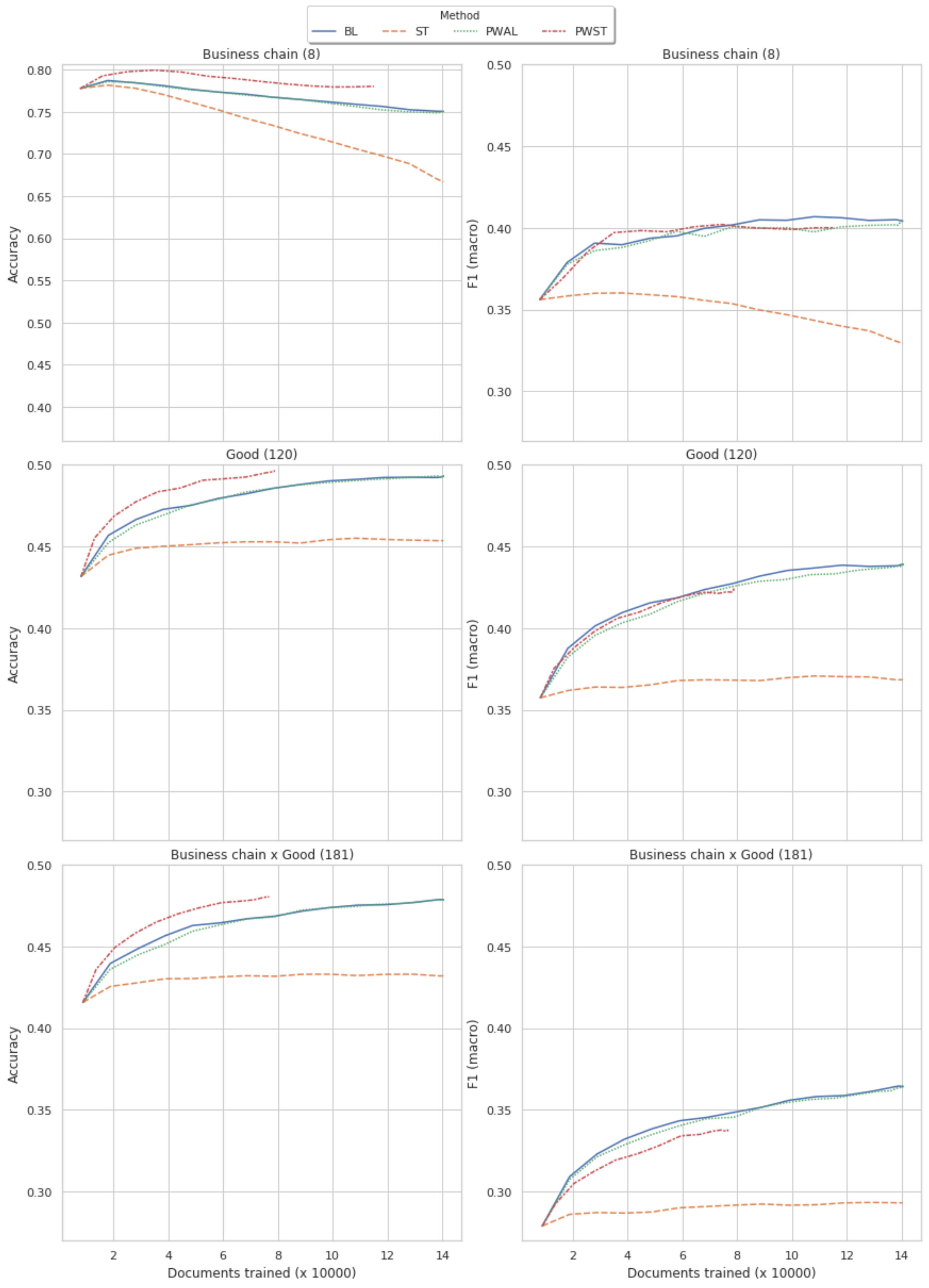

Figure 5.1: Accuracy scores (left) and macro F_1 scores (right) for the supervised baseline (BL) and the semi-supervised self-training (ST), proportional weakly active learning (PWAL) and the proportional weakly self-training (PWST) methods for the gold set ‘Size S, type S (no type L)’ for each of the three classification schemes.

to be answered is whether the query strategies in our semi-supervised methods are an improvement compared to the random sampling strategy in the supervised baseline method. For evaluation, the metrics used are both accuracy and macro-averaged F_1-score.

5.1.2. Experimental set-up

Still unmentioned are choices for the classification schemes, the evaluation metrics and the golds sets for the comparison of the semi-supervised methods. The results for all three classification schemes are compared. For the performance evaluation, the accuracy and macro-averaged F_1 scores are examined. The gold set used is 'Size S, type S (no type L)', a small-sized sample of the small companies and where the large companies are excluded from the noisy set. The comparison with other gold set choices is the main question in the next section.

5.1.3. Results

For each of the four classification methods and each of the three classification schemes accuracy and macro-averaged F_1 scores are shown in Figure 5.1. For the presentation of the results, instead of only the final performance scores, the learning curves are shown. The learning curves allow to compare for each method the performance by the number of documents trained. The curves therefore show the initial performance (i.e., the performance after training on the small gold set) of the methods as well as the final performance and the final number of documents used.

The results show, in all settings, that ST is under-performing compared to BL. For the 'Type of business chain' scheme ST is even reducing the final performance compared to the performance when trained on the gold set. BL and PWAL perform similarly. Their final performance scores are exactly similar since the algorithms differ only in the order of the sample selection; the final set of selected instances is always the full noisy set N with the corresponding noisy labels. PWST only adds instances if the predicted label matches the noisy label and is therefore using less training instances. When looking at the accuracy performance scores PWST improves upon BL. The final scores are higher than those of BL while requiring far less training instances. When looking at the macro-averaged F_1-scores something interesting happens. The advantage of PWST compared to BL disappears. Macro-averaged scores give equal weights to all individual classes and averages them. The lower macro-averaged scores suggest that PWST favours the correct prediction for large classes at the cost of misclassifying instances for small classes.

5.2. Comparison of gold sets

The gold set is the set of labelled training instances used to train the supervised NB classifier before applying the query strategy in the iterative semi-supervised methods. The query strategies in the methods all use the (sorted) prediction probabilities of the NB classifier which was trained on the gold set. Therefore, besides the initial performance, the gold set strongly influences the performance in all further iterations.

5.2.1. Research objective

The current experiment aims at evaluating the effect of the choice of gold set. The experimental results could give insights into the best choice for the type and size of the gold set.

5.2.2. Experimental set-up

The characteristics of four gold set choices were shown in Table 4.2. In short, the construction of the gold sets is based on the desired size of the gold set and the content type of the gold set. The content type is based on the number of employees of an enterprise. Small enterprises (type S) have less than five employees while large enterprises (type L) have five or more employees.

The best performing method in terms of accuracy is PWST. This method is chosen to compare the accuracy performance scores for each of the four gold sets and each of the three classification schemes. Results are presented in Figure 5.2.

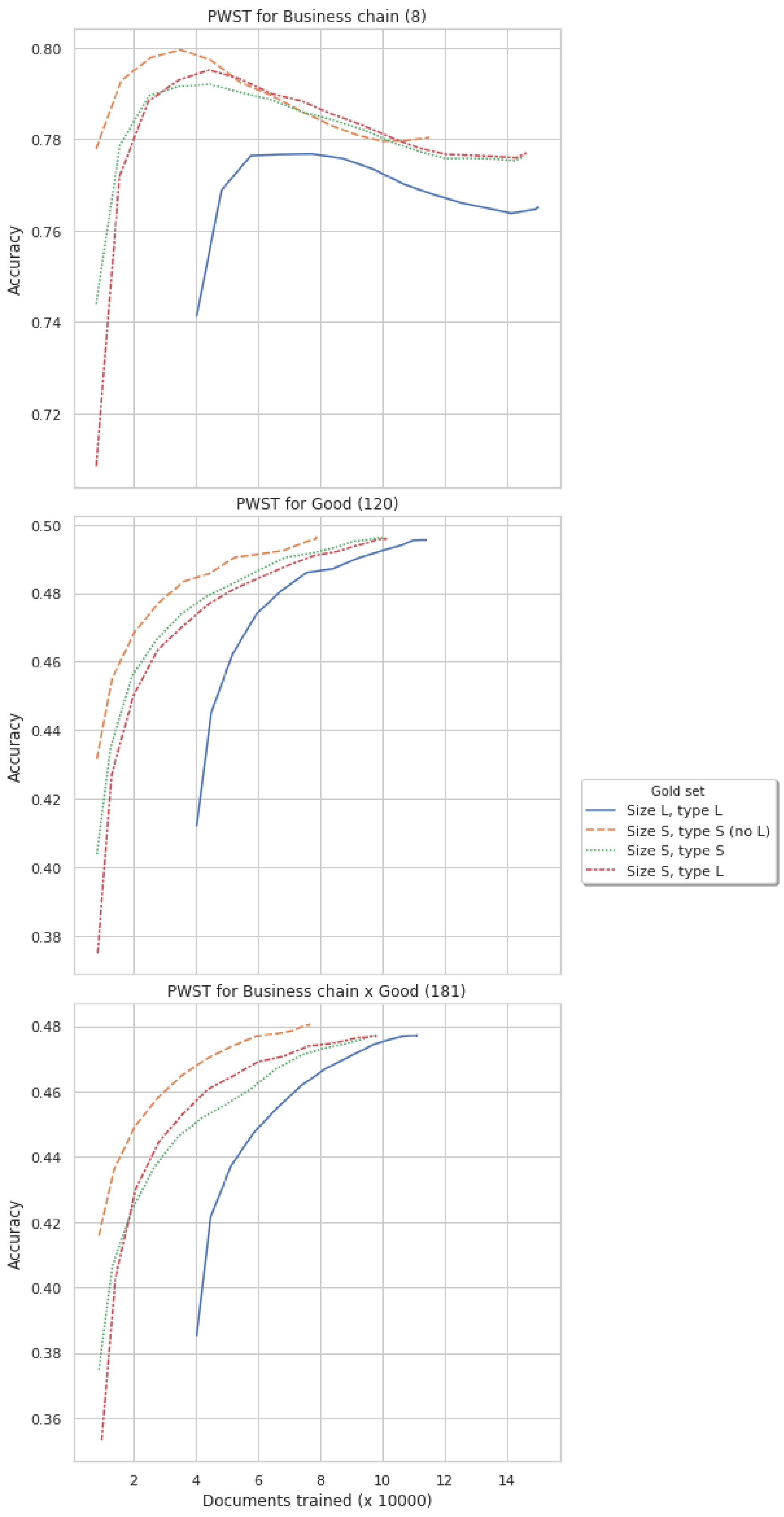

Figure 5.2: Accuracy scores for the proportional weakly self-training (PWST) method for the four gold sets for each of the three classification schemes.

5.2.3. Results

The results show surprisingly that the large-sized gold set performs worse than small-sized gold sets. Specifically, the initial accuracy performance score when training on the 'size L, type L' gold set is low compared to the small-sized gold set scores. Another observation is that the best performing set appears to be the 'size S, type S (no type L)' gold set. This observation is most clear when comparing the initial accuracy performance.

Both observations indicate that besides the size of the gold set, the type of the gold set affects the performance scores. More specifically the gold sets with small enterprises perform better compared to the gold sets with large enterprises. Moreover simply not using large enterprises at all (i.e., gold set 'size S, type S (no type L)') leads to the highest accuracy performance scores. When only the final accuracy scores are observed the difference between the four gold sets seems marginal. However note that the gold set 'size S, type S (no type L)' achieved a similar final accuracy score with far fewer documents trained.

The 'size S, type S (no type L)' gold set has the highest initial performance and has a good final performance and is therefore the best choice as gold set. This result can be explained by the contents of the test set T, which contains only small enterprises. With these results, we cannot hold our hypotheses that data of large enterprises benefit the prediction of small enterprises. Instead, the results suggest that not using the large enterprises at all is the best choice.

5.3. Comparison of top-k scores

5.3.1. Research objective

Until now the experimental results only focused on the top-1 performance scores for each prediction. Enterprises often engage in multiple economic activities. When the main webpage of an enterprise describes multiple economic activities the top prediction of our method could be one of the enterprises' side activities instead of its main activity. The predicted rank of the main activity could then be in the second or lower place. Therefore the current experiment aims at using the top-k predictions of the NB classifier to evaluate the top-k performance scores. The main question to be answered is whether the top-k performance scores are high enough for a semi-automatic classification system.

5.3.2. Experimental set-up

The top-k performance scores for the current experiment are shown for settings that were used in the two previous sections. The settings used are the classification scheme 'Type of business chain x Type of good', the PWST method and the 'size S, type S (no type L)' gold set. Results are shown with learning curves in Figure 5.3 as well as final performance scores in Table 5.2.

5.3.3. Results

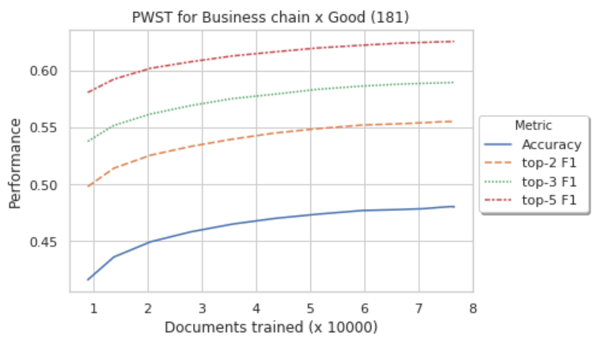

Figure 5.3: Top-k F_1 scores for the proportional weakly self-training (PWST) method, the gold set 'Size S, type S (no type L)' and the 'Type of business chain x Type of good' scheme with 181 classes.

Table 5.2: Final top-k performance scores for the proportional weakly self-training (PWST) method, the gold set 'Size S, type S (no type L)' and the 'Type of business chain x Type of good' scheme with 181 classes.

Level	Metric	Top-1	Top-2	Top-3	Top-5	Top-10
Micro	F1-score	0.48	0.56	0.59	0.63	0.66
	Precision	0.48	0.53	0.55	0.57	0.59
	Recall	0.48	0.58	0.64	0.69	0.76
	Rec. Rank	0.48	0.53	0.55	0.56	-
Macro	F1-score	0.34	0.40	0.44	0.48	0.54
	Precision	0.34	0.40	0.43	0.46	0.51
	Recall	0.35	0.44	0.48	0.53	0.61
	Rec. Rank	0.35	0.39	0.41	0.42	-

Results in Figure 5.3 and Table 5.2 show, as mathematically derived in Lemma 1, the top-k F_1 scores are increasing as a function of k. in Table 5.2, compared to top-k precision, the top-k recall scores are increasing at a faster rate. This observation can be explained by our remark in Section 4.4.4: the number of true positives goes up with the same amount as the number of false negatives goes down. The fractional expression for precision has the true positives in the denominator while the expression for recall has the true positives and the false negatives in the denominator (Equations 4.9 and 4.10). Since the number of false negatives decreases for a higher k value, the recall value increases.

Performance scores in Table 5.2 can be compared with previous research. For instance, our micro-averaged F_1-score of 0.48 is much lower compared to the score of 0.81 obtained by Berardi et al. (2015). Kühnemann (2019) obtained micro-averaged F_1-scores of around 0.5. The higher score of 0.5 can mostly be explained by the smaller classification scheme used (111 classes versus our 181 classes). Our macro-averaged F_1-score of 0.34 is lower compared to the score of 0.49 obtained by Berardi et al.. Our macro-averaged top-5 reciprocal rank score of 0.42 is also lower compared to the score of 0.54 obtained by Berardi et al.. Possible explanations for the higher performance scores by Berardi et al.. are that besides the main webpage also the first 10 subpages were crawled and that exogenous features were used (e.g., using Alexa queries).

The top-k performance scores are successfully evaluated, both mathematically and empirically. As in the first experiment, the micro-averaged scores are higher than the macro-averaged scores. Higher micro-averaged scores can be explained by higher performance scores for large classes and lower performance scores for small classes. For building a semi-automatic classification system a F_1-score of over 0.9 is necessary. Since our top-10 F_1-score is 0.66 we can conclude that the performance of our method is still too low.

5.4. Quantitative analysis

5.4.1. Research objective

In the previous sections, results were compared for different semi-supervised methods and different gold sets. However, the significance of these results was not evaluated. In this section, we aim to evaluate the significance of our experimental results with a quantitative analysis. The analysis should indicate the significance of the experimental results.

5.4.2. Experimental set-up

For the quantitative analysis, standard deviations for the learning curves are calculated by applying 1000 bootstrap resamples to the class predictions and true class labels of the test set T (Section 4.5.1). For the largest classification scheme (i.e., 'Type of business chain x Type of good') error bars are shown for each gold set and each classification method. First each classification method is compared for the best performing gold set (i.e., 'Size S, type S (no type L)') (Figure 5.4). Secondly for the best performing classification method (i.e., PWST) each gold set is compared (Figure 5.5).

5.4.3. Results

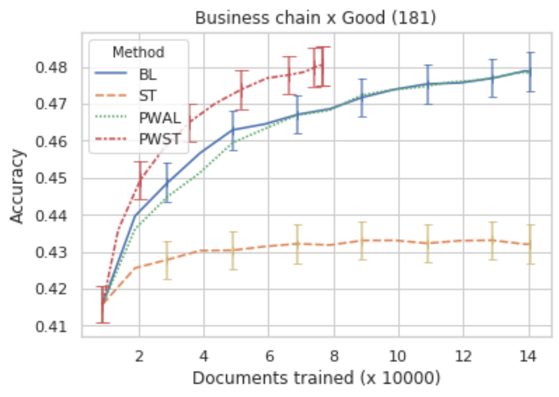

Figure 5.4: Accuracy scores of each method for the gold set 'Size S, type S (no type L)' and the 'Type of business chain x Type of good' scheme with 181 classes. Error bars represent ±2 standard deviations. (Error bars for PWAL are left out)

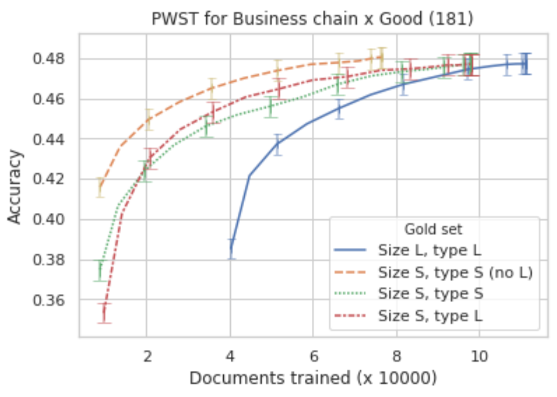

Figure 5.5: Accuracy scores of proportional weakly self-training (PWST) for the four gold sets and the 'Type of business chain x Type of good' scheme with 181 classes. Error bars represent ±2 standard deviations.

To test whether results are different the learning curves with their standard deviations can be used. First, we define θ_i^* as the estimated accuracy score after training of approximately i documents. Then our null hypothesis for evaluating PWST against BL is defined as $H_0 : \hat{\theta}_i^{BL} = \hat{\theta}_i^{PWST}$ and our alternative hypothesis as $H_1 : \hat{\theta}_i^{BL} \neq \hat{\theta}_i^{PWST}$. If $|\hat{\theta}_i^{BL} - \hat{\theta}_i^{PWST}| \geq 2S_{\mathcal{A},i}(A-B)$ the null hypothesis can be rejected at a 0.05 level of significance. Here $S_{\mathcal{A},i}(A-B)$ denotes the standard deviation of the difference of the accuracy scores after the training of i documents for BL and PWST.

ST performs consistently worse than the other methods (Figure 5.4). Classic self-training in itself is therefore not a suitable semi-supervised method for webpage classification. PWAL performs almost similar compared to BL. Since the learning curves mostly overlap the performance of the methods can not be considered significantly different. The practical difference is the order in which training documents are added; BL selects documents randomly while PWAL selects the least confident predicted documents.

PWST performs consistently better than BL. To quantify the difference we calculate the standard deviation of the difference of the accuracy scores according to Equation 4.25 for one measurement of the learning curves. The accuracy measurement closest to $78\,000$ training documents both for BL and PWST is chosen. We define sequence A as the performance scores for BL and sequence B as the performance scores for PWST. Then $S_{\mathcal{A},78000}(A) \approx 0.003$, $S_{\mathcal{A},78000}(B) \approx 0.003$ and $S_{\mathcal{A},78000}(A-B) \approx 0.002$. The standard deviation of the difference is smaller than the standard deviation of the individual sequences because of the high correlation of 0.75 for sequences A and B. The high correlation can be explained by the methods using the NB classifier and the same gold set. Since the instances and its features are shared among the two test sets the predictions of the classifiers are highly correlated and thus have positive covariance. Returning to our hypotheses, since $|\hat{\theta}_{78000}^{BL} - \hat{\theta}_{78000}^{PWST}| = |0.469 - 0.480| = 0.011 \geq 0.004 = 2S_{\mathcal{A},78000}(A-B)$ the null hypothesis can be rejected, thus our PWST method indeed improves upon a supervised method.

5.5. Qualitative analysis for groups of classes

5.5.1. Research objective

This section examines the data set and results for PWST in more detail. The aim is to evaluate the performance of PWST in more depth, i.e., for groups of individual classes. The research question to be answered is whether there are specific groups of classes that are hard to predict by our classification method. For evaluation of the results, we use the accuracy scores on individual class level.

5.5.2. Experimental set-up

The first qualitative analysis is done on groups of individual classes. Specifically, we select four different quarters based on their final performance in the BL method. Quarter 1 consists of all classes with a final accuracy of less than 0.25, quarter 2 of all classes with a final accuracy of 0.25 or higher and less than 0.5, quarter 3 of all classes with a final accuracy of 0.5 or higher and less than 0.75 and quarter 4 consists of all classes with a final accuracy of 0.75 or higher. Our hypothesis with this selection of classes is that high-performing classes will not increase a lot while low-performing classes might significantly increase their performance during the semi-supervised learning, dependent on the documents trained for that class. To examine the accuracy scores for each quarter of classes we use the largest classification scheme (i.e., 'Type of business chain x Type of good'), the best-performing gold set (i.e., 'Size S, type S (no L)') and the PWST method.

5.5.3. Results

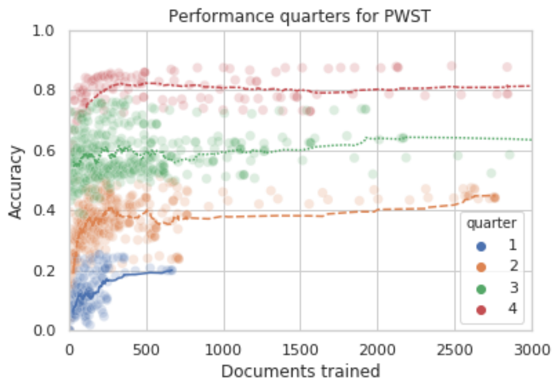

Figure 5.6: Accuracy scores per class of four performance quarters for proportional weakly self-training (PWST) with the gold set 'Size S, type S (no type L)' and the 'Type of business chain x Type of good' scheme with 181 classes.

The results for the accuracy scores of PWST for each quarter of classes are shown in Figure 5.6. The shaded dots can be seen as elements of the learning curve of each class. To prevent cluttering, the learning curves of the 181 individual classes are not shown. Instead, for each quarter the moving averages with a window width of 25 are drawn. The x-axis is cut at 3000 because only a few large classes have that much training documents added.

The results show that for the two best performing quarters the absolute and relative increase in performance is lower compared to the absolute and relative increase in performance for the two worst-performing quarters. Especially when a limited number of documents is trained and when the accuracy is low, the accuracy improves when more training documents are added. The good performance of self-training when a limited amount of initial training data is available is well-known in literature (Rosenberg et al., 2005, Stikic et al., 2008). When for a class a considerable number of documents have been trained our PWST method stabilises in terms of accuracy, i.e., training more documents has little to no effect on the final score.

5.6. Qualitative analysis for individual classes

5.6.1. Research objective

This section examines the data set and results for the semi-supervised methods at the lowest level of detail. The aim is to evaluate the performance of the methods for individual classes. The research question to be answered is whether there are specific types of classes that are hard to predict by our classification methods. For evaluation of the results, we use the accuracy scores on individual class level.

5.6.2. Experimental set-up

To examine the performance for individual classes learning curves for the accuracy scores are shown for some characteristic example classes of all the 181 classes in the largest classification scheme (i.e., ‘Type of business chain x Type of good’). As in the first experiment, the learning curves for each of the four classification methods are compared, but now on individual class level. The experiments use the best-performing gold set (i.e., ‘Size S, type S (no L)’).

5.6.3. Results

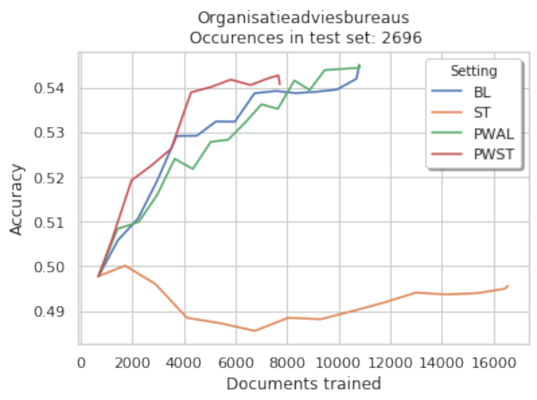

Figure 5.7: Accuracy scores for the class ‘Organisatieadviesbureaus’

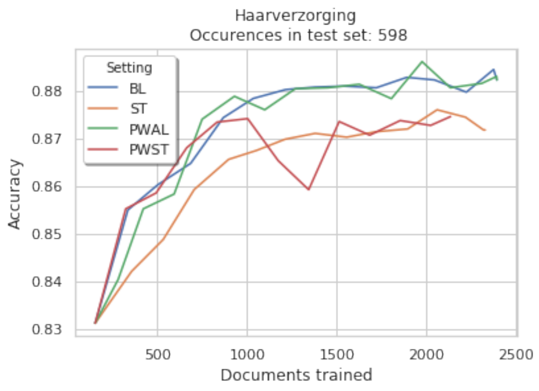

Figure 5.8: Accuracy scores for the class ‘Haarverzorging’

Figures 5.7 and 5.8 show results for the largest class ‘Organisatieadviesbureaus’ and for another large class ‘Haarverzorging’. Similar as in the first experiment, ST performs worse than BL and PWAL. In Figure 5.8 PWST barely outperforms ST and performs worse than BL. The worse performance of PWST can only be explained by the filter step, where documents are neglected if the predicted label does not match the noisy label. Surely without the filter step, the final performance of PWST would be similar as the BL and PWAL final performances. As shown in the previous qualitative analyses PWST seems to stabilize at large accuracy scores. The worse performance of PWST for the ‘Haarverzorging’ class could therefore be explained by the high accuracy scores, which is also the case for other good-performing classes such as ‘Schoonheidsverzorging’. The high accuracy scores for these classes could be due to the high homogeneity of the class documents.

Figure 5.7 shows the accuracy scores for the largest class. ST has self-trained over $16\ 000$ documents while in the noisy set less than $11\ 000$ documents belong to the class. This ‘over-training’ is an example of why classic self-training is prone to error propagation. Our PWST method does not suffer from this type of error propagation. Instead, the error propagation is reduced by the filter step. While the filter step reduces the number of training documents, the final accuracy score of PWST is similar to BL for the largest class ‘Organisatieadviesbureaus’.

Figures 5.9 and 5.10 show results for two smaller classes with somewhat lower performance scores. ST underperforms against BL while PWST outperforms BL. These two characteristic cases show when our PWST method performs best; when the initial accuracy is low and when the initial number of training documents is low.

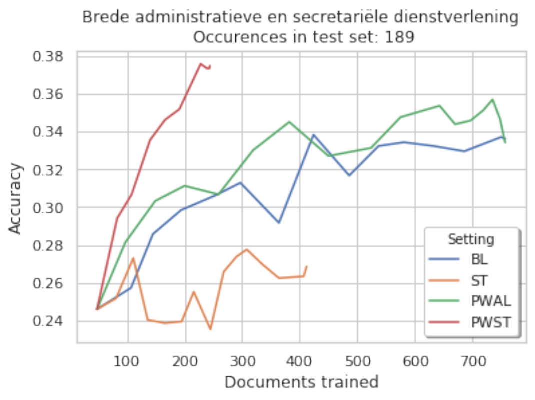

Figure 5.9: Accuracy scores for the class 'Brede administratieve en secretarïële dienstverlening'

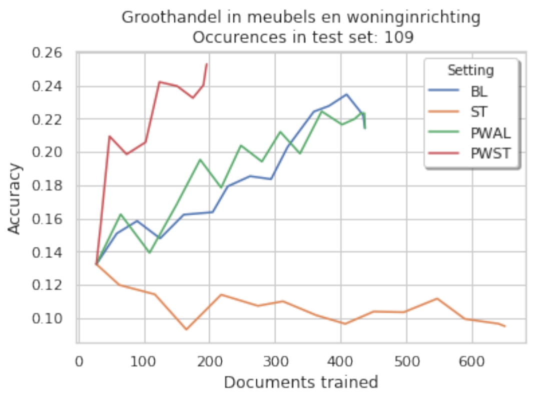

Figure 5.10: Accuracy scores for the class 'Groothandel in meubels en woninginrichting'

6 Discussion

The research objective of the current study was to leverage the abundance of noisy labelled data to increase the size of the training data. To that end, several semi-supervised machine learning methods were evaluated. The main question in the semi-supervised setting was which query strategy to apply for expanding the training data. Other questions that arose were the influence of the size of the gold set, the influence of the type of the gold set and the influence of the used classification scheme. The results indicate that 1) the proposed proportional weakly self-training method improves upon the supervised baseline method in terms of accuracy, 2) the best choice for the gold set is not using the large enterprises at all and 3) the top-k performance scores improve results but are not yet sufficient to apply semi-automatic classification in practice.

The first experiment aimed at evaluating the effect of the query strategies. The improvement of the proposed PWST method upon the supervised baseline indicates that the query strategy of high-probability sampling was successful. More specifically the heuristic filtering strategy to only add instances that agree with the prediction was successful. PWST improved in terms of accuracy while requiring far less training instances, suggesting that part of the available noisy data is not useful as training data.

The second experiment compares different types and sizes of the gold set for PWST. The best choice for the gold set to not use the large enterprises at all was unexpected. We hypothesised that the documents of large enterprises would help increase performance for predicting small enterprises since the documents of large enterprises are labelled more reliably. An explanation for the false hypothesis could be that the features on webpages of large enterprises differ from those on webpages of small enterprises. Then the model could learn wrong feature weights and thus make wrong predictions.

The third experiment aimed at evaluating top-k performance scores to assess whether these are high enough for a semi-automatic classification system. A top-k F_1-score was first defined and shown to be a monotonically increasing function of k. The experimental results indeed show the top-k scores are improving upon top-1 scores with a top-10 F_1 score of 0.66 for the classification scheme with 181 classes. However, for building a semi-automatic classification system a F_1-score of over 0.9 is necessary. In the experiment, the micro-averaged scores are higher than the macro-averaged scores which can be explained by higher performance scores for large classes and lower performance scores for small classes. A limitation of the experiment is that the top-k scores were not integrated into the iterative training algorithm. When the algorithm optimises the top-k scores these scores could increase.

The qualitative analyses aimed at evaluating the performance of classes in more detail. The finding that PWST performs well when for a class a limited amount of initial training data is available is in line with the literature (Rosenberg et al., 2005, Stikic et al., 2008). Another finding of the qualitative analyses is that classic semi-supervised self-training is prone to error propagation while PWST reduces error propagation through the filter step. The filter step aims to prevent mislabelled documents from entering the training set. Compared to the supervised baseline PWST performs best when the initial accuracy is low and when the initial number of training documents is low. However when for a class a considerable number of documents have been trained the training of more documents has little to no effect on the final performance score, even when this score lies below 0.5.

The final performance results are comparable to previous research on web-based prediction of economic activity. Roelands (2017) used enterprises' websites and text mining to predict 29 categories of economic activity with a NB classifier, obtaining a maximum accuracy (micro-averaged F_1) score of 0.42. Kühnemann (2019) obtained higher maximum accuracy scores of around $.5$ when predicting 111 categories of economic activities based on webpage texts. Even higher accuracy scores of 0.81 were obtained by Berardi et al. (2015) when predicting 216 categories using enterprises' websites. The high micro-averaged F_1 scores obtained by Berardi et al. could be explained by severe class imbalance indicated by their lower macro-averaged F_1 score of 0.50 and macro-averaged top-5 reciprocal rank of 0.54. Other explanations are that more complete input data was used, for example, the first 10 subpages were crawled and exogenous features were used (e.g., results of Alexa queries).

When using text mining to predict the economic activity of enterprises one should take into account the size of the enterprise. Roelands already showed that one-man-enterprises can be predicted much easier than others. Our findings add that for predicting the economic activity of small enterprises the webpages of large enterprises are less suitable. Indeed, the use of large enterprises' webpages reduced the performance scores on the test set.

The final performance scores for the classification scheme with 181 classes of $.48$ for the F_1-score and $.66$ for the top-10 F_1-score are insufficient for building a semi-automatic classification system. Higher performance scores are necessary to be able to evaluate and edit the NACE codes of small enterprises in the GBR.

Concerning the semi-supervised learning approach, there are several points for improvement. First, the availability of an accurate gold set and test set is important. While PWST is less prone to error propagation than semi-supervised self-training, PWST still heavily depends on the accuracy of the initial gold training set. Secondly, the (initial) performance of PWST could be improved by reducing the batch-size at the cost of increasing the run-time of the algorithm. Thirdly, PWST is a heuristic semi-supervised approach. Other semi-supervised approaches could be to use an ensemble of classifiers (Didaci and Roli, 2006) or use the more general co-training technique Blum and Mitchell (1998). The most promising approach is using semi-supervised neural networks, which have been shown to consistently outperform their supervised counterparts (van Engelen and Hoos, 2019) and have already been applied to website classification (Du et al., 2018). Neural networks can additionally handle noisy labels and can be incorporated with the hierarchical NACE system because of their hierarchical nature.

Concerning the noisy labels in the GBR, it would be worthwhile to inspect the errors in the NACE labels. For instance, the error rate of the five-digit NACE labels could give an indication of the maximum classification performance. Additionally, if error rates for specific classes or industries are high actions could be taken to manually improve the labels. Moreover, actions could be taken automatically by using error detection or even error correction methods. The filter step in PWST can be seen as a heuristic semi-supervised error detection method. Further examination of error detection methods such as data cleaning and label noise-tolerant methods as specified by Frénay and Verleysen (2013) could improve classification performance.

Concerning the text mining pipeline, the most important part is the features extracted from an enterprise's website. We expect the classification performance can be improved by extracting additional features from websites such as the content of meta tags, the images, the content of subpages or exogenous features from search engines.

Our results provide important insights into handling text classification problems with noisy labels. However, the web-based classification of economic activity remains a challenging research objective. The objective could be achieved by promising future work such as further examination of error detection methods or the further development of (semi-supervised) neural networks.

References

G. Attardi, M. Simi, and A. Zanelli. (2012). Domain adaptation by active learning. In *International Workshop on Evaluation of Natural Language and Speech Tool for Italian*, pages 77–85. Springer. URL `https://doi.org/10.1007/978-3-642-35828-9_9`.

G. Berardi, A. Esuli, T. Fagni, and F. Sebastiani. (2015). Classifying websites by industry sector: A study in feature design. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, pages 1053–1059, New York, NY, USA. ACM. ISBN 978-1-4503-3196-8. URL `https://doi.acm.org/10.1145/2695664.2695722`.

A. Blum and T. Mitchell. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, pages 92–100, New York, NY, USA. ACM. ISBN 1-58113-057-0. URL `https://doi.acm.org/10.1145/279943.279962`.

J. Burger, A. van Delden, and S. Scholtus. (2015). Sensitivity of mixed-source statistics to classification errors. *Journal of Official Statistics*, 31(3):489–506. URL `https://doi.org/10.1515/jos-2015-0029`.

G. Caterini. (2018). Classifying Firms with Text Mining. DEM Working Papers 2018/09, Department of Economics and Management. URL `https://ideas.repec.org/p/trn/utwprg/2018-09.html`.

CBS. (2020). Statline - bedrijven; bedrijfstak. Retrieved January 20, 2020 from `https://opendata.cbs.nl/statline/#/CBS/nl/dataset/81589NED/table?ts=1579515727489`.

O. Chapelle, B. Schölkopf, and A. Zien, editors. (2006). *Semi-Supervised Learning*. MIT Press, Cambridge, MA. URL `https://doi.org/10.1109/TNN.2009.2015974`.

J. L. Christensen. (2008). Questioning the precision of statistical classification of industries. In *DRUID Conference on Entrepreneurship and Innovation*, pages 17–20.

M. Collins. (1999). Unsupervised models for named entity classification. *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural LanguageProcessing and Very Large Corpora*. URL `https://ci.nii.ac.jp/naid/10020680126/en/`.

S. Dasgupta. (2001). Pac generalization bounds for co-training. *Proc. Neural Information Processing Systems, 2001*. URL `https://ci.nii.ac.jp/naid/10020991979/en/`.

L. Didaci and F. Roli. (2006). Using co-training and self-training in semi-supervised multiple classifier systems. In D.-Y. Yeung, J. T. Kwok, A. Fred, F. Roli, and D. de Ridder, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, pages 522–530, Berlin, Heidelberg. Springer Berlin Heidelberg. ISBN 978-3-540-37241-7. URL `https://doi.org/10.1007/11815921_57`.

M. Du, Y. Han, and L. Zhao. (2018). A heuristic approach for website classification with mixed feature extractors. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 134–141. URL `https://doi.org/10.1109/PADSW.2018.8645042`.

S. Dumais, J. Platt, D. Heckerman, and M. Sahami. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of the Seventh International Conference on Information and Knowledge Management*, CIKM '98, page 148–155, New York, NY, USA. Association for Computing Machinery. ISBN 1581130619. URL `https://doi.org/10.1145/288627.288651`.

B. Frénay and M. Verleysen. (2013). Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869. URL `https://doi.org/10.1109/TNNLS.2013.2292894`.

B. Frénay, A. Kabán, et al. (2014). A comprehensive introduction to label noise. In *ESANN*. URL `https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2014-10.pdf`.

J. Hernández-González, I. Inza, and J. A. Lozano. (2016). Weak supervision and other non-standard classification problems: a taxonomy. *Pattern Recognition Letters*, 69:49–55. URL `https://doi.org/10.1016/j.patrec.2015.10.008`.

H. Kühnemann. (2019). Web-based text mining approaches to the classification of economic activity. Master's thesis, Utrecht University. Unpublished, available upon request.

M. Li and Z.-H. Zhou. (2005). Setred: Self-training with editing. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 611–621. Springer. URL `https://doi.org/10.1007/11430919_71`.

C. D. Manning, P. Raghavan, and H. Schütze. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA. ISBN 0521865719, 9780521865715. URL `https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf`.

D. McClosky, E. Charniak, and M. Johnson. (2006). Effective self-training for parsing. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 152–159, Stroudsburg, PA, USA. Association for Computational Linguistics. URL `https://doi.org/10.3115/1220835.1220855`.

R. Mihalcea. (2004). Co-training and self-training for word sense disambiguation. *Proceedings of CoNLL 2004*. URL `https://acl-arc.comp.nus.edu.sg//~antho/W/W04/W04-2405.pdf`.

N. M. Müller and K. Markert. (2019). Identifying mislabeled instances in classification datasets. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. URL `https://doi.org/10.1109/IJCNN.2019.8851920`.

F. Olsson. (2009). A literature survey of active machine learning in the context of natural language processing. Technical report, Swedish Institute of Computer Science. URL `http://soda.swedishict.se/3600/`.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830. URL `https://arxiv.org/abs/1201.0490`.

N. N. Pise and P. Kulkarni. (2008). A survey of semi-supervised learning methods. In *2008 International Conference on Computational Intelligence and Security*, volume 2, pages 30–34. IEEE. URL `https://doi.org/10.1109/CIS.2008.204`.

J. Pomikálek. (2011). *Removing Boilerplate and Duplicate Content from Web Corpora*. Doctoral theses, dissertations, Masaryk University, Faculty of Informatics, Brno. URL `https://is.muni.cz/th/o6om2/`.

M. Roelands. (2017). Classifying economic activity with business websites using text mining techniques. Master's thesis, Tilburg University. URL `https://tilburguniversity.on.worldcat.org/oclc/1029112723`.

C. Rosenberg, M. Hebert, and H. Schneiderman. (2005). Semi-supervised self-training of object detection models. In *Seventh IEEE Workshop on Applications of Computer Vision*, pages 29–36. doi: 10.1184/R1/6560834.v1. URL `https://kilthub.cmu.edu/articles/Semi-Supervised_Self-Training_of_Object_Detection_Models/6560834`.

B. Settles. (2009). Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison. URL `https://digital.library.wisc.edu/1793/60660`.

N. Shuyo. (2010). Language detection library for java. URL `https://code.google.com/p/language-detection/`.

M. Stikic, K. Van Laerhoven, and B. Schiele. (2008). Exploring semi-supervised and active learning for activity recognition. In *2008 12th IEEE International Symposium on Wearable Computers*, pages 81–88. URL https://doi.org/10.1109/ISWC.2008.4911590.

J. Tanha, M. van Someren, and H. Afsarmanesh. (2017). Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, 8(1):355–370. URL https://doi.org/10.1007/s13042-015-0328-7.

O. ten Bosch, A. Delden, and D. Windmeijer. (2019). Searching for business websites. Retrieved January 20, 2020 from https://www.cbs.nl/en-gb/background/2020/01/searching-for-business-websites.

I. Triguero, J. A. Sáez, J. Luengo, S. García, and F. Herrera. (2014). On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification. *Neurocomput.*, 132:30–41. ISSN 0925-2312. URL https://doi.org/10.1016/j.neucom.2013.05.055.

V. van Asch. (2013). Macro-and micro-averaged evaluation measures. *Tech. Rep.* URL https://pdfs.semanticscholar.org/1d10/6a2730801b6210a67f7622e4d192bb309303.pdf.

A. van Delden, S. Scholtus, and J. Burger. (2016). Accuracy of mixed-source statistics as affected by classification errors. *Journal of Official Statistics*, 32(3):619–642. URL https://doi.org/10.1515/jos-2016-0032.

J. E. van Engelen and H. H. Hoos. (2019). A survey on semi-supervised learning. *Machine Learning*, pages 1–68. URL https://doi.org/10.1007/s10994-019-05855-6.

H. Zhang. (2004). The optimality of naive bayes. *Proceedings of the the 17th International FLAIRS conference (FLAIRS2004)*. URL https://www.cs.unb.ca/profs/hzhang/publications/FLAIRS04ZhangH.pdf.

Z.-H. Zhou. (2017). A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53. ISSN 2095-5138. URL https://doi.org/10.1093/nsr/nwx106.

X. Zhu. (2008). Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison. URL http://pages.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf.