Leipzig University
Institute of Computer Science
Degree Programme Computer Science, B.Sc.

Investigating Core Set-based Active Learning for Text Classification

Bachelor's Thesis

Yannick Brenning
Born Aug. 27, 2002 in Bamberg

Matriculation Number 3732848

1. Referee: Prof. Dr. Martin Potthast
2. Referee: Christopher Schröder
3. Referee: Christian Kahmann

Submission date: April 26, 2024

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Leipzig, April 26, 2024

.....
Yannick Brenning

Abstract

This is the LaTeX template for Bachelor and Master theses at Webis. This template contains several hints and conventions on how to structure a thesis, how to cite the work of others, and how to display your results besides plain text.

Contents

Chapter 1

Introduction

Text is one of the most widespread and important sources of information, but extracting data and tangible knowledge from it can be a difficult and expensive task. With the advent of the digital age, enormous amounts of unstructured texts have become available with more being generated by the day. Due to this increasingly large amount of textual data, manually processing information at a larger scale becomes infeasible and thus demands the use of computer-driven approaches.

The classification of text, meaning the assignment of a category or class to a document or piece of text is one of the most common and useful ways to gain information from a piece of text. As the amount of available text content continues to grow, text classification tasks become an increasingly important area of research within the field of natural language processing.

Thanks to machine learning and data science, many methods of extracting information from text have been developed with the purpose of performing text classification at a larger scale. This possibility for automated organization of data can enhance insights and decision-making across industries such as healthcare, finance, and social sciences, among many others. Active Learning (AL) is a subfield of machine learning in which the learning algorithm is able to perform queries on an information source in order to reduce the total amount of annotated data (Settles [2009]). This method can offer significant advantages in improving model performances and especially in reducing labeling costs. Though there is no universally optimal strategy, AL has been proven to be useful in many cases where annotating data is expensive or the total amount of data is very large (Settles [2009]).

Oftentimes, there is a large amount of unlabelled data available for an AL model to learn from. In this case, selecting the most effective data points to be labelled by the information source from this large pool becomes a crucial, but difficult challenge to overcome.

One attempt at improving the effectiveness of AL in this regard is the Core-Set approach (Sener and Savarese [2018]). This method uses core-set selection to counter the issue of AL ineffectiveness on convolutional neural networks (Sener and Savarese [2018]). The proposed approach selects a set of points the pool such that a model learning over this subset can be consistent when provided the remaining data points. The method was shown to have improved results when compared to other approaches in the field of computer vision (Sener and Savarese [2018], Caramalau et al. [2021]), which encompasses tasks that focus on enabling computers to interpret and understand visual information from the world. This field involves a variety of different methods including image classification, object detection, and semantic segmentation, all of which have significant importance in scientific research, classification tasks, and pattern recognition.

However, the Core-Set approach has been shown to have mixed results in cases of text classification using BERT (Prabhu et al. [2021], Ein-Dor et al. [2020]) and binary text classification using deep neural networks (Liu et al. [2021]). In some cases, Core-Set performs poorly even when compared to the random sampling strategy (Prabhu et al. [2021], Coleman et al. [2022]). Additional research suggests that the approach may even be less effective in computer vision tasks with higher numbers of classes as well as higher-dimensional feature spaces (Sinha et al. [2019]). The theoretical analysis shown in Sener and Savarese [2018] briefly mentions the potential issue of higher class numbers; however, it does not attempt to provide a potential solution to the problem.

The mixed results for Core-Set just mentioned motivated three different research questions for this thesis.

1. Can we improve Core-Set's mixed results within the field of text classification using dimensionality reduction techniques?

2. Does an uncertainty-based approach improve the performance of Core-Set?

3. Can we balance the class distribution within the Core-Set selection, and how will this affect the learning process?

By first explaining Core-Set's functionality and the possible reasons for why it tends to underperform in certain classification tasks, I aim to then examine the performance of Core-Set in comparison to various baseline approaches on large datasets of text content in order to verify this claim. Furthermore, this thesis looks to modify and potentially improve the Core-Set approach for text classification with the previous three research questions in mind and demonstrate the results of these modifications as a part of its experiment.

In the following, Chapter 2 explains the background and related work on the topics of text classification (Section 2.1), active learning in general (Section 2.2), the Core-Set approach (Section 2.3), and dimensionality reduction (Section 2.4) more specifically. In Chapter 3, I describe my approach to modifying Core-Set for text classification using three general approaches. In Chapters 4 and 5, I present my experiment as well as show and discuss its results. Finally, Chapter 6 concludes the thesis and provides insights on potential future developments.

Chapter 2

Background/Related Work

This chapter first provides an overview of the fields of text classification and active learning, which are crucial for understanding Core-Set and its usage scenario within this thesis. I then give a brief review of Core-Set on both a conceptual and functional level. The concepts and notations introduced here will be necessary to understand the reasoning and functionality behind the following approaches employed in the experiment. Finally, this chapter outlines the field of dimensionality reduction, placing emphasis on techniques that will pertain to the subsequent chapters.

2.1 Text Classification

Text classification is one of the most fundamental and important tasks in the field of Natural Language Processing (NLP). As a result, developing efficient automatic text classification methods has become an important research topic.

One of the most common applications of text classification is determining whether the opinion associated with a certain document has a positive or negative sentiment, also known as sentiment analysis. This has a wide range of uses, including the possibility for businesses to better gauge customer opinions on products and services (Liu and Zhang [2012]) in order to adapt accordingly. This application is a binary classification task, meaning the classifier has two classes with which each document can be labelled (positive or negative). Similarly, one might apply this binary classification task to the problem of spam filtering in e-mails, text messages and more.

Beyond that, many applications of text classification require multiple classes, such as news and content categorization. In this case, text classification algorithms can organize documents into specific topics or themes (e.g. Sports, Business, Politics, ...) (Sebastiani [2002]). Other applications include information retrieval, recommender systems, and document summarization (Kowsari et al.

[2019]).

Generally, text classification methods can be divided into the following phases: data collection and preprocessing, feature extraction, classifier selection, and model evaluation (Kowsari et al. [2019], Mironczuk and Protasiewicz [2018], Ikonomakis et al. [2005]).

In the first stage, some form of text data necessary to complete some classification objective is acquired, ideally of a sufficient amount. Several open data sets are publicly available to this end (Pang and Lee [2005], Zhang et al. [2015], Li and Roth [2006]). The preprocessing phase includes steps such as lemmatization, removal of stop words, tokenization and stemming.

The preprocessed text data must them be converted into numerical feature vectors. There are a number of techniques for accomplishing this such as Bag-of-Words, TF-IDF (Term Frequency-Inverse Document Frequency), and word embedding methods such as Word2Vec or GloVe. In addition to the word embedding methods just mentioned which are known as static word embeddings, there exist context-sensitive methods as of the late-2010s such as BERT and ELMo (Devlin et al. [2019], Peters et al. [2018]), which can better represent the varied senses encompassed by words depending on their contexts.

The next phase, classifier selection, is one of the most crucial steps in the text classification pipeline. Without a comprehensive grasp of the underlying concepts of each algorithm, we cannot effectively determine an appropriate model for the task. Commonly known algorithms include Logistic Regression, Naïve Bayes and Support Vector Machines. More recently deep learning models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Transformer Models have become established as state-of-the-art approaches, especially when considering large text classification tasks (Chen [2015], Liu et al. [2016], Lai et al. [2015], Devlin et al. [2019], Sun et al. [2019]). These different deep learning algorithms are increasingly popular due to their ability to model more complex and non-linear relationships within data (LeCun et al. [2015]). The Core-Set approach was also originally developed for the deep learning domain, more specifically CNNs. As a result, this thesis focuses on the use of deep learning models, specifically transformers.

Model evaluation is usually the "final" phase of the text classification process. This encompassees the assessment of the classifier's performance, for which a myriad of metrics such as accuracy, F1-score and AUC can be employed. Based on the evaluation, one can select a suitable model/strategy as well as attempt to optimize it, e.g. by tuning hyperparameters.

The process of text classification clearly includes many steps which can be optimized and examined. For this reason, going over the entire process in detail would exceed the scope of this thesis. With this in mind, this thesis' experiment does not focus on the initial steps such as data collection, preprocessing and

feature extraction, but rather the model training and evaluation steps.

2.2 Active Learning

Active Learning has become an increasingly important field when considering the need for efficient model construction as well as the labelling bottleneck in various machine learning tasks. Many fields, such as speech recognition, information extraction and classification suffer from this bottleneck as a result of their instance labels being expensive or time-consuming to obtain (Settles [2009]).

Generally, active learning takes place in one of three main scenarios, namely *membership query synthesis*, *stream-based selective sampling* and *pool-based active learning*.

In the case of membership query synthesis, a membership query is created in the form of some unlabelled instance from the original dataset (Angluin [1987],Wang et al. [2015]). In other words, this approach synthesizes a query *de novo*, rather than selecting some query from the instance space. One challenge of this approach is ensuring that the synthesized query is consistent with meeting the constraints imposed by the real data. In the case of a human oracle, this can result in unrecognizable queries, for example in the case of computer vision tasks. Similarly, this issue could also apply to text classification, where the query synthesis produces unintelligible texts (Lang and Baum [1992], Settles [2009]).

Stream-based selective sampling, in contrast, typically examines each unlabelled instance one at a time and decides whether to query the oracle or to assign a label (Settles [2009]).

Finally, we consider the pool-based approach as one of the commonly known applications of AL. This scenario assumes a large set of unlabelled instances and a small set of labelled instances, from which queries can be selectively drawn using a *query strategy*. Generally, most approaches begin with some random selection of points to initialize the classifier. With each AL iteration, the query strategy selects new instances on which to train the classifier. This process is then repeated until some stopping criterion is met.

The key difference to the stream-based approach is that pool-based learning considers the entire collection and attempts to select the most informative instances, as opposed to making individual query decisions. This can be useful in many real-world problems where plenty of unlabelled data is available and the pool is assumed to be closed (though this is not necessarily always the case) (Settles [2009]).

This thesis considers the case of the pool-based scenario, as does the original

Core-Set paper. It is worth mentioning however, that Core-Set has also been applied in the stream-based context (Saran et al. [2023]).

2.3 Core-Set

The Core-Set approach as proposed by Sener and Savarese [2018] was originally designed for CNNs to tackle the problem of AL algorithms on large datasets. The empirical studies conducted by Sener and Savarese [2018] show that Core-Set outperforms other established query strategies when applied in the field of computer vision.

The task of AL is defined as a *core-set selection* problem in which the algorithm selects a smaller subset of points with increased diversity to learn from such that the model can be competitive over a larger dataset. This problem ends up being equivalent to the *k-Center* problem (Sener and Savarese [2018]), which is also referred to as the minimax facility location problem.

In mathematical terms, given a set of points N and a budget $k \leq |N|$, k-Center finds a subset $\mathbf{s} \subseteq N$ of k points such that the maximum distance of any point in $N \setminus \mathbf{s}$ to its closest center in $\mathbf{s}$ is minimal (Har-Peled [2008]). Core-Set applies this concept to the field of AL in that the k centers selected from N will be the instances selected to be labelled by our oracle.

Another way of viewing this problem is by placing circles around the points in our set of centers. If we denote the maximum distance of any point in N to its nearest center in $\mathbf{s}$ with $\delta_{\mathbf{s}}$ and we place circles with a $\delta_{\mathbf{s}}$-radius around each center in $\mathbf{s}$, we can "cover" the entire set of points N. In other words, k-Center attempts to find the minimum $\delta_{\mathbf{s}}$ such that all points lie within the union of the $\delta_{\mathbf{s}}$-circles when placed upon each center (depicted in Figure 2.1). Due to the nature of k-Center, the selected points tend to be spread out throughout the dataset in order to "cover" all the unselected points, making this a diversity-based approach.

This problem has been shown to be NP-hard (Hsu and Nemhauser [1979], Hochbaum [1984]). As a result, Core-Set uses a greedy approach to solve the k-Center problem. However, it is worth noting that any approximation of k-Center is bound by twice the optimal solution (Hsu and Nemhauser [1979]). Let OPT denote the maximum distance of a center to a point in the optimal solution to k-Center, meaning the solution where $\delta_{\mathbf{s}}$ is minimal. Then the greedy approximation's resulting maximum distance $\delta_{\mathbf{s}}$ to any center in $\mathbf{s}$ is at the worst $2 \cdot OPT$ (Mount [2017]). The pseudocode used by Sener and Savarese [2018] for k-Center greedy within the active learning context can be seen in Algorithm 1.

The original paper by Sener and Savarese [2018] attempts to further im-

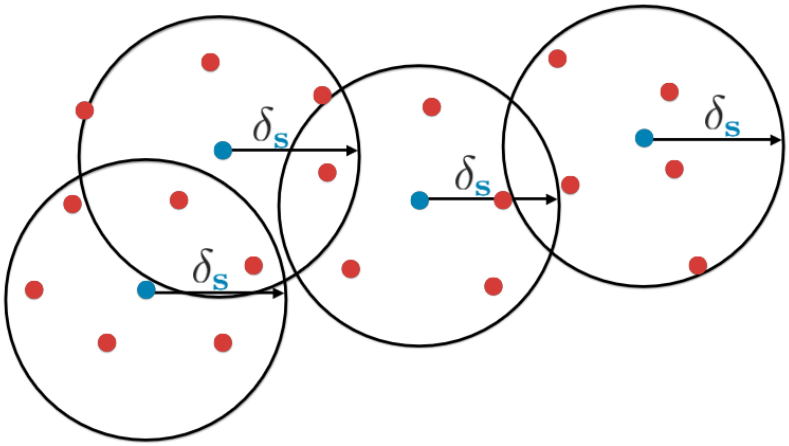

Figure 2.1: Visualization of the k-Center approach from Sener and Savarese [2018]. $\mathbf{s}$ denotes the set of k selected points, $\delta_{\mathbf{s}}$ denotes the maximum distance of any point in N to its nearest center in $\mathbf{s}$.

Algorithm 1 k-Center-Greedy (adopted from Sener and Savarese [2018])

Input: data $\mathbf{x}_i$, existing pool $\mathbf{s}^0$, budget b
  Initialize $\mathbf{s} = \mathbf{s}^0$
  repeat
    $u = \text{argmax}_{i \in [n] \setminus \mathbf{s}} \text{min}_{j \in \mathbf{s}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
    $\mathbf{s} = \mathbf{s} \cup \{u\}$
  until $|\mathbf{s}| = b + |\mathbf{s}^0|$
  return $\mathbf{s} \setminus \mathbf{s}^0$

prove the robustness of this approximation by placing an upper limit on the number of outliers that can be selected. This thesis focuses on and uses the regular k-Center greedy algorithm, which is generally better established due to ease of implementation and interpretability.

2.4 Dimensionality Reduction

As pointed out earlier, one of the major challenges of the Core-Set approach is handling data points with a higher dimensionality (Sinha et al. [2019]). Broadly speaking, this is a phenomenon coined by Richard E. Bellman known as the *curse of dimensionality* (François [2007]). As a result, many algorithms have been developed to transform data from a high-dimensional space into a low-dimensional space. This task directly poses another challenge: managing to reduce the dimensionality of the data while still being able to retain the highest possible amount of information.

One such method is the Principal Components Analysis (PCA), which is one of the most popular linear techniques for dimensionality reduction (Van Der Maaten et al. [2009]). In essence, PCA linearly transforms the data into a representation that attempts to closely describe the variance of the initial data (Jolliffe and Cadima [2016]). Other commonly used linear techniques include Linear Discriminant Analysis (LDA), Multidimensional Scaling (MDS), and Non-negative Matrix Factorization (NMF).

These techniques can be powerful; however, they often miss important non-linear structures in high-dimensional data. Therefore, non-linear techniques have been developed such as Isometric Mapping (Isomap), t-distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP).

2.4.1 t-distributed Stochastic Neighbor Embedding

t-SNE is a relatively modern probablistic approach that has improved upon many other non-linear techniques in creating a single map that reveals structure on many different scales. In addition, it manages to reduce the tendency of Stochastic Neighbour Embedding (SNE) to crowd data points together at the center by using a different cost function (Van der Maaten and Hinton [2008]).

First, it converts the high-dimensional Euclidean distances into conditional probabilities, such that similar data points are assigned higher probabilities and dissimilar data points are assigned very low probabilities. It then creates a similar probability distribution over the lower-dimensional map such that the *Kullback–Leibler divergence* (a measure of how one probability distribution differs from another) is minimized.

The caveat of this technique is that its efficiency on large datasets relies on a variant of the Barnes–Hut algorithm, which results in reductions only being possible to the two- or three-dimensional space (van der Maaten [2013]). Because of this drawback, the method is generally used for the visualization of high-dimensional data, for which a reduction to two- or three-dimensional spaces is ideal.

2.4.2 Uniform Manifold Approximation and Projection

UMAP exists within a similar vein to t-SNE, but offers important improvements in some areas. This technique models the data with a fuzzy topological representation using simplices, then attempts to find some low-dimensional projection with the closest possible equivalent fuzzy topological structure (McInnes and Healy [2018]). Another key difference is that UMAP accomplishes this projection by attempting to minimize the cross-entropy between the topological representations, whereas t-SNE uses the Kullback-Leibler divergence.

This paper also elaborates that the resulting reductions are competitive with t-SNE in the field of visualization and offer superior runtime performances. In addition, there exists no computational restriction on the embedding dimension, as opposed to t-SNE. As a result, McInnes and Healy [2018] also mentions it as a useful tool not only for visualization, but for machine learning tasks in general.

Chapter 3

Approach

The following chapter describes my approach to the subsequent experiment. The attempt to improve upon and modify Core-Set resulted in multiple different iterations of the original k-Center greedy algorithm, which I have grouped into three different categories. Each of these different categories is designed to address one of the three research questions central to this thesis.

I will first outline the dimensionality reduction-based approach, considering the two different nonlinear techniques discussed previously. After that, I will present two uncertainty-based approaches which take pointwise probabilities into account. The final approach for the experiment will attempt to use class balancing in combination with Core-Set.

3.1 Dimensionality Reduction-Based

As mentioned, distance-based methods such as Core-Set can be ineffective in higher dimensions. To overcome this challenge, I propose the application of a non-linear dimensionality reduction technique on the training data before selecting new points using Core-Set. My aim with this is attempting to preserve any non-linear structures within the datasets. In essence, this would modify the original k-Center greedy approach in that the distances computed between the points in $\mathbf{s}$ and the unselected points would be different, since the algorithm would take in reduced embeddings (this modification is noted in Algorithm 2).

This resulted in two dimensionality-reduction based approaches. The first strategy, which I refer to as CS-tSNE, will use t-SNE in order to reduce the embeddings down to two dimensions. Additionally, I propose the use of UMAP as an alternative, which I will use to reduce the data to a 256-dimensional space.[1] This strategy will be referred to as CS-UMAP.

[1]Both of these approaches assume 768-dimensional embeddings.

Algorithm 2 Dimensionality-Reduced k-Center-Greedy

Input: data $\mathbf{x}_i$, existing pool $\mathbf{s}^0$, budget b
  Initialize $\mathbf{s} = \mathbf{s}^0$
  repeat
    $u = \text{argmax}_{i \in [n] \setminus \mathbf{s}} \text{min}_{j \in \mathbf{s}} \Delta_{Reduced}(\mathbf{x}_i, \mathbf{x}_j)$ ▷ Computing distances on
    $\mathbf{s} = \mathbf{s} \cup \{u\}$ ▷ reduced embeddings
  until $|\mathbf{s}| = b + |\mathbf{s}^0|$
  return $\mathbf{s} \setminus \mathbf{s}^0$

3.2 Uncertainty-Based

Next, I will examine the effect of pointwise probabilities on the Core-Set algorithm. Core-Set currently operates using the distances between its points – in this experiment, I propose an approach where Core-Set considers the probability distributions in addition to the distances.

To accomplish this, I will be using an uncertainty-based approach. Uncertainty sampling, which was introduced by Lewis and Gale [1994], essentially attempts to select the unlabelled examples with the lowest classification certainty. In this case, I specifically opted for an approach similar to that of Breaking-Ties (Luo et al. [2005]), where the instances with the smallest margin between their top two most likely classification probabilities are selected, essentially "breaking the tie" between the two most likely classes. In other words, if $\mathbf{p}^*_{j,k}$ denotes the probability of the k-th most likely class label for the j-th instance, then Breaking-Ties seeks to select instances where $\mathbf{p}^*_{j,1} - \mathbf{p}^*_{j,2}$ is minimal.

This approach resulted in two different implementation ideas: the first, which I will refer to as "Weighted Core-Set" (WCS), calculates the uncertainties prior to executing the standard Core-Set algorithm, then multiplies the resulting Core-Set distances and probabilities using weights (in this case, I used an 80–20 weighting in favour of the Core-Set distances).

The second, which I will call "Re-ranked Core-Set" (RWCS), opts to first compute a Core-Set twice the size of the original sample size b, then "re-ranks" the resulting distances according to Breaking-Ties uncertainties and finally selects the points with the b highest uncertainties.

In theory, these two approaches aim to use uncertainties to improve Core-Set's selections such that the newly selected set of points should ideally contain instances which also have high uncertainties.
$\mathbf{p}^*_{j,k}$ denotes the probability of the k-th most likely class label for the j-th instance.

Algorithm 3 Weighted k-Center-Greedy

Input: $\mathbf{x}_i$, $\mathbf{s}^0$, b, breaking-ties probabilities $\mathbf{p}_{bt}$
  Initialize $\mathbf{s} = \mathbf{s}^0$
  repeat
    $d = \min_{j \in \mathbf{s}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
    $u = \text{argmax}_{i \in [n] \setminus \mathbf{s}} 0.8 \cdot d + 0.2 \cdot \mathbf{p}_{bt}$ ▷ Weigh results using linear
    $\mathbf{s} = \mathbf{s} \cup \{u\}$ ▷ combination
  until $|\mathbf{s}| = b + |\mathbf{s}^0|$
  return $\mathbf{s} \setminus \mathbf{s}^0$

Algorithm 4 Re-ranked k-Center-Greedy

Input: $\mathbf{x}_i$, $\mathbf{s}^0$, b, class probabilities $\mathbf{p}_i$
  Initialize $\mathbf{s} = \mathbf{s}^0, r = \emptyset$
  repeat
    $u = \text{argmax}_{i \in [n] \setminus \mathbf{s}} \min_{j \in \mathbf{s}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
    $\mathbf{s} = \mathbf{s} \cup \{u\}$
  until $|\mathbf{s}| = 2b + |\mathbf{s}^0|$ ▷ Compute Core-Set of size $2b$
  repeat
    $u = \text{argmin}_{j \in \mathbf{s} \setminus r} \mathbf{p}^*_{j,1} - \mathbf{p}^*_{j,2}$
    $r = r \cup \{u\}$
  until $|r| = b$ ▷ Compute the b-highest BT-scores
  return r

3.3 Class Balance-Based

Finally, I will take into account the class distributions within each Core-Set and select points based on the representation of each class. This approach will be called "Class-Balanced Core-Set" (CB-CS).

The basic idea with this approach is that the class distribution within out pool of labelled instances should, in theory, have an even class distribution such that the models learns from similar amounts of instances from each class. In other words, the query strategy (in this case Core-Set) should attempt to select more or less instances of a certain class depending on the discrepancy between the class distribution of the labelled pool and some balanced distribution of that class.

One way of determining the degree to which our pool of instances is "class-balanced" is by using the normalized entropy metric. This metric is based on the Shannon entropy H (Shannon [1948]), which is defined in Equation 3.1 over some discrete random variable X, which is in this case the class distribution.

$$H(X) = -\sum_{x \in X} p(x) \log p(x) \tag{3.1}$$

Dividing by the logarithm of the number of classes n for the class distribution, we get the normalized entropy H_norm(Equation 3.2), also referred to as to as H_{REL} (Wilcox [1967]).

$$H_{norm}(X) = \frac{-\sum_{x \in X} p(x) \log p(x)}{\log n} \tag{3.2}$$

This metric scales the output to lie between 0 and 1, such that 1 indicates a uniform distribution. This can be applied to the measurement of class imbalances, where a normalized entropy value closer to 1 indicates a more balanced class distribution and is thus a desirable result.

To this end, I will divide a single query into multiple Core-Set computations, essentially constructing disjoint Core-Set selections for each class. For each class' Core-Set selection, the budget b (e.g., the amount of selected instances from this class) will depend on the desired class distribution.

First, the class distributions of the model's predictions and of the true labels are used to determine how many instances should ideally be queried for each class in order to achieve a "balance" in the labelled pool. In the corresponding pseudocode (CB-CS), this calculation has been denoted by the use of `target_dist`, which uses an existing implementation from the Python library Small-Text (Schröder et al. [2023]).

After this, k-Center greedy is used to select the desired number of points of each class, such that if k_i denotes the number of newly selected instances

for the ith class, then $\sum_{c_i \in C} k_i = k$.

Algorithm 5 Class-Balanced k-Center Greedy

Input: $\mathbf{x}_i$, $\mathbf{s}^0$, b, true class labels y_i, predicted class labels y_i^{pred} classes C
  $d = \text{target\_dist}_{i \in [n] \setminus \mathbf{s}}(y_i, y_i^{pred}, C)$
  Initialize $\mathbf{s} = \mathbf{s}^0$
  for $c_i \in C$ **do**
    repeat
      $u = \text{argmax}_{i \in [n] \setminus \mathbf{s}} \text{min}_{j \in \mathbf{s}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
      $\mathbf{s} = \mathbf{s} \cup \{u\}$
    until $|\mathbf{s}| = b + |\mathbf{s}^0|$
    $\mathbf{s} = \mathbf{s} \cup \mathbf{s}_{c_i}$
  end for
  return $\mathbf{s} \setminus \mathbf{s}^0$

Chapter 4

Experiment

With these approaches in mind, this chapter will detail the experiment, first describing the data and setup of the experiment, and finally delving into its results.

This experiment will take place within a simulated AL environment, meaning the oracle will not be a human expert or annotator, but rather an already labelled pool of data. To this end, I will select three established pre-labelled text classification datasets of different varieties. The AL will take place on these datasets using two different classifier models, each of which I will then combine with one of eight different query strategies. The query strategies to be compared will contain two baseline approaches, Core-Set itself, and the implementations of the previously discussed Core-Set variations.

The following two sections will explain in greater detail the data and setup used as a basis for the conducted experiment. The performances will then be measured using two metrics, accuracy and AUC, throughout five runs for each combination of dataset, classifier and query strategy. In the final section, I will present and evaluate the experiment results.

4.1 Data

This experiment was conducted across three datasets commonly used in the field of text classification. These datasets are of three different types: sentiment analysis, questions, and news.

For binary classification, I used **Movie Review**, a sentiment analysis dataset (Pang and Lee [2005]) containing 5,331 positive and 5,331 negative online movie reviews. For multi-class text classification, I performed the AL on **AG's News** (Zhang et al. [2015]), a news dataset comprised of 120,000 training samples and 7,600 test samples, and **TREC** (Li and Roth [2006]), a question dataset containing 5,500 training samples and 500 test samples. The

test set was provided in the case of AG's News and TREC, in the case of Movie Review I employed a split of the 10,662 samples myself, which can be seen alongside additional information in Table 4.1. Example instances from each dataset as well as their corresponding classes can also be seen in Table 4.2.

Dataset Name (ID)	**Classes**	**Train**	**Test**	**MNC (*)**
Movie Review (MR)	2	9,596	1,066	114.16
AG News (AGN)	4	120,000	7,600	236.41
TREC (TREC)	6	5,452	500	49.39

Table 4.1: Information on the different datasets. (*) Mean number of characters in a single instance.

Dataset Name (ID)	**Class**	**Example Instance**
Movie Review (MR)	1 (positive)	`if you sometimes like to go to the movies to have fun , wasabi is a good place to start .`
AG News (AGN)	2 (Business)	`Wall St. Bears Claw Back Into the Black (Reuters) Reuters - Short-sellers, Wall Street's dwindling\band of ultra-cynics, are seeing green again.`
TREC (TREC)	2 ('DESC')	`How did serfdom develop in and then leave Russia ?`

Table 4.2: Examples of instances and their corresponding labels from each dataset. The example label 'DESC' signifies a description or abstract concept.

4.2 Experiment Setup

I selected two state-of-the-art transformers, BERT (Devlin et al. [2019]) and SetFit (Tunstall et al. [2022]) to fine-tune as models for this experiment. BERT

is an established language model pre-trained on 3,300M english words. It uses bidirectional self-attention in order to incorporate context from both directions of a given token, unlike many previous context-sensitive approaches which only considered a single direction. For this experiment, I will be using $\text{BERT}_{\text{BASE}}$, which consists of 12 layers, hidden units of size 768 with 110M parameters in total. SetFit, on the other hand, is based on sentence transformers (Reimers and Gurevych [2019]) which are then fine-tuned in a contrastive manner. This model is similarly made up of 12 layers and generates 768-dimensional embeddings, but has a parameter count of 109M.

The experiment is conducted by performing 20 queries on 25 instances, with a validation set size of 10%. The results will be averaged over five runs of queries per combination of dataset, model and query strategy. The training and evaluation of these models is run using an AL experiment setup based on the Python library Small-Text (Schröder et al. [2023]).

Alongside the original Core-Set (CS), I will be comparing my different approaches with Random Sampling (RS) and Breaking-Ties (BT). RS offers a good view of a baseline performance and will be interesting when compared to Core-Set. BT, on the other hand, may serve as a good upper bound when looking at improving Core-Set's performance.

4.3 Experiment Results

Based on the experimental setup outlined in Section 4.2, this chapter presents the outcomes for each experiment run. I first present the learning curves for the each training run, then report on the corresponding result tables.

The following figures show the learning curves for each combination of dataset, model and query strategy.

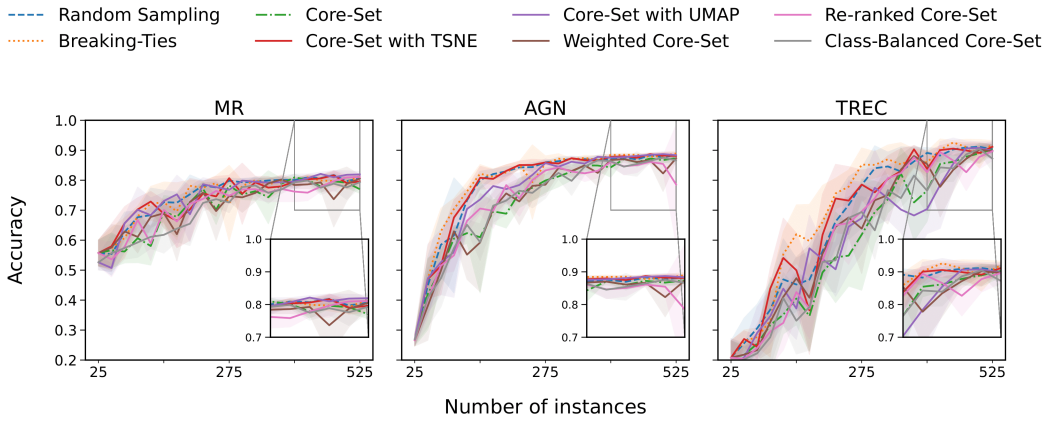

Figure 4.1: Active Learning curves of BERT on each dataset when combined with seven query strategies: Random Sampling, Breaking-Ties, Core-Set, Core-Set with t-SNE, Core-Set with UMAP, Weighted Core-Set, Re-ranked Core-Set and Class-Balanced Core-Set. The lines represent the mean accuracy and the surrounding tubes represent the standard deviation over five runs.

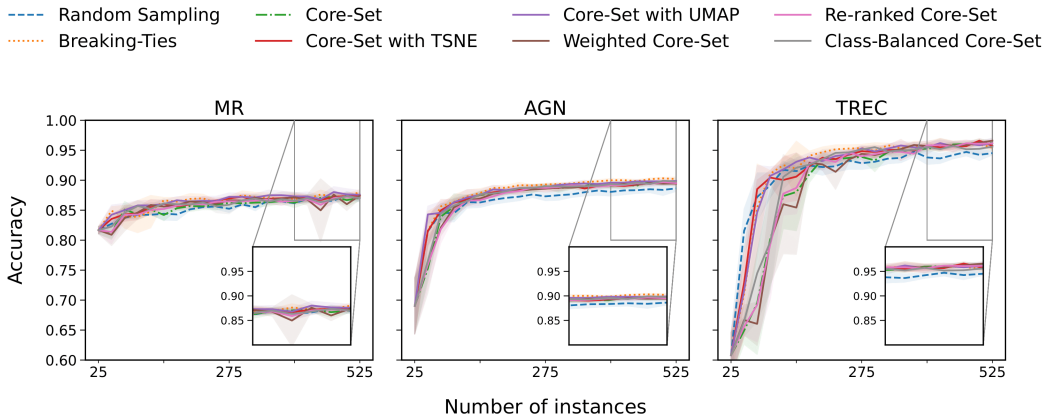

Figure 4.2: Active Learning curves of SetFit on each dataset when combined with eight query strategies: Random Sampling, Breaking-Ties, Core-Set, Core-Set with t-SNE, Core-Set with UMAP, Weighted Core-Set, Re-ranked Core-Set and Class-Balanced Core-Set. The lines represent the mean accuracy and the surrounding tubes represent the standard deviation over five runs.

Overall, the SetFit learning curves generally outperform the BERT model and seem to show much less variation between the different query strategies. This is reflected not only in the mean curve progressions of each model, but also in the smaller standard deviations of each individual line. Even in the case of BERT's resulting AL curves, the most noticeable accuracy differences seem to mainly occur in early iterations. Generally, the mean accuracy scores of the different strategies tend to lie in a similar range by the final iteration.

I will first compare and contrast the different results for the training of the BERT model (Figure 4.1). We can clearly see general upwards trends to varying degrees depending on the dataset. The highest final accuracies are achieved on the TREC dataset, whereas the learning curves for AGN have the fastest increase in early iterations. Both seem to resemble a saturation curve, with TREC's mean accuracy values starting to slowly level out around 90%, whereas AGN's stop around 85% near the second half of the learning process. From this point on, the curves of both datasets gradually rise by around 5% towards the end of training. On the other hand, the learning curves on MR show a more gradual, steady rise throughout the entire process and finish with slightly lower overall accuracies, with all results coming out to just under or around 80%.

When comparing the curves of the different query strategies, we can see that specifically in the case of BERT, CS does indeed seem to underperform when compared to the baseline approaches. This is mainly noticeable when looking at the earlier iterations (between queries 1 to 10). The underperformance of CS becomes even more apparent when considering the fact that the learning curves of RS are consistently above those of CS. This further reinforces the sentiment that Core-Set may have weaknesses within the domain of text classification.

In the case of AGN and TREC, we can clearly see the curve of BT achieve high performances throughout, although CS-tSNE is able to compete very closely on AGN. Interestingly enough, RS has managed to achieve similar scores to the strategies mentioned previously in these cases.

We also observe that in most cases, the different variations of Core-Set seem to be on par with or more performant than the original Core-Set. Especially CS-tSNE looks to be a strong contender in all cases and manages to perform similarly to or better than Core-Set and the remaining variants.

CS-UMAP also performs favorably, especially in the case of MR, where it remains on the upper end of accuracy scores throughout and manages to outperform all other strategies by the final iteration. In the case of AGN and TREC, we still find improvements to Core-Set during training, although these datasets seem to generally favour the t-SNE approach. Interestingly, the learning curve takes a dip in the latter half of training on the TREC dataset and rises rapidly again in the final iterations. Notably, a comparable sharp

decrease is apparent in the first half of training on the same dataset for CS-tSNE's curve.

The class-balanced approach, on the other hand, does not seem to show significant accuracy improvements during training in comparison to the regular Core-Set approach. One discernible difference with the learning curve of CB-CS is noticeable on the MR dataset, where its curve seems to have much less fluctuations than CS and its other variations. Generally, however, CS and its class-balanced version show very similar learning curve trends across all three datasets.

Weighted Core-Set and Re-ranked Core-Set are in a similar vein to, albeit oftentimes slightly above, the Class-Balanced approach. Both curves fluctuate around the same level or slightly higher than CS, but do not manage to surpass the baseline approaches in early iterations. We can see that in the case of AGN, the re-ranked strategy's learning curve even takes a dip in mean accuracy towards the end, resulting in a mean final accuracy below that of all other strategies by the final iteration despite being relatively competitive in prior iterations.

The SetFit learning curves (Figure 4.2) show much less fluctuation throughout the individual learning curves, as well as smaller discrepancies between the different query strategies during training.

In the case of MR, all strategies have very similar accuracy curves and bundle tightly around the final iterations, with all final values somewhere around 87,5%. For this dataset, the increase throughout the entire process is very minimal, amounting to only around 5% from the first query to the end of training. As for AGN and TREC, more learning progress is visible from beginning to end. Moreover, the margins between the curves are slightly wider, and some differences can be noted when comparing the different strategies. For these two datasets, the curves again show large early increases around the first five iterations, with the values starting to level out somewhere between instance 150 and 275. For these datasets, the values by the last iteration are also higher than for MR, with final results around the 90% mark on AGN and around 95% on TREC.

Again, BT stands out as one of the best strategies, with CS-TSNE generally following a similar trend. In contrast to BERT, CS itself does not seem to underperform throughout in any significant way. Only in the first half of the learning process on TREC can we see some kind of noticeable difference between CS and the other curves. In these earlier instances, CS is outperformed by the baseline strategies as well as the dimensionality reduction-based strategies.

In this diagram we can also observe the different strategies fall into two "groups" of curve progressions in the beginning iterations. Similarly to Fig-

ure 4.1, we can see WCS, RCS and CB-CS following a very close trend to that of CS. The other query strategies (RS, BT, CS-tSNE, CS-UMAP) seem to follow a slightly higher, but similarly bundled, trend. This bundling of query strategies is only noticeable on the TREC dataset curves, whereas the other graphs seem to show all or most strategies bundled closer together overall.

One slight outlier in these graphs is RS, of which the accuracy values stay visibly below the other query strategies in the second half of each run, specifically when considering AGN and TREC. For these two datasets, RS seems to be competitive in early iterations, outperforming CS and many of the other strategies, but progress rapidly slows, resulting in lower final scores than the other strategies.

The following page shows two tables containing the mean accuracy (Table 4.3) and mean AUC values (Table 4.4) corresponding to the learning curves.

Dataset	Model	Query Strategy							
		RS	BT	CS	CS-TSNE	CS-UMAP	WCS	RCS	CS-CB
AGN	BERT	0.884 ± 0.004	$\mathbf{0.889 \pm 0.010}$	0.874 ± 0.012	0.881 ± 0.010	0.884 ± 0.010	0.873 ± 0.011	0.785 ± 0.221	0.866 ± 0.016
	SetFit	0.886 ± 0.006	$\mathbf{0.902 \pm 0.004}$	0.895 ± 0.003	0.895 ± 0.003	0.899 ± 0.001	0.895 ± 0.005	0.895 ± 0.004	0.898 ± 0.006
MR	BERT	0.806 ± 0.011	0.815 ± 0.009	0.770 ± 0.015	0.796 ± 0.020	$\mathbf{0.819 \pm 0.011}$	0.806 ± 0.014	0.811 ± 0.013	0.793 ± 0.021
	SetFit	0.869 ± 0.006	$\mathbf{0.880 \pm 0.005}$	0.871 ± 0.005	0.874 ± 0.005	0.876 ± 0.008	0.874 ± 0.007	0.870 ± 0.004	0.870 ± 0.006
TREC	BERT	0.904 ± 0.018	$\mathbf{0.920 \pm 0.014}$	0.902 ± 0.021	0.912 ± 0.009	0.898 ± 0.033	0.897 ± 0.027	0.897 ± 0.036	0.872 ± 0.048
	SetFit	0.945 ± 0.004	0.954 ± 0.006	0.962 ± 0.004	0.957 ± 0.007	0.962 ± 0.010	$\mathbf{0.966 \pm 0.004}$	0.962 ± 0.003	0.956 ± 0.007

Table 4.3: Final accuracy per dataset, model, and query strategy. We report the mean and standard deviation over five runs. The best result per dataset is printed in bold.

Dataset	Model	Query Strategy							
		RS	BT	CS	CS-TSNE	CS-UMAP	WCS	RCS	CS-CB
AGN	BERT	0.790 ± 0.015	$\mathbf{0.800 \pm 0.009}$	0.735 ± 0.020	0.792 ± 0.007	0.772 ± 0.019	0.731 ± 0.014	0.741 ± 0.015	0.733 ± 0.017
	SetFit	0.865 ± 0.007	$\mathbf{0.881 \pm 0.004}$	0.871 ± 0.005	0.875 ± 0.004	0.879 ± 0.004	0.870 ± 0.003	0.870 ± 0.002	0.873 ± 0.003
MR	BERT	$\mathbf{0.750 \pm 0.007}$	0.746 ± 0.011	0.718 ± 0.009	0.741 ± 0.017	0.748 ± 0.005	0.720 ± 0.004	0.718 ± 0.005	0.706 ± 0.015
	SetFit	0.854 ± 0.002	$\mathbf{0.864 \pm 0.005}$	0.856 ± 0.003	0.862 ± 0.003	0.866 ± 0.005	0.858 ± 0.007	0.857 ± 0.003	0.859 ± 0.002
TREC	BERT	0.674 ± 0.029	$\mathbf{0.709 \pm 0.008}$	0.594 ± 0.022	0.676 ± 0.037	0.609 ± 0.029	0.629 ± 0.024	0.624 ± 0.012	0.598 ± 0.025
	SetFit	0.914 ± 0.011	$\mathbf{0.923 \pm 0.007}$	0.896 ± 0.015	0.919 ± 0.005	0.921 ± 0.008	0.893 ± 0.012	0.897 ± 0.012	0.905 ± 0.010

Table 4.4: Final AUC per dataset, model, and query strategy. We report the mean and standard deviation over five runs. The best result per dataset is printed in bold.

In Table 4.3, BT dominates for almost every dataset on both models. Only in two cases does another strategy achieve the highest overall final mean accuracy. Even though the differences are predominantly characterized by very small margins throughout, there are still noteworthy observations to be made.

First, we can again see that RS outperforms CS when used with BERT throughout all datasets. CS seems to encounter minor difficulties specifically on the MR dataset with BERT, where it marks the lowest result of all the strategies. Despite this, CS' training with the SetFit model yields improved results, particularly when contrasted with those achieved by RS with SetFit.

The results of the CS variations show small improvements in accuracy at various points. CS-tSNE and CS-UMAP most notably show slightly higher scores than CS in almost all cases. For CS-tSNE, the only exception is on TREC with SetFit. This row of results has some other noteworthy differences between query strategies. We can see that is the only combination of model and dataset for which CS, along with all of its variations, scores a higher result than BT. In the case of CS-UMAP, the only instance where we find no improvement to CS is on the TREC dataset. However, we observe quite consistent accuracy scores above 80% for this strategy across the board. Additionally, this approach manages to improve upon Core-Sets relatively low accuracy score on the MR dataset with BERT, increasing from 77% to nearly 82%.

Although WCS does have the best result in the case of the TREC dataset with SetFit, the remaining results in its column do not show consistent improvements when compared to CS. A similar sentiment applies to RCS and CS-CB, whose results in comparison to CS seem to be a mixed bag.

Even so, all of the CS variations also show some kind of improvement in the case of MR with BERT, in which CS has the lowest accuracy overall. Notably, AGN is the only dataset where CS-TSNE, WCS and RCS do not have any impact on the accuracy score in the case of SetFit. Even in cases where results are not exactly the same, they often remain within a one-standard-deviation range of each other. One row where this is most evident is the combination of SetFit and MR across all query strategies.

Table 4.4 contains the reported mean AUC (area under curve) and the corresponding standard deviation for each combination of dataset, model and query strategy. The top performing AUC results, similarly to the accuracy table, show BT as having the best results in almost all cases. Unexpectedly, RS manages to reach the highest AUC among all query strategies in the case of MR with BERT.

Again, the CS variations offer some minor improvements in many cases. Here, the similarity between the curves of CS, WCS, RCS and CB-CS becomes more apparent, as the values of these strategies are almost equal in most cases.

Nonetheless, CS-TSNE shows higher AUC values than CS across the board. The largest improvement was achieved with BERT, namely on TREC (8.2%), with the second largest being on MR (5.7%).

Chapter 5

Discussion

Although none of the proposed Core-Set variations manage to get scores consistently as high as the Breaking-Ties approach, we can still gather interesting insights from the conducted experiment.

One presumption prior to the conducting of the experiment was that Core-Set may have mixed results in cases of text classification. Not only that, but according to Prabhu et al. [2021], Core-Set's results with BERT sometimes lie even below the performance of Random Sampling. This experiment's results further support this statement, with Core-Set being outperformed by Random Sampling with BERT, both in accuracy and AUC.

In order to discuss the research questions posed in Chapter 1, I have structured the rest of this chapter according to the results of each approach presented in Chapter 3. Pertaining to the third research question, I have also conducted further analysis of the experiment results in order to determine whether the approach did indeed result in a more balanced distribution of classes in the labelled pool.

5.1 Dimensionality Reduction-Based Approach

The experiment has also offered further insights with regard to the research questions. The first modification to Core-Set, which used t-SNE to reduce word embeddings prior to the selection with k-Center greedy, generally managed to improve Core-Set over the three given datasets. Though the differences in the case of SetFit were very minor, we can observe more noticeable improvements in efficiency with BERT, where Core-Set seemed to have its weaknesses originally.

5.2 Uncertainty-Based Approach

In the case of the two attempts to combine an uncertainty-based approach with Core-Set, the results have shown minor, but generally insignificant improvements. Though WCS and RCS had generally similar results overall, the results are still slightly in favour of WCS due to the improvements on TREC with SetFit and AGN with BERT, in which RCS fell short. Another aspect that favours WCS over RCS is the ease of implementation and interpretability, which is very straightforward in the case of the weighted approach. In addition, WCS does not perform what are essentially two instance selection runs per query, which is the case in the re-ranked approach.

5.3 Class Balance-Based Approach

Finally, with regard to the third research question, it becomes clear that the proposed class balance-based approach does not manage to improve upon the Core-Set selection. The accuracy and AUC values are, in most cases, either on par with or slightly below those of Core-Set.

In order to investigate the functionality of the class-balanced approach, I tracked the class distributions after each query of the previously conducted experiment. As a result, I was able to observe an overall positive impact of the balancing on the class distribution within the labelled pool. This impact was measured using the normalized entropy within the labelled pool after each iteration, as explained in Chapter 3. Similar to the learning curves in Chapter 4, I have plotted the mean over five runs in Figure 5.1.

We can see that for both strategies, the normalized entropy generally increases or stays relatively constant with each query. In the case of MR, we observe only minor changes in the class distribution throughout the learning process. This may be in part attributed to the fact that this dataset only has two classes that are distributed evenly. Nonetheless, we observe the class-balanced approach to have a more stable mean normalized entropy in the first half of the learning process, whereas the regular Core-Set selection has slight fluctuations. Nevertheless, both curves run somewhere within the range of 0.95 and 1, indicating a very close to even distribution in both cases.

For AGN, the difference between Core-Set and the class-balanced approach seem to be even less evident. In both cases, we observe a saturation curve beginning around 0.9 and evening out close to 1 around the fifth query. Although AGN has twice as many classes as MR, it is also a class balanced dataset with equal amounts of instances from each class, which may explain the depicted results.

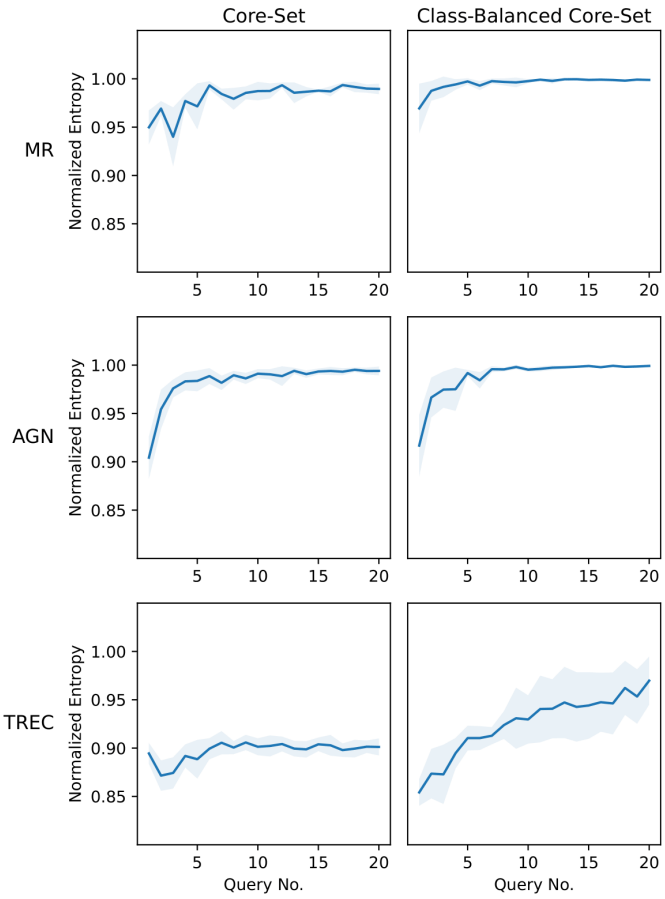

Figure 5.1: Normalized entropy curves for instances selected during AL with BERT using Core-Set and Class-Balanced Core-Set. The lines represent the mean accuracy and the surrounding tubes represent the standard deviation over five runs.

The dataset in which a change in class distribution within the labelled pool is most distinct is TREC. Here, we note a relatively constant curve around or below a normalized entropy of 0.9 in the case of Core-Set, whereas the class-balanced approach manages to select an increasingly even distribution. In this case, the normalized entropy gradually rises from roughly 0.85 to around 0.97. Still, the standard deviation increases in later iterations, becoming clearly higher than in all other cases.

One reason for these differences in the efficacy of the class-balanced approach are the underlying class distributions of each datast. In the case of MR and AGN, we have uniform distributions across all classes, whereas TREC is an inherently class-imbalanced dataset. Consequently, the class-balanced approach has very little impact on the distributions of the selected instances, which are nearly uniformally distributed in the first place. TREC, in contrast, has slightly imbalanced classes, resulting in the possibility for improvement using the class-balanced approach.

5.4 Limitations

Naturally, there are some limitations to consider with this experiment. On one hand, the training data used only encompassed three different datasets. These datasets do attempt to cover a variety of purposes, class numbers and sizes, but in order to further verify the effectiveness of the proposed approaches, more datasets should be used.

Another aspect that was not considered by my experiment is the effect of each approach on the respective runtime. This may be especially relevant when considering a dimensionality reduction algorithm such as t-SNE, which could have a significant impact when training on larger datasets. Furthermore, this experiment's result evaluation only took into account the respective accuracy and AUC metrics. For the purpose of evaluating the results of each strategy, it may be worth considering other commonly used metrics such as the F1-score.

Finally, it is worth underlining the fact that this thesis examined Core-Set as the basic k-Center greedy approach, whereas Sener and Savarese [2018] propose a slightly optimized, robust k-Center approach that minimizes outliers. This approach was not used for implementation reasons, as the solution for robust k-Center's feasability check depends on Gurobi (Gurobi Optimization, LLC [2023]), a licensed optimization framework. As a result, the results of Core-Set and the corresponding modifications used in this experiment may not be optimal.

5.5 Future Research

As mentioned earlier, this experiment was conducted on three datasets, two of which have uniform class distributions. With regard to the class-balanced approach, it may be worth examining the effect on various datasets with more imbalanced class distributions. Not only may this result in more significant improvements in normalized entropy, but possibly some more noticeable changes for the learning process itself.

Furthermore, it could be worth investigating the affect of combining several of the approaches presented here. If two strategies such as CS-TSNE and CB-CS only offer minor improvements in some cases, maybe a combination of the two approaches could contribute to more efficient learning.

Of course, some of the aspects mentioned in the previous section could be added to this experiment for further research. Examining other metrics such as the F1-score and runtimes, as well as considering other classifiers and datasets may offer additional insights in determining the usability of the presented approaches.

Chapter 6

Conclusion

In this thesis, I examined the effect of Core-Set as an AL query strategy for text classification tasks and compared it with two baseline approaches using two models across three different datasets. I then created four different variations of Core-Set, each with some modification around the original k-Center greedy algorithm and examined their performance within the same experiment setup. The motivation for doing this was to see if Core-Set does indeed achieve mixed results within the field of text classification and to attempt to work toward a possible improvement in this regard.

The experiment's results show Core-Set producing mixed results in many cases when compared to the baseline strategies. This aligns with what can be observed in Prabhu et al. [2021]. We can observe some slight improvements of Core-Set's performances when used in conjunction with t-SNE. Moreover, the uncertainty-based approach and class balance-based approach do not seem to significantly impact Core-Set's results.

Appendix A

Implementation Details

The experiment setup is based on a pre-existing AL environment created with Schröder et al. [2023]. This also includes the implementation of k-Center greedy as well as various methods concerning class redistributions. In addition, well-known machine learning libraries such as Scikit-Learn, HuggingFace, NumPy, and SciPy were used for the implementation of the query strategies. These libraries were used within a Python 3.8 environment.

The models that were fine-tuned within the experiment are bert-base-uncased and paraphrase-mpnet-base-v2. The datasets were also obtained via huggingface datasets.

Bibliography

Dana Angluin. Queries and concept learning. *Mach. Learn.*, 2(4):319–342, 1987. doi: 10.1007/BF00116828. URL https://doi.org/10.1007/BF00116828.

Razvan Caramalau, Binod Bhattarai, and Tae-Kyun Kim. Sequential graph convolutional network for active learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 9583–9592. Computer Vision Foundation / IEEE, 2021. doi: 10.1109/CVPR46437.2021.00946. URL https://openaccess.thecvf.com/content/CVPR2021/html/Caramalau_Sequential_Graph_Convolutional_Network_for_Active_Learning_CVPR_2021_paper.html.

Yahui Chen. Convolutional neural network for sentence classification. Master's thesis, University of Waterloo, 2015.

Cody Coleman, Edward Chou, Julian Katz-Samuels, Sean Culatana, Peter Bailis, Alexander C. Berg, Robert D. Nowak, Roshan Sumbaly, Matei Zaharia, and I. Zeki Yalniz. Similarity search for efficient active learning and search of rare concepts. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 6402–6410. AAAI Press, 2022. doi: 10.1609/AAAI.V36I6.20591. URL https://doi.org/10.1609/aaai.v36i6.20591.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics,

2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. Active learning for BERT: an empirical study. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7949–7962. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.638. URL https://doi.org/10.18653/v1/2020.emnlp-main.638.

Damien François. High-dimensional data analysis. 2007.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL https://www.gurobi.com.

Sariel Har-Peled. Geometric approximation algorithms. *Lecture Notes for CS598, UIUC*, 2008.

Dorit S. Hochbaum. When are np-hard location problems easy? *Ann. Oper. Res.*, 1(3):201–214, 1984. doi: 10.1007/BF01874389. URL https://doi.org/10.1007/BF01874389.

Wen-Lian Hsu and George L. Nemhauser. Easy and hard bottleneck location problems. *Discret. Appl. Math.*, 1(3):209–215, 1979. doi: 10.1016/0166-218X(79)90044-1. URL https://doi.org/10.1016/0166-218X(79)90044-1.

M Ikonomakis, Sotiris Kotsiantis, Vasilis Tampakas, et al. Text classification using machine learning techniques. *WSEAS transactions on computers*, 4 (8):966–974, 2005.

Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura E. Barnes, and Donald E. Brown. Text classification algorithms: A survey. *Inf.*, 10(4):150, 2019. doi: 10.3390/info10040150. URL https://doi.org/10.3390/info10040150.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2267–2273. AAAI Press, 2015. doi: 10.1609/AAAI.V29I1.9513. URL https://doi.org/10.1609/aaai.v29i1.9513.

K. Lang and E. Baum. Query learning can work poorly when a human oracle is used. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 335–340, 1992.

Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nat.*, 521(7553):436–444, 2015. doi: 10.1038/NATURE14539. URL https://doi.org/10.1038/nature14539.

David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and C. J. van Rijsbergen, editors, *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 3-6 July 1994 (Special Issue of the SIGIR Forum)*, pages 3–12. ACM/Springer, 1994. doi: 10.1007/978-1-4471-2099-5_1. URL https://doi.org/10.1007/978-1-4471-2099-5_1.

Xin Li and Dan Roth. Learning question classifiers: the role of semantic information. *Nat. Lang. Eng.*, 12(3):229–249, 2006. doi: 10.1017/S1351324905003955. URL https://doi.org/10.1017/S1351324905003955.

Bing Liu and Lei Zhang. A survey of opinion mining and sentiment analysis. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 415–463. Springer, 2012. doi: 10.1007/978-1-4614-3223-4_13. URL https://doi.org/10.1007/978-1-4614-3223-4_13.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879. IJCAI/AAAI Press, 2016. URL http://www.ijcai.org/Abstract/16/408.

Qiang Liu, Yanqiao Zhu, Zhaocheng Liu, Yufeng Zhang, and Shu Wu. Deep active learning for text classification with diverse interpretations. In Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong, editors, *CIKM '21: The 30th ACM International Conference on*

Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021, pages 3263–3267. ACM, 2021. doi: 10.1145/3459637.3482080. URL https://doi.org/10.1145/3459637.3482080.

Tong Luo, Kurt Kramer, Dmitry B. Goldgof, Lawrence O. Hall, Scott Samson, Andrew Remsen, and Thomas Hopkins. Active learning to recognize multiple types of plankton. *J. Mach. Learn. Res.*, 6:589–613, 2005. URL http://jmlr.org/papers/v6/luo05a.html.

Leland McInnes and John Healy. UMAP: uniform manifold approximation and projection for dimension reduction. *CoRR*, abs/1802.03426, 2018. URL http://arxiv.org/abs/1802.03426.

Marcin Mironczuk and Jaroslaw Protasiewicz. A recent overview of the state-of-the-art elements of text classification. *Expert Syst. Appl.*, 106:36–54, 2018. doi: 10.1016/j.eswa.2018.03.058. URL https://doi.org/10.1016/j.eswa.2018.03.058.

Dave Mount. Greedy approximation algorithms: The k-center problem, September 2017.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Kevin Knight, Hwee Tou Ng, and Kemal Oflazer, editors, *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 115–124. The Association for Computer Linguistics, 2005. doi: 10.3115/1219840.1219855. URL https://aclanthology.org/P05-1015/.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018. doi: 10.18653/V1/N18-1202. URL https://doi.org/10.18653/v1/n18-1202.

Sumanth Prabhu, Moosa Mohamed, and Hemant Misra. Multi-class text classification using bert-based active learning. In Eduard C. Dragut, Yunyao Li, Lucian Popa, and Slobodan Vucetic, editors, *3rd Workshop*

on Data Science with Human in the Loop, DaSH@KDD, Virtual Conference, August 15, 2021, 2021. URL https://drive.google.com/file/d/1xVy4p29UPINmWl8Y7OospyQgHiYfH4wc/view.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1410. URL https://doi.org/10.18653/v1/D19-1410.

Akanksha Saran, Safoora Yousefi, Akshay Krishnamurthy, John Langford, and Jordan T. Ash. Streaming active learning with deep neural networks. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 30005–30021. PMLR, 2023. URL https://proceedings.mlr.press/v202/saran23a.html.

Christopher Schröder, Lydia Müller, Andreas Niekler, and Martin Potthast. Small-text: Active learning for text classification in python. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 84–95, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.eacl-demo.11.

Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002. doi: 10.1145/505282.505283. URL https://doi.org/10.1145/505282.505283.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Claude E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27(3):379–423, 1948. doi: 10.1002/J.1538-7305.1948.TB01338.X. URL https://doi.org/10.1002/j.1538-7305.1948.tb01338.x.

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 5971–5980. IEEE, 2019. doi: 10.1109/ICCV.2019.00607. URL https://doi.org/10.1109/ICCV.2019.00607.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune BERT for text classification? In Maosong Sun, Xuanjing Huang, Heng Ji, Zhiyuan Liu, and Yang Liu, editors, *Chinese Computational Linguistics - 18th China National Conference, CCL 2019, Kunming, China, October 18-20, 2019, Proceedings*, volume 11856 of *Lecture Notes in Computer Science*, pages 194–206. Springer, 2019. doi: 10.1007/978-3-030-32381-3_16. URL https://doi.org/10.1007/978-3-030-32381-3_16.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. Efficient few-shot learning without prompts. *CoRR*, abs/2209.11055, 2022. doi: 10.48550/arXiv.2209.11055. URL https://doi.org/10.48550/arXiv.2209.11055.

Laurens van der Maaten. Barnes-hut-sne. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Conference Track Proceedings*, 2013. URL http://arxiv.org/abs/1301.3342.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Laurens Van Der Maaten, Eric O Postma, H Jaap van den Herik, et al. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10(66-71):13, 2009.

Liantao Wang, Xuelei Hu, Bo Yuan, and Jianfeng Lu. Active learning via query synthesis and nearest neighbour search. *Neurocomputing*, 147:426–434, 2015. doi: 10.1016/J.NEUCOM.2014.06.042. URL https://doi.org/10.1016/j.neucom.2014.06.042.

Allen R Wilcox. Indices of qualitative variation. Technical report, Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), 1967.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657, 2015. URL `https://proceedings.neurips.cc/paper/2015/hash/250cf8b51c773f3f8dc8b4be867a9a02-Abstract.html`.