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: Christopher Schröder
2. Referee: Christian Kahmann

Submission date: April 21, 2024

Declaration

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

Leipzig, April 21, 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 are 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, we have been able to develop many methods of extracting information from text, and as a result, perform 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. This method can offer significant advantages in improving model performances and especially in reducing labeling costs. Though there is no universally good 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 available data 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. 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, Core-Set 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 DNNs (Liu et al. [2021]). Particularly in the first case, the experiments show that Core-Set performs poorly even when compared to the random sampling strategy. In addition, the approach has even been shown to be less effective in computer vision tasks in cases with higher numbers of classes as well as higher-dimensional data points (Sinha et al. [2019]). The theoretical analysis shown in Sener and Savarese [2018] briefly mentions this within the context of higher class amounts, however it does not attempt to provide a potential solution to the problem.

This thesis aims to explore the possibility of improving the Core-Set approach for text classification tasks. By first explaining Core-Set's functionality and the theoretical reasons for why it tends to underperform in certain classification tasks, I aim to then demonstrate the performance difference 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 attempt to improve the Core-Set approach within the context of text classification tasks 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), and the Core-Set approach to AL more specifically (Section 2.3). In Chapter 3, I will explain my approach to improving the performance of Core-Set for text classification using dimensionality reduction. In Chapters 4 and 5, I will present my experiment as well as discuss its results. Finally, Chapter 6 will conclude the thesis and provide insights on potential future developments of the method.

Chapter 2

Background/Related Work

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. There are several open data sets that are publicly available to this end. 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, Naive 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. 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 will also be focusing 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.

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 models 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, on the other hand, 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 instaces, 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 in Sener and Savarese [2018] was originally designed for convolutional neural networks (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 active learning 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 minimum distance of any point in $N \setminus \mathbf{s}$ to its closest center in $\mathbf{s}$ is maximal. 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 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.

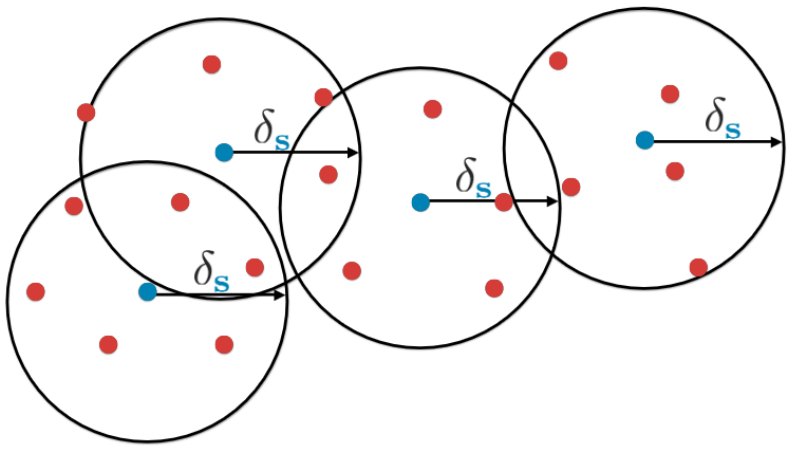

Figure 2.1: Visualization of the k-Center approach from Sener and Savarese [2018]. $\mathbf{s}$ denotes the set of k selected points, $\delta_\mathbf{s}$ denotes the maximum distance of any point in N to its nearest center in $\mathbf{s}$.

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 for k-Center greedy adapted to the active learning context can be seen in CS.

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 $\mathbf{s} \setminus \mathbf{s}^0$
```

The original paper by Sener and Savarese [2018] attempts to further improve the robustness of this approximation by placing an upper limit on the number of outliers that can be selected. This thesis will be focusing on and using the regular k-Center greedy algorithm, which is generally more established due to ease of implementation and interpretability.

2.4 Dimensionality Reduction

As mentioned earlier, one of the major challenges of the Core-Set approach (among others) 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) and t-distributed Stochastic Neighbor Embedding (t-SNE). The latter 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 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.

This method is especially useful for the visualization of high-dimensional data, in which the aim is to display and view the underlying structure in a given dataset. t-SNE plots are strongly influenced by the chosen hyperparameters however, and thus a good understanding for the influence of these parameters is important. Particularly *perplexity*, a measure of the effective number of neighbours, has a complex effect on the resulting reductions. According to Van der Maaten and Hinton [2008], typical values for perplexity are between 5 and 50 and larger/denser datasets often generally require higher perplexities (Van der Maaten [2023]). Other commonly adjusted parameters include the learning rate and number of iterations for the optimization.

Chapter 3

Approach

As mentioned, distance-based methods such as Core-Set can be ineffective in higher dimensions. To overcome this challenge, I propose the application of t-SNE as a dimensionality reduction technique on the training data before employing the Greedy Core-Set strategy. In esssence, this would modify the original k-Center greedy approach in that the distances computed between the points in **s** and the unselected points would be different, since the algorithm would take in reduced embeddings (this modification is noted in the pseudocode for CS-TSNE).

Algorithm 2 t-SNE with k-Center-Greedy

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

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 Greedy 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 a 80-20 weighting in favour of the Core-Set distances).

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 $\mathbf{s} \setminus \mathbf{s}^0$
```

The second, which I will call "Ranked Weighted Greedy 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.

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 $r$
```

Where $\mathbf{p}^*_{j,k}$ denotes the probability of the k-th most likely class label for the j-th instance

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 idea with this approach is that the class distribution within our pool of labelled instances should, in theory, reflect the underlying class distribution of the entire dataset. 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 the “real” distribution of that class.

To this end, I will divide a single query into multiple Core-Set computations, essentially constructing disjunct 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. Then, k-Center-greedy is used to select the desired amount of points of each class, such that if k_i denotes the amount of newly selected instances for the ith class, then $\sum_{i \in C} k_i = k$.

In the corresponding pseudocode (CB-CS), I did not include an additional code block for this calculation, which has instead been denoted by the use of `target_dist`. For the implementation, I used existing methods from the Python library Small-Text (Schröder et al. [2023]).

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 $C$
    Initialize $\mathbf{s} = \mathbf{s}^0$
    $d = \text{target\_dist}_{i \in [n] \setminus \mathbf{s}}(y_i, y_i^{pred}, C)$
    for $c \in C$ do
        <do stuff>
    end for
    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 $\mathbf{s} \setminus \mathbf{s}^0$
```

The experiment will be conducted with two models on three well-known classification datasets and will measure the performances throughout 5 runs per combination.

Chapter 4

Experiment

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 4.1. Example instances from each dataset as well as their corresponding classes can also be seen in 4.2.

4.2 Experiment Setup

I have chosen 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

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.

manner.

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, 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

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

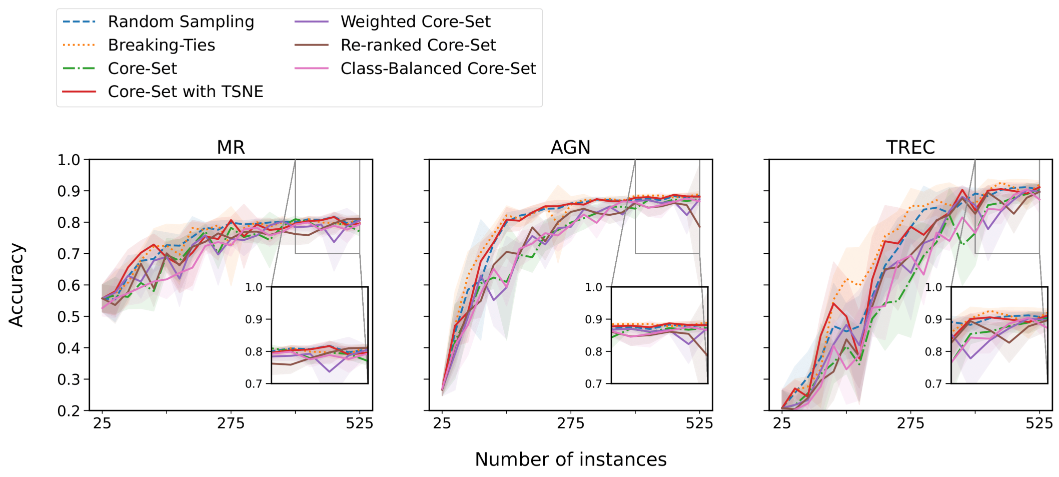

Figure 4.1: Active Learning curves of BERT on each dataset when combined with seven query strategies: Random Sampling, Breaking-Ties, Greedy Core-Set, Greedy Core-Set with t-SNE, Weighted Greedy Core-Set, Re-ranked Greedy Core-Set and Class-Balanced Core-Set. The lines represent the mean accuracy and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

The SetFit learning curves generally outperform the BERT model and seem to show much less variation between the different query strategies. Even in the case of the AL curves with BERT, accuracy differences seem to mainly occur in early iterations. By the final iteration, the mean accuracy scores of the different strategies tend to lie in a similar range.

Dataset	Model	Query Strategy						
		RS	BT	CS	CS-TSNE	WCS	RCS	CS-CB
AGN	BERT	0.884 ± 0.004	**0.889 ± 0.010**	0.874 ± 0.012	0.881 ± 0.010	0.873 ± 0.011	0.785 ± 0.221	0.866 ± 0.016
	SetFit	0.886 ± 0.006	**0.902 ± 0.004**	0.895 ± 0.003	0.895 ± 0.003	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 ± 0.02	0.806 ± 0.014	0.811 ± 0.013	0.793 ± 0.021
	SetFit	0.869 ± 0.006	**0.880 ± 0.005**	0.871 ± 0.005	0.874 ± 0.005	0.874 ± 0.007	0.870 ± 0.004	0.870 ± 0.006
TREC	BERT	0.904 ± 0.018	**0.920 ± 0.014**	0.902 ± 0.021	0.912 ± 0.009	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	**0.966** ± 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	WCS	RCS	CS-CB
AGN	BERT	0.790 ± 0.015	**0.800 ± 0.009**	0.735 ± 0.020	0.792 ± 0.007	0.731 ± 0.014	0.741 ± 0.015	0.733 ± 0.017
	SetFit	0.865 ± 0.007	**0.881 ± 0.004**	0.871 ± 0.005	0.875 ± 0.004	0.870 ± 0.003	0.870 ± 0.002	0.873 ± 0.003
MR	BERT	**0.750 ± 0.007**	0.746 ± 0.011	0.718 ± 0.009	0.741 ± 0.017	0.720 ± 0.004	0.718 ± 0.005	0.706 ± 0.015
	SetFit	0.854 ± 0.002	**0.864 ± 0.005**	0.856 ± 0.003	0.862 ± 0.003	0.858 ± 0.007	0.857 ± 0.003	0.859 ± 0.002
TREC	BERT	0.674 ± 0.029	**0.709 ± 0.008**	0.594 ± 0.022	0.676 ± 0.037	0.629 ± 0.024	0.624 ± 0.012	0.598 ± 0.025
	SetFit	0.914 ± 0.011	**0.923 ± 0.007**	0.896 ± 0.015	0.919 ± 0.005	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.

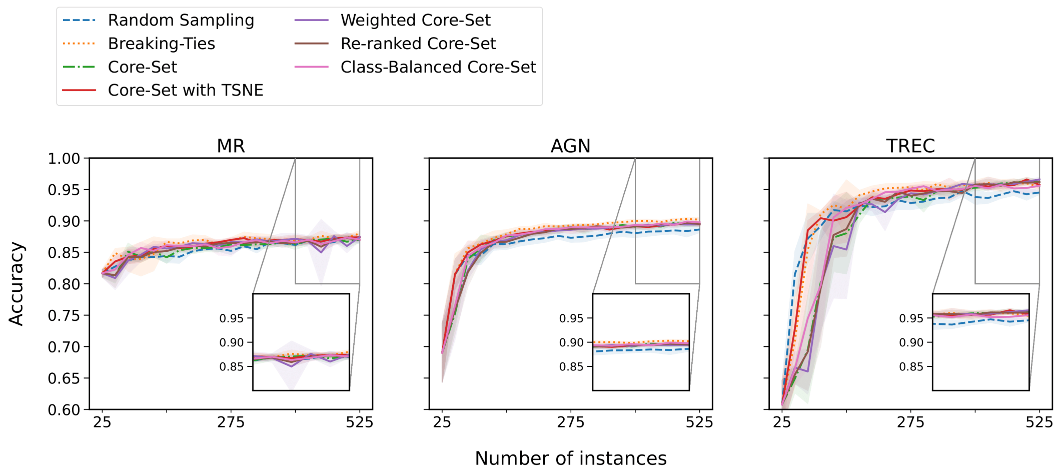

Figure 4.2: Active Learning curves of SetFit on each dataset when combined with seven query strategies: Random Sampling, Breaking-Ties, Greedy Core-Set, Greedy Core-Set with t-SNE, Weighted Greedy Core-Set, Re-ranked Greedy Core-Set and Class-Balanced Core-Set. The lines represent the mean accuracy and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

I will first compare and contrast the different results for the training of the BERT model (4.1).

In the case of AGN and TREC, we can clearly see the curve of Breaking-Ties achieve a higher performance throughout. However, CS-TSNE is able to compete very closely on AGN. Interestingly enough, Random Sampling has managed to achieve similar scores in these cases to the strategies mentioned previously.

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

We can see 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 Breaking-Ties looks to be a strong contender in all cases and manages to perform similarly to or better than Core-Set and its variants.

Class-Balanced Core-Set, on the other hand, does not seem to show significant accuracy improvements during training in comparison to the regular Core-Set approach. The final results of the respective curves do seem to slightly favour CB-CS over CS. Generally, the two strategies' learning curves show a very similar trend across all three datasets.

Weighted Core-Set and Re-ranked Core-Set are in a similar vein to, albeit generally 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.

The SetFit learning curves (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. As for AGN and TREC, the margins between the curves are slightly wider, and some differences can be noted when comparing approaches. Again, Breaking-Ties 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 in any significant way. Additionally, Random Sampling's accuracy values stay visibly below those of the other query strategies in the second half of each run when considering AGN and TREC.

In 4.3, I show the reported mean final accuracy across all runs, as well as the standard deviation for each combination of dataset, model and query strategy. Overall, Breaking-Ties dominates for almost every dataset on both models. Only for SetFit does another strategy achieve the highest overall final mean accuracy. Furthermore, we can see that RS outperforms CS when used with BERT throughout all datasets.

However, the results still show small improvements in accuracy for many of the Core-Set variation's results. CS-TSNE most notably shows slightly higher scores than CS in almost all cases, the one exception being TREC with SetFit.

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.

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.

As also made evident by the learning curves earlier, the differences in AUC are most drastic in the case of BERT on the TREC dataset.

Chapter 5

Discussion

...

Chapter 6

Conclusion

...

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`.

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.

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`.

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`.

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`.

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.

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`.

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. Laurens Van der Maaten home page. `https://lvdmaaten.github.io/tsne/`, 2023. Accessed: 2023-09-24.

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`.

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`.