UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Few-shot Classification of Ukrainian News

Author:
Yevhenii SAPOLOVYCH

Supervisor:
Oleksii IGNATENKO

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2024

Declaration of Authorship

I, Yevhenii SAPOLOVYCH, declare that this thesis titled, "Few-shot Classification of Ukrainian News" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

Few-shot Classification of Ukrainian News

by Yevhenii SAPOLOVYCH

Abstract

In the past few years, substantial progress has been made in the development of text classification algorithms that can efficiently generalize on limited training data (a setting called few-shot learning). However, the effort has primarily been focused on the English language, with research on few-shot classification of texts in the Ukrainian language being largely limited. We shorten this gap by suggesting a modified version of Sentence Transformer Fine-tuning (SetFit), and comparing its performance to versatile baselines on a corpus of news articles. Our solution, which we call SetFit Modified, involves data augmentations, self-training, and evaluation based on synthetic data. It outperforms all baselines in a setting with 8 training examples per class. A benchmark of speed and computational costs shows that both original and modified SetFit provide the fastest and most efficient inference among the tested methods, which makes them applicable in a general low-resource scenario[1].

[1]Code to reproduce experiments from this project is available at https://github.com/ysapolovych/few-shot-masters

Acknowledgements

First of all, I would like to express my gratitude to my supervisor, Oleksii Ignatenko, who provided guidance throughout the project and made this work possible.

I would also like to thank everyone at UCU Applied Sciences Faculty for the exceptional master's program, especially Ruslan Partsey and Oleksii Molchanovskyi for their support throughout this journey.

Special thanks go to Ares AI for providing the environment and the hardware that most experiments were run on.

Last but not least, I am grateful to my family and all the people who helped me keep my spirit up in these trying times.

Contents

List of Figures

List of Tables

List of Abbreviations

NLP	**N**atural **L**anguage **P**rocessing
PLM	**P**retrained **L**anguage **M**odel
MLM	**M**asked **L**anguage **M**odel
LLM	**L**arge **L**anguage **M**odel
SotA	**S**tate **o**f **t**he **A**rt
BoW	**B**ag **o**f **W**ords
TF-IDF	**T**erm **F**requency - **I**nverse **D**ocument **F**requency
NN	**N**eural **N**etwork
ICL	**In**-**C**ontext **L**earning
GPU	**G**raphics **P**rocessing **U**nit
FLOPs	**F**loating **P**oint **O**peration**s**
ST	**S**entence **T**ransformers
LoRA	**Lo**w **R**ank **A**daptation
VRAM	**V**ideo **R**andom-**A**ccess **M**emory
CPU	**C**entral **P**rocessing **U**nit

Chapter 1

Introduction

1.1 Background

Text classification is the task of assigning labels to textual documents. It has a wide range of applications in different fields, and encompasses a large set of problems in natural language processing (NLP): sentiment analysis, document categorization, question answering, natural language inference, relation classification [Li et al., 2022] and more.

One of the key challenges of machine learning and text classification in particular is that algorithms typically require large amounts of examples to perform well on unseen data [Li et al., 2022]. However, under many real-life scenarios, obtaining a sizable training dataset is problematic or impossible. Data collection and labeling may incur high costs, or the data may be distributed in such a way that there are rare classes with only a few instances. This issue has spurred the emergence of a framework called few-shot learning, which aims at efficient machine learning with limited training data [Parnami and Lee, 2022].

Few-shot text classification has gained much momentum in the past several years, in particular, due to the advent of pretrained language models (PLMs). A key aspect of PLMs is their ability to interact with inputs written in natural language. They are capable of filling in missing words (like BERT [Devlin et al., 2019] and similar masked language models [MLMs]), or completing and generating text (like GPT-3 [Brown et al., 2020] and other autoregressive models). Much progress in few-shot text classification has been driven by the development and application of prompts, i.e. texts that guide PLMs' answers with instructions or examples. However, prompting can be quite challenging, as it requires domain expertise and an understanding of the PLM's internal workings [Schick et al., 2020]. Therefore a large body of works on the subject is dedicated either to making prompt engineering more efficient and/or automated, or fully abandoning prompts.

1.2 Motivation

Ukrainian NLP has advanced substantially in recent years, largely thanks to the active community that trains SotA models, develops new linguistic tools, and creates datasets. However, many problems that have been extensively written on in English remain completely or almost untouched. One such example is few-shot text classification, the vast majority of the scholarly works on which have been focused on the English language. To the best of our knowledge, there are only a few works that touch upon few-shot classification of Ukrainian texts. This presents a notable research gap, as it has just as wide a variety of applications in Ukrainian as in English. The issue of data scarcity is arguably even more acute for the Ukrainian language

due to the lack of publicly available datasets and fewer sources to collect data from. Bridging this gap would be beneficial to both researchers and practitioners from different fields in need of robust classification models under low-data restrictions. Even when annotation of sufficient data is possible, few-shot text classification may help reduce development costs and save (often inordinate) time spent on labeling.

1.3 Thesis Structure

This paper is structured as follows. Chapter 2 will provide a review of related work and describe existing research gaps. Chapter 3 will present the methodology of the research. Chapter 4 will provide experimental setup, an evaluation of the results, and their discussion. The paper concludes with a summary and future work in Chapter 5.

Chapter 2

Related Work

This chapter will start with a short inquiry into general text classification, as a solid grasp of it is necessary for a deep dive into few-shot algorithms. An overview of few-shot text classification will then be provided, with approaches divided into two major groups based on whether they rely on prompts or not. The chapter will close with an analysis of research gaps.

2.1 Text Classification

From the 1960s to the 2010s, traditional "shallow" learning methods, such as Support Vector Machine, Naive Bayes, K-Nearest Neighbor, and Decision Tree were dominant in text classification [Li et al., 2022]. They heavily rely on extensive feature engineering for the construction of text representation. The most common technique is called the bag of words (BOW), a document-term matrix representing word frequencies in documents [Dogra et al., 2022]. To adjust word weights for their hypothetical importance in a corpus (i.e. more frequent words tend to be less informative/indicative of a class and vice versa), a technique called TF-IDF (Term Frequency - Inverse Document Frequency) is often applied over BOW. TF-IDF and BOW typically require pre-processing like stemming or lemmatization to reduce related words to common stems or lexemes, as well as a reduction of vocabulary size [Palanivinayagam et al., 2023]. Another downside of BOW and TF-IDF is that they do not account for word ordering and context.

The early 2010s saw the rise of deep-learning methods for text classification based on neural networks (NNs), which typically involve end-to-end learning [Li et al., 2022]. One of the most important changes brought by NNs were word embeddings, i.e. mappings of textual data to continuous vectors that allow encoding semantic and syntactic similarities. Textual embedding models like word2vec [Mikolov et al., 2013], doc2vec [Le and Mikolov, 2014], and GloVe [Pennington et al., 2014] allowed for more robust text representations that could be used in conjunction with both "shallow" and NN-based methods. The latter include such architectures as multi-layer perceptron, feed-forward, convolutional, recurrent, and attention-based NNs [Li et al., 2022].

In the late 2010s, transformer-based PLMs emerged, such as BERT [Devlin et al., 2019], GPT [Radford and Narasimhan, 2018], and XLNet [Yang et al., 2019]. PLMs are pre-trained on large textual data mostly in a self-supervised manner and can be adapted to numerous NLP tasks with supervised fine-tuning. Due to the versatility and superior performance of PLMs, the focus of NLP research started shifting towards their application circa 2019, and to this day they remain the state of the art in text classification [Galke et al., 2023].

Model training procedures largely depend on a PLM type, which falls into three main categories. Fine-tuning for text classification with encoder-only models like

BERT, RoBERTa [Liu et al., 2019], ALBERT [Lan et al., 2020], as well as other BERT-family models, typically involves optimizing model weights to receive better embeddings that can be classified with a dedicated layer added on top of a model [Sun et al., 2019]. Encoder-decoder models, such as T5, require all NLP tasks to be in a text-to-text format [Raffel et al., 2020]. For text classification, the model is tasked to predict a word corresponding to a label, and the results are iteratively improved by adjusting the model's parameters. Decoder-only models, like GPT-3 [Brown et al., 2020], take examples or instructions written in natural language as an input and generate text as an output. Modern decoder-only models tend to have billions of parameters and are commonly referred to as large language models (LLMs), therefore this acronym will be used further in the text.

Another paradigm that has gained significant traction is graph NN-based text classification [Galke et al., 2023]. Their application involves the transformation of documents into a graph that can be either homogeneous (containing only document or word-level nodes), or heterogeneous (where multiple features may serve as nodes simultaneously, such as documents, words, labels [Malekzadeh et al., 2021], entities, modeled topics [Linmei et al., 2019], etc.). Graphs are propagated through a graph NN which returns predictions. Galke et al., 2023 conducted experiments showing that so far graph NNs have been unable to outperform transformer-based methods on most of the benchmarks.

As both transformer and graph NN-based methods are typically computationally heavy [Galke and Scherp, 2022], over the past few years some researchers suggested a return to simpler architectures. Liu et al., 2021 proposed a model based on multilayer perceptron showing results competitive with BERT. Some "shallow" machine learning methods have demonstrated an ability to perform on par or even better than PLMs, e.g. Support Vector Machines with TF-IDF [Wahba et al., 2023] and K-Nearest Neighbor on documents compressed with gzip [Jiang et al., 2023b].

2.2 Few-shot text classification

2.2.1 Prompt-based Approaches

Brown et al., 2020 introduced a method called in-context learning (ICL) capable of addressing diverse few-shot textual tasks. ICL relies on the ability of LLMs such as GPT-3 to generate text by following examples and/or explicit instructions in human language called prompts. In its base form, ICL is simple, as it does not involve parameter updates, and is capable of producing solid results, but is also prone to several issues, like sensitivity to prompt design and various biases [Zhao et al., 2021]. Another issue with ICL is that a model must process a whole set of examples to make each prediction.

MLMs, unlike LLMs, cannot follow explicit textual instructions and therefore require a different prompt design. A text classification task for an MLM can be expressed as a cloze phrase (a phrase where one or multiple words are masked) that exploits the model's ability to fill in missing words [Schick and Schütze, 2021a]. For instance, in a prompt "I had a lot of fun! It was [MASK]" a model should predict the best replacement for the [MASK] token. There are two necessary components required for such prompts to work. Firstly, a function is required to convert a classified text into an answerable prompt called a pattern [Schick and Schütze, 2021a] or a template [Hu et al., 2022b]. In the above case, "It was [MASK]." serves as a pattern. Secondly, considering many possible semantically similar words that may be put in place of a mask token, a separate function called verbalizer is defined to map label

words (e.g. "positive") to words from the PLM's vocabulary (e.g. "good," "great", "awesome" etc.). This setup was first implemented by Schick and Schütze, 2021a in an approach called Pattern-Exploiting Training (PET). They fine-tune an ensemble of models, each with a different pattern, and use them to assign soft labels to unlabeled documents. A set of examples expanded with high-confidence predictions is used to either train a "real" final classifier, or intermediate models that would label more texts for training the next generation of MLMs. Tam et al., 2021 modify PET by utilizing more complex loss functions, thus improving the performance on natural language inference tasks and mitigating the need for unlabeled examples and ensembling. Gao et al., 2021 fine-tune a single MLM in PET fashion, but also use ICL-like demonstrations in prompts at both training and inference time for better accuracy. Wang et al., 2021 suggest reformulating text classification and other NLP tasks under few-shot restriction into an entailment problem, based on an assumption that it can serve as a unified task closer to the language understanding task a model was pre-trained on.

Manual prompt design is challenging, and poorly constructed prompts can be detrimental to the results. Multiple methods have been proposed for more efficient and/or automated prompt engineering. Schick et al., 2020 maximize the likelihood of training data provided a verbalizer. Gao et al., 2021 generate patterns with a text-to-text language model and use brute-force search to select the best label words for verbalizers. Deng et al., 2022 formulate prompt engineering as a reinforcement learning problem. Hu et al., 2022b utilize knowledge bases to collect synonymous or semantically related words for verbalizers. Logan IV et al., 2021 explore fine-tuning with null prompts, i.e. prompts with just an input text, a mask, and a verbalizer (which can also be picked from random tokens to minimize manual engineering). While performing worse than meticulously crafted and validated manual prompts, they show results competitive to prompt tuning and do not require training or validation with a large dataset unavailable in a true few-shot setting.

Lester et al., 2021 replaced human-language discrete prompts with continuous numeric prompts. In an approach called prompt tuning, embedded text inputs are concatenated with special tokens, the weights of which are updated during training. Various methods have been suggested to improve the performance of continuous prompts, such as pre-training with unsupervised tasks [Gu et al., 2022], encoding discrete prompts with added adversarial perturbations [Hambardzumyan et al., 2021], and using continuous prompts concatenated with discrete tokens [Liu et al., 2023]. Hambardzumyan et al., 2021 and Gu et al., 2022 also note that either initializing a continuous prompt with a manual template or combining continuous prompts with well-crafted discrete ones improves the performance of the classifier in few-shot scenarios. These results suggest that continuous prompts cannot be seen as an outright replacement for discrete ones.

A large point of critique of works on few-shot text classification is that the assumption of scarce data is often relaxed to take advantage of large test sets for hyperparameter tuning and prompt selection. Perez et al., 2021 argue that under *true* few-shot learning, when such data is not available, models often show results only marginally better than random selection. It has since been proven that effective classifiers can be developed with limited test and validation data [Karimi Mahabadi et al., 2022; Schick and Schütze, 2022], but the model performance can significantly benefit from them.

Another issue of prompt-based approaches comes from the large size of PLMs, which makes them impractical or impossible to use without high-end hardware. Existing solutions suggest using smaller or distilled PLMs [Schick and Schütze, 2021b]

or optimizing the fine-tuning process. The latter can be achieved by freezing the model's weights and training embeddings [Lester et al., 2021; Hambardzumyan et al., 2021], or training just a few new parameters introduced into the model's body [Liu et al., 2022]. Recent developments in quantization and task adaptation with injected low-rank matrices [Hu et al., 2022a; Dettmers et al., 2023] have also made fine-tuning of LLMs with billions of parameters more accessible, albeit still demanding powerful GPUs.

2.2.2 Promptless Approaches

Older promptless methods do not rely on PLMs, which makes them language-agnostic and less hardware-demanding. Two primary groups can be distinguished: semi-supervised and meta-learning-based.

Semi-supervised methods leverage unlabeled data along with labeled examples to improve the performance of a model. Linmei et al., 2019 suggest heterogeneous networks that can capture relations between labeled texts and additional information, such as entities and topics, to enrich the semantics of texts for better representation. Meng et al., 2018 generate pseudo-documents from different seed information (label names, keywords, and labeled documents), and use them to pre-train a convolutional or recurrent NN classifier. At the next stage, a model is first trained on the initial small training set, and then on an iteratively expanded set of predictions from unlabeled data. Xie et al., 2020 apply augmentations on unlabeled data, such as back-translation and replacement of words with low TF-IDF scores. A classifier is trained with a combination of two loss functions: supervised cross-entropy loss for labeled examples, and consistency loss to enforce the assignment of the same labels to augmented and non-augmented versions of unlabeled documents. The latter results in a model less sensitive to changes in the input spaces. Gururangan et al., 2019 pretrain variational autoencoders on in-domain unlabeled data to efficiently encode and decode texts. The vectors of labeled documents are concatenated with weighted combinations of internal states of the encoder and used as features to train a supervised classifier.

Meta-learning aims at learning an algorithm, often called a meta-model, on a variety of tasks so that it can be efficiently adapted to new tasks [Lee et al., 2022]. Yu et al., 2018 train clusters of related tasks on various metrics for easier adaptation to diverse new tasks. The resulting meta-model picks suitable linear combinations of these metrics weighted by parameters trained during a few-shot target task. Han et al., 2018 adapt meta networks originally introduced by Munkhdalai and Yu, 2017 for few-shot image classification. Their model is made up of two components, a meta learner and a base learner. The former acquires knowledge across different tasks and provides fast weights, i.e. quickly evolving parameters, which help a base learner generalize on a new task. Prototypical networks [Snell et al., 2017] is another technique originally introduced for image classification. Its core idea is the construction of class prototypes through averaging vectors of the training set and making predictions based on the distance between them and new data. Gao et al., 2019 build robust class prototypes by applying an attention mechanism that assigns larger weights to more important features and instances. Bao et al., 2020 suggest improving the adaptability of a meta-model by learning specific word distributions, namely word frequency across the training data, and inverse entropy of conditional likelihood of a class label given a word. These statistics are translated into attention and used to train a ridge regressor, which makes predictions and provides the attention generator with a loss for optimization.

More recent promptless methods utilize PLMs. Xie et al., 2020 use a model variant with BERT-initialized weights in their semi-supervised approach. Chen et al., 2020 utilize a similar semi-supervised approach and add augmentations from interpolations of documents' hidden states between multiple layers of BERT. Karimi Mahabadi et al., 2022 suggested an MLM-based setup that alleviates the need for prompts. As a replacement for patterns, task-specific adapter layers inform a model of a given task, and instead of verbalizers, they introduced multi-token trainable label embeddings that learn label representations. Tunstall et al., 2022 leverage Sentence Transformers models [Reimers and Gurevych, 2019], which map sentences to continuous vectors. They perform contrastive fine-tuning to pull example pairs of one class closer in the embedded space, and push pairs from different classes further apart. A fine-tuned encoder produces embeddings to train a supervised classifier.

2.3 Few-shot Classification of Ukrainian Texts

There are several works dealing with few-shot classification of texts in Ukrainian that we are aware of. Dementieva et al., 2024b employ ICL with Mistral 7B [Jiang et al., 2023a] and Llama 2 [Touvron et al., 2023] LLMs for toxicity detection and compare them to full-shot benchmarks. ICL with Llama 2 is also utilized by Dementieva et al., 2024a for three classification tasks (toxicity, formality, and natural language inference). Both papers use two to three examples in each prompt. In their experiments, full-shot methods show superior performance, although in Dementieva et al., 2024b ICL rivals XLM-RoBERTa Large [Conneau et al., 2020] in some settings. Isaienkov and Paramonov, 2020 evaluates classifiers typically used in a full-shot setting (support vector machine, long-short term memory NN and BERT) on a small set (60 training and 60 test examples) of short word descriptions. Their best model, support vector machine with Radial Basis Function kernel, reaches an accuracy of 0.72.

2.4 Research Gaps

The review of related work has identified several research gaps that we intend to address in our work. First of all, with the exception of ICL, it is not apparent which PLM-based approaches can be adapted — and to which degree of success — to the Ukrainian language. The choice of PLMs is much more constrained compared to English, the (primarily multilingual) existing options are not guaranteed to perform as well as English-only monolingual models, and the previous guidelines on prompt building may not be applicable. This calls for an exploration of available linguistic resources and extensive experimentation. We are especially interested in comparing the LLMs, which dynamically evolve and regularly establish new SoTA in a wide range of NLP tasks, to smaller PLMs fine-tuned specifically to text classification.

The shortage of prior research on the topic in Ukrainian implies the lack of baselines. This further necessitates adaptation and experiments with existing methods that will serve as a basis for the comparison with our proposed solution.

Many previous publications on the subjects rely on large validation or holdout sets of data. The speed and computational efficiency of few-shot text classification methods are also not reported or are of secondary importance in many of the reviewed works. The first factor limits the usefulness of the approaches in true few-shot setting, while the second impedes applicability in general low-resource scenarios (low data and no high-end hardware available), or when inference must be done

at a large scale. These two factors need to be addressed for the results to be practical in real life.

Chapter 3

Methodology

3.1 Problem Formulation

Given a dataset with K documents x and corresponding labels y belonging to a class c, $\mathcal{D}_L = \{(x, y)\}^K$, our task is to train a supervised model A to accurately map previously unseen documents $\mathcal{D}_U$ to labels. Provided that for a class $c \in \mathcal{C}$ the number of training examples is equal to P, and $|\mathcal{C}| = M$, a problem can be worded as M-way P-shot classification. For the sake of brevity, we will further use $N = P$ to indicate that P training examples are available per class.

$\mathcal{D}_L$ is typically broken into three subsets, $\mathcal{D}_T$ for training, $\mathcal{D}_V$ for validation, and $\mathcal{D}_H$ for testing the performance of a trained model. Under the few-shot setting, we assume that large validation or holdout sets are not available, and the size $\mathcal{D}_V$ is the same as the size of $\mathcal{D}_T$. When selecting prompts or tuning hyperparameters, we cannot base our choices on $\mathcal{D}_H$.

Model A is to be compared to several baselines $\hat{A}_1, ..., \hat{A}_P$. P and the criteria for baseline selection will be presented in chapter 4.

As we are interested in both optimal performance and computational costs, we will use two measures to assess the latter. Firstly, we will evaluate time T it takes for a method to predict labels for 1000 documents. Secondly, we will measure floating point operations (FLOPs) per input token for each model, which is a frequently employed indicator for computational costs and complexity [Liu et al., 2022; Tunstall et al., 2022]. We are only considering inference and not training for two reasons. Firstly, training for different methods is to be done on different hardware due to the varying requirements. Secondly, inference efficiency is more important as typically most of the compute is spent on it [Amodei and Hernandez, 2018].

3.2 Approach to Solution

3.2.1 Base Method

Our proposed solution is based on Sentence Transformer Fine-tuning (SetFit) [Tunstall et al., 2022]. This method leverages Sentence Transformers (STs), a neural network architecture that maps texts to fixed-size continuous vectors [Reimers and Gurevych, 2019]. Original STs were based on BERT and RoBERTa [Liu et al., 2019] and used a pooling layer to derive sentence embedding. Fine-tuning was done with Siamese networks (two subnetworks with tied parameters outputting comparable embeddings) using cosine similarity loss (training a model to keep similar sentences a and b closer in the embedded space, while pushing a dissimilar sentence c further away). The resulting models have demonstrated strong performance in tasks involving sentence comparison, while drastically reducing the computational time compared to BERT and RoBERTa.

SetFit is comprised of two components: an ST model body and a classification head. The training involves two stages. During the first one, the model body is fine-tuned using contrastive learning. Given a set of labeled examples $\mathcal{D}_T = \{(x_i, y_i)\}^K$, where x_i is a text and y_i is its corresponding class label, sets of positive $\mathcal{T}_p$ and negative $\mathcal{T}_n$ triplets are created. Each positive triplet $\mathcal{T}_p = \{(x_i, x_j, 1)\}$ contains two texts belonging to the same class $(y_i = y_j)$, and a negative triplet $\mathcal{T}_n = \{(x_i, x_j, 0)\}$ contains examples from different classes $(y_i \neq y_j)$. A final fine-tuning set $\mathcal{L}$ is compiled from all generated triplets. The maximum potential size of $\mathcal{L}$ is derived from all unique combinations of available examples F, $F(F-1)/2$. By default, $|\mathcal{L}| = 2R|\mathcal{C}|$, where $|\mathcal{C}|$ is the number of classes and R is a hyperparameter usually set to 20. The resulting $\mathcal{L}$ is therefore considerably larger than just F.

A model body is fine-tuned to encode positive pairs closer to each other in the vector space and to maximize the distance between the negative pairs using cosine similarity loss:

$$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - \frac{x_i \cdot x_j}{\|x_i\|\|x_j\|}) \tag{3.1}$$

where x is a vector of embeddings and $y \in \{0, 1\}$ is a label.

When the fine-tuning of the ST is completed, it generates embeddings for the training examples, creating a set $\mathcal{D}_H = \{(Emb(x_i), y_i)\}^N$ used to train a classification head. Any classification algorithm capable of taking text embeddings as an input can be used, although the authors recommend logistic regression. Another option provided in the SetFit Python package is an NN with a single feed-forward layer trained with cross-entropy loss.

The main reason we chose SetFit as a basis for our solution is the strong results reported in the paper combined with relatively low computational costs and fast training and inference times. This, along with a modular architecture (body and head are separate models), allows for relative ease of modification and quick experimentation.

3.2.2 Modifications

We suggest a few modifications meant to improve the classification performance upon the base SetFit.

Augmentations

Data augmentation is a technique that increases the number of examples through the generation of artificial data by application of transformations on the available examples. If applied properly, augmentations can lead to improved model performance and better regularization. We find that relatively few few-shot text classification works experiment with data augmentation, despite its potential to mitigate the lack of training data.

While the small size of data for training sentence transformer's body is compensated by contrastive learning, the classification head is still trained on a very limited number of examples, which we assume is a major bottleneck. We will therefore use augmentations to train a classifier head only.

There is a wide variety of augmentation methods for text classification used either directly on textual data or on embeddings [Bayer et al., 2023]. However, only a handful of them are applicable in our case, largely due to the lack of linguistic resources like a lexical database of semantic relations for synonym replacement, or a

labeled dataset to fine-tune a transformers-based model for word/phrase/sentence replacements (such as Conditional BERT) [Wu et al., 2018]. We picked several methods that could be beneficial to the model performance and are relatively easy to apply:

- Back-translation: textual data is first translated into another language, and then translated back to the original language. This results in paraphrases that largely preserve the semantics, although potentially adding some noise [Bayer et al., 2023].
- Easy Data Augmentation (EDA): suggested by Wei and Zou, 2019, this method utilizes four techniques: synonym replacement, random insertions of synonyms, random word swaps, and random word deletion. Due to the aforementioned lack of linguistic resources, we only apply the latter two.
- Sentence swap: we randomly shuffle sentences in our texts.
- TF-IDF-based replacement: words that have low TF-IDF scores in sentences are replaced with other words with low TF-IDF scores [Xie et al., 2020].
- Mixup: this method comes from the domain of computer vision [Zhang et al., 2018]. Its primary idea is the generation of examples from pairwise weighted combinations of vectors and their labels. For a pair of text embeddings x_i, x_j and corresponding labels y_i, y_j, the following transformation is applied:

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda)x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda)y_j \end{aligned} \tag{3.2}$$

  where $\lambda \in [0,1]$ and is usually sampled from a Beta distribution, $\lambda \sim Beta(\alpha, \alpha)$, and α is a hyperparameter.
- Cutmix: another method from computer vision, it extends Mixup by adding another augmentation type, Cutout, which randomly replaces a portion of a vector or a matrix with zeros [Yun et al., 2019]. Given text embeddings x_i, x_j with length L and their labels y_i, y_j, the algorithm cuts out a random segment of x_i and replaces it with a segment of y_i with the corresponding indices. Labels are mixed proportionally to the size of the substitution:

$$\begin{aligned} \tilde{x} &= \mathbf{M} \odot x_i + (1-\mathbf{M}) \odot x_j \\ \tilde{y} &= \lambda y_i + +(1-\lambda)y_j \end{aligned} \tag{3.3}$$

  where $\mathbf{M} \in \{0,1\}^L$ denotes a binary mask indicating whether a value should be dropped or left in place, and $\odot$ is an element-wise multiplication. As in Mixup, λ is sampled from a Beta distribution. Sentence embeddings, unlike images, are vectors with length L and width 1, therefore we slightly modify the algorithm for sampling M. We pick the first coordinate for a cut c and the size of the cut r the following way:

$$\begin{aligned} c &\sim \text{Unif}(0, L) \\ r &= L\sqrt{1-\lambda} \end{aligned} \tag{3.4}$$

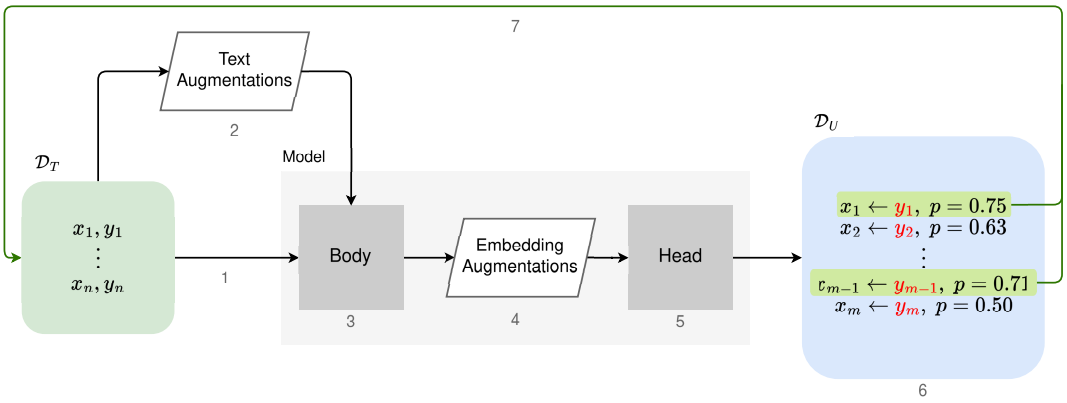

FIGURE 3.1: Schematic representation of SetFit$_M$

Use of synthetic data

Text generation is largely considered an augmentation technique, however, we view it separately due to the intended use. As the previous two modifications expand the size of the training data, synthetic examples are to be used as a test set for evaluating intermediate results of applied modifications under conditions where a large test dataset is considered unavailable. We will further refer to it as an artificial holdout set $\mathcal{D}_H$ to distinguish it from the original test set (the terms holdout and test are often used interchangeably). While many methods and model architectures have been suggested for generating artificial texts [Bayer et al., 2023], a variety of LLMs capable of text completion make the process particularly easy. To soften high computational requirements, one may either use moderately sized models combined with quantization, or inference APIs for access to the models hosted in the cloud.

Self-training

Another way to mitigate the small size of a training set is inspired by iPET (iterative PET) from Schick and Schütze, 2021a. We assume that apart from the initial training set $\mathcal{D}_T^0$, there is a set of unlabeled data $\mathcal{D}_U$ drawn from the same distribution. First, a model is trained on $\mathcal{D}_T^0$ and then assigns pseudo-labels to texts $x \in \mathcal{D}_U$. High-confidence predictions $\mathcal{D}_P^1$ (above a probability threshold θ) are added to $\mathcal{D}_T^0$ and used to train a new model. This process is applied iteratively, with model A^k at step k trained on $\mathcal{D}_T^k = \mathcal{D}_T \cup \mathcal{D}_P^1 \cup \ldots \cup \mathcal{D}_P^k$. We increase the number of pseudo-labeled examples for each class c rather conservatively, with $|\mathcal{D}_P^1(c)| = |\mathcal{D}_T^0(c)|$, and at each consecutive step

$$|\mathcal{D}_P^k(c)| = |\mathcal{D}_P^{k-1}(c)| + \frac{|\mathcal{D}_P^{k-1}(c)|}{2} \tag{3.5}$$

Schick and Schütze, 2021a continue training until a PLM is fine-tuned on 1000 examples. Unlike them, we use performance on an artificial holdout set as a stopping criterion. That is, we stop when the accuracy of two consecutive steps $k+1, k+2$ deteriorates or remains the same as in step k. iPET also leverages ensembling to make use of different prompts. We only train a single model per generation as SetFit does not utilize prompts.

For each N, we train three configurations: a full-fledged variant Modified Setfit (SetFit$_M$) (augmentations + multiple generations) and two ablations: a single generation with augmentations, Augmented SetFit (SetFit$_A$), as well as multiple generations with a logistic regression classifier and no augmentations, Multi-generational SetFit (SetFit$_G$). A schematic depiction of SetFit$_M$ is presented in fig. 3.1:

1. Training set $\mathcal{D}_T$ is used to fine-tune an ST model body.
2. Augmentations are applied to the input data.
3. Augmented texts are fed into the fine-tuned body to extract embeddings.
4. Additional augmentations are applied to the embedded texts.
5. Embeddings are used to train a model head.
6. A trained model predicts texts for unlabeled texts $\mathcal{D}_U$.
7. High-confidence predictions are added to $\mathcal{D}_T$ and a new model is trained with the same procedure.

The above modifications are directed toward the expansion of the available labeled data for different stages of training and evaluation. We acknowledge several limitations of our approach. First, additional in-domain data is not always accessible, and there is no way to guarantee that high-confidence predictions will provide a similar number of examples for all classes, potentially leading to an unbalanced training set. The predicted labels may also be incorrect to a certain extent, and the only way to validate them is through a manual check, which is only viable while the set remains relatively small. The diversity of artificial texts is limited by the seed texts used as examples in prompts, and they require manual verification to ensure correct labeling. Although our modifications will not affect the inference time, the training time will increase (quite significantly in the case of multi-generation training).

Chapter 4

Implementation and Evaluation of Results

4.1 Dataset

We chose to base our experiments on the UA-News dataset [Ivanyuk-Skulskiy et al., 2021]. It is comprised of news articles divided into five rubrics: business, politics, sports, technology, and news (an umbrella category for texts that do not fit into others). It is split into test (30105 records) and training (120417 records) sets. Apart from a text body and a category, the records also contain headlines and tags associated with each document. It is worth noting that the data distribution is imbalanced, as can be seen in fig. 4.1b.

We considered a few different datasets among few options available in Ukrainian but ultimately opted for UA-News. Other choices we considered included two datasets gathered from the Yakaboo online book store[1]. The first consists of reviews with user scores (on a scale from 1 to 5), which essentially presents a five-way sentiment classification. Most few- and even full-shot classifiers struggle with such a task [Gao et al., 2021; Deng et al., 2022; Gu et al., 2022; Karimi Mahabadi et al., 2022], so it might not be the best choice for the first benchmark to compare the methods. The second dataset contains book descriptions with categories that do not fall into a single or even a few taxonomies. Both of these datasets are also only partially in Ukrainian.

Apart from UA-News, we are aware of two other news datasets in Ukrainian. Panchenko et al., 2022 compiled a dataset labeled by source news websites. The second option is a news subcorpus of the UberText 2.0 dataset [Chaplynskyi, 2023]. Although its publicly available version lacks classification labels, the author of the dataset kindly agreed to share a portion of it containing tags, which, similarly to the book description dataset, are very diverse. All of the above options would have been valid for our task, despite complicating it in one way or another. Nevertheless, we ultimately chose UA-News as classification by rubric is quite common and therefore comparable to such popular benchmarks as AG News[2].

4.2 Experimental Setup

4.2.1 Training Data Size and Evaluation

There is no rule as to what number of examples constitutes a problem as a "few-shot". Reviewed papers experiment with varying numbers of examples per class,

[1] https://github.com/osyvokon/awesome-ukrainian-nlp
[2] https://huggingface.co/datasets/ag_news

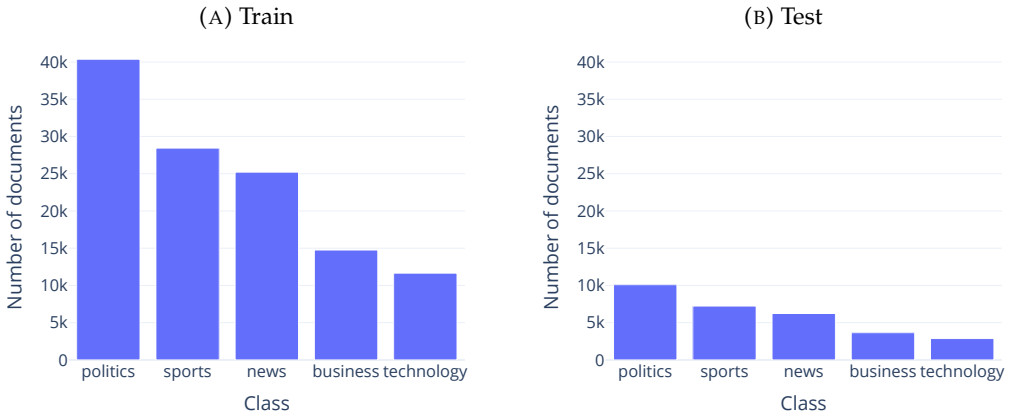

FIGURE 4.1: Class distribution in UA-News dataset

from 1 [Hu et al., 2022b] to 1250 [Chen et al., 2020]. Assuming that annotating more than a few dozen texts per label can become quite tedious, especially when some classes are rare and tasks are numerous, we experiment with $N = 8$ and $N = 16$ for both training and validation sets. They are sampled from the full training set, the remainder of which is treated as unlabeled data for self-training.

Most of the works on few-shot text classification we reviewed report only accuracy as the performance metric. We find it insufficient due to the imbalanced data. Therefore we also report macro F1 and weighted F1 scores. However, we mostly rely on accuracy during hyperparameter tuning and model selection, as keeping track of all metrics and deciding which should be given preference is quite challenging. Inference times will also be compared on the same hardware.

Due to the lack of prior work on text classification utilizing UA-News dataset, there are no external results to which we can compare ours. Although comparisons to somewhat similar experiments on data in English, e.g. AG News, are possible, they would be uninformative given differences in the data and fundamental models.

4.2.2 Baselines

We utilized subsequent criteria to select the baselines for comparing against our solution:

- Availability of linguistic resources. For the selected method to work on a dataset in the Ukrainian language, there should be compatible language models and other resources (e.g. part of speech taggers) if necessary.
- Diversity. We wanted our selection of baselines to cover different approaches and model architectures. We outlined the following groups of approaches, with at least a single method representing each: prompt-based (in-context learning, in-context learning with fine-tuning, MLM-based), and promptless (MLM).
- Performance. The baselines must provide a strong competition to our solution. However, selecting the best-performing options is problematic given the lack of a unified benchmark. To mitigate this issue at least partially, we compiled a table with the reported performance of the methods from the reviewed literature on a variety of datasets. The table A.1 is available in appendix A.

Our choice of baselines landed on the following methods:

- **SetFit** [Tunstall et al., 2022]: we compare our modifications to the basic SetFit without any added modifications.
- **ADAPET** [Tam et al., 2021]: a prompt-based method utilizing an MLM. It is a modification of PET [Schick and Schütze, 2021a] which breaks fine-tuning an MLM into two tasks. Firstly, all words in the MLM's vocabulary are considered as candidates for a label, and a loss function that encourages correct choices and penalizes incorrect ones is employed. Secondly, a model is trained to predict an input provided a label. This combination results in faster training, eliminating the need for ensembling and training multiple model generations, and outperforms PET in multiple natural language inference tasks.
- **Perfect** [Karimi Mahabadi et al., 2022]: a promptless method utilizing an MLM. The authors introduce adapter layers into a model's body while freezing all pretrained weights. Adapters serve as a replacement for prompt patterns, informing a model of a task while reducing the computational costs of training. They also utilize trainable multi-token label embeddings that learn label representations. At inference time, a text is classified based on the distance to a class prototype, similar to Gao et al., 2019.
- **In-context learning** [Brown et al., 2020]: we run vanilla ICL as suggested in the original GPT-3 paper.
- **In-context learning + fine-tuning**: we use Low-Rank Adaptation (LoRA) [Hu et al., 2022a] to fine-tune a model for the task. LoRA introduces low-rank decomposition matrices into the body of a PLM and trains them while keeping the pre-trained weights frozen. This drastically reduces the compute required for fine-tuning.

4.2.3 Model, Hyperparameters, and Prompt Selection

Due to the availability of multiple foundational models for each selected text classification method, we conducted experiments to choose the ones yielding the optimal performance. For SetFit, we chose Multilingual-E5-small [Wang et al., 2022]. Larger E5 models required the reduction of training batch size below 8 and slowed down both training and inference, while two other multilingual options, paraphrase-multilingual-mpnet-base-v2[3] and paraphrase-multilingual-MiniLM-L12-v2[4], demonstrated inferior results. Three MLMs were tested as base models for ADAPET, one unilingual (ukr-RoBERTa-Base[5]), and two multilingual (XLM-RoBERTa-Base and XLM-RoBERTa-Large [Conneau et al., 2020]). We opted for the last option based on the performance. We reused XLM-RoBERTa-Large to train Perfect. For ICL, our choice landed on Mistral 7B Instruct 0.2 [Jiang et al., 2023a], as it can run on affordable hardware and has a decent command of Ukrainian. Other models with comparable hardware requirements, namely Llama2 7B [Touvron et al., 2023] and Gemma 2B [Team et al., 2024], returned considerably worse (often unintelligible) outputs when provided with the same prompts. We quantized Mistral 7B to 4 bit for it to fit into VRAM.

Quick training time with SetFit allowed us to do an extensive hyperparameter search on the validation set with Optuna Python library [Akiba et al., 2019] using

[3]https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2
[4]https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
[5]https://huggingface.co/youscan/ukr-roberta-base

Tree-structured Parzen Estimator. We set the body learning rate to 1.93×10^{-5} and the number of epochs to 3. During training, we compute evaluation loss and save model checkpoints at the end of each epoch to load the best model at the end. We chose logistic regression as a classifier head for SetFit$_G$ and an NN classifier (trained for 24 epochs) for other two variants, as the most efficient combination of augmentations required a loss function that could accept continuous values for class probabilities. For ADAPET and Perfect, generic hyperparameters provided in the source code were used due to fine-tuning requiring a very long time. Several temperature values in the range from 0.1 to 2.0 were tried for ICL but did not affect the performance in any way, so we ended up using a "standard" temperature of 1.0. Various training data configurations were explored for fine-tuning of Mistral, including $N = 8$, $N = 16$, and a combination of larger training and validation sets ($N = 32$). Due to insufficient time and computational resources for running inference on the full test set for all variants, we tested each one (with and without demonstrations) on a sample of 3000 examples. Adapters with the best performance were then chosen (with $N = 32 + 3$ demonstrations per class taken from the training set) for inference on all test examples.

We tested three prompt pattern variants (“Category:”, “Text rubric:”, and a null prompt) with ADAPET. Results for the best one ("Category:") are reported in table 4.1. Class labels were used as verbalizers. For ICL, we experimented with various prompts on a unified set of training and validation data and ended up using three randomly selected examples from each class per prompt for demonstration, each followed by a question asking to identify a category of the text, followed by a label. The prompt template for ICL and its variations can be found in the appendix B.

4.2.4 SetFit Modifications

We started the implementation of our primary solution with the generation of synthetic data. Our prompt consisted of three example texts (randomly sampled from $\mathcal{D}_T$ and $\mathcal{D}_V$) belonging to the same class prepended with a pattern "News text on *topic*:" with class name in place of *topic* and finished with the pattern to force the model to complete it. We first generated 500 texts (100 per each class). After deduplication and manual filtering of unintelligible outputs and results that could not be confidently attributed to a particular class, their number was reduced to 248. We found this amount insufficient and therefore decided to generate more data with a more powerful cloud-hosted model. We opted for GPT 3.5 Turbo[6] due to relatively cheap inference compared to other similar services. The same sampling strategy, prompt template, and filtering procedures were applied. We then combined artificial data generated by two models and, based on the cosine similarity of ST embeddings, dropped records that were complete or near duplicates of the ones from $\mathcal{D}_T$, $\mathcal{D}_V$ and $\mathcal{D}_U$. A cosine similarity threshold of 0.92 was chosen empirically to cut off texts with high resemblance while keeping the ones on close topics. The resulting artificial holdout set $\mathcal{D}_H$ contained 698 texts. Class distribution can be seen in fig. 4.2. Despite it being non-uniform and different from the distribution of the real data, it was decided to proceed with it, as adding more examples incurred additional costs (which is undesirable under a low-resource setting we were aiming for) and removing meant decreasing an already small number of examples.

[6] https://platform.openai.com/docs/models/gpt-3-5-turbo

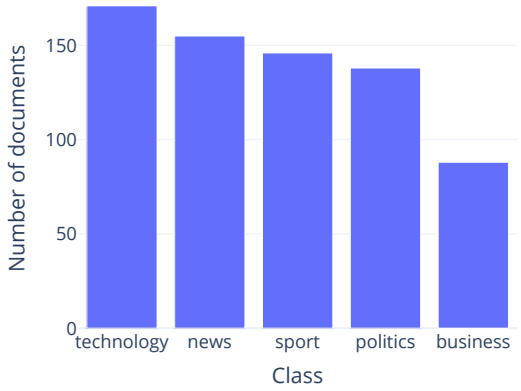

FIGURE 4.2: Distribution of artificial holdout dataset

New $\mathcal{D}_H$ allowed us to conduct a hyperparameter search to find the most efficient composition of augmentations. To this end, we used vanilla SetFit body fine-tuned on $N = 8$ as text encoder and only re-trained classifiers (either a logistic regression or an NN) for each trial. Back-translation, being the most resource- and time-demanding augmentation, was done beforehand with nllb-200-distilled-1.3B model [Team et al., 2022] and three intermediate languages (English, French, and German). The best accuracy was yielded by an NN classifier and a combination of back-translation (a single set of examples translated from Ukrainian to English and back), EDA (with 5 maximum word swaps and a maximum chance of word deletion equal to 0.6), TF-IDF replacement, and Mixup (with $\alpha = 0.141$). Optimization also suggested a text augmentation multiplicity of 2 (i.e. two augmented texts per each original or back-translated), and an embedding augmentation multiplicity of 3. This set of parameters was used in all further experiments with augmentations.

We then proceeded to train SetFit with modifications. We trained three variants for each N. The same hyperparameters were reused for all generations, except the number of pseudo-labeled examples added to $\mathcal{D}_T$ according to the formula from chapter 3 and the probability threshold θ incremented by 0.05 for each next generation until reaching 0.85. We set the initial θ to 0.70, however, the first generation of SetFit$_M$ trained on $N = 8$ only predicted a single class (sports) with sufficient confidence, therefore we had to make an exception in this particular case.

4.3 Results

4.3.1 Performance

Performance metrics for the principal approach, ablations, and baseline models are available in table 4.1. For $N = 8$, the best results across all metrics are achieved by SetFit$_G$ trained for two generations (+0.03 accuracy and weighted F1 and +0.2 macro F1 compared to the vanilla SetFit), followed by SetFit$_M$ (+0.02 across all scores). Conversely, for $N = 16$, the vanilla SetFit surpasses all other approaches. Evaluation of SetFit$_G$ with $N = 16$ on $\mathcal{D}_H$ suggested picking the first model generation, hence *the scores are the same as for SetFit with no modifications*. SetFit$_M$ improves upon the vanilla SetFit for $N = 8$, but falls short of it for $N = 16$. Overall, the use of the NN classifier head with augmentations had a detrimental effect on the model performance, and the logistic regression head consistently provided better results.

	N = 8			**N = 16**		
method	accuracy	macro F1	weighted F1	accuracy	macro F1	weighted F1
SetFit$_M$	0.83	0.81	0.83	0.83	0.84	0.84
SetFit$_G$	**0.84**	**0.82**	**0.84**	0.85*	0.84*	0.85*
SetFit$_A$	0.81	0.79	0.81	0.80	0.81	0.81
SetFit	0.81	0.79	0.81	**0.85**	**0.84**	**0.85**
ADAPET	0.81	0.78	0.81	0.81	0.79	0.81
Perfect	0.75	0.73	0.76	0.69	0.67	0.70
	N = 3 (demonstrations)					
ICL	0.77	0.71	0.76			
ICL + fine-tuning	0.73	0.71	0.74			

TABLE 4.1: *the first model generation was selected based on holdout dataset accuracy, making it the same as SetFit

Fig. 4.3 demonstrates that in three out of four cases, the second generation improved the accuracy upon the first, but further training led to degradation of the performance. In the case of SetFit$_G$ with $N = 16$, the decline starts immediately with the second generation. The primary reason for this is that the addition of more data leads to overfitting, as evidenced by fig. 4.4. We attempted to compensate for this by strengthening the regularization, namely increasing the weight decay parameter of the Adam optimizer (controlling L2 regularization) and adding L1 regularization, but did not reach significant improvements. Another factor that could affect the outcomes is inaccurate pseudo-labels. As can be seen from fig. C.2, a certain proportion of pseudo-labels predicted by each generation is incorrect. With the exception of SetFit$_M$ with $N = 16$, the number of mislabeled examples is low, however, it is still likely to have an impact given the small training size. It should also be noted that many mislabeled examples are somewhat ambiguous, e.g. a text on investments into cryptocurrency with the true label "technology" was classified as "business", which is also a plausible option considering the lexicon. At the same time, the high confidence of this prediction can be seen as erroneous from a human perspective.

The decline of performance with the addition of more training goes against the results from Tunstall et al., 2022, where the accuracy increases with a transition from $N = 8$ to $N = 64$ across all benchmarks. We hypothesized that this could be caused by the diminishing value of new pseudo-labeled data, i.e. the model does not improve significantly by learning from the examples it was confident about in the previous iteration. A potential indicator could be the lack of intra-class diversity between the preceding and the next training sets. To test this assumption, we used a method for comparing corpora similarity suggested by Kilgarriff, 2001. He ranks words in two corpora based on frequency, takes the difference d in rank orders for each of n most common words between the two corpora, and calculates the Spearman's rank-order correlation:

$$\text{Sim} = 1 - \frac{6 \sum d^2}{n(n^2 - 1)} \tag{4.1}$$

which provides an easily interpretable score on a scale from 0 to 1. Our idea was that the similarity grew more rapidly when a preceding set of examples for a class was

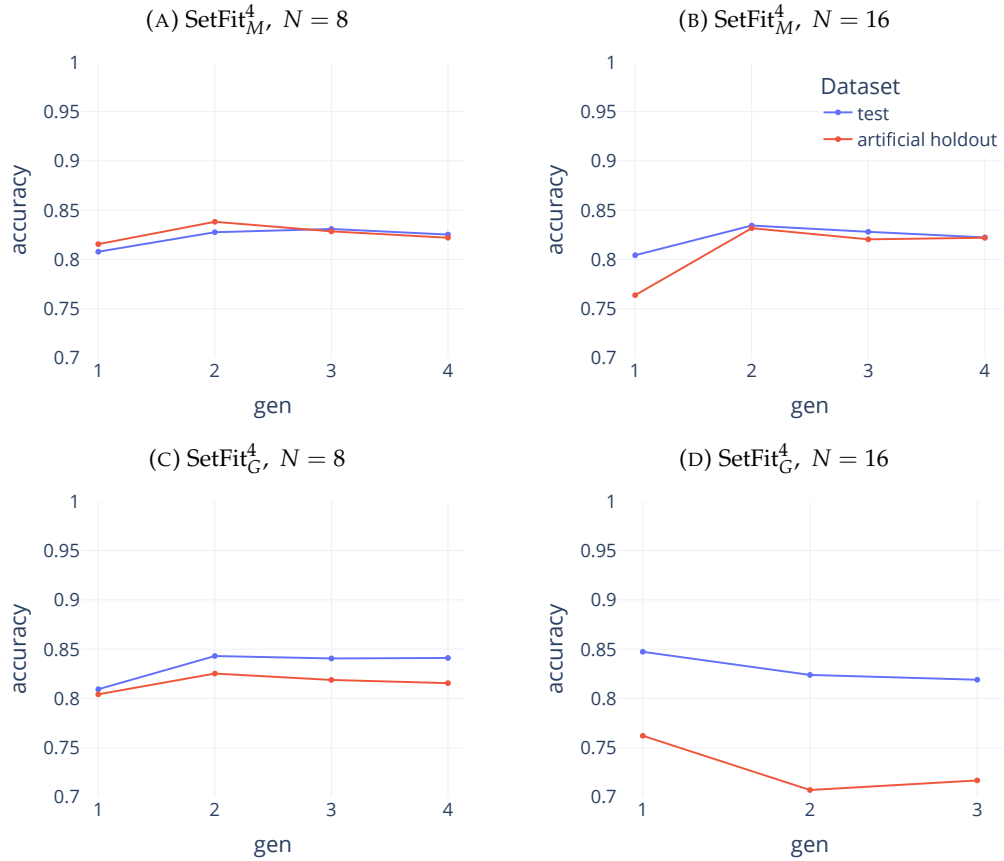

FIGURE 4.3: Accuracy of SetFit$_M$ and SetFit$_G$ generations

supplemented by high-confidence predictions compared to when randomly sampled documents from the full-size $\mathcal{D}_T$ were added. However, this was disproved in an experiment, as in the latter case the similarity grew at approximately the same or a higher rate (0.08 vs 0.16 on average in 10 trials). For an alternative check, we embedded 4-generational (the original data is counted as generation 0) $N = 8$ train set for SetFit$_M$ with Multilingual-E5-small. We plotted a heatmap of the cosine similarity matrix for each class (fig. C.3), hoping that if semantic diversity declined over generations, similarity values should visibly increase from left to right (at least compared to the first generation). However, a visual investigation of the plots revealed no such pattern. Although our hypothesis found no confirmation, we assume the outcomes of self-training could still be improved through advanced techniques and heuristics, like applying curriculum learning to shift the value of θ or using multiple classifiers [Amini et al., 2023], which might be a promising line of future work.

As for the baselines other than base SetFit, the best results in both settings are demonstrated by ADAPET. Doubling the size of the dataset to 16 examples per label only slightly improves macro F1 by +0.01. The worst overall performance is shown by Perfect with $N = 16$, which could be caused by both overfitting and insufficient size of validation data. Mistral 7B, being by far the largest model, scores significantly lower than both ADAPET and all SetFit variants. It is worth noting that despite multiple demonstrations and a specific instruction requesting to provide the most suitable option among the ones listed, it often returned a completely different label. Fine-tuning Mistral leads to a decrease in performance (-0.04 accuracy and -0.02 weighted F1). From this, we may conclude that moderately sized LLMs, while

being very versatile and easy to use, struggle to keep up with smaller fine-tuned MLMs and STs in the task of text classification.

For each model, the macro F1 score is lower than the accuracy and weighted F1 by about 0.01-0.02, which means that some classes are harder to predict. As can be seen from fig. C.1 in appendix C, all models predict the sports category consistently well. While other classes are mislabeled more often, there is no universal pattern among the models, e.g. ICL struggles more with the business category and Perfect has difficulty with the news class. A more detailed analysis of classification errors is provided in appendix C.

4.3.2 Effect of the Artificial Holdout

Model evaluation on $\mathcal{D}_H$ served as quite an effective criterion for stopping the training and selecting a model generation. As evidenced by fig. 4.3, classification accuracy on it generally (although not perfectly) aligns with the accuracy on the full test set. However, to confidently say that artificial texts can be a reliable substitute for a real test set would require a lot more experimentation with diverse data. The lacking test results of SetFit$_A$ suggest the artificial set was not adequate to find the most efficient augmentations or their combination.

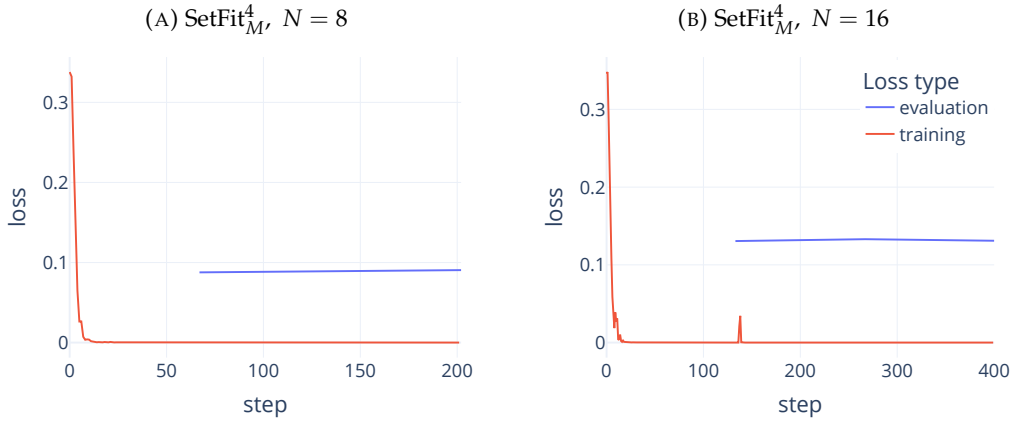

FIGURE 4.4: Training and validation losses of SetFit$_M$ generation 4. Training loss recorded for every 100 steps, validation loss for each epoch

4.3.3 Inference Time and Computational Costs

To compare the inference times of the employed methods, we run experiments on two sets of hardware: first, on a virtual machine with 2 vCPU, 16 GB RAM, and 20GB VRAM (1/4 of H100 graphic processing unit); second, to test potential usability without GPU acceleration, on a laptop with AMD Ryzen 5 5500U CPU (6 cores and base clock speed 2.1 GHz), 16 GB RAM and no graphical processor. For each method, inference times were measured on three random samples of size 1000 from the test dataset, except for ICL without GPU acceleration, running which took extremely long (therefore we offer time extrapolated from 100 examples). Moreover, quantization in our experiments relies on bitsandbytes Python library[7] which requires GPU. Without it we resorted to using a pre-quantized version of Mistral 7B

[7] https://github.com/TimDettmers/bitsandbytes

method	model	T (GPU)	T (no GPU)	FLOPs/token
SetFit	Multilingual-E5-small	5.03	171.60	5.18E+07
ADAPET	XLM-RoBERTa-Large	376.35	8846.26	6.29E+08
Perfect	XLM-RoBERTa-Large	117.02	4545.95	6.29E+08
ICL	Mistral 7B	1883.82	451759.31*	1.42E+10
ICL + fine-tuning	Mistral 7B + LoRA	2418.12	-	1.49E+10

TABLE 4.2: Inference times (in seconds) and FLOPs of the tested classification methods. * - time is extrapolated from 100 examples

Instruct 0.2[8]. For this reason, we were unable to run a fine-tuned model on the local machine.

Calflops Python library [Xiaoju, 2023] was used to measure FLOPs. The measure of FLOPs only accounts for the PLM model body, leaving out SetFit's logistic regression/NN head and Perfect's prototypical network classifier.

The results are recorded in table 4.2. SetFit is by far the fastest and most cost-efficient classification method among the ones tested. Perfect comes second, followed by ADAPET. Although Perfect and ADAPET both utilize the same MLM, the reliance of the former on the prototypical evaluation, according to its authors, gives it a speed advantage over PET-like models which predict verbalizers in an autoregressive fashion [Karimi Mahabadi et al., 2022]. ICL lags far behind the other methods, and the introduction of adapters into Mistral 7B led to about 28% additional slowdown. Larger prompts for PLMs also result in much longer inference times. In our case, the average input length for Mistral 7B was around 3440 tokens, while the maximum sequence length for XLM-RoBERTa-Large and Multilingual-E5-small was limited to 512 tokens. Without GPU acceleration, only SetFit demonstrates inference time that could still be tolerable in a production environment.

It is worth noting that the gap could have been reduced somewhat if we had chosen a larger ST model and a smaller MLM and LLM. However, smaller model sizes would most likely result in worse performances. Mistral 7B is considered rather a compact LLM, as more capable decoder-only models tend to have dozens or hundreds of billions of parameters.

[8] https://huggingface.co/TheBloke/Mistral-7B-Instruct-v0.2-GGUF

Chapter 5

Conclusions

5.1 Discussion

In this work, we considered the problem of the few-shot classification of texts in Ukrainian language. Our contributions are the following:

1. To the best of our knowledge, this is the first published work on few-shot classification of Ukrainian texts that utilizes methods based on different PLM architectures (MLM, LLM and ST). We have provided an extensive review of the existing approaches, selected the baselines based on the availability of linguistic resources and the reported metrics, and tested their performance and computational efficiency, thus laying the groundwork for future inquiries into the subject. The results of our experiments can serve as a reference point for the comparison with new approaches and different language models.

2. We have established the first SoTA for few-shot classification on UA-News dataset, with our proposed solution providing the best results in 8-shot setting. Of two modifications to the training procedure of SetFit proposed by us, namely self-training with multi-generation models and raw text and embedding augmentations, the former and the combination of the two were beneficial to the performance in 8-shot setting, with Modified, Augmented, and Multi-generational SetFit improving upon the base SetFit. However, potential model overfitting should be kept in mind.

3. Our experimental results speak favorably of using (albeit with caution) synthetic data generated by LLMs for model evaluation when a full-fledged test dataset is not available, although, for a conclusive statement, this must be verified on more data.

It should be noted that our method is not limited to either news texts or the Ukrainian language. It can easily be adapted to other corpora and languages as long as the latter are supported by Sentence Transformers and large language models that can generate texts of sufficient quality.

5.2 Future Work

We suggest several directions that future research might take to expand upon our work.

1. The selection of approaches to few-shot text classification and PLMs tested in this paper is not exhaustive and was largely limited by the constraints of time and computational resources, so further research should explore more

available options and suggest novel techniques. Possible lines of work could be automated prompt generation and tuning, experiments with more powerful LLMs, search for more robust self-training algorithms, and exploration of other data augmentation methods.

2. The methods we tested in this paper should be run on more datasets, preferably from different domains and representing other types of text classification tasks, such as natural language inference, sentiment analysis and question answering. An obstacle to this is a general lack of publicly available text datasets in the Ukrainian language. We, however, expect to see more made available in the near future, which will present a great opportunity for new studies.

3. There is a need for a reliable assessment of synthetic text quality for classification, as the potential utility of generated data is vast, but it also presents many pitfalls when used either for training or evaluation.

Appendix A

Reported Accuracy of the Reviewed Methods

			Dataset					
Method	PLM used	*N*	Yelp5	AGNews	Yahoo	MNLI	BoolQ	CB
ADAPET [Tam et al., 2021]	ALBERT-xxlarge	32					80	92
Adaptive ROBUST TC-FSL [Yu et al., 2018]	-	5						
Distributional Signatures [Bao et al., 2020]	-	5						
EFL [Wang et al., 2021]	EFL	8						
HGAT [Linmei et al., 2019]	-	20		72.1				
ICL + contextual calibration [Zhao et al., 2021]	GPT-3	8		84.3				65
iPET [Schick and Schütze, 2021a]	RoBERTa large	50	60.7	88.4	69.7	67.4		
Knowledgeable Prompt-tuning [Hu et al., 2022b]	RoBERTa large	10		86.3	68			
LM-BFF [Gao et al., 2021]	RoBERTa large	16				70.7		
MixText [Chen et al., 2020]	BERT	10		88.4	67.6			
P-tuning [Liu et al., 2023]	ALBERT	?					76.55	88.39
Perfect-rand [Karimi Mahabadi et al., 2022]	RoBERTa large							90.3
PET [Schick and Schütze, 2021a]	RoBERTa large	50	60	86.3	66.2	63.9		
PETAL [Schick et al., 2020]	RoBERTa large	10+	56.5	84.9	62.9	62.4		
PPT [Gu et al., 2022]	T5-XXL	32					76	82.2
RLPROMPT [Deng et al., 2022]	RoBERTa large	16		80.2				
SetFit [Tunstall et al., 2022]	MPNet	8		82.9				
UDA [Xie et al., 2020]	BERT large	20+						
VAMPIRE [Gururangan et al., 2019]	-	200		83.9	59.9			
WARP [Hambardzumyan et al., 2021]	RoBERTa large	32				88		
WeSTClass-CNN [Meng et al., 2018]	-	10+	77.6	84.1				

	Dataset									
Method	RTE	WiC	DBpedia	IMDB	Amazon (sentiment)	TREC	SST-2	SST-5	MR	CR
ADAPET	75	53.5								
Adaptive ROBUST TC-FSL					83.12					
Distributional Signatures										
EFL				87.1			90.8		86.2	92.3
HGAT										
ICL + contextual calibration			86.9			66.9	95.3			
iPET										
Knowledgeable Prompt-tuning			98.0	92.9	93.8					
LM-BFF						89.4	93	50.6	87.7	91
MixText			98.5	78.7						
P-tuning	63.27	55.49								
Perfect-rand	60.4	53.8				90.6	90.7	42.8	86.3	90
PET										
PETAL										
PPT	65.8						94.4	46		
RLPROMPT							92.5	41.4	87.1	89.5
SetFit					40.3					88.5
UDA										
VAMPIRE				82.2						
WARP	84.3						96.3			
WeSTClass-CNN										

TABLE A.1: Reported Accuracy of the reviewed methods. Only benchmarks used 3+ times are shown

Appendix B

Prompts for In-context Learning

Текст: 29-річний український півзахисник «Гента» Роман Безус поділився думкою про те, що головний тренер збірної України Андрій Шевченко може очолити італійський «Мілан». «Звичайно, Шевченко зможе тренувати «Мілан». Зараз він створив топ-атмосферу в збірній України. Все на топ-рівні: тактика, аналіз, весь підхід повністю. Від «Гента» відрізняється набагато. У збірної України на голову сильніше. У збірної України, думаю, найвищий рівень, який може бути. Тому, я думаю, якщо Шевченко запросять до «Мілана» і керівники підуть на його умови, то він зможе там легко побудувати команду високого рівня», — зазначив Безус.
Катеогорія тексту: політика, технології, бізнес, спорт, чи новини?
Категорія: спорт
...
Текст: Згідно з новим витоком, наступне покоління iPhone має вміщати бездротову зарядку, яка дозволить смартфону заряджати бездротовий годинник та навушники Apple. За даними японського новинного блога про Apple Macotakara, наступне покоління iPhone, яке представлять восени 2019-го, буде включати бездротову зарядку інших пристроїв, повідомляє 9to5Mac. Раніше про це також заявляв наближений до компанії Apple та її інвесторів аналітик Мінь-Чі Куо, що підтверджує чутки про майбутню двосторонню бездротову зарядки iPhone.
Катеогорія тексту: політика, технології, бізнес, спорт, чи новини?
Категорія:

FIGURE B.1: A shortened example of a prompt used for ICL

Fig. B.1 shows an example of a prompt we used for ICL. A light blue backgrounds highlights a demonstration, an example of a task that helps guide a model's predictions. It consists of a text, a question that suggests available class labels, a pattern that forces a model to output a label, and a label itself. In each prompt we used three demonstrations per class (15 in total). An actual task is put at the end of the prompt, its structure is identical to a demonstration, except that it does not contain a label. We used this template for inference on the test data, as it demonstrated the best accuracy on the validation set. Other prompt variants included different structures (question, text, pattern, label; text, pattern, label [no question]; question, text, label [no pattern] omitting the "Текст: " prefix; no demonstrations on a fine-tuned model) as well as different wordings for questions ("Питання: До якої категорії належить даний текст: політика, технології, бізнес, спорт, чи новини?", "Текст належить до

однієї з категорій: політика, технології, бізнес, спорт, новини."), fewer (1, 2) and more (4, 5) demonstrations per class.

Appendix C

Additional Result Statistics

C.1 Analysis of Classification Errors

Fig. C.1 provides confusion matrices of predictions made by different methods. Although errors vary from one method to another, some common patterns can be discerned. A common mistake by SetFit (both vanilla and with modifications) is labeling politics as news and vice versa. A manual inspection suggests a partial explanation. Texts tagged with politics and predicted as news are often dedicated to topics such as crime, incidents, warfare, and terrorism. In the opposite case, many texts are related to combat clashes, exchange rates, taxes, trade, catastrophes, and police reports. The topics listed above are often quite similar and, from a human standpoint, classifying them as either of two rubrics would not be incorrect. Without the guidelines, we can only presume that an original choice of a label was largely dictated by a subjective preference of an annotator. The same is true for quite a few other cases, e.g. documents labeled as technology and predicted as business frequently contain messages about tech companies and startups, technological innovations for business, and cryptocurrency. At the same time, the sports category, a lot less ambiguous by its nature, was predicted consistently well by all classifiers. Therefore many (though by no means all) errors can be attributed to the dataset design, which largely capped the performance of SetFit and its modifications. Similar conclusions can be drawn about some common mistakes made by other methods. For instance, in cases where news was misclassified as technology and vice versa, the documents were often dedicated to scientific discoveries, tech companies, and innovation. While being a considerable drawback of the UA-News dataset, we acknowledge that such overlaps are hard to avoid while keeping the number of categories limited to only five. This calls for further evaluation of the methods explored in this paper on other datasets with less room for ambiguity and subjectivity.

C.2 Other visualizations

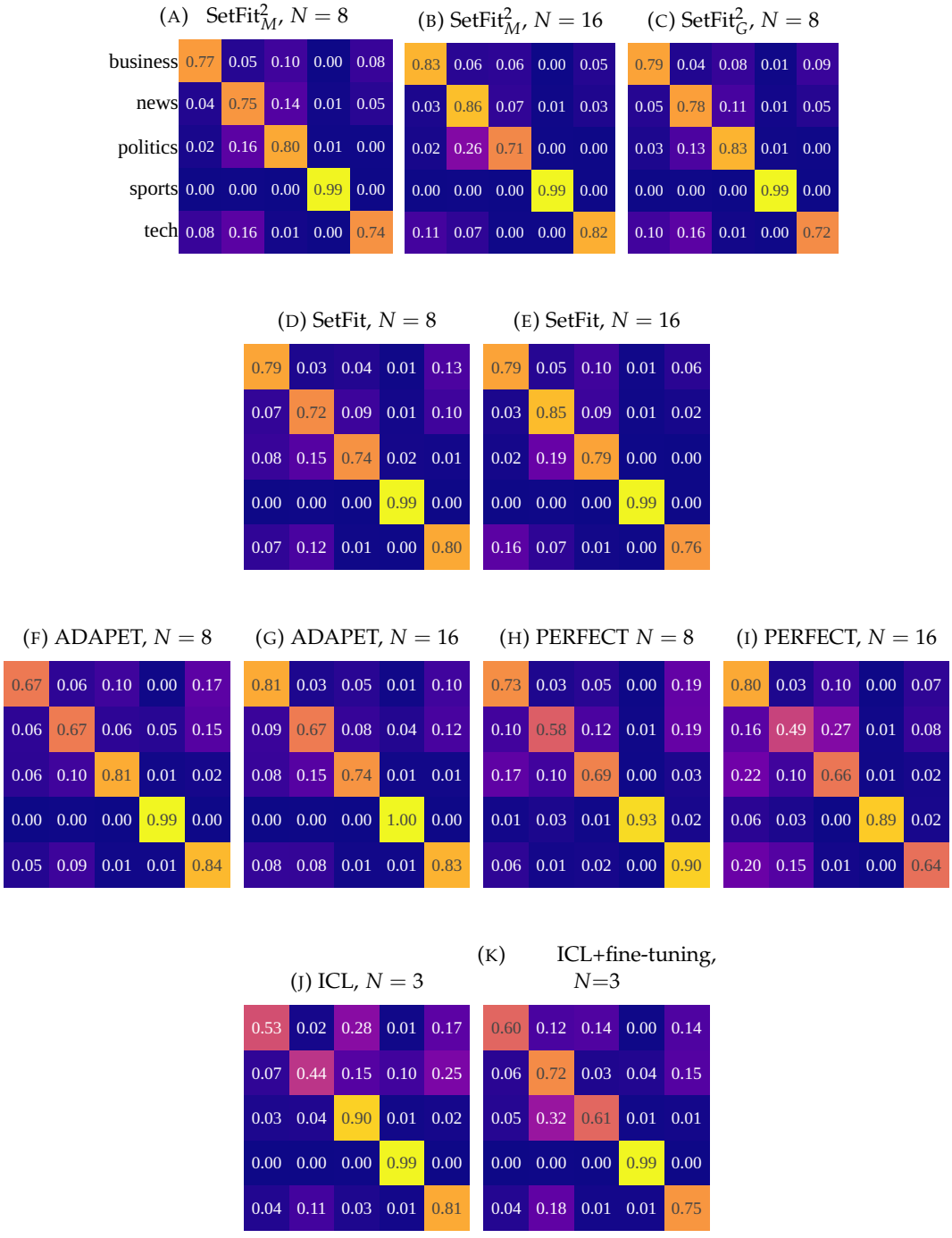

FIGURE C.1: Confusion matrices of the models' predictions (normalized over true labels)

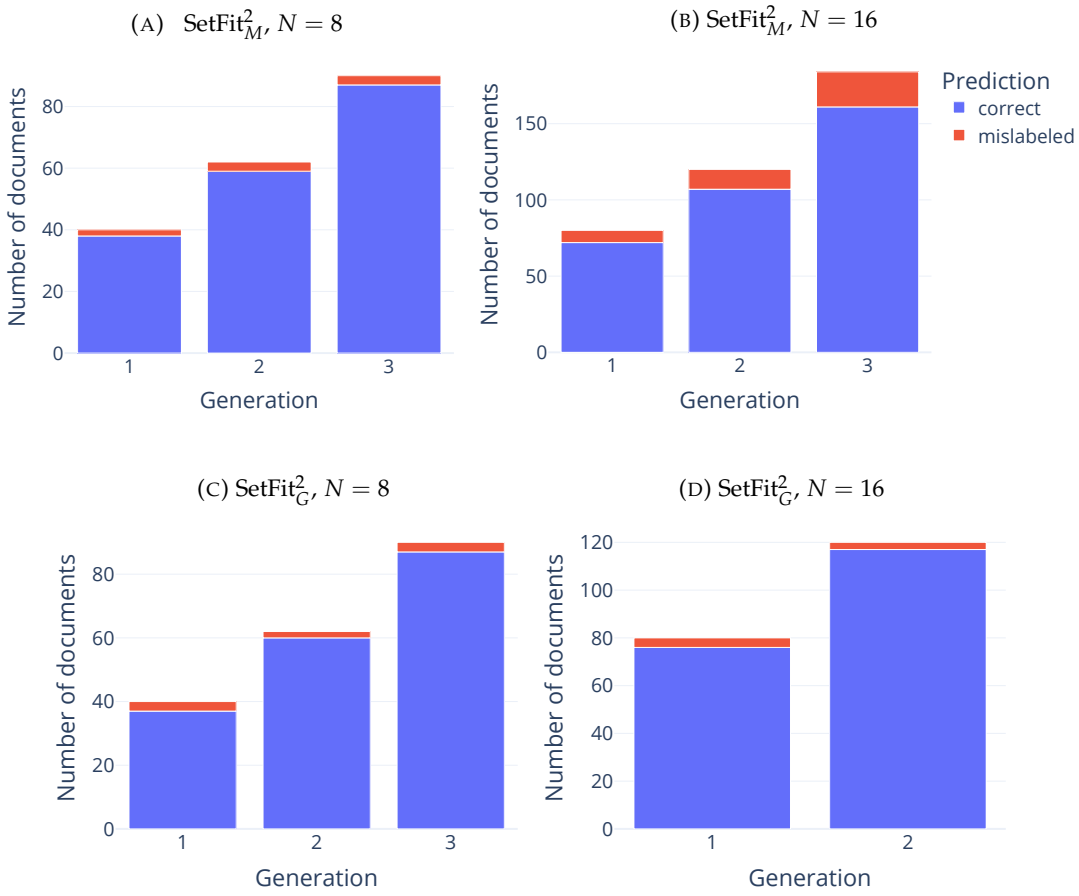

FIGURE C.2: Number and proportion of correct and incorrect predictions made by each model generation

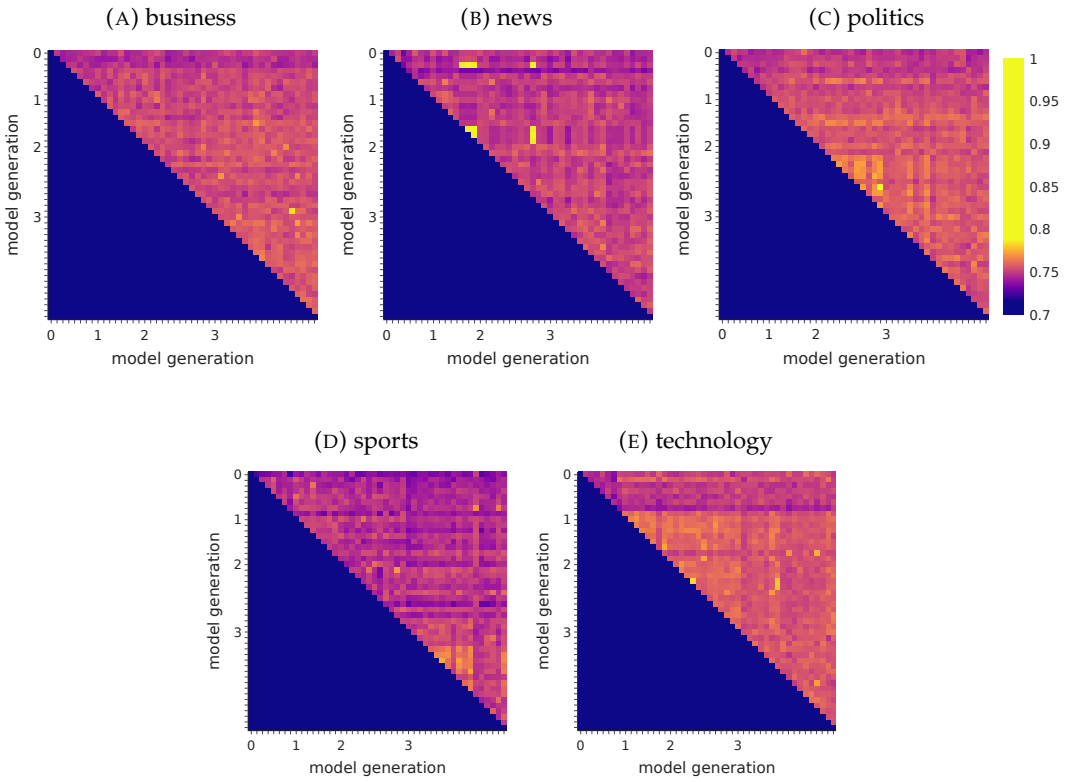

FIGURE C.3: Cosine similarity of 4 generations training examples, with generation 1 onward pseudo-labeled by SetFit$_M$ ($N = 8$). The main diagonal is removed not to show the similarity of texts with themselves; the lower triangle is removed as the matrices are symmetric.

Bibliography

Akiba, Takuya et al. (2019). "Optuna: A Next-Generation Hyperparameter Optimization Framework". In: *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631.

Amini, Massih-Reza et al. (Sept. 18, 2023). *Self-Training: A Survey*. DOI: 10.48550/arXiv.2202.12040. arXiv: 2202.12040[cs]. URL: http://arxiv.org/abs/2202.12040.

Amodei, Dario and Danny Hernandez (May 16, 2018). *AI and compute*. OpenAI. URL: https://openai.com/index/ai-and-compute.

Bao, Yujia et al. (Feb. 18, 2020). *Few-shot Text Classification with Distributional Signatures*. DOI: 10.48550/arXiv.1908.06039. arXiv: 1908.06039[cs]. URL: http://arxiv.org/abs/1908.06039.

Bayer, Markus, Marc-André Kaufhold, and Christian Reuter (July 31, 2023). "A Survey on Data Augmentation for Text Classification". In: *ACM Computing Surveys* 55.7, pp. 1–39. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3544558. URL: https://dl.acm.org/doi/10.1145/3544558.

Brown, Tom et al. (2020). "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 1877–1901. URL: https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

Chaplynskyi, Dmytro (May 2023). "Introducing UberText 2.0: A Corpus of Modern Ukrainian at Scale". In: *Proceedings of the Second Ukrainian Natural Language Processing Workshop*. Dubrovnik, Croatia: Association for Computational Linguistics, pp. 1–10. URL: https://aclanthology.org/2023.unlp-1.1.

Chen, Jiaao, Zichao Yang, and Diyi Yang (July 2020). "MixText: Linguistically-Informed Interpolation of Hidden Space for Semi-Supervised Text Classification". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, pp. 2147–2157. DOI: 10.18653/v1/2020.acl-main.194. URL: https://aclanthology.org/2020.acl-main.194.

Conneau, Alexis et al. (Apr. 7, 2020). *Unsupervised Cross-lingual Representation Learning at Scale*. DOI: 10.48550/arXiv.1911.02116. arXiv: 1911.02116[cs]. URL: http://arxiv.org/abs/1911.02116.

Dementieva, Daryna, Valeriia Khylenko, and Georg Groh (Apr. 2, 2024a). *Ukrainian Texts Classification: Exploration of Cross-lingual Knowledge Transfer Approaches*. DOI: 10.48550/arXiv.2404.02043. arXiv: 2404.02043[cs]. URL: http://arxiv.org/abs/2404.02043.

Dementieva, Daryna et al. (Apr. 27, 2024b). *Toxicity Classification in Ukrainian*. DOI: 10.48550/arXiv.2404.17841. arXiv: 2404.17841[cs]. URL: http://arxiv.org/abs/2404.17841.

Deng, Mingkai et al. (Dec. 2022). "RLPrompt: Optimizing Discrete Text Prompts with Reinforcement Learning". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2022. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for

Computational Linguistics, pp. 3369–3391. DOI: 10.18653/v1/2022.emnlp-main.222. URL: https://aclanthology.org/2022.emnlp-main.222.

Dettmers, Tim et al. (May 23, 2023). *QLoRA: Efficient Finetuning of Quantized LLMs*. DOI: 10.48550/arXiv.2305.14314. arXiv: 2305.14314[cs]. URL: http://arxiv.org/abs/2305.14314.

Devlin, Jacob et al. (June 2019). "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)*. NAACL-HLT 2019. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

Dogra, Varun et al. (June 9, 2022). "A Complete Process of Text Classification System Using State-of-the-Art NLP Models". In: *Computational Intelligence and Neuroscience* 2022. Publisher: Hindawi, e1883698. ISSN: 1687-5265. DOI: 10.1155/2022/1883698. URL: https://www.hindawi.com/journals/cin/2022/1883698/.

Galke, Lukas and Ansgar Scherp (Apr. 12, 2022). *Bag-of-Words vs. Graph vs. Sequence in Text Classification: Questioning the Necessity of Text-Graphs and the Surprising Strength of a Wide MLP*. DOI: 10.48550/arXiv.2109.03777. arXiv: 2109.03777[cs]. URL: http://arxiv.org/abs/2109.03777.

Galke, Lukas et al. (June 4, 2023). *Are We Really Making Much Progress in Text Classification? A Comparative Review*. DOI: 10.48550/arXiv.2204.03954. arXiv: 2204.03954[cs]. URL: http://arxiv.org/abs/2204.03954.

Gao, Tianyu, Adam Fisch, and Danqi Chen (2021). "Making Pre-trained Language Models Better Few-shot Learners". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Online: Association for Computational Linguistics, pp. 3816–3830. DOI: 10.18653/v1/2021.acl-long.295. URL: https://aclanthology.org/2021.acl-long.295.

Gao, Tianyu et al. (2019). "Hybrid attention-based prototypical networks for noisy few-shot relation classification". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. Issue: 01, pp. 6407–6414. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4604.

Gu, Yuxian et al. (2022). "PPT: Pre-trained Prompt Tuning for Few-shot Learning". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin, Ireland: Association for Computational Linguistics, pp. 8410–8423. DOI: 10.18653/v1/2022.acl-long.576. URL: https://aclanthology.org/2022.acl-long.576.

Gururangan, Suchin et al. (2019). "Variational Pretraining for Semi-supervised Text Classification". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, pp. 5880–5894. DOI: 10.18653/v1/P19-1590. URL: https://www.aclweb.org/anthology/P19-1590.

Hambardzumyan, Karen, Hrant Khachatrian, and Jonathan May (Aug. 2021). "WARP: Word-level Adversarial ReProgramming". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint*

Conference on Natural Language Processing (Volume 1: Long Papers). ACL-IJCNLP 2021. Ed. by Chengqing Zong et al. Online: Association for Computational Linguistics, pp. 4921–4933. DOI: 10.18653/v1/2021.acl-long.381. URL: https://aclanthology.org/2021.acl-long.381.

Han, Xu et al. (2018). "FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, pp. 4803–4809. DOI: 10.18653/v1/D18-1514. URL: http://aclweb.org/anthology/D18-1514.

Hu, Edward J. et al. (2022a). "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations*. Online. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

Hu, Shengding et al. (2022b). "Knowledgeable Prompt-tuning: Incorporating Knowledge into Prompt Verbalizer for Text Classification". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin, Ireland: Association for Computational Linguistics, pp. 2225–2240. DOI: 10.18653/v1/2022.acl-long.158. URL: https://aclanthology.org/2022.acl-long.158.

Isaienkov, Yaroslav and Anton Paramonov (2020). "Comparison of text classification methods for the Ukrainian language". In: URL: https://card-file.ontu.edu.ua/handle/123456789/15749.

Ivanyuk-Skulskiy, Bogdan et al. (Oct. 2021). *ua_datasets: a collection of Ukrainian language datasets*. Version 0.0.1. URL: https://github.com/fido-ai/ua-datasets.

Jiang, Albert Q. et al. (Oct. 10, 2023a). *Mistral 7B*. DOI: 10.48550/arXiv.2310.06825. arXiv: 2310.06825[cs]. URL: http://arxiv.org/abs/2310.06825.

Jiang, Zhiying et al. (July 2023b). ""Low-Resource" Text Classification: A Parameter-Free Classification Method with Compressors". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Findings 2023. Toronto, Canada: Association for Computational Linguistics, pp. 6810–6828. DOI: 10.18653/v1/2023.findings-acl.426. URL: https://aclanthology.org/2023.findings-acl.426.

Karimi Mahabadi, Rabeeh et al. (May 2022). "Prompt-free and Efficient Few-shot Learning with Language Models". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2022. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Dublin, Ireland: Association for Computational Linguistics, pp. 3638–3652. DOI: 10.18653/v1/2022.acl-long.254. URL: https://aclanthology.org/2022.acl-long.254.

Kilgarriff, Adam (Nov. 9, 2001). "Comparing Corpora". In: *International Journal of Corpus Linguistics* 6. DOI: 10.1075/ijcl.6.1.05kil.

Lan, Zhenzhong et al. (Feb. 8, 2020). *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. arXiv: 1909.11942[cs]. URL: http://arxiv.org/abs/1909.11942.

Le, Quoc and Tomas Mikolov (2014). "Distributed representations of sentences and documents". In: *International conference on machine learning*. PMLR, pp. 1188–1196. URL: http://proceedings.mlr.press/v32/le14.html?ref=https://githubhelp.com.

Lee, Hung-yi, Shang-Wen Li, and Thang Vu (July 2022). "Meta Learning for Natural Language Processing: A Survey". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL-HLT 2022. Ed. by Marine Carpuat, Marie-Catherine

de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, pp. 666–684. DOI: 10.18653/v1/2022.naacl-main.49. URL: https://aclanthology.org/2022.naacl-main.49.

Lester, Brian, Rami Al-Rfou, and Noah Constant (2021). "The Power of Scale for Parameter-Efficient Prompt Tuning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 3045–3059. DOI: 10.18653/v1/2021.emnlp-main.243. URL: https://aclanthology.org/2021.emnlp-main.243.

Li, Qian et al. (Apr. 30, 2022). "A Survey on Text Classification: From Traditional to Deep Learning". In: *ACM Transactions on Intelligent Systems and Technology* 13.2, pp. 1–41. ISSN: 2157-6904, 2157-6912. DOI: 10.1145/3495162. URL: https://dl.acm.org/doi/10.1145/3495162.

Linmei, Hu et al. (Nov. 2019). "Heterogeneous Graph Attention Networks for Semi-supervised Short Text Classification". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. EMNLP-IJCNLP 2019. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, pp. 4821–4830. DOI: 10.18653/v1/D19-1488. URL: https://aclanthology.org/D19-1488.

Liu, Hanxiao et al. (June 1, 2021). *Pay Attention to MLPs*. DOI: 10.48550/arXiv.2105.08050. arXiv: 2105.08050[cs]. URL: http://arxiv.org/abs/2105.08050.

Liu, Haokun et al. (2022). "Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning". In: *Advances in Neural Information Processing Systems* 35, pp. 1950–1965. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/0cde695b83bd186c1fd456302888454c-Abstract-Conference.html.

Liu, Xiao et al. (Aug. 26, 2023). "GPT understands, too". In: *AI Open*. ISSN: 2666-6510. DOI: 10.1016/j.aiopen.2023.08.012. URL: https://www.sciencedirect.com/science/article/pii/S2666651023000141.

Liu, Yinhan et al. (July 26, 2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. DOI: 10.48550/arXiv.1907.11692. arXiv: 1907.11692[cs]. URL: http://arxiv.org/abs/1907.11692.

Logan IV, Robert L. et al. (July 1, 2021). *Cutting Down on Prompts and Parameters: Simple Few-Shot Learning with Language Models*. DOI: 10.48550/arXiv.2106.13353. arXiv: 2106.13353[cs]. URL: http://arxiv.org/abs/2106.13353.

Malekzadeh, Masoud et al. (2021). "Review of graph neural network in text classification". In: *2021 IEEE 12th annual ubiquitous computing, electronics & mobile communication conference (UEMCON)*. IEEE, pp. 0084–0091. URL: https://ieeexplore.ieee.org/abstract/document/9666633/.

Meng, Yu et al. (Oct. 17, 2018). "Weakly-Supervised Neural Text Classification". In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. CIKM '18. New York, NY, USA: Association for Computing Machinery, pp. 983–992. ISBN: 978-1-4503-6014-2. DOI: 10.1145/3269206.3271737. URL: https://dl.acm.org/doi/10.1145/3269206.3271737.

Mikolov, Tomas et al. (Sept. 6, 2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv: 1301.3781[cs]. URL: http://arxiv.org/abs/1301.3781.

Munkhdalai, Tsendsuren and Hong Yu (2017). "Meta networks". In: *International conference on machine learning*. PMLR, pp. 2554–2563. URL: https://proceedings.mlr.press/v70/munkhdalai17a.html.

Palanivinayagam, Ashokkumar, Claude Ziad El-Bayeh, and Robertas Damaševičius (May 2023). "Twenty Years of Machine-Learning-Based Text Classification: A Systematic Review". In: *Algorithms* 16.5. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute, p. 236. ISSN: 1999-4893. DOI: 10.3390/a16050236. URL: https://www.mdpi.com/1999-4893/16/5/236.

Panchenko, Dmytro et al. (2022). "Ukrainian News Corpus as Text Classification Benchmark". In: *ICTERI 2021 Workshops*. Ed. by Oleksii Ignatenko et al. Vol. 1635. Series Title: Communications in Computer and Information Science. Cham: Springer International Publishing, pp. 550–559. ISBN: 978-3-031-14840-8 978-3-031-14841-5. DOI: 10.1007/978-3-031-14841-5_37. URL: https://link.springer.com/10.1007/978-3-031-14841-5_37.

Parnami, Archit and Minwoo Lee (Mar. 7, 2022). *Learning from Few Examples: A Summary of Approaches to Few-Shot Learning*. DOI: 10.48550/arXiv.2203.04291. arXiv: 2203.04291[cs]. URL: http://arxiv.org/abs/2203.04291.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543. URL: https://aclanthology.org/D14-1162.pdf.

Perez, Ethan, Douwe Kiela, and Kyunghyun Cho (May 24, 2021). *True Few-Shot Learning with Language Models*. DOI: 10.48550/arXiv.2105.11447. arXiv: 2105.11447[cs,stat]. URL: http://arxiv.org/abs/2105.11447.

Radford, Alec and Karthik Narasimhan (2018). *Improving Language Understanding by Generative Pre-Training*. URL: https://www.mikecaptain.com/resources/pdf/GPT-1.pdf.

Raffel, Colin et al. (2020). "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *The Journal of Machine Learning Research* 21.1. Publisher: JMLRORG, pp. 5485–5551. URL: https://dl.acm.org/doi/abs/10.5555/3455716.3455856.

Reimers, Nils and Iryna Gurevych (2019). "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China: Association for Computational Linguistics, pp. 3980–3990. DOI: 10.18653/v1/D19-1410. URL: https://www.aclweb.org/anthology/D19-1410.

Schick, Timo, Helmut Schmid, and Hinrich Schütze (Dec. 2020). "Automatically Identifying Words That Can Serve as Labels for Few-Shot Text Classification". In: *Proceedings of the 28th International Conference on Computational Linguistics*. COLING 2020. Ed. by Donia Scott, Nuria Bel, and Chengqing Zong. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 5569–5578. DOI: 10.18653/v1/2020.coling-main.488. URL: https://aclanthology.org/2020.coling-main.488.

Schick, Timo and Hinrich Schütze (2021a). "Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume. Online: Association for Computational Linguistics, pp. 255–269. DOI: 10.18653/v1/2021.eacl-main.20. URL: https://aclanthology.org/2021.eacl-main.20.

— (2021b). "It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Online: Association for Computational Linguistics, pp. 2339–2352. DOI: 10.18653/v1/2021.naacl-main.185. URL: https://aclanthology.org/2021.naacl-main.185.

— (June 17, 2022). "True Few-Shot Learning with Prompts—A Real-World Perspective". In: *Transactions of the Association for Computational Linguistics* 10, pp. 716–731. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00485. URL: https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00485/111728/True-Few-Shot-Learning-with-Prompts-A-Real-World.

Snell, Jake, Kevin Swersky, and Richard Zemel (2017). "Prototypical Networks for Few-shot Learning". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/cb8da6767461f2812ae4290eac7cbc42-Abstract.html.

Sun, Chi et al. (2019). "How to Fine-Tune BERT for Text Classification?" In: *Chinese Computational Linguistics*. Ed. by Maosong Sun et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 194–206. ISBN: 978-3-030-32381-3. DOI: 10.1007/978-3-030-32381-3_16.

Tam, Derek et al. (2021). "Improving and Simplifying Pattern Exploiting Training". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 4980–4991. DOI: 10.18653/v1/2021.emnlp-main.407. URL: https://aclanthology.org/2021.emnlp-main.407.

Team, Gemma et al. (Apr. 16, 2024). *Gemma: Open Models Based on Gemini Research and Technology*. DOI: 10.48550/arXiv.2403.08295. arXiv: 2403.08295[cs]. URL: http://arxiv.org/abs/2403.08295.

Team, NLLB et al. (Aug. 25, 2022). *No Language Left Behind: Scaling Human-Centered Machine Translation*. DOI: 10.48550/arXiv.2207.04672. arXiv: 2207.04672[cs]. URL: http://arxiv.org/abs/2207.04672.

Touvron, Hugo et al. (July 19, 2023). *Llama 2: Open Foundation and Fine-Tuned Chat Models*. DOI: 10.48550/arXiv.2307.09288. arXiv: 2307.09288[cs]. URL: http://arxiv.org/abs/2307.09288.

Tunstall, Lewis et al. (Sept. 22, 2022). *Efficient Few-Shot Learning Without Prompts*. DOI: 10.48550/arXiv.2209.11055. arXiv: 2209.11055[cs]. URL: http://arxiv.org/abs/2209.11055.

Wahba, Yasmen, Nazim Madhavji, and John Steinbacher (2023). "A Comparison of SVM Against Pre-trained Language Models (PLMs) for Text Classification Tasks". In: *Machine Learning, Optimization, and Data Science*. Ed. by Giuseppe Nicosia et al. Vol. 13811. Series Title: Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, pp. 304–313. ISBN: 978-3-031-25890-9 978-3-031-25891-6. DOI: 10.1007/978-3-031-25891-6_23. URL: https://link.springer.com/10.1007/978-3-031-25891-6_23.

Wang, Liang et al. (Dec. 7, 2022). *Text Embeddings by Weakly-Supervised Contrastive Pre-training*. DOI: 10.48550/arXiv.2212.03533. arXiv: 2212.03533[cs]. URL: http://arxiv.org/abs/2212.03533.

Wang, Sinong et al. (Apr. 29, 2021). *Entailment as Few-Shot Learner*. DOI: 10.48550/arXiv.2104.14690. arXiv: 2104.14690[cs]. URL: http://arxiv.org/abs/2104.14690.

Wei, Jason and Kai Zou (Aug. 25, 2019). *EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks*. arXiv: 1901.11196[cs]. URL: http://arxiv.org/abs/1901.11196.

Wu, Xing et al. (Dec. 17, 2018). *Conditional BERT Contextual Augmentation*. DOI: 10.48550/arXiv.1812.06705. arXiv: 1812.06705[cs]. URL: http://arxiv.org/abs/1812.06705.

Xiaoju, Ye (2023). *calflops: a FLOPs and Params calculate tool for neural networks in pytorch framework*. URL: https://github.com/MrYxJ/calculate-flops.pytorch.

Xie, Qizhe et al. (Dec. 6, 2020). "Unsupervised Data Augmentation for Consistency Training". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., pp. 6256–6268. ISBN: 978-1-71382-954-6.

Yang, Zhilin et al. (Dec. 8, 2019). "XLNet: generalized autoregressive pretraining for language understanding". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 517. Red Hook, NY, USA: Curran Associates Inc., pp. 5753–5763.

Yu, Mo et al. (2018). "Diverse Few-Shot Text Classification with Multiple Metrics". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). New Orleans, Louisiana: Association for Computational Linguistics, pp. 1206–1215. DOI: 10.18653/v1/N18-1109. URL: http://aclweb.org/anthology/N18-1109.

Yun, Sangdoo et al. (Aug. 7, 2019). *CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features*. DOI: 10.48550/arXiv.1905.04899. arXiv: 1905.04899[cs]. URL: http://arxiv.org/abs/1905.04899.

Zhang, Hongyi et al. (Apr. 27, 2018). *mixup: Beyond Empirical Risk Minimization*. DOI: 10.48550/arXiv.1710.09412. arXiv: 1710.09412[cs, stat]. URL: http://arxiv.org/abs/1710.09412.

Zhao, Zihao et al. (2021). "Calibrate before use: Improving few-shot performance of language models". In: *International Conference on Machine Learning*. PMLR, pp. 12697–12706. URL: http://proceedings.mlr.press/v139/zhao21c.html.