ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

School of Information Sciences and Technology
Department of Informatics
Athens, Greece

Master Thesis
in
Computer Science

Examining how teacher-student approaches can benefit few-shot learning for toxicity detection tasks

Natalia Avramidou

Supervisors: Ion Androutsopoulos
John Pavlopoulos

December 2022

Natalia Avramidou

Examining how teacher-student approaches can bene t few-shot learning for toxicity detection tasks

December 2022

Supervisors: Ion Androutsopoulos, John Pavlopoulos

Athens University of Economics and Business

School of Information Sciences and Technology

Department of Informatics

Natural Language Processing Group, Information Processing Laboratory

Athens, Greece

Abstract

In recent years, the evolution of social media platforms has introduced the need for systems that detect the toxic behavior of users. A Toxicity Detection system tries to detect user posts that are offensive and abusive. Abusive content can take many forms, including threatening, harassment, intimidation, or discrimination over a person or a group. Bias against users based on race, color, national origin, sex, sexual orientation, religion or other characteristics frequently occurs online. The field of Natural Language Processing (NLP) contributes to detecting this hateful content by automating it with classification models that categorize user posts as offensive or not offensive. In addition, classifiers can label abusive content according to the nature of the insult (e.g., racist, sexist etc.). As annotating thousands of examples for training (NLP) models is expensive and time-consuming, it is a significant challenge to train a model effectively with the least amount of labeled data. Although there is a lack of fully annotated datasets for many different tasks, usually, there is a much larger pool of task-specific unlabeled instances that could be used to improve a system's performance. In this thesis, we will focus on: toxicity detection in Greek tweets and sexism detection in English posts crawled from platforms such as Reddit and Gab. For instance, a system that tries to detect hate speech could benefit from the abundant online unlabeled data as there is a much smaller number of labeled instances in toxicity datasets.

There are many methods explored in literature for few-shot learning scenarios. Self-training is a semi-supervised method where a Teacher model is initially trained on the few available labeled instances. Subsequently, it generates silver labels for the bigger pool of task-specific unlabeled data. In each round, it samples a number of silver-labeled examples, in most cases, based on the model's confidence. These examples and their silver labels act as additional supervision to train a stronger Student model iteratively. In a large pool of unlabeled instances, not all are equally useful for learning a specific task. Active Learning tries to maximize the system's performance gain by identifying the most informative examples to be labeled by a human annotator or, in our case, to be selected among those silver-labeled by the Teacher. In this thesis, we focus on applying the Teacher-Student approach to detect toxic and sexist content when the initial training examples are limited. We experiment with different machine-learning models and apply various sampling techniques to augment

our initial dataset. We also employ Active Learning criteria in the Self-training algorithm to examine if they could further benefit our system.

Περίληψη

Τα τελευταία χρόνια, η εξέλιξη των πλατφορμών κοινωνικής δικτύωσης έχει δημιουργήσει την ανάγκη ύπαρξης συστημάτων που ανιχνεύουν την τοξική συμπεριφορά χρηστών των εν λόγω πλατφορμών. Ένα σύστημα Εντοπισμού Τοξικότητας προσπαθεί να εντοπίσει προσβλητικές και υβριστικές δημοσιεύσεις χρηστών. Το online περιεχόμενο θεωρείται τοξικό όταν περιέχει απειλές, εκφοβισμό, σχόλια που έχουν στόχο την παρενόχληση ή όταν αναπαράγει στερεότητα και διακρίσεις εις βάρος ενός ατόμου ή μίας ομάδας ατόμων. Η προκατάληψη με βάση τη φυλή, το χρώμα, την εθνικότητα, το φύλο, τον σεξουαλικό προσανατολισμό, τη θρησκεία και άλλα χαρακτηριστικά παρατηρείται συχνά στις δημόσιες αναρτήσεις. Ο τομέας της Επεξεργασίας Φυσικής Γλώσσας (ΕΦΓ) συμβάλει στην ανίχνευση κακοποιητικού περιεχομένου, αυτοματοποιώντας τη διαδικασία με μοντέλα ταξινόμησης που κατηγοριοποιούν τις αναρτήσεις των χρηστών σε προσβλητικές ή μη προσβλητικές. Επιπλέον, οι ταξινομητές μπορούν να επισημάνουν το περιεχόμενο μιας τοξικής ανάρτησης με βάση τη φύση της προσβολής (ρατσιστική, σεξιστική κ.λπ.). Καθώς η επισημείωση χιλιάδων παραδειγμάτων για την εκπαίδευση των (ΕΦΓ) μοντέλων είναι μια ακριβή και χρονοβόρα διαδικασία, η αποδοτική εκπαίδευση τους με τον ελάχιστο αριθμό δεδομένων με ετικέτα πρόκειται για μία σημαντική πρόκληση. Παρά την έλλειψη επαρκών επισημειωμένων βάσεων δεδομένων για πολλές διαφορετικές εργασίες, συνήθως, υπάρχει ένα πολύ μεγαλύτερο σύνολο δεδομένων χωρίς ετικέτα που σχετίζονται με την κάθε εργασία. Τα εν λόγω δεδομένα θα μπορούσαν να χρησιμοποιηθούν για τη βελτίωση της απόδοσης ενός συστήματος ταξινόμησης. Σε αυτή τη διπλωματική θα επικεντρωθούμε σε δύο εργασίες: τον εντοπισμό τοξικότητας σε Ελληνικά tweets και τον εντοπισμό σεξιστικής συμπεριφοράς σε αναρτήσεις στα Αγγλικά που έχουν συλλεχθεί από πλατφόρμες όπως το Reddit και το Gab. Ενδεικτικά, ένα σύστημα που προσπαθεί να ανιχνεύσει περιεχόμενο που περιέχει ρητορική μίσους θα μπορούσε να επωφεληθεί από τα άφθονα δεδομένα χωρίς ετικέτα που είναι διαθέσιμα online, καθώς ένας πολύ μικρότερος αριθμός δεδομένων με ετικέτα είναι διαθέσιμος για τη συγκεκριμένη εργασία.

Πολλές μέθοδοι έχουν διερευνηθεί στη σχετική βιβλιογραφία για σενάρια μάθησης με περιορισμένα δεδομένα εκπαίδευσης. Η Αυτό-εκπαίδευση είναι μία μέθοδος ημι-επιβλεπόμενης μάθησης κατά την οποία ένα μοντέλο Καθηγητής εκπαιδεύεται αρχικά στα λίγα δι-

αθέσιμα δεδομένα με ετικέτα. Στη συνέχεια, δημιουργεί ψευδο-ετικέτες για ένα πολύ μεγαλύτερο σύνολο μη επισημειωμένων δεδομένων από τον ίδιο τομέα με τα επισημειωμένα δεδομένα. Σε κάθε γύρο επιλέγεται ένας αριθμός δεδομένων με ψευδο-ετικέτα, στις περισσότερες περιπτώσεις με κριτήριο τη πιθανότητα που δίνει το μοντέλο Καθηγητής να είναι σωστή η πρόβλεψή του, για να χρησιμοποιηθούν ως επιπρόσθετη επίβλεψη στην εκπαίδευση ενός ισχυρότερου μοντέλου Μαθητή. Σε ένα τεράστιο σύνολο παραδειγμάτων χωρίς ετικέτα, δεν είναι όλα εξίσου χρήσιμα ώστε να μάθει μία εργασία ο ταξινομητής. Η Διαδραστική Μάθηση προσπαθεί να μεγιστοποιήσει την απόδοση ενός συστήματος προσδιορίζοντας τα μη-επισημειωμένα δεδομένα που περιέχουν την πιο χρήσιμη πληροφορία για την εκμάθηση της συγκεκριμένης εργασίας. Τα δεδομένα αυτά συνήθως επισημαίνονται από έναν άνθρωπο μεσολαβητή ή, στην περίπτωση της δικής μας μελέτης, επιλέγονται ανάμεσα στα δεδομένα για τα οποία έχει δημιουργήσει ψευδο-ετικέτες το μοντέλο Καθηγητής. Σε αυτή την εργασία, εστιάσαμε στην εφαρμογή της προσέγγισης Καθηγητή-Μαθητή για την ανίχνευση τοξικού και σεξιστικού περιεχόμενου όταν τα παραδείγματα εκπαίδευσης είναι περιορισμένα. Για τα πειράματά μας χρησιμοποιήσαμε διαφορετικά μοντέλα μηχανικής μάθησης και εφαρμόσαμε διαφορετικές τεχνικές δειγματοληψίας για την επαύξηση του αρχικού συνόλου εκπαίδευσης. Ακόμα εφαρμόσαμε κριτήρια εμπνευσμένα από την Διαδραστική Μάθηση στον αλγόριθμο Αυτό-εκπαίδευσης για να εξετάσουμε αν μπορούν να αυξήσουν την απόδοση του συστήματός μας.

Acknowledgements

I would like to sincerely thank my supervisors Ion Androutsopoulos and John Pavlopoulos for their guidance, continuous support and beneficial comments throughout the work of this thesis.

Contents

1 Introduction

1.1 Toxicity Detection

In recent years, as social media platforms are evolving rapidly with billions of users worldwide, offensive posts and comments are increasingly common in online discussions. Social media content can be perceived as toxic when it contains insults, threats, intimidation, identity attacks and, in general, aims to abuse individuals or groups of people. Hate speech and xenophobia are increasingly prevalent despite living in the age of globalization. Users that feel superior to others because of specific characteristics such as color, race, gender, religion and sexual orientation often target and harass other users with the power of online anonymity.

A specific category of hate speech is sexist behavior. It concerns prejudice and discrimination that targets mostly women but also homosexual, transgender or non-binary individuals. As mentioned in Combating Sexist Hate Speech, a report of the Council of Europe, "the aim of sexist hate speech is to humiliate or objectify women, to undervalue their skills and opinions, to destroy their reputation, to make them feel vulnerable and fearful, and to control and punish them for not following a certain behavior" [Chi+20].

Toxicity Detection systems aim to automate the process of offensive content detection to facilitate human moderators to ban this content and to ensure healthy online conversations. Using Natural Language Processing (NLP), a sub-field of Artificial Intelligence, to build efficient detection systems has very promising results. Hateful content is a complex concept that is hard to define and can take various forms. Attacks and threats can be made directly or indirectly, and the meaning of specific words can change when placed in a different context. Hence it is difficult to specify handcrafted rules and conditions for rule-based systems to identify such offensive content. For instance, a rule that classifies user comments as toxic if they contain specific words or patterns generally considered insulting can be misleading if the comment's context is not considered. On the contrary, online content can be abusive without consisting of affronting words. Thus it is often necessary to develop and train more complex models to detect such cases.

1.2 Few-shot learning and pretrained models

Deep learning models can learn different concepts for several NLP and computer vision tasks, including text or image classification, intent recognition, sequence generation and information extraction. Such remarkable performance is usually the result of high-quality supervision as models are trained with thousands of examples available for a specific task. As deep learning methods rely on large annotated datasets, a challenge encountered in many implementations is the absence of labeled data. The cost of labeling an efficient amount of instances is very high: it takes a great deal of human effort, is time-consuming or even requires some scientific expertise. For example, in the digital pathology domain, exhaustively annotating large image datasets to train a histology image classifier is challenging when medical specialists are a scarce resource [Sha+20]. Hence, few-shot learning, the ability to learn a task with the least amount of labeled task-specific data, is increasingly important and valuable for many practical applications in low-resource scenarios.

A lot of research has been conducted by NLP scientists to train models with compatible performance to those with fully supervised learning when training examples are limited. In recent years pre-trained language models (LMs) have been widely used in few-shot learning scenarios as studies have proved that they are efficient few-shot or zero-shot learners [Bro+20]. In the NLP field, the task-agnostic knowledge that is acquired through pretraining over a large general corpus seems to help state-of-the-art LMs to achieve high few-shot performance. Intuitively, it provides a strong baseline that significantly improves all the tasks. The most common approach when training examples for end-tasks are limited is to use pre-trained transformers [Dev+19], possibly pre-trained on a domain-specific corpus [Lee+19], and further train them on the limited examples of the end-tasks with a low learning rate (fine-tuning).

Related research has been conducted on prompting LMs by reformulating tasks as natural language "prompts" and conditioning on those prompts. By transforming a classification task as a language modeling problem, LMs can accomplish great results after being fine-tuned on a few annotated examples or even without any training process [Che+21]. Specifically, [Bro+20] show that this method achieves remarkable performance in large-scale LMs (GPT-3, 175B parameters) on many NLP datasets without any gradient updates by incorporating task-specific prompts [Log+22; Bro+20; Che+21]. However, it is hard to use large-scale LMs in many real-world applications where computational resources are limited. Corresponding methods have been proposed by NLP researchers to use smaller LMs like BERT and ROBERTa that are resource efficient for few-shot learning. They can achieve consistent performance to GPT-3 by conditioning on task-specific prompts after being fine-tuned on a few annotated samples [Che+21]. But even though using smaller LMs is more efficient, manually-written prompts require exhaustive tuning using large

validation sets, as model accuracy depends on prompts engineering and is influenced by simple prompt modifications [Log+22].

Another common approach for learning low-resource tasks is to create artificial examples by transforming the original training dataset. Data augmentation (DA) techniques are widely used to increase training examples' diversity and to prevent the model from overfitting without collecting new data [PMA22]. In NLP, many DA methods are applied to enhance the system's robustness even when models with high capacity, such as large-scale pre-trained LMs, are used. Most common DA techniques are based on word replacement. Specifically, the augmented examples are created by replacing some tokens from the original sequences with related words crawled from a thesaurus or with tokens that have similar embeddings. Despite the benefits of such methods, [KCC20] note that researchers should be cautious about preserving class labels when replacing words that could change the meaning of an instance.

1.3 Proposed Approach

Labeled data that can be used for supervised learning is not always available in practice, but there is usually a much larger pool of task-specific unlabeled data. Intuitively, unlabeled in-domain instances also carry valuable information about the task that the model could benefit from. Hence, it could be beneficial to apply a learning method that utilizes this large set of available data to improve the system's performance [Che+21]. For instance, a toxicity detection system trained on a few annotated examples from toxicity datasets could significantly profit from thousands of user posts that are available online and can be crawled from social network platforms.

Semi-supervised learning techniques are adapted to settings where few labeled instances are available [Mi+21]. In this thesis, we implement Teacher-Student, a common semi-supervised approach that exploits the large pool of unlabeled data for few-resource tasks. It employs a Teacher model originally trained on a few annotated data, which creates silver labels for the unlabeled dataset [Mi+21]. A portion of the silver-labeled data augments the training set to further train a more robust Student model. This process is repeated iteratively, and at the end of each round, the Student becomes the Teacher. This process is executed for a fixed number of iterations or until the system reaches convergence.

In the related bibliography, Active Learning is often reviewed to deal with few-resource challenges [Aro07]. Active Learning is the process of querying a human annotator to create labels for a portion of unlabeled examples that carry the richest information about the task. These instances are extracted from the unlabeled data pool according to different

criteria. Subsequently, the sampled instances, along with their true labels (created by a human specialist), are added to the training set to improve the model.

The purpose of this thesis is to explore the benefits of Self-training for few-shot learning. We simulate few-resource scenarios by only considering a few samples per class for the two tasks we focus on: Offensive/Not Offensive and Sexist/Not Sexist. We consider the remaining data as unlabeled for the purpose of the experiment. We employ Self-training in a machine learning model (a Logistic Regression classifier) and test different sampling techniques to expand our training data. These techniques are based on methods applied to Self-training in related work by NLP researchers and on Active Learning criteria found in the related bibliography. We present these methods in a detailed manner in Chapters 2 and 3. Finally, we test the best sampling techniques on a deep learning model, specifically a BERT-based model, to explore whether our methods can be beneficial when high-capacity models are used.

1.4 Main Findings

The experiments we conducted in this thesis proved that the Teacher-Student approach could improve the performance of both a Logistic Regression classifier and a state-of-the-art BERT-based classifier when the models are trained with limited labeled examples. The pseudo-labeled instance confidence score (Section 3.2) is the most important factor when sampling the silver-labeled instances that will augment the initial labeled set in each round of the Teacher-Student process. The sampling techniques that did not consider the instance confidence score failed to improve the system's robustness or even degraded its performance (compared to the few-shot learning scenario without the Teacher-Student application).

When confidence sampling is applied, the probability over a threshold is a more efficient sampling technique than the top-k technique as it achieves equal or higher scores with fewer resources.

Augmenting the training set with the k most confident examples in imbalanced datasets may improve only the evaluation of the majority class. In these cases, it is important to experiment with the class ratio of the most confident examples to be added to the labeled set. Alternatively, tuning the number of Teacher-Student rounds based on the score of the minority category or on macro-averaged scores is crucial to ensure that the evaluation of both classes is improved.

Active Learning criteria could benefit the silver-labeled instance sampling when combined with the confidence sampling technique (Section 3.6). Although this method was not as

beneficial as confidence sampling in the case of the Logistic Regression Teacher-Student framework, for the BERT Teacher-Student framework, it was the most resource-efficient sampling technique. In the case of the Sexism detection task, it reached the highest results of all the sampling methods.

Finally, compared to full-supervision (where true labels for a bigger number of examples are available), the Teacher-Student approach obtained higher scores in some of the evaluation metrics we used in the case of the Logistic Regression classifier. On the contrary, a BERT classifier trained under full supervision outperformed the BERT Teacher-Student framework. However, Teacher-Student obtained promising results with the least amount of training examples.

1.5 Thesis Structure

Chapter 2

Provides background information about models and methods used and presents previous work related to the topic of this thesis.

Chapter 3

Analyzes the proposed methods.

Chapter 4

Provides statistics regarding the dataset, the evaluation measures and the experimental results.

Chapter 5

Contains the conclusions and possible future work.

2 Background and Related Work

In this chapter, we provide background information regarding the models we used for our experiments: a Logistic Regression classifier and a pre-trained BERT Model. We concentrate on previous related work along four dimensions. First, we examine previous approaches for the task of abusive content/sexism detection. Second, we focus on Teacher-Student approaches and sampling techniques. Third, we present DA techniques combined with Self-training for few-shot scenarios. Finally, we describe Active Learning approaches.

2.1 Logistic Regression

In this thesis, we implemented a toxicity detection system that tries to predict for every user post if it belongs to the Offensive/Sexist class. Otherwise, it classifies the instance as Not Offensive/Not Sexist. We can interpret this task as a binary classification problem where we want to calculate the value of a variable Y that corresponds to the class label given an input text x. We will use the convention that the binary variable $Y \quad \{0, 1\}$, where 1 corresponds to the Offensive/Sexist class.

The first classifier we employed in our experiments is a Logistic Regression classifier. It is a well-known supervised linear classifier that takes as input a feature vector $x = (x_1, x_2, ..., x_k)$ of k dimensions. Text features can be represented by boolean vectors, term frequency (TF) vectors etc. At the training phase, the algorithm uses stochastic gradient ascent to update the coefficients (θ) of the linear function to maximize the (conditional) log-likelihood of the training examples. The model uses the sigmoid function to calculate each input vector's probability of belonging to each class. The classifier assigns a label to the input based on the predicted probability of each class.

$$Pr(1|x\theta) = \frac{1}{1 + exp(-x\theta)} \tag{2.1}$$

$$Pr(0|x\theta) = 1 - Pr(+1|x\theta) = \frac{exp(-x\theta)}{1 - exp(-x\theta)} \tag{2.2}$$

More information about the Logistic Regression classifier can be found in [Jan05].

2.2 Pre-trained Language Models (LMs)

In recent years the role of LMs has evolved from generating or evaluating the fluency of natural text to being a powerful tool for text understanding [Jia+21]. The shortage of large datasets with sufficient supervised data has been a significant challenge for many NLP tasks. This challenge led to the emergence of pre-trained LMs. They are deep learning models pre-trained on tasks that do not require human supervision (manual annotations), typically predicting masked words based on the surrounding context. As these tasks do not require human effort, LMs are usually trained on huge general corpora, such as documents from Wikipedia pages.

The objective of pretraining is to learn universal language representations that can benefit specific downstream tasks. Typically these models consist of millions or even billions of parameters learned while pretraining on abundant unsupervised data. The concept of transfer learning is applied, where the task-agnostic knowledge that the model has acquired is used as a starting point for a second specific task, and there is no need to train models from scratch. The most common approach using LMs is fine-tuning. A task-specific head, a fully connected layer or a Multi-layer Perceptron (MLP), is attached on top of the LM, and the model is fine-tuned (further trained), usually with a low learning rate. During this process, typically, the weights of the LM and of the task-specific head are updated. In other approaches, the LM layers are kept frozen, or only some of its top layers unfreeze.

Transformers are LMs that use an encoder-decoder architecture. As explained in [Vas+17], the encoder maps an input sequence $x = (x_1, x_2, ..., x_n)$ to a hidden representation $z = (z_1, z_2, ..., z_n)$. Given z, the decoder generates an output sequence $y = (y_1, y_2, ..., y_m)$. Both the encoder and decoder are stacks of layers. Every encoder layer is composed of two sub-layers, a multi-head self-attention mechanism and a fully-connected feed-forward network. Residual connection and Layer normalization are applied after each of these sub-layers. This means that the output of each sub-layer is equal to $\text{LayerNorm}(x + F(x))$ where $F(x)$ is the function of the sub-layer [Vas+17]. In addition to the two sub-layers of the encoder layer, the decoder has an extra sub-layer that performs multi-head attention over the output of the encoder. Encoder-decoder transformers are widely used in NLP tasks related to sequence generation, such as text summarization, machine translation and question answering.

We will use a BERT model pre-trained on a general corpus in our experiments. BERT stood for Bidirectional Encoder Representations from Transformers [Dev+19] and was introduced by the Google AI team. BERT follows a transformer architecture by implementing an Encoder stack. BERT Base has 12 layers in the Encoder stack, while BERT Large has 24 layers in the Encoder stack. Due to its architecture, BERT can be fine-tuned and be used

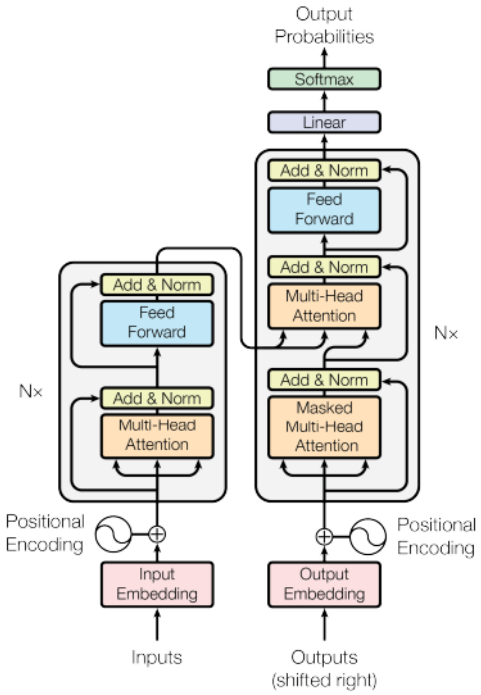

Fig. 2.1: Tranformer model architecture. Figure is taken from [Vas+17]

in any downstream task simply by adding just one additional output layer on top of the encoder stack, resulting in state-of-the-art performance.

2.3 Abusive Content Detection

Identifying toxic language on social media platforms is difficult, as offensive content can take many forms. Users can experience hate-speech attacks for their race, gender, religion or political beliefs. In some cases, discrimination or harassment over specific groups of people can take place without insults or profane phrases. Counter to that, online comments could contain irony or sarcasm for entertainment reasons (e.g., in the case of stand-up comedians' posts), so the perceived meaning of the words can change based on their context. Social media are trying to restrict harmful content but mostly rely on tools that detect frequent patterns and phrases that, in many cases, could be misleading in the classification of online posts [PMA17].

Researchers have experimented with different approaches for the task of Abusive context Detection. Initially, machine learning methods with lexical or syntactic features were adapted. [Dav+17] have applied machine learning models such as Logistic Regression and Naive Bayes with the use of morpho-syntactic features (Part-Of-Speech tags). [KW13] trained a Naive Bayes hate-speech classifier with a bag-of-word approach that employed unigrams to construct the vocabulary of the training set. In the first case, although the

model reached high micro-averaged precision (0.90), recall (0.90) and f1-score (0.91), it did not perform equally well for the hateful class as it misclassified most of the hate tweets (0.44 precision). In the second case, the model tended to over-classify tweets as hateful in cases where they contained words that, when placed out-of-context, could be perceived as hateful.

In recent approaches, deep learning techniques with word embeddings were employed for detecting toxic content. [PF17] trained three Convolutional Neural Networks (CNN) with multiple filter sizes, large feature map sizes, and a max-pooling layer after the convolution to capture the feature with the most powerful signal. They focused on a single-label multi-class classification problem with three class labels: {Sexist, Racist, None}. Their deep learning models had similar overall results as the Logistic Regression baseline classifier but outperformed it at the offensive categories evaluation, where the baseline had a low recall score. [PMA17] developed a Recurrent Neural Network (RNN) operating on word embeddings that outperformed Detox, the previous state-of-the-art comment moderation system. Detox used Logistic Regression or an MLP classifier that operated on n-grams [WTD17]. They experimented with two different deep learning models (RNN, CNN) and reproduced Wulczyn et al.'s system [WTD17] as a comparison measure. They reported Area Under Curve (AUC) score as their evaluation metric. Their RNN model, especially when a self-attention mechanism was added (Figure 2.2) to compute the weighted sum of all the hidden states, outrun the Detox system.

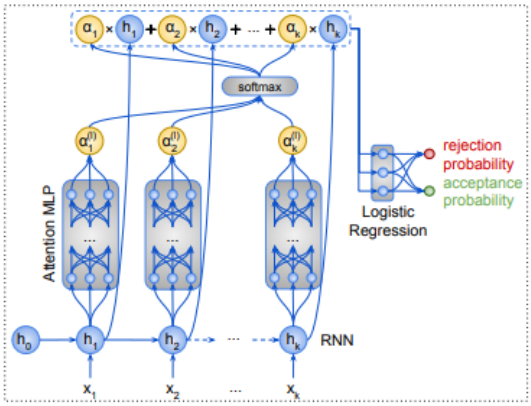

Fig. 2.2: Illustration of RNN with an attention mechanism. Figure is taken from [PMA17]

2.4 Self-training

Semi-supervised learning benefits from unsupervised data that are available for a specific task (Section 1.3). A common implementation of semi-supervised learning is Self-training. It is an iterative method that employs a Teacher model initially trained on the labeled instances to create silver labels for the unlabeled dataset [Yar95; BM00]. A subset of the silver-labeled instances, along with the pseudo-labels, is sampled and added to the labeled

dataset. The Student, usually a model with the same architecture as the Teacher, is then trained on the augmented labeled dataset. Subsequently, the Student model becomes the new Teacher to sample and silver-label additional instances for the next round [Li+21; Mi+21]. These steps can be repeated in cycles for a fixed number of iterations or until the evaluation metric starts to drop or it reaches convergence.

This approach is called Self-training when the Student model has a similar or higher capacity than the Teacher, and knowledge distillation [HVD15] when the Student model is smaller than the Teacher. Self-training utilizes unlabeled data in a task-specific way during the pseudo-labeling process to train a more robust Student model and has been applied in a variety of NLP, computer vision and speech recognition tasks. More specifically, recent research has been conducted on Self-training in tasks such as sentiment analysis [Du+21], intent-classification and dialog-state tracking [Mi+21], evidence-extraction [Niu+20], natural language inference [Li+21], toxic span detection [SJ21], text classification and rationale extraction [BSM21].

2.4.1 Pseudo-label process

Self-training utilizes artificial labels generated by the trained Teacher to act as additional supervision to the Student [Che+21]. During the pseudo-labeling process, silver labels are assigned to the unannotated examples by picking up the class with the higher predicted probability by the Teacher model. The confidence score of the prediction is defined as the predicted probability score. In binary classification problems, the prediction score is the probability generated by the model to the predicted label. In multi-label multi-class classification problems, [Mi+21] use the mean of the prediction scores corresponding to the predicted labels to acquire a confidence score for each unannotated example.

2.4.2 ST sampling methods

The most common approach for selecting the subset of unlabeled data to augment the training dataset is based on the model's confidence score [Yar95; Niu+20]. This method samples k instances from the unlabeled data pool that produced the highest confidence scores [Mi+21; Du+21], or all instances with probability (confidence score, Section 2.4.1) above a certain threshold at the end of each round [Yar95]. This technique is often compared in literature with random sampling, where random k instances are added to the training set iteratively or with least-k sampling, where k instances that produced the lowest confidence scores in the previous round are selected. The Student is then trained on the augmented dataset. Algorithm 1 describes the Teacher-Student iterative process steps when the top-k technique is applied.

[Mi+21] implemented Self-training for four downstream tasks, including intent classification, dialog state tracking, dialog act prediction, and response selection. They used large-scale pre-trained LMs with different task-specific heads for each task and experimented with different sampling techniques such as the top-k, the least-k and the random-k technique. The top-k sampling outperformed all other sampling methods. [Mi+21] suggested that the initial Teacher trained on limited labeled data is not good enough to assign reliable labels to a large number of unlabeled data, so the instance confidence score should be considered when sampling the silver-labeled data to be added to the training set.

[Du+21] employed Self-training and knowledge-distillation for Natural Language Understanding in dialogue systems using RoBERTa-Large. In the case of knowledge distillation, they used a Student model that had an order of magnitude fewer parameters than the RoBERTa Teacher model. They also produced few-shot settings as their third experiment by considering only a few samples per class. Self-training outperformed the baselines, which were large pre-trained LMs in all of their experiments. [Du+21] observed that the most significant improvement was in the few-shot scenario, where the model's accuracy increased by 3.5%. [Niu+20] performed experiments on the Teacher-Student approach by applying both confidence and random sampling. Based on the evaluation of their system when both these techniques were used, they concluded that the top-k sampling strategy is more efficient as it tends to prevent the model from learning the wrong knowledge of the wrong Teacher predictions made in previous rounds.

Algorithm 1 Self-training (ST), K most confident instances

Require: Labeled data: L, Unlabeled data: U
Require: Teacher: F_t, Student: F_s
Require: Number of pseudo-labeled data in an iteration: k
Ensure: A trained Student F_s
  Initialize F_t and train F_t on L
  while F_s not good enough and $U = \emptyset$ **do**
    Initialize F_t, $L \leftarrow Priority\_list()$
    while $x \quad U$ **do**
      Compute prediction label $y_x = F_t(x)$
      Compute confidence score s_x
      $L.insert(x, y_x, s_x)$
    $L \leftarrow L.top(k)$
    $L \leftarrow L \quad L$
    $U \leftarrow U \setminus L$
    Train F_s on L
    $F_t \leftarrow F_s$

Another method used to augment the labeled dataset is the select-all technique. [Li+21; Du+21]. At the end of each round, the Student is trained with the union of the labeled data and the entire set of unlabeled data, provided with soft labels by the Teacher (Algorithm 2). However, this method may be effective only when a good fraction of the predictions on the unlabeled samples are correct. Otherwise, early mistakes made by the Teacher can

reinforce themselves by generating incorrectly pseudo-labeled data. Re-training with this data will lead to an even worse Student model in the next round [Li+21].

Algorithm 2 Self-training (ST), Select-all instances

Require: Labeled data: L, Unlabeled data: U,
Require: Teacher: F_t, Student: F_s
Ensure: A trained Student F_s
  Train F_t on L
  while F_s not good enough **do**
    while $x \quad U$ **do**
      Compute prediction label $y_x = F_t(x)$
    $U = \{(x_j, y_x) | \; x_j \quad U\}$
    $L \leftarrow L \quad U$
    Train a Student model F_s on L
    $F_t \leftarrow F_s$

[Li+21; Du+21] propose task-adaptive pre-training as a complementary semi-supervised method to deal with this challenge. By pre-training masked LMs with a large number of unlabeled in-domain training examples and then fine-tuning the Teacher model on labeled data in a standard supervised way, they achieve a better initialization for the Teacher model. As the initial model is pre-trained with task-specific data, it performs better at early stages, can avoid early mistakes and generate more accurate predictions. Algorithm 3 describes the steps of task adaptive pretraining and Self-training with the select-all technique. In addition, [BSM21] introduced a weighted pseudo-labeled loss function used during Student training. A weight is assigned to each training example based on the confidence score of the Teacher's prediction. These weights are normalized across each mini-batch during the training process. Their experiments showed that upweighing most confident examples and downweighing noisy ones are beneficial for the Student's training. It prevents the model from relying on wrong knowledge obtained from previous rounds.

Algorithm 3 Task adaptive pre-training and Self-training, Select-all instances

Require: Labeled data: L, Unlabeled data: U,
Require: Teacher: F_t initialized with p_θ, Student: F_s
Ensure: A trained Student F_s
  Update model p_θ with Task-adaptive pre-training on U
  Train F_t initialized with p_θ by fine-tuning on L
  while F_s not good enough and $U = \emptyset$ **do**
    while $x \quad U$ **do**
      Compute prediction label $y_x = F_t(x)$
      $U = \{(x_j, y_x) | \; x_j \quad U\}$
    Train F_s on $L \quad U$
    $F_t \leftarrow F_s$

2.5 Data augmentation (DA)

DA refers to strategies for increasing the diversity of training examples without explicitly collecting new data [Fen+21]. It is a common technique used in machine learning to prevent models from overfitting on training data and improve generalization. For few-shot learning where training examples are hard to obtain, models can significantly benefit from DA techniques. [PMA22] showed that DA can lead to very significant performance gains, even when using large pre-trained transformers, by experimenting with seven different DA techniques for question answering in the biometrical domain.

DA has been commonly used in Computer vision, where techniques like cropping, flipping, and color jittering are a standard component of model training [Fen+21]. However, in NLP, where the input includes complex syntactic and semantic structures, the way of generating effective augmented examples that preserve the class label is less obvious. In addition, [Fen+21] note that the distribution of the augmented data should be neither too similar nor too different from the original dataset. In the first case, the risk of overfitting is not reduced when using the augmented dataset, while in the second case, the model's performance could drop further through training on examples not representative of the given domain.

Two of the most common DA techniques used in several NLP tasks are Back Translation and Word Substitution. Back Translation [Fen+21] refers to the case where training examples are machine-translated from a source to a pivot language and back, obtaining paraphrases of the original example. Word substitution, a DA method widely applied in NLP, replaces words from the original sentence with synonyms or other relevant words drawn from a thesaurus or words with similar embeddings [PMA22]. In more recent works, word substitution is performed using large pre-trained masked LMs, which suggest the replacement of randomly masked words from the initial training example (Masked LM Word Substitution).

2.5.1 DA techniques applied with Self-training

An important goal of Self-training is to improve the robustness of the Student model trained from potentially noisy pseudo-labeled samples. DA techniques are widely used for that purpose [Mi+21]. DA acts as a form of regularization as the model is encouraged to generate consistent predictions on original sequences and augmented ones, based on the assumption that a model that has not learned to memorize the training data should produce similar predictions for relevant inputs. Some implementations of Self-training combined with DA methods are presented below.

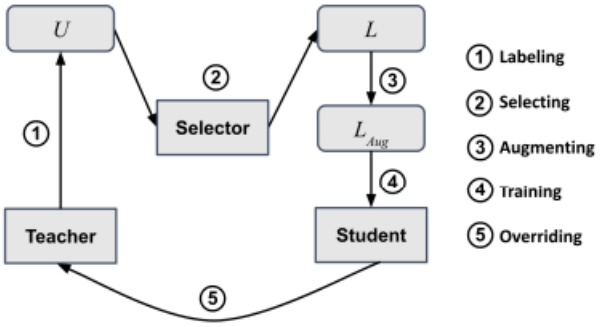

Fig. 2.3: Self-Training pipeline when DA is included. The Teacher generates pseudo labels for data in U. Then, the Selector chooses the most confident samples based on the Teacher's predictions and adds them to L. Afterwards, L is augmented by a DA technique to train the Student. Lastly, the trained Student becomes the Teacher in the next iteration. Figure is taken from [Mi+21].

[Du+21] have created a sentence encoder that outputs similar hidden representations for sentences of similar meaning to produce task-specific sentence embeddings. These embeddings are then used as queries for retrieving in-domain sentences from a large bank of sentences crawled from the web. [Niu+20; Mi+21] have used Masked LMs to create label-preserving augmentations. [Niu+20] applied augmentation to the examples with the highest confidence scores that are more likely to be reliable and performed random masking (15% of the sequence tokens). They replaced masked tokens with tokens suggested by the pre-trained LM based on the assumption that the label would be preserved because of the context-aware representations on top of the LM. Contrary to that assumption, [Mi+21] suggested that masking and replacing the most crucial tokens for each task might lead to changing the semantics after the sequence reconstruction. In order to preserve the label of the initial example, they suggested measuring the importance of each token x_i by accumulating the gradients of all elements in its embedding by differentiating $F_t(x)$ w.r.t. x_i ($F_t(x)$ is the Teacher prediction score). Based on their intuition, they suggested that tokens with large gradients are important to the label y.

[Vu+21] have proposed the implementation of "task augmentation" for few-shot learning. This approach utilizes a BERT model fine-tuned on a dataset with thousands of examples for an auxiliary task, such as Natural Language Inference (NLI), before the target task. For the NLI task, they used MNLI, a dataset that contains sentence pairs labeled as {entailment, contradiction, neutral}. They transformed each training example from its basic form [textA, textB] -> label to [textA, label] -> textB. They trained the BERT LM with the transformed examples in order to generate textB. The fine-tuned data generator could then be used to augment training examples for any downstream task.

Dropout is also used in related research as an additional form of augmentation. [Xie+20] in the task of image classification suggested that adding model noise to the Student makes it more powerful. More specifically, when dropout is used as noise, the Teacher behaves like

an ensemble at inference time, whereas the Student behaves like a single model. In other words, the Student is forced to "mimic a more powerful ensemble model" [Xie+20].

2.6 Active Learning

As discussed in Sections 1.2 and 1.3, large datasets of supervised data are scarce for many NLP tasks. Manually annotating thousands of examples takes a great deal of human effort and is error-prone and expensive. In addition, not all examples are equally useful in order to learn a specific task. For example, instances very similar to what the model has already seen do not provide significant new information. Active Learning is the task of reducing the amount of labeled data required to learn the target concept. By querying the user to annotate only the most informative instances, the concept is learned with fewer examples [Aro07]. The success of an active learner is demonstrated by showing that it needs to be trained with fewer examples than the traditional learner to achieve the desired performance.

In a typical Active Learning scenario, there is a limited amount of labeled data and a large pool of unlabeled data available for a specific task. A classifier is initially trained on the labeled examples. Selective sampling is then used to select a subset from the unannotated set in order to be labeled by a human annotator. The classifier is subsequently further trained with the newly manually-labeled instances. This iterative process of training, selective sampling and annotation is repeated until convergence [Aro07]. Algorithm 4 presents the steps of the Active Learning process. Active Learning has been successfully applied in a wide range of NLP tasks, including text classification [MN98a], named entity recognition [She+04], semantic role labeling [RS06] and parsing [Hwa00].

2.6.1 Selective Sampling

Several different techniques for selective sampling have been explored in the literature. Uncertainty-based sampling selects examples that the model is least certain about and presents them to the user for correction/verification [Aro07]. Several definitions of uncertainty have been used, but all are based on estimating how likely a classifier trained on previously labeled data would be to produce the correct class label for a given unlabeled example. [LG94] use a probabilistic text classifier for uncertainty sampling. Specifically, the classifier samples a subset in each iteration based on the max-entropy decision rule. [GG16] apply Monte Carlo Dropout (MC Dropout) to a Neural Network to generate the prediction and compute the uncertainty of each training example. MC Dropout refers to the case where Dropout is also applied at the inference time instead of only being applied during training. By predicting the label of a test instance in many rounds, the same model will produce different probability scores because each time, different neurons of the

network would be randomly switched off. The instance's uncertainty is then computed by the variance of the model's predictive probabilities in each round of the inference time.

The query-by-committee sampling method generates a group of predictions and selects a set of unlabeled examples based on the disagreement among these predictions. In particular, it selects the examples on which the disagreement within the committee is the highest [Aro07]. The committee consists of k classifiers acting like an ensemble of models. Vote Entropy, defined as "the entropy of the class label distribution obtained when each committee member votes with probability 1/k for its winning class" [Aro07], and the KL divergence of each committee member's predictive probability to the mean of all committee members' predictive probability [MN98b] are the two metrics often used to measure disagreement among the committee of classifiers.

In literature, the diversity criterion is often used to measure the training utility of a batch of examples. Specifically, it is suggested that a batch of examples with high variance may be pretty informative and useful to the training process. [She+04] propose two methods for diversity sampling, global and local sampling. For global consideration, they create clusters of the remaining unlabeled examples (using k-means clustering) based on similarity and then select examples from different clusters. For local consideration, examples that are most different from instances already in the training pool of the labeled data are sampled in each iteration.

Another technique of sampling valuable examples from the unlabeled data pool is representativeness sampling. This method is based on the assumption that the most informative examples are the ones that best cover the dataset. The representativeness of an example can be calculated as the number of unlabeled examples that are similar to it [Aro07]. Examples with high representativeness are less likely to be an outlier, and adding them to the training set will have an effect on a large number of unlabeled examples. If the examples were clustered together based on similarity, the centroids of the clusters would be the most representative examples according to [Aro07]. During their experiments, [She+04] used cosine similarity and Dynamic Time Warping to measure the similarity of examples.

In our experiments, we applied the Active Learning criteria that measure the representativeness and diversity of the silver-labeled instances described above to augment our training set in each Teacher-Student round. Instead of using confidence sampling, we reproduced Self-training by sampling the most representative or diverse instances from the unlabeled data pool and the generated labels. We performed this experiment to examine whether the informative score g_x of an instance (representativeness/diversity score) could be a more critical factor than the Teacher's prediction confidence score s_x when expanding the training set with artificial labels. In addition, we examined if these two factors (s_x, g_x) could be combined when sampling the silver-labeled instances to train a stronger Student model. More details about our experiments can be found in Chapter 3.

Algorithm 4 Active Learning

Require: Inital model $f(x; \theta_0)$, Unlabeled data U, number of iterations T, sampling algorithm A
  $D = \{\}, t = 0$
  while iterations $t < T$ **do**
    $Q_t \leftarrow$ Apply A on model $M_t(x)$, data U
    $D_t \leftarrow$ Label queries Q_t
    $D \leftarrow D \quad D_t$
    $U \leftarrow U \setminus D_t$
    $\theta_t \leftarrow$ Fine-tune $f(x; \theta_0)$ on D
    $t = t + 1$
  return $f(x; \theta_T)$

2.6.2 Combining Active Learning with Self-training

In related research, various implementations combine Active Learning with Self-training. The combination of these two learning techniques aims to reduce annotation costs. The Self-training method discovers highly reliable instances based on a trained classifier, while Active Learning queries the most informative instances based on active query algorithms. [Wu+06] developed a system for spoken language understanding in domain-specific dialogue systems that consisted of two parts. The first part was a topic classifier used to reduce the search space of the correct answer by identifying the topic of the slot. The second classifier (semantic classifier) was trained to extract the corresponding slot-value pairs using the restriction of the recognized target topic. They employed the strategy of combining Active Learning and Self-training for training the topic classifier. For Active Learning, they selected uncertainty-based sampling where the most unconfident examples were selected for a human to label and then added to the training set. For Self-training, the examples with classification confidence scores over a certain threshold and their predicted labels were added to the training set to retrain the classifier. They used the class probability as the confidence score of the example to apply both methods and repeated this process iteratively until no unlabeled examples were left in the pool. To evaluate their approach, they compared it with random sampling as their baseline and with the implementation only of Active Learning. Their experiments showed that Active Learning significantly reduces the amount of labeled data needed for the task, as it almost reached the performance of the baseline classifier using only 1/3 of the labeled examples. The combination of Active Learning and Self-training further boosted the baseline classifier performance by using 1/3 of the labeled examples.

[GKP11], followed a similar approach for the task of argumentative zoning (i.e., analysis of the argumentative structure of a scientific paper). In their experiments, they implemented supervised training with Support Vector Machines as their baseline and compared it with various weakly supervised techniques, including Active Learning alone and in combination with Self-training. The sampling method used for Active Learning was uncertainty sam-

pling, where the unlabeled instances with the lowest posterior probabilities were queried for the next round of learning. In each round, the model was jointly trained with the machine-labeled and manually annotated data from the Active Learning sampling. When using only 10% of the labeled data from the initial training set, their method outperformed the fully-supervised learning and the weakly-supervised learning that was based only on Active Learning.

[Tra+17] combined Active Learning and Self-training for the named entity recognition task from tweet streams. They employed Self-training queries based on both diversity and uncertainty sampling to select the most informative instances. The probability of individual token labels was considered while looking for uncertainty instances. They considered that the model is uncertain about its prediction if it assigns to the instance at least one label with a probability less than a predefined threshold. The diversity of instances was examined based on their context and content to select instances that differed from the current training data. A vector model was used to measure the context similarity of instances, where each instance was represented as a vector. The vector's dimensions represented the POS tags of the sequence tokens. Diversity was then evaluated by counting the number of tokens with the same POS tag sequence as the instances already in the training set. For content diversity, they constructed word vectors using the Word2Vec model and then added the word vectors to compose an instance vector. They evaluated similarity for two instances using a function based on cosine similarity. In each round, the instances with similarity scores less than or equal to the predefined similarity threshold were human labeled. In addition, the most confident instances were machine-labeled and added to the training data.

3 System Design and Implementation

As discussed in Section 1.3, in this thesis, we implement a Teacher-Student approach for the two following tasks: Toxicity detection on Greek tweets and Sexism Detection on English posts. We interpreted these tasks as binary classification problems where the class labels for the input user posts are 1 (Offensive/Sexist) and 0 (Not Offensive/Not Sexist). We implement Self-training by applying different sampling techniques to augment our initial labeled dataset. First, we examine different techniques using a Logistic Regression classifier. Second, we apply Self-training with the best sampling techniques tested on the Logistic Regression Teacher-Student framework to a BERT model pre-trained on a large general corpus.

To reproduce few-shot settings, we randomly sampled a small number of instances from the dataset's training examples as our initial labeled set (L). We detached 10% from the original training dataset for our validation set (V) and 10% as a test set to evaluate our final results. The rest of the supervised set was used as our unlabeled Dataset (U).

We will first describe the different sampling techniques applied to the Logistic Regression Teacher-Student framework. We will then present our experiments on the Teacher-Student framework based on a pre-trained LM (Section 3.7).

3.1 Teacher-Student with Logistic Regression classifier

In our approach, both the Teacher F_t and the Student F_s are Logistic Regression classifiers with the same hyperparameter values. During training, two data pools are maintained, denoted as U (unlabeled data) and L (labeled data). We have two class labels: 1 for Offensive/Sexist and 0 for Not Offensive/Not Sexist. Following Algorithm 1, we initially train the Teacher model on L. Then the Teacher generates predictions for U by assigning each input sequence x to the class with the highest probability. In the case of a Logistic Regression classifier with weights vector w :

Probability for positive class: $P(1|x) = \frac{1}{1+exp(-xw)}$

Probability for negative class: $P(0|x) = 1 - P(1|x) = \frac{exp(-xw)}{1+exp(-xw)}$

The predicted class is assigned as the silver label for the input $x : y_x = F_t(x)$. The Selector (the function that is responsible for the sample selection) samples a number of silver-labeled instances based on different criteria to expand the initial labeled set (L). The Student is trained on the augmented training set, and at the end of each round, he becomes the new Teacher. We evaluate the Student's performance on the validation set at the end of each round. At the next iteration, the new Teacher predicts the class labels for the remaining instances on U. These steps are repeated in rounds until the evaluation metric starts to drop by evaluating our system on the validation dataset. The final model is then evaluated on the test set.

3.2 Confidence sampling

3.2.1 Top - k instances

In each round of the Teacher-Student approach, the Teacher generates predictions for the entire unlabeled set U. The confidence score s_x of each instance in U is the probability of the classifier's predicted class. In the top-k technique, the Selector subtracts the k instances with the highest confidence score from U and inserts them in L. We considered k a hyperparameter and experimented with different values [20, 50, 100, 200]. For each k we performed Teacher-Student training iteratively until the round that the system achieved the max micro F1-score[1] based on the validation set and reported the results on the test set.

For the second dataset, although the micro F1-score increased after applying Self-training with confidence sampling, we observed that the F1-Score of the minority class[2] (Sexist instances) decreased after the addition of the silver-labeled examples. Hence we considered both the micro F1-Score and the F1-Score of the minority class to monitor the number of Teacher-Student iterations. More specifically, we set the number of rounds equal to the iteration where the max F1-Score of the minority class is achieved while micro F1-Score does not drop (compared to the micro F1-score achieved in few-shot learning). Given the class imbalance of our datasets, an alternative could have been to monitor the macro F1-Score that assigns equal weight to each class regardless of the number of instances that belong to each category.

In addition, for the second dataset, we examined if the class ratio of the top-k instances added in each iteration is responsible for the performance drop in the minority class

[1]Information about the micro F1-score evaluation metric can be found in Section 4.3.
[2]Information about the F1-score of each class can be found in Section 4.3.

detection. Instead of adding the top k instances irrespective of their silver labels, we selected a batch that contains 24% minority class instances and 76% majority class instances to preserve the class balance of the initial labeled set. We also examined if adding batches of different fixed class ratios when expanding our training set could have better results. Hence we performed the same experiment by adding batches of 25%, 26%, 27% and 28% sexist instances and 75%, 74%, 73%, 72% not sexist instances, respectively.

3.2.2 Instances with confidence above a certain threshold

Another quite common technique when confidence sampling is applied is to sample all instances with s_x above a certain threshold t. We performed this experiment by assigning different thresholds in the interval [0.81, 0.99] and repeated Teacher-Student training until the round in which our system had the best performance on the validation set. Specifically, for the Toxicity dataset, we set as monitor micro F1-Score, while for the Sexism dataset, we repeated the iterations monitoring both micro F1-Score and F1-Score of the minority class.

3.2.3 Adaptive number of top confident instances

The next experiment we performed based on confidence sampling was adding an adaptive number of top confident examples. The idea was to examine how the Student's performance would be affected if we augmented the labeled dataset by a variable rather than a fixed number of examples in each round. Based on the assumption that the Student becomes more robust after each iteration and it generates more accurate predictions for the unlabeled dataset, we examined two approaches that increase k in each round: $k + 20, 2 \quad k$.

In addition, we tried to adapt k based on the difference in the score of the Student's micro F1-Score after each iteration. In each round, if the Student was evaluated with the same or lower micro F1 score as before, we kept k stable. Instead, if the Student's micro F1-score increased by adding k examples in round l, we doubled the examples inserted in the Labeled dataset in round $l + 1$.

To tune the Teacher-Student rounds, for the Toxicity dataset, we set as monitor micro F1-Score, while for the Sexism dataset, we repeated the iterations until the max F1-Score of the minority class is obtained, and the micro F1-Score does not drop compared to the few-shot results (based on the validation set).

3.3 Select - all instances

Following algorithm 2, the Self-training process follows the same steps described in Section 3.1 except for the Selector function. The entire dataset U and the predicted labels are combined with the L in each round ($L \leftarrow L \quad U$). Then the Student model is trained on the augmented dataset L and generates pseudo-labels for U. The number of Teacher-Student rounds was set equal to the iteration where the micro F1-score started to drop based on the evaluation of the validation set.

3.4 Active Learning

After experimenting with different Self-training techniques extracted from the related bibliography, we tried to examine if Active Learning criteria could be useful when selecting instances from the unlabeled data pool U to augment our training set. As described in Section 2.6.1, selective sampling tries to locate the most informative instances, i.e., the instances that have the maximum training utility for the concept that the model is trying to learn. In most cases, a human annotator is then queried to assign labels to these examples instead of annotating a huge task-specific dataset. This process can be repeated in rounds to further boost the performance of the model.

We applied this method iteratively to our framework to study if Active Learning criteria are helpful when sampling and assigning true labels to the selected unlabeled instances to augment L. To conduct the Teacher-Student experiments, we removed the ground-truth labels of the remaining training set to use it as the unlabeled set U. Hence, for the Active Learning application, we did not need a human annotator to create ground-truth labels for the examples, as their true labels were already available. We augmented the training set in each round with top-k instances (based on Active Learning criteria) and their gold labels. The criteria we employed are described in the following two subsections.

3.4.1 Representativeness sampling

The first Active Learning criterion we implemented is representativeness sampling. According to this criterion, the most informative instances of the Unlabeled dataset U are the most representative of the dataset. They are less likely to be outliers, and they represent a significant portion of the unlabeled examples. As the representativeness score of an instance in U, we define the average distance from its nearest neighbors in U. The metric for distance was set as cosine similarity. Hence, we implemented the K-NN (K-Nearest Neighbors) algorithm fitted on the unlabeled dataset to find the ten nearest neighbors of each instance. Then we calculated the average distance from these neighbors to produce

the instance score r_x. Specifically, we assigned the highest r_x to the instances in U that had the lowest average distance from their neighbors in U. In each iteration, we sampled k instances that had the highest r_x and added them to the labeled set.

3.4.2 Diversity sampling

The diversity criterion suggests that the instances that are most different from the examples already in the training pool L are the ones with the highest training utility. This technique is popular in Active Learning based on the assumption that the model could benefit more from instances different from those it has already seen so it can gain new knowledge for the task. We calculated the diversity score of an instance in U from the training set as the average distance of the instance from its nearest neighbors in L. We used our implementation of k-NN (fitted on the Labeled dataset) search to find the ten nearest neighbors of each unlabeled instance. Then we calculated the average distance from them and set it as the instance diversity score d_x. During the iterative process, we selected k instances with the highest d_x along with their true labels and inserted them into the training data pool.

Figure 3.1 illustrates each sequence's average distance from its nearest neighbors in L and the average distance from its nearest neighbors in U, respectively. In the left figure, as the average distance increases, the samples are more diverse from those already in the labeled dataset. Correspondingly in the right figure, as the average distance decreases, the samples are more representative of the examples that have not yet been added to the training set.

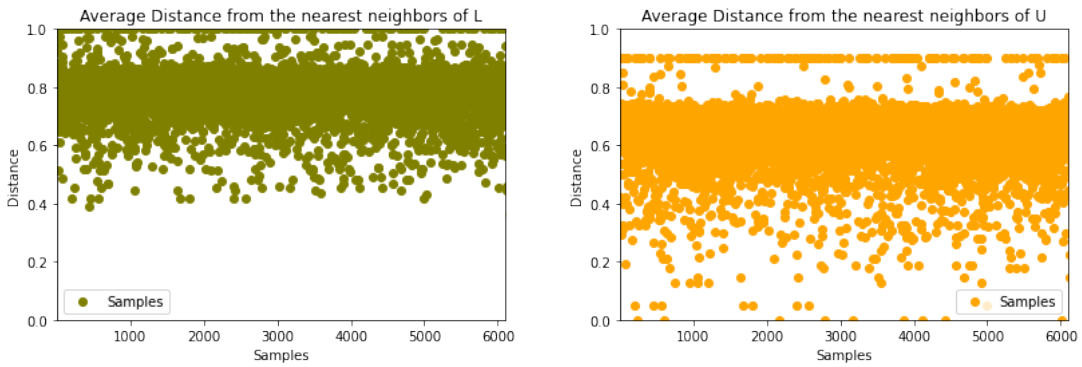

Fig. 3.1: Toxicity Dataset: Samples Diversity from L and Representativeness of U

As mentioned in Section 3.4, since the true labels of the instances in U were available, we used the ground truth labels to simulate a human annotator that would be consulted during Active Learning. We applied Active Learning to demonstrate whether expanding the training set with the k most informative instances could lead to a more robust model with fewer training examples than other sampling techniques. We performed the same

experiment by selecting in each round the k most confident samples based on the model's predictions and random k examples, along with their true labels, to compare the results.

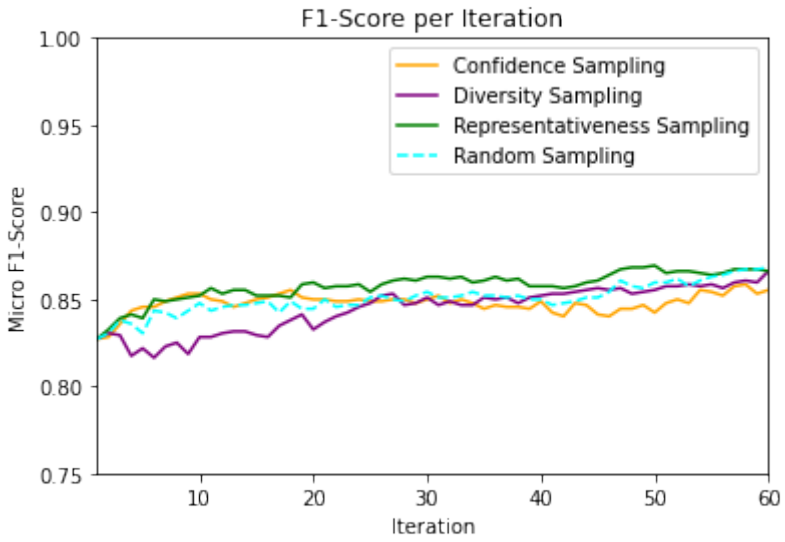

Fig. 3.2: Toxicity Dataset: Micro F1-Score per iteration when Active Learning with different techniques is applied.

As shown in Figure 3.2, when ground-truth labels are available, the representativeness score of the instance is the most beneficial criterion when expanding the initial few-resource dataset. The diversity criterion did not work as well as the other three techniques at first, but after the first 20 iterations, it performed better than confidence sampling. Random sampling performs better than diversity sampling during the first TS rounds and better than confidence sampling as the iterations increase. The curve of the micro F1-Score when representativeness sampling is applied is on top of the other curves in most iterations, and it is probably a more reliable option.

3.5 Teacher - Student with Active Learning Criteria

Following algorithm 5, we reproduced Teacher-Student with Active Learning criteria as sampling techniques for a few-shot learning scenario. Similar to the Teacher-Student approach, when confidence sampling is applied, two data pools are maintained: L, the initial annotated dataset and U, the unsupervised data. A Teacher model is originally trained on L and generates predictions for all instances in U. Then a Selector function samples k most informative instances based on two different criteria (representativeness/diversity). These instances and their silver labels are added to the training set. After adding the queried examples, the Student is initialized and trained on the extended set. These steps are iteratively repeated until the Student's micro F1-score starts to drop based on the

evaluation of the validation set. At the end of this process, the Student is evaluated on the test dataset.

Algorithm 5 Teacher - Student with Active Learning criteria

Require: Labeled data: L, Unlabeled data: U,
Require: Number of pseudo-labeled data in an iteration: k
Require: Teacher: F_t, Student: F_s
Require: Informativeness criterion: i
Ensure: A trained Student F_s
  Train F_t on L
  while F_s not good enough and $U = \emptyset$ **do**
    Initialize F_t, $L \leftarrow Priority\_list()$
    while $x \quad U$ **do**
      Compute prediction label $y_x = F_t(x)$
      Compute informativeness score g_x based on the informativeness criterion i
      $L.insert(x, y_x, g_x)$
    $L \leftarrow L.top(k)$
    $L \leftarrow L \quad L$
    $U \leftarrow U \setminus L$
    Train F_s on L
    $F_t \leftarrow F_s$

3.6 Active Learning criteria combined with confidence sampling

We attempted to combine Self-training and Active Learning criteria to employ a Teacher-Student method that samples instances based on both these techniques (Algorithm 6). In the first experiment, we converted the Selector function to extract instances from the unlabeled data pool U based on a linear combination of confidence (s_x) and representativeness (r_x) of the instance. Specifically, we used the equation $a_x = \lambda \cdot s_x + (1 - \lambda) \cdot r_x$. We considered λ a hyperparameter and manually tuned it by testing different values in the space $[0.1, 0.9]$. When $\lambda = 0.1$, the Selector pays little attention to the confidence score of the instance, whereas when $\lambda = 0.9$, the model takes into little account the representativeness score of the instance. For the case of confidence and diversity of the samples, the Selector again sampled instances based on the score of the linear combination of these two criteria $a_x = \lambda \cdot s_x + (1 - \lambda) \cdot d_x$.

Figure 3.3 illustrates micro F1-Score per iteration for different values of λ in the interval $[0, 1]$ for both datasets when evaluating the validation set. The figure on top shows the results of the Teacher-Student application using the Toxicity dataset, while the figure on the bottom presents the results of Teacher-Student using the Sexism dataset. As shown below, as we increase the weight assigned to the confidence score of the instance, the micro F1-Score increases. The best results in the interval $[0.1, 0.9]$ for both datasets were

Algorithm 6 Teacher - Student with confidence sampling and Active Learning criteria

Require: Labeled data: L, Unlabeled data: U,
Require: Number of pseudo-labeled data in an iteration: k
Require: Instance confidence score weight λ
Require: Teacher: F_t, Student: F_s
Require: Informativeness criterion: i
Ensure: A trained Student F_s
  Train F_t on L
  while F_s not good enough and $U = \emptyset$ **do**
    Initialize F_t, $L \leftarrow Priority\_list()$
    while $x \quad U$ **do**
      Compute prediction label $y_x = F_t(x)$
      Compute confidence score s_x
      Compute informativeness score g_x based on the informativeness criterion i
      $a_x = \lambda \cdot s_x + (1 - \lambda) \cdot g_x$
      $L.insert(x, y_x, a_x)$
    $L \leftarrow L.top(k)$
    $L \leftarrow L \quad L$
    $U \leftarrow U \setminus L$
    Train F_s on L
    $F_t \leftarrow F_s$

obtained when setting $\lambda = 0.9$ (when the largest weight is attributed to the instance's confidence score). We also present the curves of micro F1-Score per iteration when setting $\lambda = 0$ (representativeness sampling) and $\lambda = 1$ (confidence sampling) for comparison reasons.

Respectively, figure 3.4 depicts micro F1-Score per iteration for different values of λ in the interval $[0, 1]$ when evaluating the validation dataset. The figure on the top depicts the results of the Teacher-Student framework for the toxicity detection task, while the figure on the bottom shows the results of the Teacher-Student framework for the sexism detection task. For confidence and diversity, the results are comparable to confidence and representativeness sampling. The highest scores for both datasets when tuning λ in the interval $[0.1, 0.9]$ were obtained when setting $\lambda = 0.9$, and the largest weight is assigned to the instance's confidence score. We also present the curves of micro F1-Score per iteration when setting $\lambda = 0$ (diversity sampling) and $\lambda = 1$ (confidence sampling) for comparison reasons.

3.7 Self-training with BERT

Our final experiments included Self-training using a pre-trained BERT model trained on large general corpora. After experimenting with different sampling methods with a Logistic Regression classifier, we wanted to explore whether Self-training can be effective

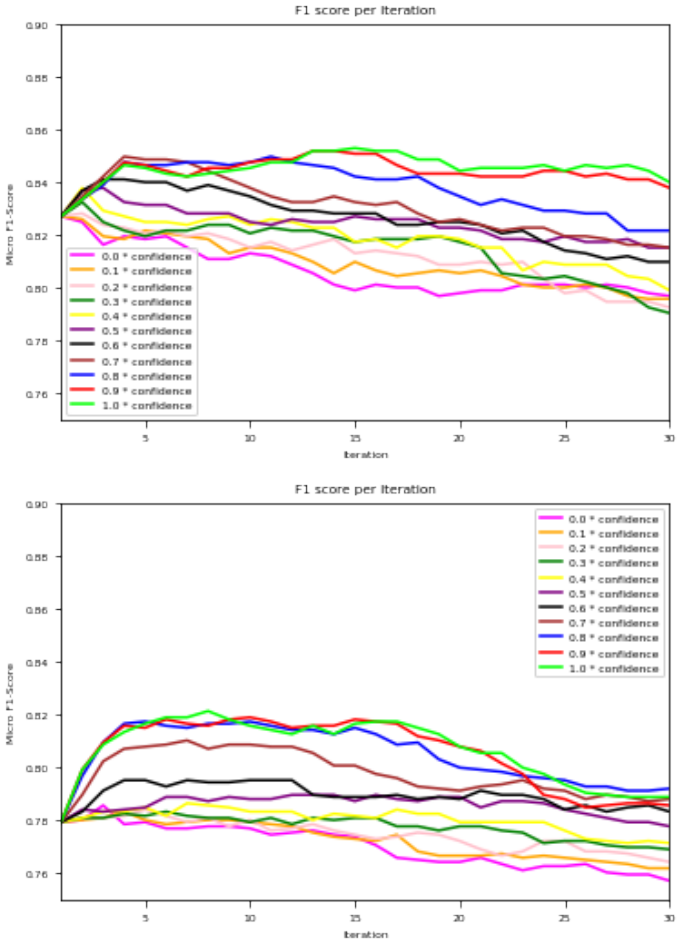

Fig. 3.3: Self-training when the sampling function is a linear combination of the instance confidence and representativeness score. Different values of confidence weight are applied (λ [0, 1]). On the top: System evaluation for the Toxicity Dataset. On the bottom: System evaluation for the Sexism Dataset.

even when applied to models with higher capacity, such as the state-of-the-art NLP models, and further boost their performance. Based on the fact that BERT models are already pre-trained on unsupervised data, a question that arose was whether Self-training captures the same information as pretraining or if these semi-supervised learning methods can be complementary and beneficial to the model when applied together. In addition, we wanted to examine if the sampling techniques that had the best results for the Teacher-Student Logistic Regression framework are equally effective in the Teacher-Student BERT framework application.

In our implementation, Teacher is a classifier that consists of a BERT base LM with a task-specific head (MLP) to classify the instances into Offensive/Sexist or Not Offensive/Not Sexist. The Student is a BERT base model with the same architecture as the Teacher. First, the Teacher is fine-tuned on the initial labeled set L. In each round, the Students weights are initialized, and the Student is fine-tuned on the union of the labeled and silver-labeled examples that the Selector sampled. The Student becomes the Teacher to

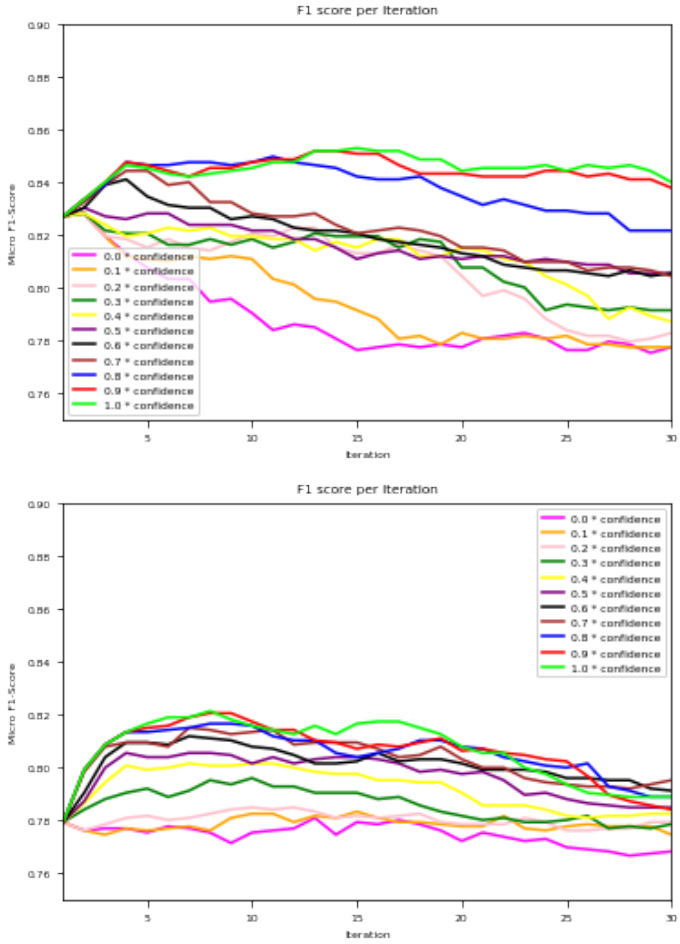

Fig. 3.4: Self-training when the sampling function is a linear combination of the instance confidence and diversity score. Different values of confidence weight are applied (λ [0, 1]). On the top: System evaluation for the Toxicity Dataset. On the bottom: System evaluation for the Sexism Dataset.

generate predictions for the rest of the unlabeled dataset U at the end of the round as the Self-training algorithm suggests .

We experimented with the sampling techniques that had the best results on the Logistic Regression framework. More specifically, we applied confidence sampling by adding the top-k most confident silver-labeled instances or all instances with a confidence score above a certain threshold iteratively to train the Student. We also applied combined confidence and representativeness scores as factors contributing to the selection of the unannotated instances.

3.8 Full - Supervision

To evaluate the results of our best Teacher-Student methods, we also employed full supervision for comparison reasons. Specifically, we trained both a Logistic Regression classifier and a BERT model with the same architecture as the Teacher of our previous experiments with the entire labeled dataset without removing the ground-truth labels of the remaining training set to be used as unlabeled data. The results of our experiments are presented in the next section.

4 Evaluation

We present the dataset statistics, the configurations of our experiments, the evaluation measures we used and our experimental results.

4.1 Datasets

4.1.1 OGTD

The Offensive Greek Tweet Dataset (OGTD) is a publicly available dataset for offensive language identification [PZR20]. Although there are many toxicity datasets in English, it was the first Greek annotated dataset for the specific task [PZR20]. It contains a total of 10,287 posts from Twitter labeled as Offensive or Not Offensive. The posts were collected using popular hashtags, mostly from television programs such as reality and entertainment shows and hashtags concerning political events like European parliament elections. The researchers note that they focused on these categories as TV and politics usually gather more disputes and insulting language than other topics.

The dataset contains offensive tweets of different types (racist, sexist, etc.). It consists of a training subset of 8,743 tweets and a test subset of 1,544 tweets. The dataset is imbalanced as most tweets are labeled as Not Offensive, as shown in Figure 4.1. The training data consist of 2,486 offensive tweets and 6,257 not offensive Tweets, with an offensive class ratio of 28% of the total set. The test dataset is even less balanced as it consists of 242 offensive tweets and 1,544 not offensive tweets with an offensive class ratio of 15% of the total set. To experiment with datasets with the same class balance, we united the training and test data and produced a random split for the test set. In addition, the average length of training and test sequences is 18 words, and the max sequence length is 72 tokens.

4.1.2 Explainable Detection of Online Sexism Dataset

The Explainable Detection of Online Sexism Dataset (EDOSD) was introduced for SemEval 2023 for the task of Sexism Detection. It consists of 20,000 user English posts sampled from Gab and Reddit. All entries are labeled as Sexist or Not Sexist by human annotators. The available training data consists of 14,000 entries. The validation and test data will be

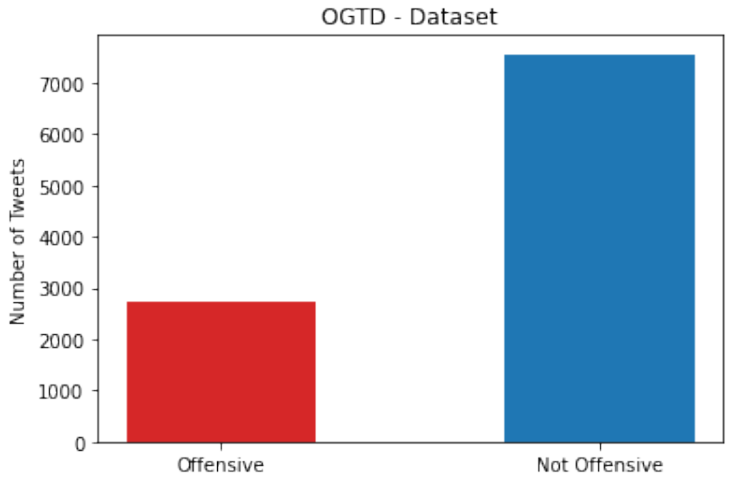

Fig. 4.1: OGTD Class distribution

released at the final phase of the competition, so they cannot be used in this thesis. The training set is imbalanced as there are 3,398 sexists samples (24% of the entire dataset) as shown in Figure 4.2. The training samples have an average length of 27 words and a max length of 84 tokens.

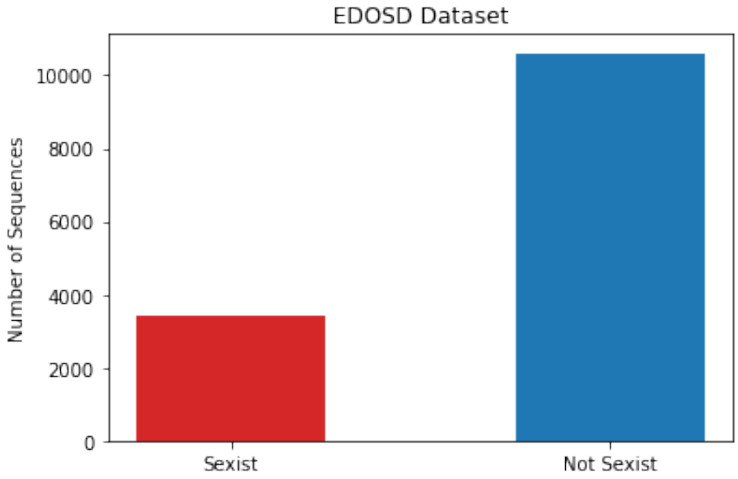

Fig. 4.2: EDOSD Class distribution

4.2 Few-shot settings

As mentioned in Chapter 3, we reproduced few-shot settings for our experiments. We randomly sampled 2,000 instances from the training subset as our initial labeled set (L) for both datasets. We detached 10% of the original training data for our validation set (V) and 10% as the test set (T) to evaluate our final results. The rest of the supervised set was used as our unlabeled dataset (U) for both datasets. Tables 4.1 and 4.2 summarize the statistics of our settings.

	Offensive	Not Offensive	Total	% Offensive
Labeled Set (L)	527	1473	2000	26%
Unlabeled Set (U)	1697	4635	6332	27%
Validation Set (V)	240	686	926	26%
Test Set (T)	264	765	1029	26%

Tab. 4.1: OGTD - Few shot settings

	Sexist	Not Sexist	Total	% Sexist
Labeled Set (L)	484	1516	2000	24%
Unlabeled Set (U)	2246	7094	9340	24%
Validation Set (V)	316	944	1260	25%
Test Set (T)	352	1048	1400	25%

Tab. 4.2: EDOSD - Few shot settings

4.3 Evaluation metrics

We evaluated the final Student model after applying Teacher-Student with the techniques and configurations described in Chapter 3 on its performance on the test set based on the evaluation metrics described in this Section.

We report the F1-Score of each class to test the success of our methods for detecting both the minority and the most frequent class. The F1-Score of each class is the harmonic mean of class precision and recall, respectively, and can be defined by the following equations:

$$precision_c = \frac{TP_c}{TP_c + FP_c} \tag{4.1}$$

$$recall_c = \frac{TP_c}{TP_c + FN_c} \tag{4.2}$$

$$F1\text{-}score_c = 2 \cdot \frac{precision_c \cdot recall_c}{precision_c + recall_c} \tag{4.3}$$

TP_c, FP_c, FN_c refer to the True Positive, False Positive and False Negative predictions respectively, for each class c.

We also set as evaluation metric micro F1-Score, which computes the global average F1-score by counting the sum of the True Positives (TP), False Negatives (FN), and False Positives (FP) of the two classes. We used this metric to evaluate our system because we wanted to give equal importance to all instances regardless of their class. Our purpose

was to train a Student model that would correctly classify the maximum number of test instances irrespective of their category. Micro-averaged F1-score constitutes the harmonic mean of micro-precision and micro-recall, which are defined as follows:

$$\textit{micro-precision} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FP_c} \tag{4.4}$$

$$\textit{micro-recall} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FN_c} \tag{4.5}$$

$$\textit{micro F1-Score} = 2 \cdot \frac{\textit{micro-precision} \cdot \textit{micro-recall}}{\textit{micro-precision} + \textit{micro-recall}} \tag{4.6}$$

We configured the number of Teacher-Student iterations for each technique based on the micro F1-Score of the Student evaluated on the validation set. However, in some of our experiments, we observed that although micro F1-Score increased, there was a considerable decrease in the minority class evaluation. That means that although our classifier labels more test examples correctly, it cannot effectively predict the Offensive/Sexist instances that occur more rarely. In these cases, we used both micro F1-Score and F1-Score of the minority class to monitor the number of Teacher-Student iterations. An alternative evaluation metric to monitor the configurations of our experiments could have been macro F1-Score, as it assigns equal weight to both categories regardless of their frequency.

This thesis aimed to examine which sampling method had the best impact on our system by leading to the highest values of the evaluation metrics. In addition, as in real case scenarios, the resources we could use for the Teacher-Student iterative process were limited. Hence, we also considered the number of rounds and the number of unlabeled samples required for each technique to reach the highest scores to conclude its success.

4.4 Training Details

4.4.1 Text prepossessing

To transform the toxicity dataset into a clean and consistent format, we followed some text preprocessing steps. Specifically, we removed tweet hashtags (tokens that start with #) and usernames (tokens that begin with @). In addition, we removed URLs and converted all sequences to lowercase. Finally, as it is a Greek dataset, we removed Greek accents and excluded stop words.

The dataset we used for the task of sexism detection contains user posts in English. We removed punctuation, numbers and single characters from the posts to preprocess the text. We converted all tokens to lowercase and used a lemmatizer which converts any word to its base form (lemma) as a text normalization technique. Finally, we removed stop words.

4.4.2 Logistic Regression

Considering our few-shot learning scenario, we set the Logistic Regression solver as "liblinear" as it is the suggested option for small datasets by scikit-learn library[1]. We performed tuning using grid search to determine the model's hyperparameter values. We specified as hyperparameters the regularization term with candidates L1 and L2 penalty, the inverse of regularization strength C (defined in the interval [0.1, 1.0]), and the weights assigned at each class with candidates: "balanced" and "None." With no weights assigned to each label, all classes are supposed to have equal weights. In contrast, with "balanced" weights, the algorithm "automatically adjusts weights inversely proportional to class frequencies in the input data" (based on the scikit-learn documentation). We applied 5-fold cross-validation on the initial labeled dataset (L) to compute the hyperparameter values. The evaluation metric was the micro F1-score. For text vectorization, we converted the sequences into TFIDF feature vectors with unigrams and bigram features.

4.4.3 BERT

For our BERT experiments on the Greek tweets dataset, we used GreekBERT. It a Greek version of uncased BERT-base introduced by AUEB's NLP Group [Kou+20]. It is pre-trained on large Greek corpora, such as the Greek part of Wikipedia. Regarding the (English) sexism dataset, we used BERT-base uncased, a masked LM introduced by Google [Dev+19], which was pre-trained on general English corpora. Our final classifier for both tasks consisted of BERT followed by a task-specific MLP head.

In both cases, we set the top-level embedding of the CLS special token as the output from the transformer model. We performed a hyperparameter search to determine the values of the following parameters: the number of layers of the task-specific MLP in the range of [1, 5], the number of units (neurons) of each layer with distinct candidates, the values: [64, 128, 256, 512, 1024] and the activation function of each layer with candidates: relu and tanh activation functions. We also considered the dropout rate between MLP layers as a hyperparameter and tuned it in the interval [0, 0.5].

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`

We first trained only the MLP layers by keeping the transformer layers frozen with a higher learning rate for 50 epochs and set early stopping with micro F1-Score as the monitor. Then, we fine-tuned the model with a lower learning rate by keeping frozen the embeddings layer and the first two encoder blocks. We set the learning rate's search space when training the MLP layers in the interval $[10^{-4}, 10^{-2}]$. For the fine-tuning of the entire network (by keeping only the lowest transformer layers frozen), we tuned the learning rate in the interval $[10^{-8}, 10^{-5}]$. We used the Adam optimizer [KB14]. We used one unit (neuron) at the network's output layer with the sigmoid activation function as it concerns a binary classification task.

4.5 Experimental Results

4.5.1 Teacher - Student with Logistic Regression

Tables 4.3 and 4.4 depict the performance of our Teacher-Student approach when top-k is applied. As stated in Section 3.2.1, the Selector function samples k instances with the highest confidence score s_x to be added to the training set at the end of each round. The number of iterations for different k values was determined by monitoring performance on the validation set. We set the number of iterations equal to the round where the micro F1-Score starts to drop based on the validation set. We present the results of our method for different values of k, and the evaluation of our system when trained only with the initial limited labeled data (L) for comparison reasons.

In addition, we report the Offensive/Sexist class ratio of the training data (labeled and pseudo-labeled that were used in each method), the number of rounds and the number of total samples required for training.

	Micro F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.831	0.644	0.889	26%	1	2000
$k = 20$	0.849	0.640	0.904	37%	50	2980
$k = 50$	**0.858**	0.662	**0.910**	39%	26	3250
$k = 100$	**0.858**	**0.663**	**0.910**	38%	15	3400
$k = 200$	0.854	0.657	0.907	37%	8	3400

Tab. 4.3: Self-training with confidence sampling (OGTD Dataset).

For the toxicity dataset, the Teacher - Student approach with confidence sampling seems to considerably impact our system's performance for the few-setting scenario. We observed that different values of k do not produce a remarkable difference in the evaluation results. However, when setting $k = 50$ or $k = 100$, our system gains 2.7 percentage points for the micro F1-score and approximately 2 percentage points for each class F1-Score. We conclude that the best method was adding the top 100 (highest confidence) instances to the

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.779	**0.577**	0.850	24%	1	2000
$k = 20$	0.816	0.524	0.886	38%	37	2740
$k = 50$	**0.817**	0.527	**0.886**	38%	16	2750
$k = 100$	0.815	0.525	0.885	38%	8	2700
$k = 200$	**0.817**	0.532	**0.886**	37%	4	2600

Tab. 4.4: Self-training with confidence sampling (EDOSD Dataset).

training set in each iteration as it reached the highest scores in fewer rounds than when setting $k = 50$.

For both datasets, we observe that as we increase k, the number of rounds needed for this process drops. If our resources are limited and it is impossible (or too costly) to apply Teacher-Student for many rounds, we should set a higher k as configuration. In addition, we observe that the class ratio of the augmented training set changed after self-training (38% Offensive instances compared to 26% Offensive samples of the labeled set and 37% Sexist samples compared to 24% initially labeled Sexist examples). The classifier seems more confident when labeling an instance as Offensive/Sexist, as it assigns a higher probability to the examples pseudo-labeled as the minority class.

Figure 4.3 presents the curves of the Student's micro F1-score per Teacher-Student iteration where the top-k sampling technique is applied (by setting $k = 100$). To compare the results of this method, we also present the Student's micro F1-score per iteration when the same instances from U are sampled, but their true labels are available. We observe that during the first twenty iterations concerning the toxicity detection task, the Teacher-Student framework performs even better than supervised learning. For the sexism detection task, the Teacher-Student framework performs similarly to training under supervision for more than ten iterations. However, as the iterations increase, the performance of the Student model drops when Teacher-Student is applied.

In the case of Teacher-Student for the sexism detection task, we observe that although micro F1-Score increased for 3.8 percentage points at the best setting ($k = 200$), the F1-Score of the Sexist Class decreased considerably (-4.5 percentage points) as shown in Table 4.4. That means that when our classifier is trained with silver labels, it struggles to detect sexist cases.

We performed two experiments based on the top-k technique for the second dataset to improve the evaluation of the minority category (Sexist class) detection, as described in Section 3.2.1. The first experiment was monitoring the Teacher-Student rounds based on the F1-Score of the Sexist class when evaluating our approach on the validation set (instead of micro F1-Score). We managed to increase the micro F1-Score and F1-Score of the majority class (Not Sexist class), compared to the system evaluation when training

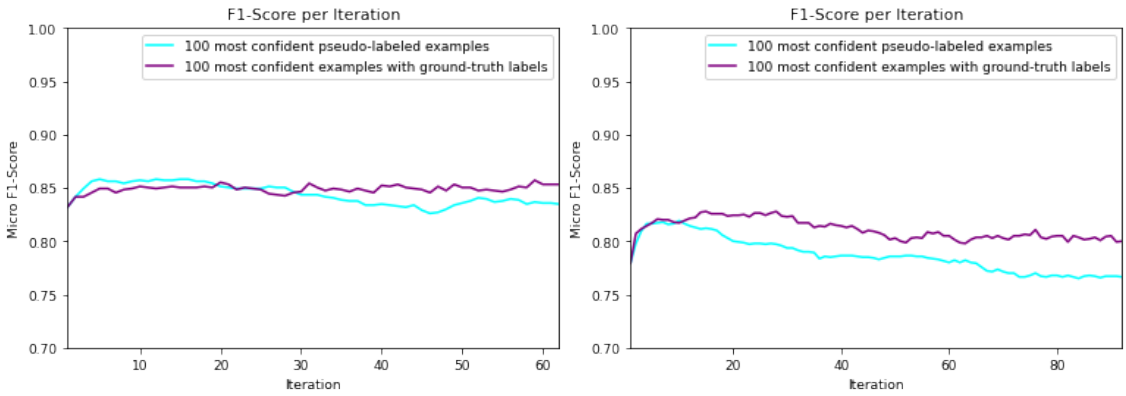

Fig. 4.3: Micro F1-Score per iteration when applying Teacher-Student with the top-k tehcnique compared to iterative supervised learning with the most confident examples from U and their true labels. On the left: System evaluation for the Toxicity Dataset. On the right: System evaluation for the Sexism Dataset.

only with L, and to maintain the F1-Score of the minority class (Table 4.5). For instance, when setting $k = 200$, our system gains 2.3 percentage units for micro F1-Score and 2.0 units regarding evaluating the majority category. The detection of the sexist instances did not deteriorate, contrary to applying Teacher-Student for the number of rounds that maximize micro F1-Score of the validation set (Table 4.4).

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.779	0.577	0.850	24%	1	2000
$k = 50$	**0.812**	0.568	**.879**	33%	30	3450
$k = 100$	0.810	0.569	0.878	32%	17	3600
$k = 200$	0.802	**0.578**	0.870	28%	14	4600

Tab. 4.5: Self-training with confidence sampling (EDOSD Dataset). Monitor F1-Score of Sexist Class to determine the number of Teacher-Student rounds.

We observe that class balance has an impact on our experimental results. Hence as mentioned in section 3.2.1, we performed a second experiment to add batches of the most confident silver-labeled examples by preserving the class balance of the training set (24% Sexist, 76% Not Sexist). We also tried to augment the initial labeled set by adding the most confident examples of different class ratios (25%, 26%, 27% and 28% sexist instances and 75%, 74%, 73%, 72% not sexist instances). Preserving the class balance (adding 24% Sexist and 76% Not Sexist silver-labeled examples at each round) improved few-shot learning results. However, when setting each batch class ratio as 28% Sexist and 72% Not Sexist, our system reached even better results after the Teacher-Student application. Micro F1-Score increased by 2 percentage units, the F1-Score of the Not Sexist class gained 1.9 percentage units, and the F1-Score of the Sexist class gained 0.4 percentage units (Table 4.6).

Tables 4.7, 4.8 show the results of Self-training with confidence sampling when we add iteratively all samples with a confidence score above a certain threshold t. We present the results of this method for different values of t.

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Round	Samples
Initial L (24% Sexist)	0.779	0.577	0.850	1	2000
24% Sexist	0.785	**0.582**	0.855	9	2800
25% Sexist	0.785	**0.582**	0.855	9	2800
26% Sexist	0.790	0.579	0.860	14	3300
27% Sexist	0.797	**0.582**	0.866	14	3300
28% Sexist	**0.800**	0.581	**0.869**	15	3400

Tab. 4.6: Self-training with confidence sampling (EDOSD Dataset). Adding k most confident samples with fixed class ratio.

	Micro F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.831	0.644	0.889	26%	1	2000
$t = 0.84$	0.855	0.659	0.908	38%	8	3060
$t = 0.87$	**0.859**	**0.668**	**0.910**	38%	2	2381
$t = 0.90$	**0.859**	0.665	**0.910**	37%	3	2340
$t = 0.93$	0.855	0.657	0.908	35%	5	2272
$t = 0.96$	0.846	0.651	0.901	32%	4	2166
$t = 0.99$	0.838	0.646	0.895	28%	2	2055

Tab. 4.7: Self-training with confidence sampling over a threshold (OGTD Dataset).

For the toxicity dataset, confidence sampling above a certain threshold significantly improved our system's performance compared to training with the labeled dataset only, as shown in table 4.7. In addition, intermediate values of thresholds were the best for our system ($t = 0.87$, $t = 0.90$). When setting $t = 0.87$, our system gained 2.8 micro F1-Score percentage units, 2.4 units at the evaluation of the Offensive category and 2.1 units regarding the F1-Score of the Not Offensive class. Although the results are comparable to the top-k technique, there is a slight improvement. In addition, these results were reported after only 1 round of Self-training, and only 381 unlabeled samples were required. Hence, this technique is also more resource efficient (compared to the top-k approach, where 14 TS rounds and 1,400 unlabeled examples were required when setting $k = 100$).

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.779	0.577	0.850	24%	1	2000
$t = 0.84$	0.760	**0.580**	0.832	25%	7	8554
$t = 0.87$	0.788	0.574	0.859	28%	28	5206
$t = 0.90$	0.810	0.556	0.879	34%	24	3321
$t = 0.93$	**0.818**	0.541	**0.884**	37%	3	2523
$t = 0.96$	0.810	0.557	0.879	32%	2	2240
$t = 0.99$	0.797	0.575	0.867	27%	2	2094

Tab. 4.8: Self-training with confidence sampling over a threshold (EDOSD Dataset).

About the sexism dataset, we observe that when setting a low threshold ($t = 0.84$) F1-score of the Sexist class is preserved; however micro F1-score and F1-Score of the majority class decrease. When setting higher thresholds ($t = 0.87$, $t = 0.90$, $t = 0.93$, $t = 0.96$), the

results are the opposite. The most effective threshold for our experiment was $t = 0.99$, where the system marked an insignificant decrease at the Sexist class evaluation but micro F1-Score and the Not Sexist class F1-Score increased by 1.8 and 1.7 percentage units, respectively. In addition, for the sexism dataset too, this technique is more resource efficient than the top-k technique as only 1 Teacher-Student round and 94 unlabeled examples are required to reach these scores.

Tables 4.9, 4.10 show the results of confidence sampling when an adaptive number k of samples is added to the training set. Specifically, we experimented by increasing the value of k in each iteration. In the first case, k increases by 20, while in the second case, k doubles at the end of each round. In the third experiment, when Student micro F1-Score decreases or stays stable at the end of a round based on the validation set, k is maintained the same and when micro F1-Score increases, k doubles.

	Micro F1	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.831	0.644	0.889	26%	1	2000
$k = k + 20$	0.853	0.654	0.906	37%	12	3320
$k = k \quad 2$	0.856	0.658	0.908	36%	5	2300
$k = k/k = k \quad 2$	**0.857**	**0.658**	**0.909**	39%	7	2660

Tab. 4.9: Self-training with confidence sampling with adaptive k (OGTD Dataset).

	Micro F1	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.779	0.577	0.850	24%	1	2000
$k = k + 20$	**0.797**	0.569	**0.867**	30%	15	4100
$k = k \quad 2$	0.794	**0.582**	0.863	26%	3	2060
$k = k/k = k \quad 2$	0.789	0.576	0.859	26%	11	5820

Tab. 4.10: Self-training with confidence sampling with adaptive k (EDOSD Dataset).

This method did not result in a better evaluation score of our system than when adding a fixed number of instances or instances with a confidence score above a certain threshold. However, increasing the number of samples in each iteration seems to reach a very good performance in a few rounds. For the first dataset, doubling k in each iteration reaches good scores in only 4 rounds of Teacher-Student compared to 7 rounds when k was stable ($k = 200$). Only 2 rounds and 60 unlabeled instances were required for the sexism dataset. In conclusion, increasing k in each iteration is more resource efficient than keeping k stable (top-k technique).

Tables 4.11, 4.12 illustrate our system evaluation when the Self-training with the select-all technique is applied. In each iteration, the entire unlabeled dataset and the pseudo-labels generated by the Teacher are added to the training set.

The select-all technique performed poorly in our system, resulting in a lower micro F1-Score on the test set compared to training only with the initial labeled set. For the toxicity

	Micro F1-Score	F1-Score Off/ve	F1-Score Not Off/ve	Round
Initial L only	**0.831**	**0.644**	**0.889**	1
Select-all	0.819	0.632	0.880	2

Tab. 4.11: Self-training with select-all technique (OGTD Dataset).

	Micro F1-Score	F1-Score Sexist	F1-Score Not Sexist	Round
Initial L only	**0.769**	0.578	**0.841**	1
Select-all	0.765	**0.581**	0.837	2

Tab. 4.12: Self-training with select-all technique (EDOSD Dataset).

dataset, the best results were reported at the first round of Teacher-Student training, and as the iterative process continued, micro F1-Score kept decreasing until round 10. For the next 20 rounds, micro F1-Score practically stayed stable, as shown in Figure 4.4 (the left sub-figure represents micro F1-Score per iteration curves for the toxicity dataset). The diagram contains the curve of Student micro F1-Score per iteration when the select-all technique is applied compared to Student micro F1-Score per iteration when the top-k technique is used. The results suggest that our initial Teacher model is not robust enough to label a significant portion of U correctly, and the Student is sensitive to wrong Teacher predictions. We reached that conclusion given that the Student's performance drops when the entire silver-labeled U is added to the training set. On the contrary, Student's performance improves when L is augmented with confident silver-labeled instances.

For the sexism dataset, the best results of the Teacher-Student approach with the select-all technique were also obtained in the first round, where the F1-Score of the Sexist category gained 0.3 percentage points (Table 4.12). However, the other two metrics dropped. Figure 4.4 shows the micro F1-Score per iteration when the select-all technique is applied compared to the top-k approach for comparison reasons (the right sub-figure represents micro F1-Score per iteration curves for the sexism dataset).

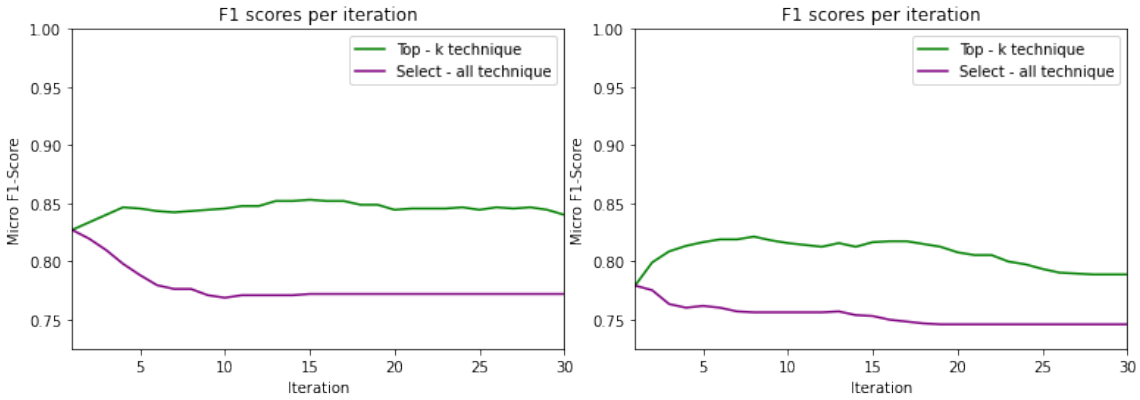

Fig. 4.4: Micro F1-Score per Teacher-Student iteration when applying the top-k and select-all technique. Evaluation of the validation set. On the left: System evaluation for the Toxicity Dataset. On the right: System evaluation for the Sexism Dataset.

Tables 4.13, 4.14 summarize experimental results when Active Learning criteria are applied for the sample selection in each round. In the first case, the Selector function samples the k most representative instances of the unlabeled dataset. In the second case, it samples the k most diverse examples compared to those already in the training data pool. We tested these techniques with different values of k and set it equal to 100 as it had the highest score at the validation set. We also present the results of the top-k technique (confidence sampling) when setting $k = 100$ for comparison reasons. For the second dataset, the 100 most confident examples follow the class distribution that had the best results on our system (28% Sexist, 72% Not Sexist).

	Micro F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.831	0.644	0.889	26%	1	2000
Top-k	**0.858**	**0.663**	**0.910**	38%	15	3400
Representativeness	0.825	0.632	0.885	27%	2	2100
Diversity	0.821	0.633	0.881	25%	2	2100

Tab. 4.13: Self-training with Active Learning Criteria (OGTD Dataset).

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.779	0.577	0.850	24%	1	2000
Top-k	**0.800**	0.581	**0.869**	28%	15	3400
Representativeness	0.778	**0.585**	0.848	25%	3	2200
Diversity	0.766	0.584	0.837	23%	13	3200

Tab. 4.14: Self-training with Active Learning Criteria (EDOSD Dataset).

Active Learning queries the most informative instances to be labeled, usually by a human annotator. This method turns the initial classifier into a more robust model with the least training examples. However, applying the Teacher-Student approach with Active Learning criteria for sampling instances from U does not seem to perform well. The representativeness criterion had better results than diversity for both datasets. Still, micro F1-Score decreased after applying Self-training based on diversity and representativeness sampling compared to the few-shot learning. In addition, almost all evaluation metrics are lower than Teacher-Student with confidence sampling. The bad performance of our approach with Active Learning criteria as a sampling technique may be due to the noisy predictions of the original Teacher model. The Student's performance seems to be affected by the initial wrong pseudo-labels created by the Teacher.

We present the diagram of the Student's micro F1-Score per Teacher-Student iteration when Active Learning criteria are applied compared to micro F1-Score per iteration when we use confidence sampling to augment L iteratively for the toxicity dataset (Figure 4.5). In addition, the diagram contains the curve of micro F1-Score per iteration when we apply random sampling (we randomly select k silver-labeled examples from U in each Teacher-Student round) for comparison reasons.

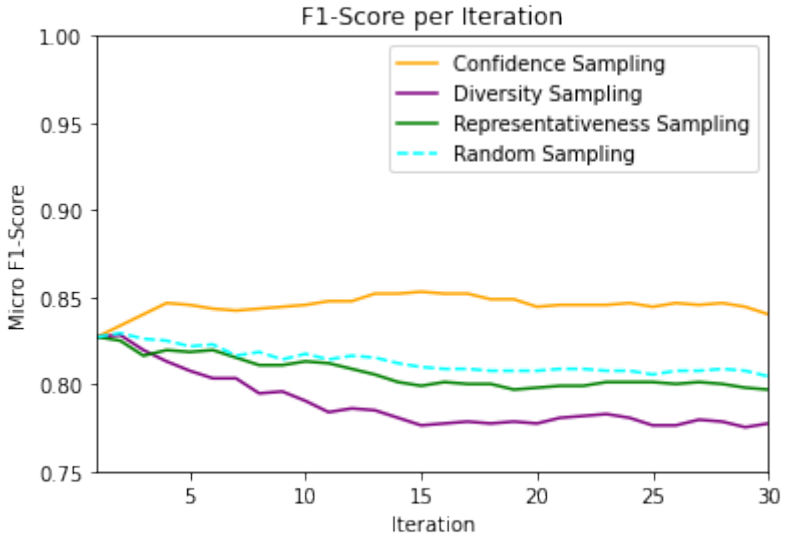

Fig. 4.5: Micro F1-Score per iteration when applying Teacher-Student with top-k, Active Learning criteria and random-k as sampling techniques. Evaluation on the validation set (OGTD Dataset).

Tables 4.15, 4.16 demonstrate the system evaluation when confidence sampling is combined with Active Learning criteria. As explained in Section 3.6, we applied a linear combination of confidence and representativeness scores to sample instances from the unlabeled data pool. We reproduced the same experiment using a linear combination of confidence score and diversity from the instances already in the labeled data pool. We weighted the model's confidence score by $\lambda = 0.9$ and the instance representativeness of U/diversity from L score by $1 - \lambda = 0.1$ (after manually tuning the weight of each factor, Section 3.6).

	Micro F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.831	0.644	0.889	26%	1	2000
Confidence	**0.858**	0.663	**0.910**	38%	15	3400
Conf/ce+Repr/ness	0.855	0.659	0.908	38%	13	3200
Conf/ce+Div/ty	**0.858**	**0.666**	0.909	38%	14	3300

Tab. 4.15: Self-training with Confidence and Active Learning Criteria (OGTD Dataset).

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.779	0.577	0.850	24%	1	2000
Confidence	**0.800**	**0.581**	**0.869**	28%	15	3400
Conf/ce+Repr/ness	**0.800**	0.574	**0.869**	30%	20	3900
Conf/ce+Div/ty	0.790	0.575	0.861	27%	36	5500

Tab. 4.16: Self-training with Confidence and Active Learning Criteria (EDOSD Dataset).

The combination of confidence and Active Learning criteria did not improve our system's performance. For both datasets, the evaluation metrics had almost the same values with confidence sampling. That result could be explained by having assigned higher importance to the instance's confidence score. The only difference we observe is that the same scores

were obtained in fewer iterations when Active Learning criteria were applied (13 and 14 iterations compared to 15 when confidence is applied) for the toxicity detection task. However, in the case of the sexism detection, more Teacher-Student rounds were required compared to confidence sampling.

4.5.2 Comparison to Full Supervision

In this Section, we compare our experimental results after applying the best Teacher-Student sampling techniques of Section 4.5.1 to fully supervised learning. In the case of full supervision, we use the entire training set (not only the limited initial labeled set L) with the ground-truth labels to train our initial classifier without applying Teacher-Student.

	Micro F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.831	0.644	0.889	26%	1	2000
Conf. $t = 0.87$	**0.859**	0.668	**0.910**	**38%**	**2**	**2381**
Full Supervision	0.856	**0.712**	0.904	27%	1	9258

Tab. 4.17: Comparison to Full Supervision (ODTD Dataset).

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.769	0.578	0.841	24%	1	2000
Conf. $k = 100$ (28% sexist)	**0.800**	0.581	**0.869**	28%	15	3400
Full Supervision	0.777	**0.602**	0.845	24%	1	11340

Tab. 4.18: Comparison to Full Supervision (EDOSD Dataset).

As shown in Tables 4.17, 4.18, our system benefits from the Teacher-Student approach in the few-shot learning scenario. Regarding our best technique for the toxicity dataset (confidence sampling over a threshold), we observe that all evaluation metrics significantly increased compared to training with limited training data. Compared to full supervision, where true labels are available for the entire dataset, Self-training performs worse as far as the minority category (Offensive class) is concerned. Specifically, with full supervision, Offensive Class F1-Score is 4.6 percentage units higher than the Teacher-Student framework. However, the other two metrics obtain higher scores than full supervision. In addition, only 2,000 labeled examples, 381 unlabeled examples and 1 round of the Teacher-Student application were needed to reach this performance, opposite to full supervision where 9,258 labeled samples were required.

Teacher-Student has proved to be an effective approach compared to training with limited data n for the task of Sexism detection too. Compared to full supervision, both micro F1-Score and Not Offensive class F1-Score reach higher values (+2.3 and +2.4 percentage units compared to full supervision scores). Again full supervision is more effective in the detection of sexist cases. However, the most competent technique of Self-Training for this dataset, adding top-k instances with fixed class ratio (28% Sexist and 72% Not Sexist) in

each iteration, has satisfying results with only 2,000 labeled examples and 1,400 unlabeled examples.

We conclude that, for imbalanced datasets, the minority class seems to benefit from a larger amount of examples for which the ground-truth labels are available because it is harder for the classifier to detect the category that occurs more rarely. However, Teacher-Student framework with confidence sampling significantly improves the performance of a Logistic Regression classifier when the initial labeled set is limited.

4.5.3 Teacher-Student with BERT

This subsection presents the Self-Training results when the Teacher and Student are pre-trained transformer models. Specifically, we employed the Teacher-Student approach to training a BERT base model. We tested the Logistic Regression Teacher-Student framework's three most efficient sampling methods (based on the criteria of confidence and confidence+representativeness). For the confidence criterion, we set $k = 200$ when the top-k technique is applied and $t = 0.90, t = 0.93$ to reproduce the confidence over a threshold technique for the toxicity and sexism dataset, respectively. For the combination of confidence and representativeness, we set $k = 200$ and the weights of the two factors as $\lambda = 0.8$ and $1 - \lambda = 0.2$, respectively. The performance on the validation set (micro F1-Score) defined the number of iterations for each technique. Table 4.19 and 4.20 show the results of our experiments.

	Micro F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.863	0.703	0.911	26%	1	2000
Top-k ($k = 200$)	**0.877**	**0.758**	0.917	53%	7	3200
Confidence ($p > 0.90$)	0.867	0.723	0.913	51%	3	3015
Conf/ce + Repr/ness	0.875	0.729	**0.919**	33%	2	2200

Tab. 4.19: BERT - Self-training with different sampling techniques (OGTD Dataset).

	Micro F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.806	0.581	0.874	24%	1	2000
Top-k ($k = 200$)	0.814	0.619	0.877	42%	4	2600
Confidence ($p > 0.93$)	0.812	0.622	0.875	47%	3	2889
Conf/ce + Repr/ness	**0.817**	**0.625**	**0.879**	38%	3	2400

Tab. 4.20: BERT - Self-training with different sampling techniques (EDOSD Dataset).

First, we observe that the Self-training boosted the performance of our system when compared to the initial few-shot training scenario for both datasets. All the sampling techniques that performed well on the Logistic Regression Teacher-Student framework also improved our system evaluation scores when Teacher-Student with BERT was applied.

Specifically for the toxicity dataset, when Teacher-Student with the top-k technique was applied, micro F1-score gained 1.4 percentage units. The system performed significantly better in detecting the minority category (+5.8 percentage units), while the results for the majority category were similar to the few-shot learning scenario (+0.06 percentage units). In addition, Self-training based on confidence over a threshold and confidence + representativeness performed pretty well. Although the F1-Score of the Offensive category had a smaller increase (+2.0 units and +2.6 units, respectively), it reached promising results in only 2 and 1 Teacher-Student iteration, respectively. Based on the evaluation of the validation set, these criteria reached higher micro F1-Score during the first 3 rounds than the top-k technique, as shown in figure 4.6. In a low-resource scenario, these criteria could

be preferred as training iteratively a LM of millions of parameters requires significant computing resources.

Results after applying Teacher-Student on top of few-shot learning were similar for the sexism dataset. Confidence sampling with both techniques increased the micro F1-score and F1-Score of the sexist class. However, the best sampling technique for this dataset was the combination of confidence and representativeness as sampling factors. It obtained the highest scores (1.1 and 4.4 percentage units rise of the micro F1-Score and the F1-Score of the Sexist class, respectively). In addition, it was the most resource-efficient technique as it required only 2 rounds of Teacher-Student application and 400 unlabeled examples.

In both scenarios, the Teacher-Student application had the most significant impact on minority class detection, unlike our experiments with Logistic Regression models. The Teacher-Student approach was more beneficial for detecting cases of the majority class when applied on top of a Logistic Regression classifier. In addition, similar to the Logistic Regression framework, confidence over a threshold was more resource efficient than the top-k technique, but in the case of BERT models, the top-k approach reached higher scores for most evaluation metrics for both datasets.

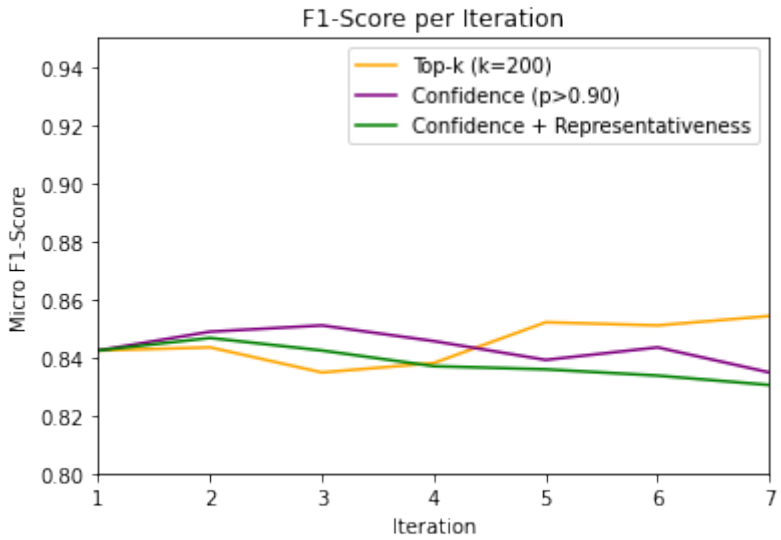

Fig. 4.6: Micro F1-Score per iteration when applying Teacher-Student with different sampling techniques on BERT. Evaluation on the validation dataset.

Finally, we compare the best sampling techniques when Teacher-Student with BERT models is applied to fully supervised learning (Tables 4.21, 4.22). In the case of full supervision, we train the BERT base classifier with the entire training dataset and the ground truth labels for comparison reasons.

Contrary to the Logistic Regression framework, Teacher-Student did not reach the results of full supervision when BERT models were used. Training BERT with the entire labeled

	F1-Score	F1 Off/ve	F1 Not Off/ve	Off/ve%	Round	Samples
Initial L only	0.863	0.703	0.911	26%	1	2000
Top-k (k = 200)	0.877	0.758	0.917	53%	7	3200
Full Supervision	**0.894**	**0.779**	**0.930**	27%	1	9258

Tab. 4.21: BERT - Few-shot learning compared to Self-training and full supervision (OGTD Dataset).

	F1-Score	F1 Sexist	F1 Not Sexist	Sexist%	Round	Samples
Initial L only	0.806	0.581	0.874	24%	1	2000
Conf/ce + Repr/ness	0.817	0.625	0.879	38%	3	2400
Full Supervision	**0.845**	**0.658**	**0.900**	24%	1	11340

Tab. 4.22: BERT - Few-shot learning compared to Self-training and full supervision (EDOSD Dataset).

dataset (9,258 labeled instances for the toxicity detection and 11,340 labeled instances for the sexism detection task) obtained the highest scores in all the evaluation metrics (Tables 4.21, 4.22). However, applying Teacher-Student with the best sampling techniques had satisfying results, with only 2,000 labeled examples available for both datasets.

5 Conclusions And Future Work

In this thesis, we examined the benefits of the Teacher-Student framework in a few-resource scenario. We applied different Self-training techniques to improve the performance of two initial classifiers trained with limited labeled examples for Toxicity and Sexism detection tasks. We studied the effects of Teacher-Student with Logistic Regression and a BERT model of higher capacity to explore whether these methods can improve the performance of even a state-of-the-art NLP model.

Teacher-Student with confidence sampling significantly improved the classifier's performance compared to training with the initial labeled set only. The confidence over a threshold technique was more efficient than the top-k technique. It reached similar or better scores concerning our evaluation metrics with fewer resources required compared to the top-k sampling technique.

In imbalanced datasets, Self-training on a Logistic Regression classifier may improve the majority class evaluation while the minority class evaluation deteriorates. In this case, tuning the class ratio of the most confident silver-labeled examples that are added to the labeled set in each round could improve the minority class detection. Alternatively, we could configure the Teacher-Student framework based on macro-averaged scores. On the contrary, for the BERT model, the most significant improvement after applying Self-training concerned the evaluation of the minority category for both datasets.

Select-all technique and Active Learning criteria (representativeness and diversity) did not perform as well as confidence sampling. The evaluation metrics decreased even when compared to few-shot learning results. When the instance's confidence score is not considered when sampling the silver-labeled examples, the Student model is harmed by the Teacher's wrong predictions. Augmenting the training set without considering the instance confidence score can add more noise to the Student training.

When confidence is combined with Active Learning criteria as a sampling technique, the greatest importance should be given to the instance's confidence score rather than its informativeness score (representativeness/diversity). For the BERT Teacher-Student framework, sampling based on the instance's confidence and representativeness score was more resource efficient than the top-k technique or even obtained higher scores (in the case of Sexism detection). The method's efficiency is important when large pre-trained transformers are used, as training them iteratively could be challenging. On the contrary,

when a Logistic Regression classifier was used, the combination of confidence and Active Learning criteria did not reach the performance of confidence sampling.

Finally, Teacher-Student approach showed promising results compared to fully supervised learning. In the case of Logistic Regression, Teacher-Student obtained even higher scores than full supervision in some of the evaluation metrics used. For the BERT classifier, fully supervised learning outperformed all Teacher-Student techniques. However, in the case of Self-training, we only used the 22% and 18% of the labeled examples used in supervised learning for the tasks of Toxicity and Sexism detection, respectively.

In future work, we would like to apply a linear combination of all factors examined in this thesis when sampling the silver-labeled instances. Specifically, we would like to compute each instance's confidence + representativeness + diversity score to examine if combining these factors could benefit our system when using this score as a sampling criterion.

In addition, we would like to test the abilities of the Teacher-Student when the initial model is trained under full supervision (the initial task-specific labeled examples are not limited). We want to examine whether Self-training could improve the performance of an initial classifier that is trained with sufficient samples andalready performs significantly well.

Finally, in cases with an abundant set of unlabeled examples, a method often applied in the corresponding bibliography is further pretraining LMs with task-specific unannotated data. It would be interesting to examine Teacher-Student abilities on top of a task-specific pre-trained transformer model to determine if these two techniques could be complementary and produce better results.

Bibliography

[Aro07] Shilpa Arora. "Active Learning for Natural Language Processing Literature Review". In: *Language Technologies Institute School of Computer Science Carnegie Mellon University*. 2007.

[BM00] Avrim Blum and Tom Mitchell. "Combining Labeled and Unlabeled Data with Co-Training". In: *Proceedings of the Annual ACM Conference on Computational Learning Theory*. 2000.

[Bro+20] Tom B. Brown, Benjamin Mann, Nick Ryder, et al. "Language Models Are Few-Shot Learners". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. 2020.

[BSM21] Meghana Moorthy Bhat, Alessandro Sordoni, and Subhabrata Mukherjee. "Self-training with Few-shot Rationalization". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021.

[Che+21] Yiming Chen, Yan Zhang, Chen Zhang, et al. "Revisiting Self-training for Few-shot Learning of Language Model". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021.

[Chi+20] Patricia Chiril, Véronique Moriceau, Farah Benamara, et al. "An Annotated Corpus for Sexism Detection in French Tweets". English. In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*. 2020.

[Dav+17] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. "Automated Hate Speech Detection and the Problem of Offensive Language". In: *Proceedings of the International AAAI Conference on Web and Social Media*. 2017.

[Dev+19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019.

[Du+21] Jingfei Du, Edouard Grave, Beliz Gunel, et al. “Self-training Improves Pre-training for Natural Language Understanding”. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021.

[Fen+21] Steven Y. Feng, Varun Gangal, Jason Wei, et al. “A Survey of Data Augmentation Approaches for NLP”. In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. 2021.

[GG16] Yarin Gal and Zoubin Ghahramani. “Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning”. In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. 2016.

[GKP11] Yufan Guo, Anna Korhonen, and Thierry Poibeau. “A Weakly-supervised Approach to Argumentative Zoning of Scientific Documents”. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. 2011.

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the Knowledge in a Neural Network”. In: *NIPS 2014 Deep Learning Workshop*. 2015.

[Hwa00] Rebecca Hwa. “Sample Selection for Statistical Grammar Induction”. In: *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. 2000.

[Jan05] Martin Jansche. “Maximum Expected F-Measure Training of Logistic Regression Models”. In: *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. 2005.

[Jia+21] Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. “How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering”. In: *Transactions of the Association for Computational Linguistics* (2021).

[KB14] Diederik Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations*. 2014.

[KCC20] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. “Data Augmentation using Pre-trained Transformer Models”. In: *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*. 2020.

[Kou+20] John Koutsikakis, Ilias Chalkidis, Prodromos Malakasiotis, and Ion Androutsopoulos. “GREEK-BERT: The Greeks Visiting Sesame Street”. In: *11th Hellenic Conference on Arti cial Intelligence*. 2020.

[KW13] Irene Kwok and Yuzhou Wang. “Locate the Hate: Detecting Tweets against Blacks”. In: *Proceedings of the Twenty-Seventh AAAI Conference on Arti cial Intelligence*. 2013.

[Lee+19] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, et al. “BioBERT: a pre-trained biomedical language representation model for biomedical text mining”. In: *Bioinformatics*. 2019.

[LG94] David D. Lewis and William A. Gale. "A Sequential Algorithm for Training Text Classifiers". In: *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1994.

[Li+21] Shiyang Li, Semih Yavuz, Wenhu Chen, and Xifeng Yan. "Task-adaptive Pre-training and Self-training are Complementary for Natural Language Understanding". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. 2021.

[Log+22] Robert Logan IV, Ivana Balazevic, Eric Wallace, et al. "Cutting Down on Prompts and Parameters: Simple Few-Shot Learning with Language Models". In: *Findings of the Association for Computational Linguistics: ACL 2022*. 2022.

[Mi+21] Fei Mi, Wanhao Zhou, Lingjing Kong, et al. "Self-training Improves Pre-training for Few-shot Learning in Task-oriented Dialog Systems". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021.

[MN98a] Andrew McCallum and Kamal Nigam. "A comparison of event models for naive bayes text classification". In: *AAAI Conference on Artificial Intelligence*. 1998.

[MN98b] Andrew McCallum and Kamal Nigam. "Employing EM and Pool-Based Active Learning for Text Classification". In: *International Conference on Machine Learning*. 1998.

[Niu+20] Yilin Niu, Fangkai Jiao, Mantong Zhou, et al. "A Self-Training Method for Machine Reading Comprehension with Soft Evidence Extraction". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020.

[PF17] Ji Ho Park and Pascale Fung. "One-step and Two-step Classification for Abusive Language Detection on Twitter". In: *Proceedings of the First Workshop on Abusive Language Online*. 2017.

[PMA17] John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. "Deeper Attention to Abusive User Content Moderation". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017.

[PMA22] Dimitris Pappas, Prodromos Malakasiotis, and Ion Androutsopoulos. "Data Augmentation for Biomedical Factoid Question Answering". In: *Proceedings of the 21st Workshop on Biomedical Language Processing*. 2022.

[PZR20] Zesis Pitenis, Marcos Zampieri, and Tharindu Ranasinghe. "Offensive Language Identification in Greek". In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*. 2020.

[RS06] Dan Roth and Kevin Small. "Margin-Based Active Learning for Structured Output Spaces". In: *Proceedings of the 17th European Conference on Machine Learning*. 2006.

[Sha+20] Shayne Shaw, Maciej Pajak, Aneta Lisowska, Sotirios A. Tsaftaris, and Alison Q. O'Neil. "Teacher-Student chain for efficient semi-supervised histology image classification". In: *CoRR, DBLP computer science bibliography*. 2020.

[She+04] Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. "Multi-Criteria-based Active Learning for Named Entity Recognition". In: *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. 2004.

[SJ21] Thakur Ashutosh Suman and Abhinav Jain. "AStarTwice at SemEval-2021 Task 5: Toxic Span Detection Using RoBERTa-CRF, Domain Specific Pre-Training and Self-Training". In: *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. 2021.

[Tra+17] Van Cuong Tran, Ngoc Thanh Nguyen, Hamido Fujita, Dinh Tuyen Hoang, and Dosam Hwang. "A combination of active learning and self-learning for named entity recognition on Twitter using conditional random fields". In: *Knowledge-Based Systems*. 2017.

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017.

[Vu+21] Tu Vu, Minh-Thang Luong, Quoc Le, Grady Simon, and Mohit Iyyer. "STraTA: Self-Training with Task Augmentation for Better Few-shot Learning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021.

[WTD17] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. "Ex Machina: Personal Attacks Seen at Scale". In: *Proceedings of the 26th International Conference on World Wide Web*. 2017.

[Wu+06] Wei-Lin Wu, Ru-Zhan Lu, Jian-Yong Duan, et al. "A Weakly Supervised Learning Approach for Spoken Language Understanding". In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. 2006.

[Xie+20] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. "Self-Training With Noisy Student Improves ImageNet Classification". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[Yar95] David Yarowsky. "Unsupervised Word Sense Disambiguation Rivaling Supervised Methods". In: *33rd Annual Meeting of the Association for Computational Linguistics*. 1995.

List of Figures

List of Tables

List of Algorithms