Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Computer Science and Media

Few Shot Learning for Text Classification

Master's Thesis

Shaour Haider
Born Sept 26, 1993 in Dera Ghazi Khan

Matriculation Number 119482

1. Referee: Prof. Dr. Benno Stein
2. Referee: Prof. Dr. Volker Rodehorst

Submission date: September 10, 2020

Declaration

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

Weimar, September 10, 2020

. . . .
Shaour Haider

Abstract

Training a classifier in data-scarce scenarios often results in having poor accuracy during testing. One of the main reasons for this is an over-fitting that is caused by the limited amount of data. Few-shot learning approaches are becoming increasingly popular in such scenarios. Their target is to accommodate the shortcoming of small training sets through pre-training. The general idea is to (1) train a model on a problem that is similar in the domain to the target problem using a large dataset as a training resource (2) then transfer to the target problem. This thesis aims at tackling the few-shot problem in the natural language domain using a few-shot learning approach called transfer learning. In transfer learning, the knowledge gained from solving one problem is transferred to another given there is a similarity between them. To improve text classification in a few-shot scenario, we investigate up to what extent the classification of Wikipedia paragraphs into section headings, for which ample data can be acquired is an effective pre-training task. To test this claim, we came up with the two experimental methods namely baseline and transfer learning. In the baseline method, we set up a binary classifier having no prior pre-training. The experiments performed with both methods had different settings such as having a varying amount of training data k, as well as different paragraph representations such as bag of words, fasttext, and bert. The results from the baseline method show a clear distinction between the accuracy scores. The pre-trained representations from the bert model demonstrate the best performance in this method followed by the fasttext model. While the bow representations demonstrate the worst accuracy scores. In the transfer learning, the baseline method is preceded with a pre-training step. The model is trained on the base dataset that consists of a huge resource of Wikipedia section headings and their paragraphs. The learned parameters from this pre-training are transferred to the target task. The results from the transfer learning method don't show any improvements while using pre-trained models such as fasttext and bert. As these models are just fine-tuned on the underlying task using the base dataset it is our observation that they already contained the knowledge that was transferred. The transfer learning method demonstrates an increase of 10-20% in the performance when using the bow representations as compared to the baseline method. These results indicate that transfer learning is benefiting the few-shot task and it also solidifies our claim that classifying Wikipedia section headings is an effective pre-training task.

Contents

Acknowledgements

My sincere thanks go to my thesis advisor Tim Gollub for giving me the opportunity to work on such an interesting topic. This work would not be possible without his dedicated support and valuable feedback.

I would also like to thank Prof. Dr. Benno Stein for accepting this work under his supervision.

Last but not least, I would like to dedicate my thesis to my family. They have always been a source of motivation and encouragement.

Chapter 1

Introduction

Classification is a supervised learning task in which the algorithm learns to predict the class of new objects based on past observations. If the objects are lexical units, the task is a text classification task. It is one of the fundamental tasks in natural language processing (NLP). NLP methods transform the human language into vectors for machines to perform calculations, with these word embeddings, researchers can perform several tasks such as sentiment analysis, spam detection and topic classification.

Labeled data is crucial for supervised learning but obtaining it is an expensive and time-consuming activity. The scarcity of data often leads to poor performance in the classification task. If only a small number of labeled examples (<10) are given for each class, we refer to this classification task as a few-shot task. More specifically, if a model is given k examples from n different classes as the train set for training a classifier, this is referred to as an n-way k-shot task. The ability of an algorithm to perform the few-shot learning is assessed by its performance on a test set which contains previously unseen instances. The algorithm must determine which of the train set classes does the test set instance belong to as demonstrated in Figure 1.1.

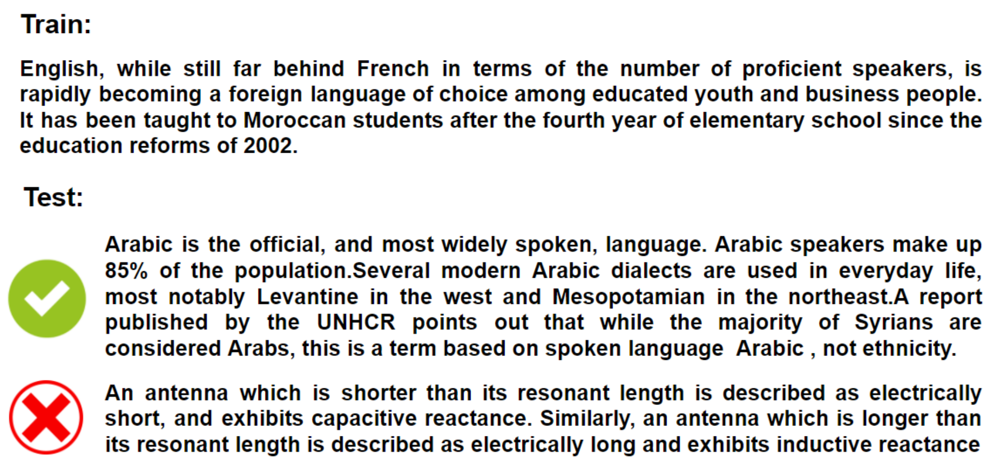
A basic approach to performing text classification is to train a classifier over the train set instances based on their bag-of-words representation. Some of the problems that arise when following this approach are overfitting and vocabulary mismatch. The reason for overfitting is that small training data is not representative of the whole topic. For example, we have a topic of "championship". Many sporting events have the same title for their competitions. Imagine in our training set we have most of the paragraphs from football. A model will become too attuned to the data related to football, therefore, losing its applicability to any other sports i.e. baseball. The bag of words representations is high dimensional feature vector due to the large size of the vocabulary. While the high variance of the model enables it to represent training data more accurately but, in the process, leads to learning the noise. The noise learned in the training dataset makes the predictions less accurate. As a result, though the performance of the classifier on the training data might be good, performance on the unseen data is notably worse. Following this example, if we have two paragraphs from football and baseball, although there is a semantic similarity between the two paragraphs but in terms of the bag of words, their similarity will be low due to vocabulary mismatch.

One way to solve these issues is to use more sophisticated representations from advanced pre-trained models such as fasttext or bert. As pointed out in Chapter 2, these approaches can be seen as transfer learning approaches in the few-shot paradigm. The models are trained on a large corpus and generate dense feature representations. These representations are then used to train the classifier so that it can then be used to predict the previously unseen instances. The workings of such an approach can be seen in Figure 1.2. The features are extracted from the train set instances. These features along with their true classes are passed on to a machine learning algorithm which then recognizes the patterns in the training data and passes this knowledge to the classifier. In the prediction, the features are extracted from the test set that contains previously unseen instances which are then used to predict the class. These pre-trained representations not only account for word occurrences but also the semantic relations between them. Along with that, the low dimensional feature vectors mitigate the overfitting making them less susceptible to the problems faced by the bag of words model.

Another way to tackle these problems and to perform well in these few-shot tasks, there are several approaches like transfer learning, metric learning, meta-learning, etc. that can train a classifier to recognize unseen classes with limited labeled examples. In Chapter 2, we discuss the workings of these approaches in detail. In this thesis, we choose to experiment with the transfer learning. It enables the transfer of knowledge from an additional dataset to the target task. We distinguish the target dataset on which the classifier is applied from an additional dataset that is used to train a paragraph representation. We refer to this additional dataset as the base dataset. A model is pre-trained on the base dataset. The learned parameters from this model are transferred to the target task. The target dataset is divided into the train set and the test set. An episode is generated for each class in the target dataset consisting of the train and test set instances. These episodes are then used to train and test the classifier.

Transfer learning has mostly been applied to image-related tasks. For example, images from a huge dataset such as MNIST or ImageNet are used to pre-train a model. The knowledge from these models is then transferred to the target task. In this thesis, we use transfer learning for text classification using the Wikipedia article corpus as a resource for pre-training a model. In Chapter 3, we reveal the experimental setup, construction of datasets, and the implementation details to further elaborate this idea. We hypothesize that using Wikipedia article headings as a resource of pretraining and transferring the knowledge gained from it is an effective way to overcome the issue of overfitting and increasing the performance in few-shot tasks.

We conduct experiments on the baseline and transfer learning methods. In the baseline method, we perform binary classification without any pre-training. As the name suggests, this method provides us with a baseline score for our experiments which is crucial to evaluate the change in performance. After calculating the accuracy scores across the different experimental settings, we move on to the transfer learning method. The idea behind this method is to transfer the knowledge gained from one task to another with a condition that there is a level of similarity between the tasks. A model is trained on thousands of varying headings and paragraphs. The knowledge gained from this trained model is transferred to our classifier for the evaluation of test paragraphs. The results from these experiments as discussed in Chapter 4 demonstrate a significant increase in the performance of a classifier while transferring the parameters learned using bow representations for pre-training a model. The results also indicate no improvements when transferring the learned parameters after fine-tuning the already pre-trained models such as bert and fasttext on the base dataset. The increase in the accuracy scores for bow representations in general also proves our hypothesis that the few-shot task is benefiting from the Wikipedia section heading classification. While our observations on the result show an improvement with the use of transfer learning methods, it leaves many intriguing questions unanswered about the possible evaluation of other few-shot learning approaches such as metric learning and meta-learning. Finally, in Chapter 5, we discussed the possibilities for future work regarding the improvements to the current approach along with the implementation of other few-shot learning approaches.



Train:

English, while still far behind French in terms of the number of proficient speakers, is rapidly becoming a foreign language of choice among educated youth and business people. It has been taught to Moroccan students after the fourth year of elementary school since the education reforms of 2002.

Test:

✓ **Arabic is the official, and most widely spoken, language. Arabic speakers make up 85% of the population.Several modern Arabic dialects are used in everyday life, most notably Levantine in the west and Mesopotamian in the northeast.A report published by the UNHCR points out that while the majority of Syrians are considered Arabs, this is a term based on spoken language Arabic , not ethnicity.**

✗ **An antenna which is shorter than its resonant length is described as electrically short, and exhibits capacitive reactance. Similarly, an antenna which is longer than its resonant length is described as electrically long and exhibits inductive reactance**

Figure 1.1: Sample classification problem. A training paragraph belonging to the topic of language is presented as a resource. The classifier has to classify which one of these two test examples is related to the training example.

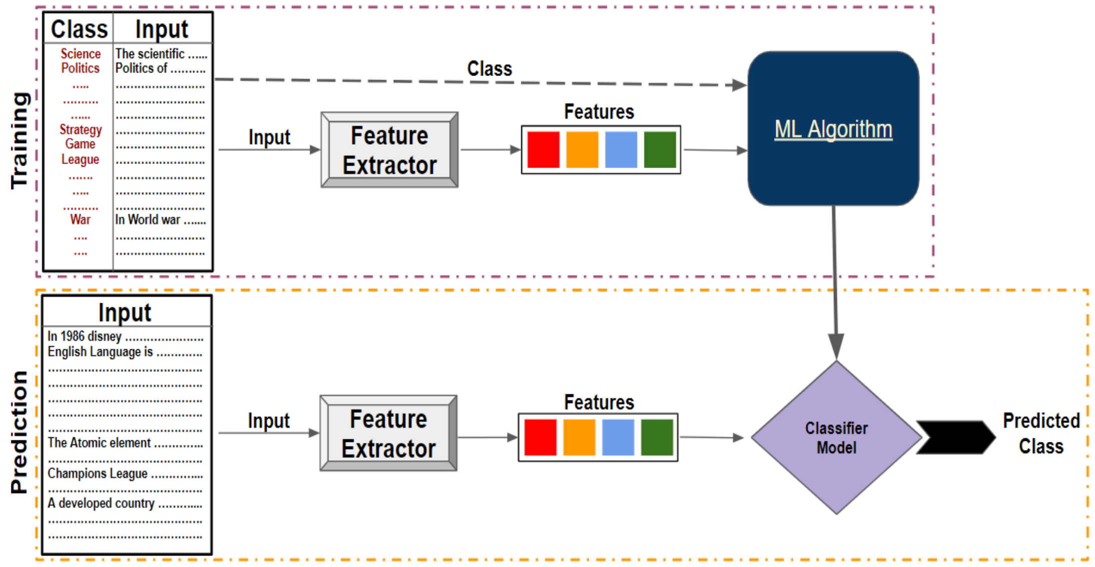


Figure 1.2: Overview of a standard classifier. The knowledge gained from the training phase is used to predict previously unseen input.

Chapter 2

Related Work

Few-shot learning algorithms aim at training a classifier that can classify previously unseen instances by using only a limited amount of labeled training examples from the target dataset. This can be done by pre-training a model by utilizing a huge amount of training examples from the base dataset. We will discuss these few-shot learning approaches by categorizing them into four categories: transfer learning, metric learning, meta-learning, and data augmentation.

2.1 Transfer Learning

Transfer learning deals with few-shot classification problem by first finding a task similar in domain to the target task and for which there is plenty of data. A model is trained on this task and the trained parameters are transferred to the few-shot task as seen in Figure 2.1.

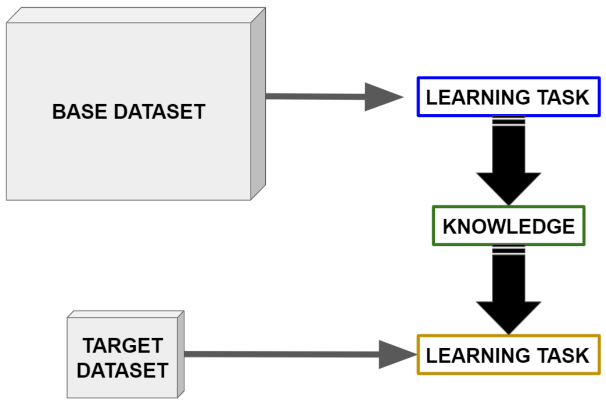


Figure 2.1: Simple illustration of transfer learning approach.

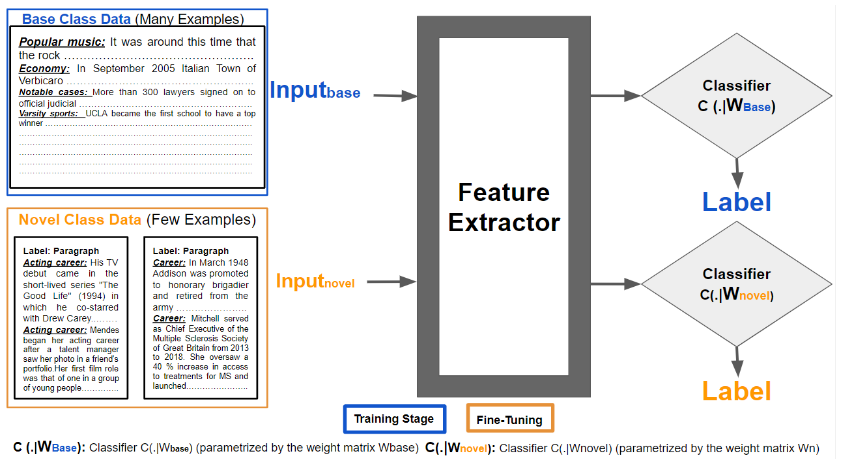

Base Class Data (Many Examples)
Popular music: It was around this time that the rock ……
Economy: In September 2005 Italian Town of Verbicaro ……
Notable cases: More than 300 lawyers signed on to official judicial ……
Varsity sports: UCLA became the first school to have a top winner ……
Novel Class Data (Few Examples)
Label: Paragraph
Acting career: His TV debut came in the short-lived series "The Good Life" (1994) in which he co-starred with Drew Carey……
Acting career: Mendes began her acting career after a talent manager saw her photo in a friend's portfolio.Her first film role was that of one in a group of young people……
Label: Paragraph
Career: In March 1948 Addison was promoted to honorary brigadier and retired from the army ……
Career: Mitchell served as Chief Executive of the Multiple Sclerosis Society of Great Britain from 2013 to 2018. She oversaw a 40 % increase in access to treatments for MS and launched……
Inputbase
Inputnovel
Feature Extractor
Classifier C (.|WBase)
Label
Classifier C(.|Wnovel)
Label
Training Stage
Fine-Tuning
C (.|WBase): Classifier C(.|Wbase) (parametrized by the weight matrix Wbase) C(.|Wnovel): Classifier C(.|Wnovel) (parametrized by the weight matrix Wn)

Figure 2.2: Illustration of the baseline method for the transfer learning introduced in Chen et al., 2019. Two Stage Procedure:(Blue) In the training stage a classifier and feature extractor are trained from scratch. (Yellow) In the fine-tuning stage, the feature extractor is fixed but the classifier is trained on new classes.

The few-shot data is used to either fine-tune this model or use the hidden layers as input for a new classifier trained with the few-shot data. The method followed in Gidaris and Komodakis [2018] learns a weight generator to forecast a target class classifier using an attention-based process and Qi et al. [2017] explicitly use target class features as their weights. Replacing the linear classifier with the distance-based classifier for the baseline approach in Chen et al. [2019] yields comparable performance with the existing state-of-the-art meta-learning algorithms. This baseline approach follows a two-stage procedure. In the training stage, a feature extractor and classifier are trained from scratch using training examples from the training class by minimizing cross-entropy classification loss. The classifier consists of a linear layer followed by a softmax function. The fine-tuning stage is used to adapt a model to recognize the target classes. The feature extractor from the training stage is fixed and a new classifier is trained on new classes as shown in Figure 2.2.

2.2 Metric Learning

Metric learning based methods tackle the issue of few-shot classification by "learning to compare". The algorithms seek to learn embeddings in which the data vector is largely unaffected by intra-class variations but retains class information. Early work focused on pairwise comparators aimed at assessing whether two examples of data are of the same or different classes, although the system may not have seen these classes before. The comparators take two instances in parallel and label them either as belonging to the same or different classes. In **siamese network** (Koch, 2015), a model is trained that gives out the probability that two examples belong to the same class. To create two embeddings, the two examples are passed through identical neural networks as shown in Figure 2.3. The component-wise absolute distance between the embedding is determined and transferred to a subsequent network of comparison, which reduces this distance vector to one number. This is passed through a sigmoidal output to classify as either belonging to the same or a different class.

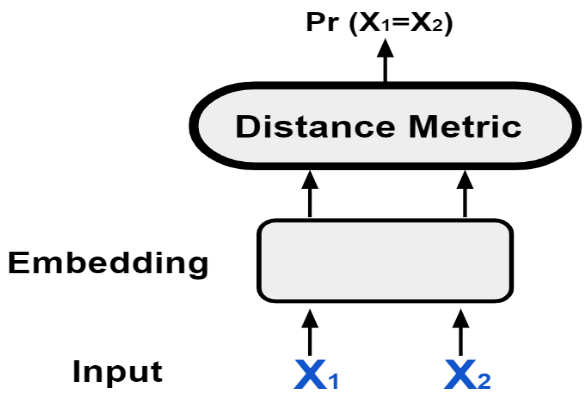


Figure 2.3: Siamese network takes two inputs and return the likelihood of these inputs being in the same class. This is passing each example through an identical network and then using as the basis of the decision the pairwise difference between the embeddings.

Each pair of examples during training are randomly drawn from a huge set of training classes. Therefore, the system learns to discriminate between classes in general, rather than any particular classes. Completely different classes are used for testing. While this doesn't have the formal N-way-K-shot task structure, the basic idea is similar enough. By assigning the class for an example in the query set based on its maximum similarity to one of the examples in the support set, comparators can be modified to the N-way-K-shot setting in pairs. Multi-class comparators, however, attempt to do the same thing but here the representations and the classifications are learned end to end. To compute the similarity Vinyals et al. [2016] proposed **matching networks** which pass each support example through a network to produce an embedding and pass the query example through a different network i.e f(*) and g(*)to produce a different embedding. The cosine similarity is then computed between these embeddings as Formula 2.1 and normalized using a softmax function to generate a positive similarity number that sums to one.

$$\mathbf{sim(q,p)} = \frac{\mathbf{f[q].g[p]}}{\mathbf{||f[q]||.||g[p]||}} \tag{2.1}$$

This system can be trained end-to-end for the N-way-K-shot learning task. The system is presented with a training task at every learning iteration; the predicted labels are calculated for the query set (calculation is based on the support set) and the loss function is the cross-entropy of ground truth and predicted labels. Matching networks measure similarities between the embedding of each support example and the query example. This has the drawback that the algorithm is not resilient to data imbalance meaning that if for a class there are more shots, it may favor those with more abundant shots. In **prototypical networks** (Snell et al., 2017) the support examples are all mapped to the embedding space to create an embedding. All the embeddings for the class are averaged to create a prototype. To classify query examples, we first compute its embedding and then base the decision on the relative distance to the prototypes. The average class embedding of examples makes it resilient to data imbalance. The similarity is calculated as a negative multiple of euclidean distance as seen in Formula 2.2.

$$\mathbf{sim(q,p)} = \mathbf{-d(q,p)} \tag{2.2}$$

To give a probability over classes, they pass these similarities on to a softmax function. This model effectively learns a metric space where a good representation of that class is the average of a few examples of a class, and class members can be assigned based on distance. Matching networks and prototypical networks focus on learning the embedding as well as comparing examples using a predefined metric (cosine and euclidean distance, respectively). **Relation networks** (Sung et al., 2017) also learn a metric for the embedding comparison. The relation network, like prototypical networks, averages the embedding of each class in the support set together to form a single prototype. Each prototype is then combined with the query embedding and transferred to a relation module that produces a similarity score between 0 and 1 where 1 indicates that the query example belongs to this class prototype as seen in Figure 2.4.

2.3 Meta Learning

Meta learning methods address the few-shot learning problem by "learning to learn." In meta-learning, an algorithm is trained by several training tasks. Each task contains a support set that imitates a N-way-K-shot classification problem. Along with the support set, there is a query set which contains

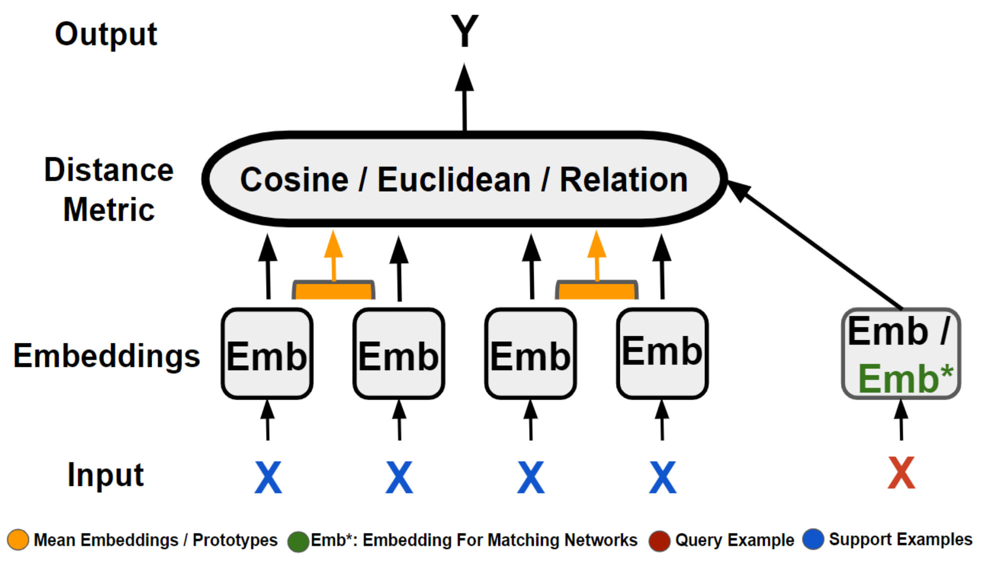

Output
Y
Distance Metric
Cosine / Euclidean / Relation
Embeddings
Emb
Emb
Emb
Emb
Emb / Emb*
Input
X
X
X
X
X
Mean Embeddings / Prototypes
Emb*: Embedding For Matching Networks
Query Example
Support Examples

Figure 2.4: Visualization of different metric learning approaches. a) Prototypical networks embed the query examples and the support examples by using the same network, but average support embeddings to make prototypes for each class. The euclidean distance is used to find difference between class prototypes and the query example. b) Matching networks compute separate embeddings for support and the query examples. The cosine similarity measure between each embedding of support and query example is used to choose the class. c) Relation networks replace the Euclidean distance with a non-linear distance metric that is learned.

previously unseen examples of the task that is used to test accuracy. Each task can be completely non-overlapping; we may never see the classes from one task in any of the others. The model parameters are updated at each step of meta-learning, based on a randomly selected training task. The cost function measures the performance on the query set of this training task. Since the network is presented with a different task at each step of the process, it must learn how to distinguish between data classes in general, rather than a specific set. Meta-learning has proven to be highly efficient in computer vision, where low-level features can be transferred across classes. One of the approaches is aimed at learning a set of weights over training tasks so that classifiers can be learned with a limited number of labeled examples and few gradient update steps is model-agnostic meta-learning(MAML) (Finn et al., 2017). Another area of research is about learning an optimizer. Examples include the LSTM-based meta-learner to replace a stochastic gradient descent optimizer with an external memory, and the weight-update mechanism (Ravi and Larochelle, 2017). The progress has been made in the direction, and these methods have been successful in achieving rapid adaptation with a limited number of training examples for novel classes as seen in Figure 2.5. Yet, the literature reviews point out some shortcomings of these methods in having difficulty handling the domain variance between base and novel classes.

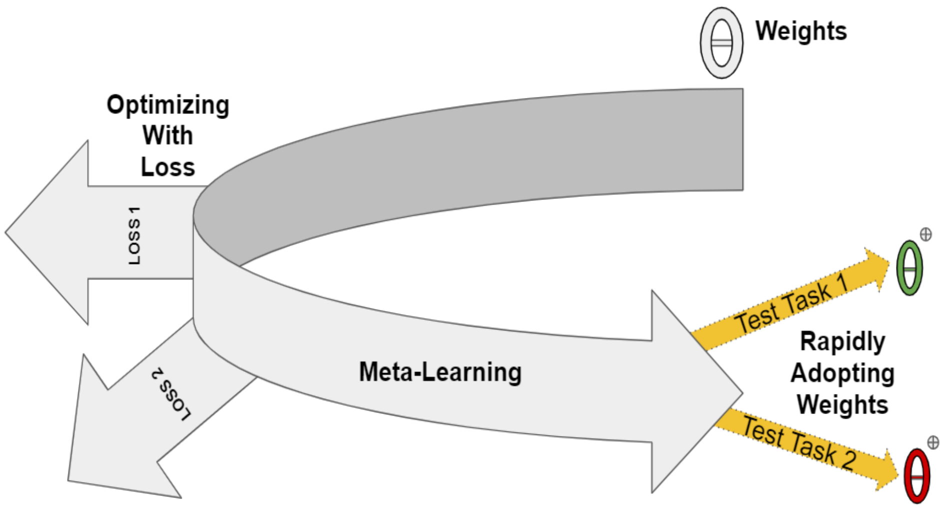


Figure 2.5: Illustration of the MAML algorithm rapidly optimizing the initial weight with loss. It quickly adopts/tunes the weights for previously unseen tasks.

2.4 Data Augmentation

Data augmentation directly tackles the issue of data scarcity by "learning to augment." This method learns a data generator from the base class data and uses the learned generator to mimic new class data. This is done by training a conditional generative adversarial network from the source to generate within source data items. The data realism of newly generated samples is not prioritized as the goal is to generate samples just for learning the classifier. This method in its essence tries to change few-shot classification problems to standard classification problems and hence deviates from the goal set for this thesis.

Chapter 3

Approach

Wikipedia is a one of the worldâs largest online encyclopedia. It is the most popular general reference source on the internet and has millions of articles available online with hundreds[1] of new articles uploading every day. Wikipedia has been the resource and inspiration for numerous research work up till now. The amount of raw information stored in this platform is easily available using the database backup dumps. This huge source of information gave us the inspiration to further investigate if we can benefit from the preexisting division of section heading within each article. We downloaded the latest Wikipedia articles dump at the time and further processed it to collect only the level 2 sections headings from each article along with their paragraphs. This implementation basically provided us with a huge collection of labelled data which was necessary for the pre-training of models. The further details of this dataset are discussed in this chapter.

3.1 Experimental Setup

3.1.1 Dataset

The dataset consists of a 3446-heading subset of the Wikipedia article corpus. The headings are chosen on a special criterion of having 20 paragraphs per heading to keep the experiments balanced. The original unaltered dataset consists of millions of headings taken from Wikipedia articles on subjects like film, religion, guns, animals, etc. Each article title has a unique id and is followed by its heading and the paragraphs present in that heading as seen in Table 3.1. We extracted only the level two headings from each Wikipedia article, keeping in mind the computational resources required for processing

[1]https://en.wikipedia.org/wiki/Wikipedia:Statistics

Nr.	Id	Title	Heading	Paragraph
1	9087	James Dyson	Introduction	Sir James Dyson bo ...
2	56321	Apple Inc.	Corporate Culture	Apple is one of sever ...
3	1630	Glenn Research	History	It was first named A ...
.	.	.	.	...

Table 3.1: Sample raw dataset file.

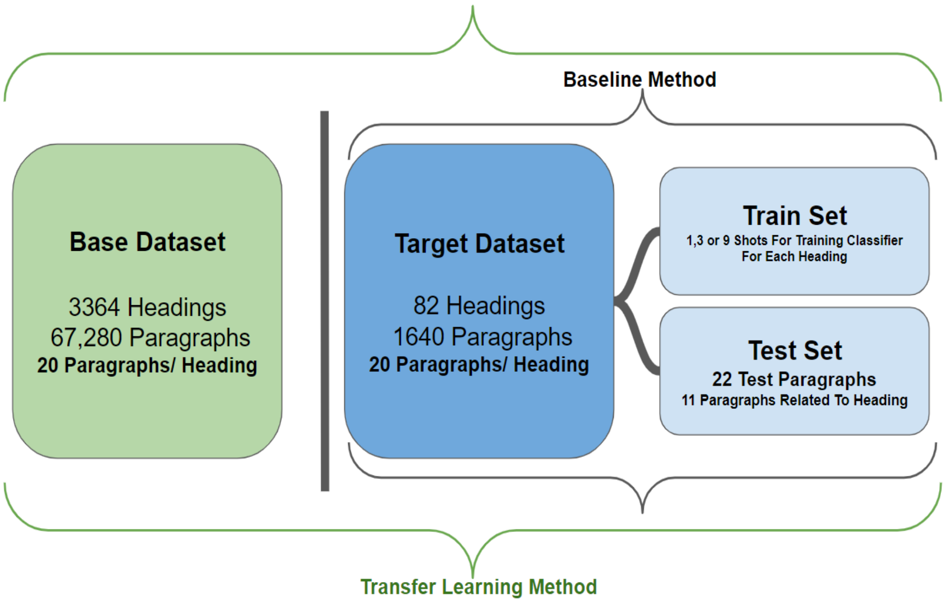


Figure 3.1: Division of final dataset into base dataset and target dataset.

and experimenting as well as keeping the scope of this thesis limited but with the possibility to easily expand it for future studies.

The headings from this subset are divided into the base dataset and target dataset. The base dataset contains most of this labelled data as it is imperative to have a large pre-training dataset for transfer learning. The target dataset refers to the few-shot dataset with a limited number of labeled data in the training set. The test set in the target dataset is used to evaluate the performance of these methods. The detailed overview of datasets and their usage can be seen in Figure 3.1.

3.1.2 Preprocessing

Each heading used in the experiments is passed through a special criterion. Following these criteria, the headings along with their paragraphs become part of the final datasets. First is the character count of each paragraph. This

Nr.	Id	Title	Heading	Paragraph
1	61721	Caroline Hirsch	Career	While collecting un ...
2	85007	Risto Hieta	Career	Hieta became famo ...
3	19715	HMS Amphitrite (1778)	Career	On 2 May "Amphit ...
4	34210	Magazine	Contents	Campbell's plan for ...
5	99715	Theodore Lukens	Forester	Lukens hired 55 me ...
6	22012	Kilgore Trout	Appearances	Trout, who has sup ...

Table 3.2: Training episode for 3-way task. This episode consists of 3 paragraphs belonging to a common heading and those not belonging to the heading as seen in the heading column.

was set to a minimum of 250 characters so that each paragraph can present enough information for the experiments. Secondly, it was encountered during the dataset creation that some of the headings have a name consisting of either single characters or a number hence they were rejected. The third and most important of these criteria was the number of repetitions for each heading. In this thesis, it was important to have a minimum of 20 paragraphs for each task to be further divided into train and test sets of the target dataset.

3.1.3 Implementation Details

Episode Creation

A single training episode consists of k (1,3 and 9) shot from each heading as our train data for training a classifier. We take 22 unlabeled paragraphs for our test set. This test set is used to predict the classifier performance for each heading. From these 22 unlabeled paragraphs, half of them belong to a common heading and the other half are randomly selected and don't belong to the heading. Hence for each episode, we will have an equal number of paragraphs as well as equal division of paragraphs in our test set. We train our model based on the train set and evaluate it on the test set. Table 3.2 shows a sample of a training episode for a k=3 shot task.

Embeddings

The experiments performed in both the methods use three distinctly different representations namely bag of words representations, fasttext, and bert.

The choice of using the bert model was obvious as the majority in the machine learning community are adopting or using it as benchmark to produce state-of-the-art results in a wide variety of NLP tasks. The true power of bert lies in its use in NLP transfer learning. Bert is trained on lower-cased English text. The amount of data it is trained on made it an ideal choice to be used in this thesis. The fasttext model though not as complicated in structure as the bert model but is trained on Wikipedia. It also proves beneficial for the transfer learning. Lastly the bag of words model provides us with a simplified representation which acts as a base point for our experiment. These different representations help us to better distinguish the effects of using simple approaches such as using bag of words models as compared to advanced models for the few-shot task.

Bag of Word For the bag-of-words representations, we are using a sklearn[2] count vectorizer function to vectorize the input paragraphs. In the case of the baseline method, the train and test set paragraphs from the target dataset are vectorized based on the term frequency. In the transfer learning method, the base dataset is vectorized and then the paragraphs from the train and test sets of target datasets are fitted with these vectors.

FastText The fasttext embeddings are obtained using pre-trained word vectors for the English language, trained on Wikipedia using FastText[3]. The vectors generated from this model are translated into 300 dimensions and are obtained using skip-gram model. We use the function get_sentence_vector which returns a vector representation of a given paragraph. we use this function over the base dataset, train set and the test set paragraphs of target dataset.

BERT We use bert-as-service[4] which is a sentence encoding service for mapping a variable-length sentence to a fixed-length vector using the original bert model. The model used to get the bert embeddings is a bert-base-uncased pre-trained model[5] which has 12-layer, 768-hidden, 12-heads, 110M parameters, and is trained on lower-cased English text. Each paragraph in the base dataset, the train and test sets of target dataset are translated to a 768-dimensional vector.

Accuracy Score

The accuracy scores are measured over the test set of the target dataset. A logistic regression classifier is trained over the labeled train set and then the prediction is made for the unlabeled test set of the target dataset. The training

[2]https://scikit-learn.org/stable/
[3]https://fasttext.cc/docs/en/pretrained-vectors.html
[4]https://github.com/hanxiao/bert-as-service
[5]https://github.com/google-research/bert

Heading	Accuracy
Power and abilities	0.81
Renovations	0.63
Spiritualism	0.86
Technology	0.59
Transport	0.81
Average Accuracy	**0.74**

Table 3.3: Sample result for the final output of each experiment. The result constitutes of headings with their corresponding accuracy scores. The accuracy scores are averaged over all heading in the target dataset.

and the prediction are done episodically as mentioned in the previous section. For each heading, the training and prediction are done separately as seen in Table 3.3. The individual accuracy scores are averaged to get the final score for each of the experiments. We use the sklearn machine learning library for the logistic regression classifier as well as for the predict function for the accuracy scores.

3.2 Methods

A chain of experiments is performed on both the baseline and transfer learning methods. The flow of experiments can be seen in Figure 3.2.

3.2.1 Baseline

As the baseline of our experiments, we investigate the performance of a binary classifier by training only on the few train set paragraphs from the target dataset for each heading. The k-shot train set paragraphs are passed on to a binary classifier along with their labels. The classifier is then trained to accurately classify the test set paragraphs as seen in Figure 3.3. This process is repeated for all the headings in the target dataset.

The baseline experiment is performed several times each time using different experimental settings such as having different representations and k values as seen in Figure 3.2. The performance after each experiment is measured on the basis of test set after training on the k-shot(1, 3, and 9) task.

In the target dataset we have a total of 82 headings. The experiment begins by passing the train set paragraphs belonging to a specific heading to the feature extractor. The paragraphs in the train set are vectorized and are used to train a classifier. The size of the train set depends on the specified k-shot value as seen in Figure 3.4. The train set is structured in a way that k-paragraphs

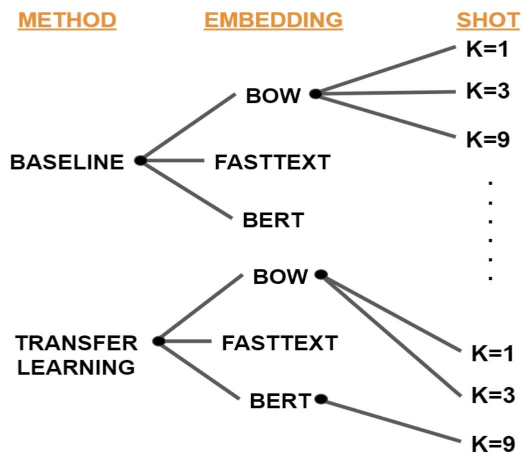


Figure 3.2: Figure illustrating the flow of experiments performed in this thesis. Each method is followed by using 3 embeddings and each embedding is used 3 times again for 3 different k-shot tasks. The total number of experiments performed is 18, excluding 9 experiments performed on a slightly different version of the transfer learning method.

belong to a common headings and equal number of random headings. After training on the train set, features from its corresponding test set are extracted and the vectorized paragraphs are forwarded to the trained classifier for the prediction. The test set contains 22 unlabeled paragraphs half of which belong to the common heading and the other half are equal number of random paragraphs as shown in Figure 3.5. After predicting and measuring the accuracy score for a single heading, the process is repeated for all the remaining headings in the target dataset. The accuracy scores over all the headings are averaged to give the final score for a single experiment setting.

3.2.2 Transfer Learning

In transfer learning, we experiment with the idea of transferring the knowledge gained from one learning task to another. The model is pre-trained on the base dataset which contains paragraphs from thousands of headings. For the transfer learning we experiment with two models as seen in Figure 3.6. One is the standard model that contains a hidden layer. In the other model there are only input and output layers. The learned weights from either the hidden layer or the output layer from the pre-training step are transferred to the fine-tuning step where they are used to optimize the classifier as shown in Figure 3.7. The episode-based training and testing as in the baseline method is used to measure the accuracy from the test set of the target dataset.

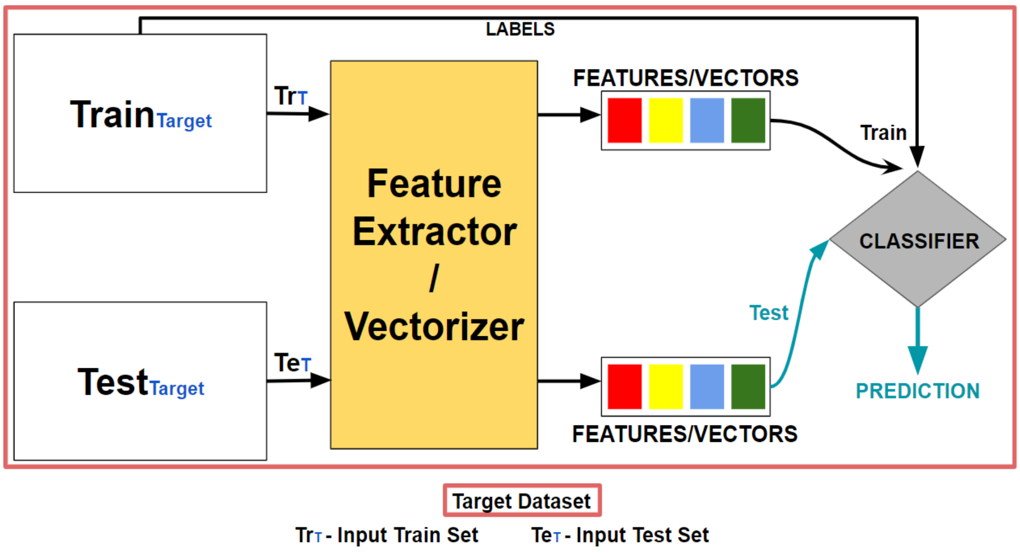
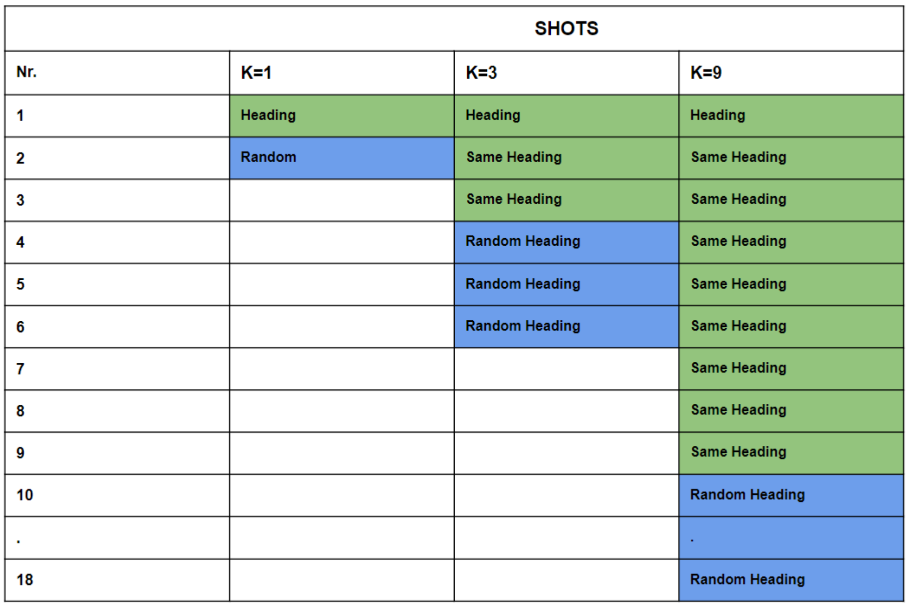


Figure 3.3: Baseline classification method trains a classifier with k-shot train set paragraphs belonging to a specific heading. The trained classifier is then used to predict the paragraphs in the test set.

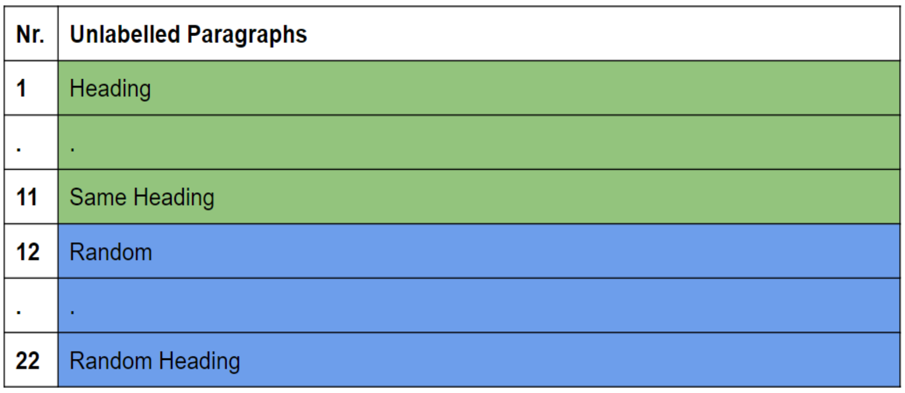
As a resource for the pre-training, we collect paragraphs from Wikipedia. This collection of labeled paragraphs is sampled from 3364 headings, with each heading having 20 paragraphs. In total, we have 67,280 paragraphs in the base dataset. The base dataset is shuffled to have a random distribution of paragraphs. A batch of 100 paragraphs is selected from the shuffled base dataset and fed to the model. For the model to be useful, we need to train it, such that the weights are meaningful, non-random values. The model makes the predictions, we then compare the output value of our model and the ground truth expected value via the loss function. The loss function we choose for our model is binary cross entropy with logits loss[6]. The adagrad optimization algorithm is used to optimize the weights. This process repeats for all the remaining batches in the base dataset. We end up with the optimized parameters after the pretraining which are transferred to the fine-tuning step.

[6]https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html



	SHOTS		
Nr.	K=1	K=3	K=9
1	Heading	Heading	Heading
2	Random	Same Heading	Same Heading
3		Same Heading	Same Heading
4		Random Heading	Same Heading
5		Random Heading	Same Heading
6		Random Heading	Same Heading
7			Same Heading
8			Same Heading
9			Same Heading
10			Random Heading
.			.
18			Random Heading

Figure 3.4: Train set training episodes for k=1, 3 and 9 shot tasks. Each episode consists of k same-heading paragraphs (green) and random (blue) paragraphs.



Nr.	Unlabelled Paragraphs
1	Heading
.	.
11	Same Heading
12	Random
.	.
22	Random Heading

Figure 3.5: Test set episode for testing the accuracy of a classifier. The number of test set paragraphs remain the same across each experimental setting. It consists of 11 related-heading paragraphs (green) and 11 random (blue) paragraphs.

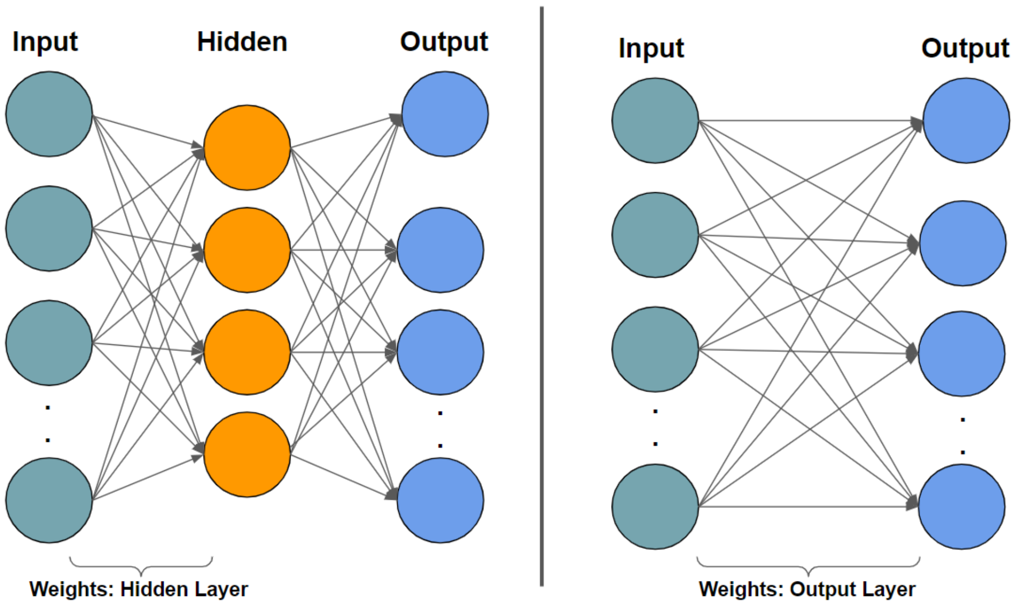


Figure 3.6: Left: Standard model with a 300-dimensional hidden layer, weights from this hidden layer are transferred. Right: Modified model with only input and output layer, weights from the output layer are transferred.

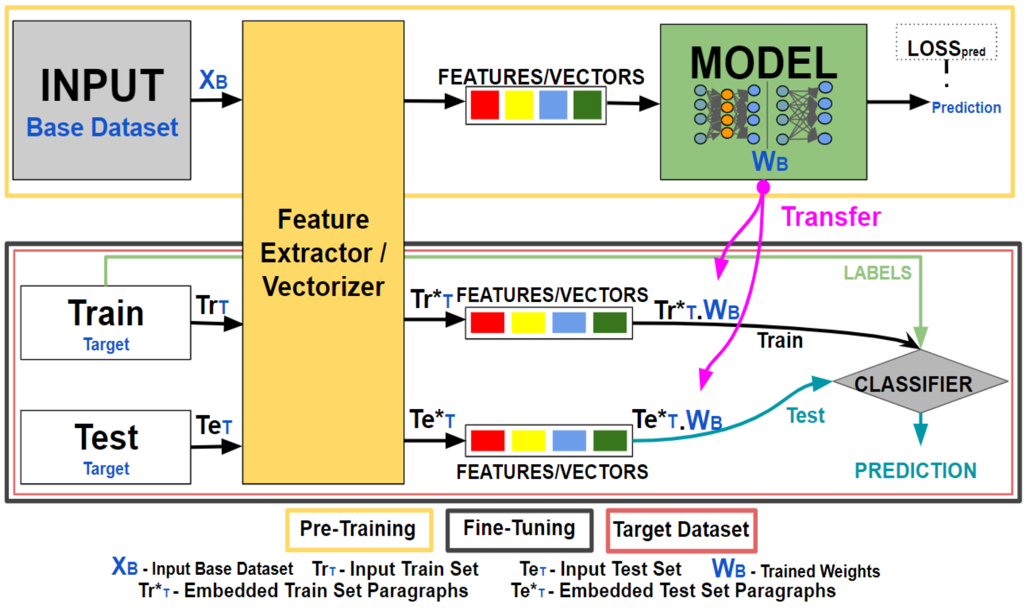


Figure 3.7: Illustration of the transfer learning method. The knowledge is gained by training a model over a large number of paragraphs in the base set. The optimized weights from the pre-training are used for fine tuning the classifier using the target dataset.

Chapter 4

Results

In this chapter we discuss the results of the two methods that we devised for this thesis. The comparison of their performance serves as an indicator that the transfer learning method shows notable increase in the performance as compared to the baseline method.

4.1 Baseline

We computed the few-shot classification accuracies for the baseline method on the Wikipedia article corpus by averaging over 82 different headings in the test set of the target dataset. The results for the baseline method are shown in Table 4.1. We achieved satisfactory results in the case of using fasttext and bert representations as expected because we used pre-trained word vectors of these models. The results from all three representations show increase in the accuracy scores when there is an increase in the training data from the train set (k). Figure 4.1 accurately depicts the difference in the performance of using k-different shots as well as representations in the baseline method. It shows that for all the embeddings, the increase of k yields an increase in the performance of the model. Bert embeddings show the best performance among all the settings. The k=9 shot bert experiment achieves 89% accuracy which, considering the baseline experiment, is already a very good score and demonstrates the ability of the pre-trained bert model.

Baseline	K=1	K=3	K=9
Bow	0.48	0.58	0.74
Fasttext	0.66	0.78	0.84
Bert	0.73	0.84	0.89

Table 4.1: Baseline method accuracy scores for the different k-shot tasks.

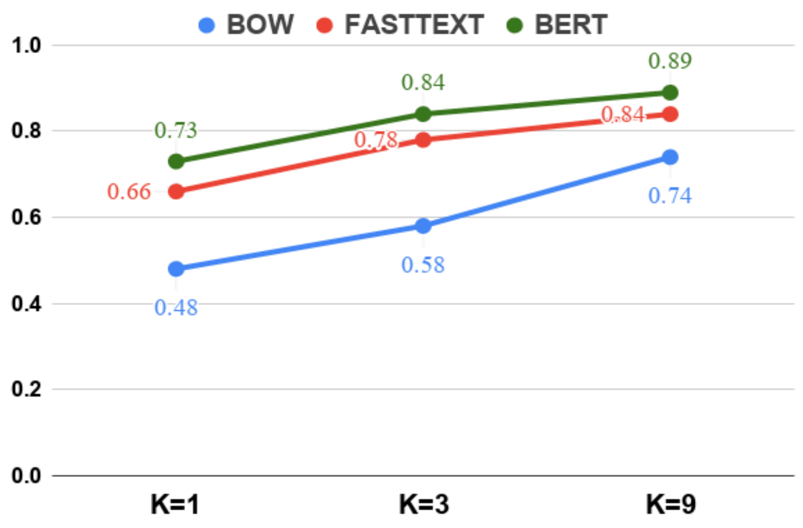


Figure 4.1: Plot illustrating the differences in the accuracy scores for the baseline method in all experimental settings.

4.2 Transfer Learning

In the transfer learning setting, the evaluation criteria are similar to the baseline method as we average the results over 82 heading from the target dataset. The difference arises in the preceding pre-training step of this method. We use the entire base dataset to train a model for a single epoch. This training yields the optimized weights which are then transferred to the target dataset to optimize the classifier.

The model used for the pre-training step had 2 variants. At first, we tried a standard model for training on base dataset. This model consisted of an input layer, a 300-dimensional hidden layer, and an output layer. Transferring the representations from the hidden layer of this model resulted in unsatisfactory results as we saw slight decrease or no change in the performance as compared to the baseline method as seen in Table 4.2.

Transfer Learning (Standard)	K=1	K=3	K=9
Bow	0.49 **(+0.01)**	0.52 **(-0.06)**	0.71 **(-0.03)**
Fasttext	0.62 **(-0.04)**	0.75 **(-0.03)**	0.81 **(-0.03)**
Bert	0.73 **(0.00)**	0.84 **(0.00)**	0.88 **(-0.01)**

Table 4.2: Transfer learning accuracy scores for k-shot tasks using the standard model to train on the base dataset. The values in brackets represent the changes in accuracy scores as compared to baseline method.

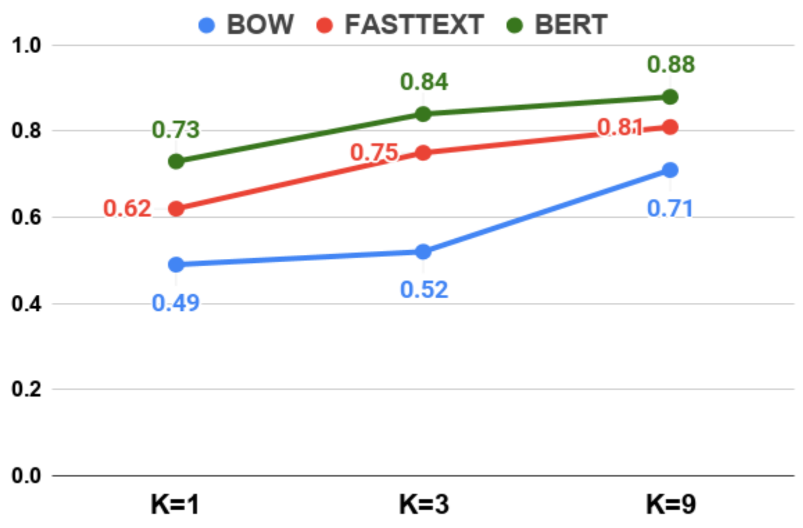


Figure 4.2: Plot illustrating the differences in the accuracy scores for the standard transfer learning method in all experimental settings. This figure shows the resemblance to the baseline method results.

Figure 4.2 shows the results of the standard transfer learning method. The result trend of this method shows similarity to the baseline method. We can clearly see the distinction between the accuracy scores when using different representations.

After analyzing the results from the standard model, we decided to alter the model and have a single linear layer. Using this model, we transferred the representations from the output layer. For the referencing purpose in this chapter we refer to the modified method as "modified transfer learning". The results after this modification as shown in Table 4.3 have a significant improvement in the case of the bag of representations. With fasttext and bert, the accuracy scores either remain the same or register minor drop or increase in the accuracy scores as seen in Figure 4.3. This was anticipated as these models are pre- trained and no new knowledge is to be learned. But in the case of the bag of word representation, there is an average increase of almost 15% in the accuracy scores as compared to the baseline method as shown in Figure 4.4. The increase in the accuracy scores makes the bag of words model comparable to the more advanced models used in this experiment. The detailed results for all the experiments can be seen in Figure 4.5. This serves as an indicator that the transfer learning in this case is highly beneficial. The results also indicate that paragraph representations which are good for the section heading classification are also good for the few-shot learning task.

Transfer Learning (Mod)	K=1	K=3	K=9
Bow	0.68 (+**0.20**)	0.75 (+**0.17**)	0.84 (+**0.10**)
Fasttext	0.69 (+**0.03**)	0.78 **(0.00)**	0.83 **(0.00)**
Bert	0.73 **(0.00)**	0.81 **(-0.03)**	0.87 **(-0.02)**

Table 4.3: Modified transfer learning accuracy scores for the different k-shot tasks. The values in brackets represent the changes in accuracy scores as compared to baseline method.

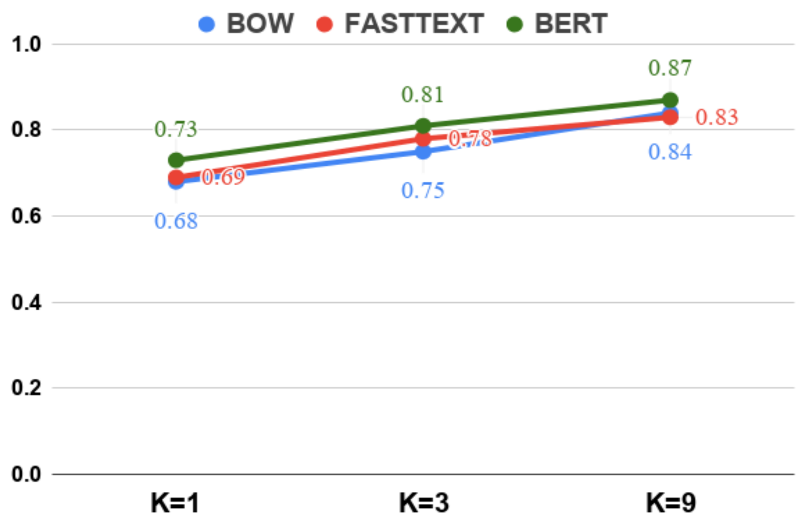


Figure 4.3: Plot illustrating the differences in the accuracy scores for the modified transfer learning method in all experimental settings. The modified transfer learning approach enables previously untrained embeddings such as bow representations to achieve comparable scores as compared to the modern pre-trained embeddings such as fasttext or bert.

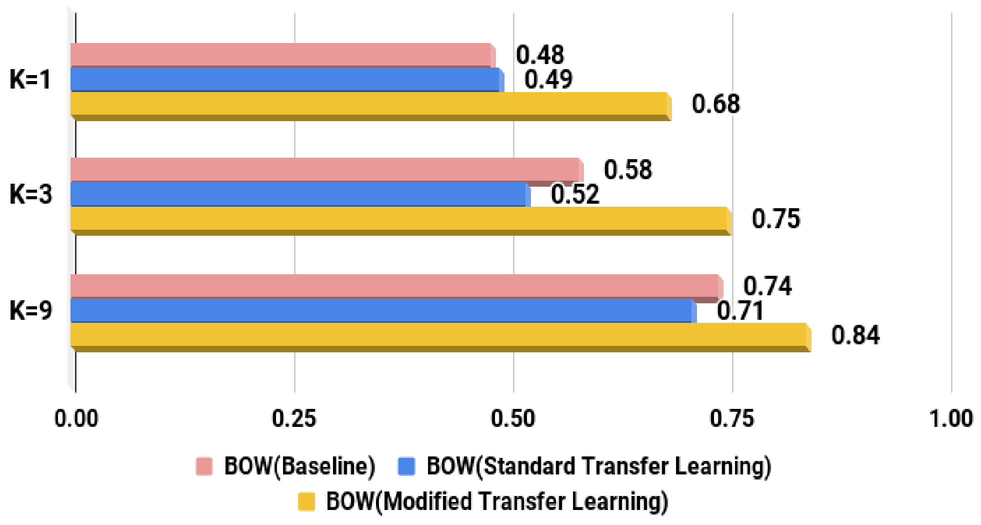


Figure 4.4: BOW result comparison between the baseline, standard transfer learning and modified transfer learning method.

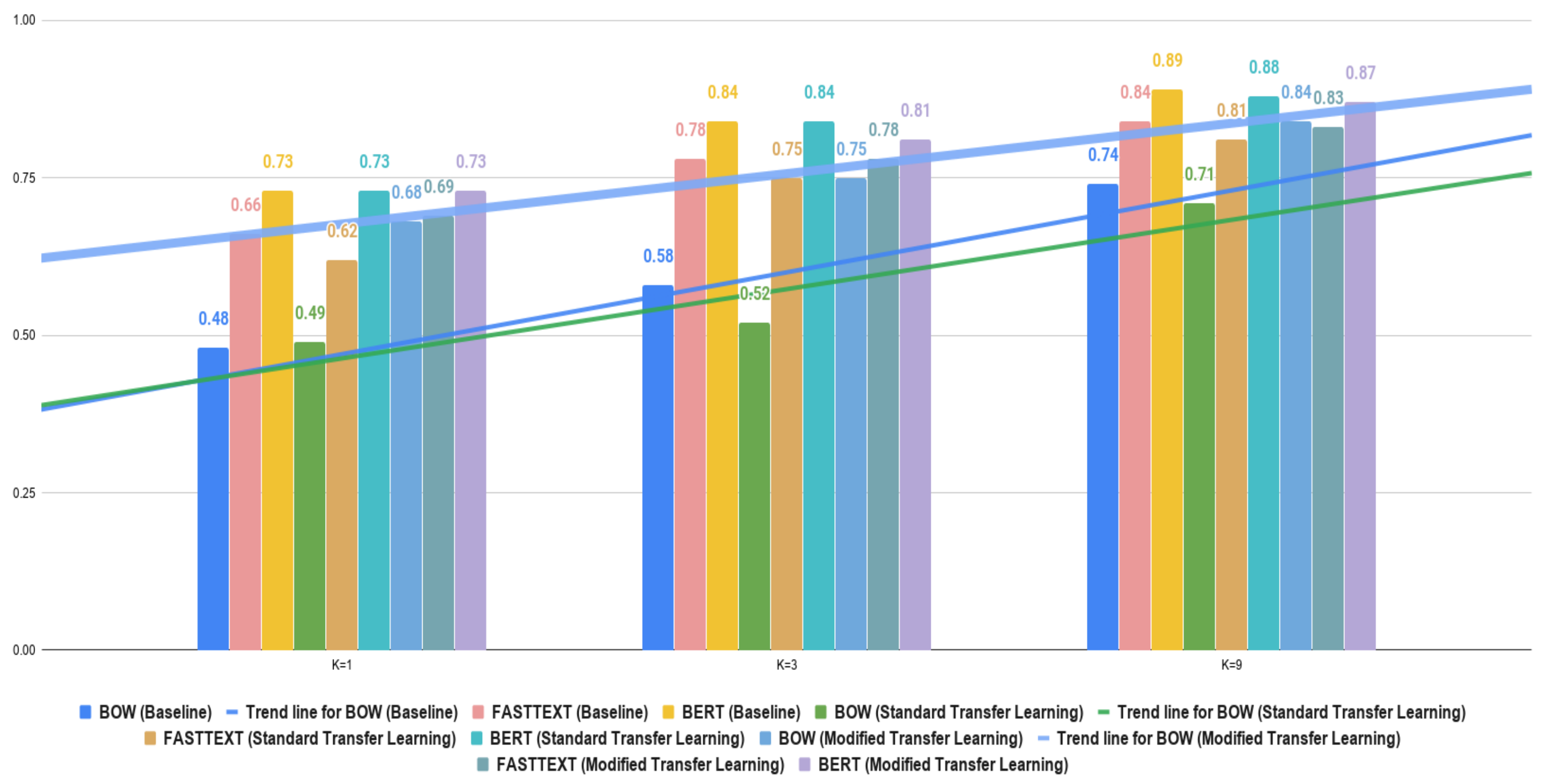


Figure 4.5: Complete results table with trend line representing the significant differences between two methods and their variants. The blue trend line represents the baseline method, the green line represents the trend of standard transfer learning method and the light blue trend line represents the modified transfer learning method results.

Chapter 5

Conclusion

In this thesis, we investigated the use of few-shot learning methods on textual data. The first contribution of this thesis is the Wikipedia section heading corpus as a resource for pre-training. This corpus consists of millions of level 2 headings along with their respective paragraphs. As proved by the experiments, this corpus can be used effectively as a resource for pre-training a model for the heading classification task. The diversity and the abundance of this resource can truly be benefited if provided with sufficient computational resources.

The next contribution of this work is the application of the transfer learning method in the textual domain. As most of the work being done in few-shot learning is based on image-related tasks, the application of few-shot methods in this domain opens the door for new research paradigms. The results gained by the application of the transfer learning method encourages the idea to use other few-shot learning methods in this domain which can produce even more promising results and research questions.

While this thesis resulted in a lot of interesting arguments and promising results, still many research questions remain unanswered.

As future work, the Wikipedia section heading corpus which in this thesis focuses on just the level 2 section headings can easily be extended to gain access to even more labeled textual data. This can benefit the training of models and theoretically increase the accuracy scores. The number of section heading used in this thesis were kept nominal as compared to the actual scale of the dataset due to the limitations of computational resources. Even with the limited resources which bounded our ability to use the current section heading corpus to its fullest, the results significantly point out the benefit of the transfer learning used in this thesis. It would be interesting to see in future that how other few-shot learning approaches such as metric learning and meta-learning can perform in the same task. It would also be interesting to experiment with

Bert-large model, as we used the Bert-base uncased model weights. The reason for doing so was that the Bert-large model weights are too big for a GPU and currently require a TPU which was not used during the experiments. It can provide us with the peak accuracy score of the bert model for this specific task. Along with that, it would be intriguing to see how our trained classifier performs on a topic classification data not from the Wikipedia.

Bibliography

Katherine Bailey and Sunny Chopra. Few-shot text classification with pre-trained word embeddings and a human in the loop. *CoRR*, abs/1804.02063, 2018. URL `http://arxiv.org/abs/1804.02063`.

Shrisha Bharadwaj. embarrsingly-simple-zero-shot-learning. *GitHub repository*, 2018.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification, 2019. 2.2, 2.1

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. 2.3

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting, 2018. 2.1

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1398. URL `https://www.aclweb.org/anthology/D18-1398`.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527â1554, July 2006. ISSN 0899-7667. doi: 10.1162/neco.2006.18.7.1527. URL `https://doi.org/10.1162/neco.2006.18.7.1527`.

Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICMLâ11, page 521â528, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Gregory R. Koch. Siamese neural networks for one-shot image recognition. 2015. 2.2

Abhishek Kumar and Hal Hal III. Learning task grouping and overlap in multi-task learning. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 2, 06 2012.

Tsendsuren Munkhdalai and Hong Yu. Meta networks, 2017.

Hang Qi, Matthew Brown, and David G. Lowe. Low-shot learning with imprinted weights, 2017. 2.1

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 2.3

Jürgen Schmidhuber, Jieyu Zhao, and Marco A Wiering. Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. *Machine Learning*, 28:105–130, 1997.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. 2017. 2.2

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. 2017. 2.2

Sebastian Thrun. *Lifelong Learning Algorithms*, page 181â209. Kluwer Academic Publishers, USA, 1998. ISBN 0792380479.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. 2016. 2.2

Yaqing Wang, Quanming Yao, James Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning, 2019.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. Diverse few-shot text classification with multiple metrics. *CoRR*, abs/1805.07513, 2018. URL `http://arxiv.org/abs/1805.07513`.