MULTI-TEACHER KNOWLEDGE DISTILLATION USING TEACHER'S DOMAIN EXPERTISE

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Arafat Bin Hossain

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

November 2022

Fargo, North Dakota

NORTH DAKOTA STATE UNIVERSITY

Graduate School

Title

MULTI-TEACHER KNOWLEDGE DISTILLATION USING TEACHER'S DOMAIN EXPERTISE

By

Arafat Bin Hossain

The supervisory committee certifies that this thesis complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Prof. Muhammad Zubair Malik
Chair

Prof. Saeed Salem

Prof. Md Mukhlesur Rahman

Approved:

11/18/2022
Date

Simone Ludwig
Department Chair

ABSTRACT

Large BERT models cannot be used with low computing power and storage capacity. Knowledge Distillation solves this problem by distilling knowledge into a smaller BERT model while retaining much of the teacher's accuracy in student. A teacher expert in predicting one class should be chosen, by student, over others for that class - we used the teacher's domain expertise like this to train the student. We calculated per-class accuracy for the Student and the Teacher and recorded the difference between the student from the teacher for all k classes. With k differences, we calculated the median of the differences to quantify the student's overall deviation from the teacher over all k classes. Student trained using our approach eventually outperformed all its teachers for the MIND dataset where it was 1.3% more accurate than its teacher, BERT-base-uncased, and 2.6% more accurate than its teacher, RoBERTA, in predicting k classes.

ACKNOWLEDGEMENTS

My heartiest thanks to Professor Muhammad Zubair Malik for having trust in me and supporting me for 2 years in NDSU. I would also like to thank all my colleagues in QBB-108 for providing a friendly work environment.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1. Overview

Pre-training a language model is an effective method for enhancing sentence-level tasks like Natural Language Inference (NLI), which involves Natural Language Processing [2] [11] [16] [17] . In other tasks like Named Entity Recognition, Question Answering, or Text Summarization, pre-trained models are proving to be quite good at predicting the relationships between sentences or even at the token level, producing a very fine-grained output. Jacob Devlin and his colleagues in Google built the groundbreaking Bidirectional Encoder Representations from Transformers (BERT) - transformer-based machine learning technique for natural language processing (NLP) . Ever since its release, BERT became one of the first choices in understanding or handling language-based tasks making it an inevitable baseline for NLP tasks [18]. In 2019, BERT was introduced in the google search engine. After the release of BERT, it has achieved state-of-the-art performance on several popular NLP tasks in the research community such as GLUE(The General Language Understanding Evaluation) and SQuAD (Stanford Question Answer Dataset).

The BERT model comprises a lot of encoder layers—12 for the Base version and 24 for the Large version—that the publication refers to as Transformer Blocks [4]. Additionally, these have larger feed-forward networks than the Transformer's reference version in the first paper (768 and 1024 hidden units, respectively), as well as more attention heads (12 and 16), respectively (6 encoder layers, 512 hidden units, and 8 attention heads). A core understanding of how BERT works lies in the pretraining of BERT. It has been pre-trained on a large natural text corpus based on two tasks- Masked Language Model(MLM) and Next Sentence Prediction(NSP). While training, the model tries to predict the next sentence of the masked sentence and it is during these two prediction tasks that the model achieves a good understanding of the language. These two trivial yet important techniques help achieve BERT a very good understanding of the language the model is trained upon. Researchers extended these ideas to build many different variants of BERT like - RoBERTa[15] and DeBERTa[9] which have proved to outperform the base BERT variant on popular benchmarking datasets like GLUE and SQuaD. To add more meaning to the usefulness of the pretraining of models, researchers went a step ahead and extended the idea to build clinical variants of BERT

by pretraining the BASE model with large medical corpora like PubMed, MIMIC III dataset, etc. Open leaderboard portals like BLURB contain the list of all the lead performers of clinical BERTs. Two of the most renowned variants are ClinicalBERT [12] and PubMedBERT [7] which are released by Microsoft. Another important idea behind BERT is its flexibility to be fine-tuned for task-specific purposes. Any pre-trained BERT model can be fine-tuned for any domain-specific data, which means its complicated and robust architecture can be used to fit domain-specific knowledge and hence be used to create a new model which has now the weights and biases of the pre-trained slightly fine tuned.

However, because of the high computational complexity and significant storage needs, it is difficult to deploy these clumsy deep learning models on devices with constrained resources, such as mobile phones and embedded devices. Numerous model compression and acceleration approaches have been created in order to achieve this. Knowledge distillation[10] effectively creates a tiny student model from a large instructor model as an example of model compression and acceleration. The basic idea is to temper the loss function of the student model by looking up to the teacher model. The lighter student model back propagates and updates the weight by gathering the knowledge differences between itself and the teacher. Generally, it is considered to be a very good student if the student can retain a good percentage(80/90 percent for example) of the teacher in terms of prediction but there has been recent work which shows that students can outperform the teacher[3].

1.2. Research Question

In our research work, we tried to combine the idea of BERT fine-tuning and Multi-Teacher Knowledge Distillation(MT-KD). There is a specific question we tried to find the answer for- ***In Multi-Teacher Distillation, can a student outperform its teachers given it knows exactly which teacher to follow depending on the training sample?*** We tried to find the answer to these questions by doing KD experiments on *General Natural Language Inference Task*, mainly the multi-class text classification problems. A very brief overview of how we have formulated the problem is discussed in Chapter 4 where we illustrate our experiments and methodologies. In Chapter 5, we will try to answer these questions while discussing the results.

1.3. Brief overview of the experiments and results

We performed an experiment for two different types of NLP tasks - **NLI task**: In the NLI task, we took four datasets- EMOTION, TWITTER-EVAL-EMOTION, TWITTER-EVAL-SENTIMENT MIND because we wanted to work with multi-class classification. We explored a technique based on the assumption *if a teacher is best in predicting a certain class, the student should follow that teacher over others for that class.* We fine-tuned multiple teachers, RoBERTa, BERT base, DeBERTa on NLI datasets and listed the best model for predicting each class label. This is the reason we tried to work with multi-class classification problems so that we can analyze the performance of fine-tuned BERT models per class. For a given training sample during distillation training, the student looks for predictions of all the teachers and picks the logits of that teacher who produced a hard prediction it is best at. If two or more teachers ended up predicting what they are best at, we took the logits of the teacher whose overall accuracy is better. DistilBERT distilled from BERT-Base+DeBERTa+RoBERTa gets better at predicting certain classes than its teachers. A detailed overview of the result is presented in chapter 5.

1.4. Additional Experiment on Clinical Patient Notes

We used clinical notes from NBME(National Board of Medical Examinations) to find medical keywords from illustrated patient history notes. In the first phase, we fine-tuned different variants of general language BERT models and Clinical BERT models, fine-tuned them on the dataset, and tried to find which BERT model works best and recorded the F1 score. For the second phase, we used fine-tuned DeBERTa and BioMedNLP-PubMedBERT as teachers to distill a smaller student model by taking average logits of teachers to represent teacher output. Additionally, we incorporate teacher loss into the traditional loss function. We used two student models for two separate experiments: Distill-SQuAD-BERT and BERT-base-cased where the former one is just Distilbert fine-tuned with the SQuAD dataset. The output from the second phase was two different distilled student models. One of the key findings was DistilBERT-Squad distilled from DeBERT+BioMedNLP-PubMedBERT achieved a better F1(0.82) on our clinical dataset than a fine-tuned BioMedNLP-PubMedBERT (0.79). Details about the results and formulation of this problem are discussed in Chapter 4.

1.5. Contribution and outline

In chapter 2, we discuss previous works in the field related to our research works. Chapter 3 elaborates on the theoretical ideas which are required to refer to our research works. It discusses how BERT works, and how the fine-tuning of BERT works and wraps up the discussion by elaborating on what is knowledge distillation and how it works. In chapter 4, we discuss the experimental setup and methodologies we used in our research work. The results are discussed in chapter 5 while chapter 6 discusses the limitation of our proposed method and room for further research.

2. RELATED WORK

Deep neural networks are currently seeing unheard-of success in a variety of applications[8] [21] [4]. However, it is challenging to apply these complicated models to embedded systems since they require a large memory footprint and computing power. The solution to this problem is knowledge distillation (KD), which increases the accuracy of a lightweight student model by extracting the knowledge from a pre-trained, hefty teacher model. Softmax of the teacher logits was considered to be the only way the transferable knowledge from the teacher to the student could be formalized. In later works, it has been shown that the intermediate teacher layers can also improve the performance of the student if incorporated[19][26][1]. Concepts like Multi teacher distillation takes advantage of multiple teachers instead of a single teacher to boost the performance of the student. The most popular techniques used for this is assigning average or weights to different teachers[25] [5] [24] or calculating the weights based on cross-entropy[14].

In this chapter, we try to discuss previous works that focus more on the ties between pre-trained language models like BERT and Knowledge Distillation. Also, we try to look into a few different techniques of MT-KD previously used to build smaller BERT models out of multiple large BERT models.

2.1. MT-BERT

Microsoft Research Asia and Tsinghua University, Beijing collaborated to build Multi-Teacher BERT(MT-BERT)[3]. With shared pooling and prediction layers, they created a multi-teacher co-finetuning approach in MT-BERT to collaboratively fine-tune several instructor PLMs in downstream tasks to align their output space for improved collaborative teaching. Additionally, we provide a multi-teacher hidden loss and a multi-teacher distillation loss to transfer the beneficial information from multiple instructor PLMs to the student model in both hidden states and soft labels. The efficacy of MTBERT in compressing PLMs has been verified by tests on three benchmark datasets. Their techniques comprise two steps:

2.1.1. Multi-Teacher Co-Fine-tuning

If we assume there are N teachers and T(i) denotes i-th teacher, for each teacher we get the output of a hidden state which is entailed in the last layer of each teacher. Using shared pooling,

N output embedding is summarized into a unified text embedding followed by a shared dense layer to produce soft probability vectors y(i) for each teacher. Now, a joint task-specific loss function is used by summing up the CrossEntropyLoss(y(i), y) for all the teachers during the fine-tuning process where y denotes the ground truth. In this way, the backpropagation is made more shared and meaningful while fine-tuning the teachers.

2.1.2. Multi-Teacher Knowledge Distillation

In the proposed MT-KD, in addition to the tradition student loss function, they introduced MT-Hidden Loss and MT-Distillation Loss. In the hidden loss, the output produced by the inner transformer layers is also taken into account while training the student model. The main goal is to have similar functions for the student with the teacher models. For the MT-Distillation loss function, they tried to change the way the traditional distillation loss is defined. Since labels for training samples are available in task-specific knowledge distillation, they suggested a distillation loss weighting method to give various samples varying amounts of weight. The weights are determined by the loss inferred from the teacher's predictions when compared to the gold labels.

The equation above shows the proposed distillation loss function. y(s) represents student prediction whereas y(i) and y represent teacher prediction and ground truth respectively. If a teacher's prediction is close to the ground truth, its corresponding distillation loss will gain a higher weight. Lastly, the traditional student loss function is also added to the mentioned loss functions above making the overall loss function as Loss Function = MT-Hidden Loss + MT-Distillation Loss + Student Loss. The t in the equation refers to the temperature which is usually set to 1. They tested their approach on SST-2, RTE, and MIND dataset and could show that, with fewer parameters, MT-BERT can perform very close to the teacher models and sometimes even better than the teachers.

2.2. CA-MKD(Confidence-Aware Multi-Teacher Knowledge Distillation)

CA-MKD aims to reduce the impact of low-quality teacher predictions while dealing with Multi Teacher knowledge distillation [27]. The idea is pretty straightforward which is, the teachers whose predictions are close to the one hot encoded ground truth are assigned larger weights or given more priority in deciding what the students should learn. Besides, CA-MKD aims to incorporate the features in the intermediate layers to make the learning process more insightful for the students. The works presented in the paper are based on pre-trained neural networks for computer vision and

hence the teacher models used are - WRN40, ResNet56, VGG13, and ResNet32 and all the teacher models have the same architecture.

The idea behind CA-MKD is to use ensemble teachers and tune the way the weights are provided to each teacher while producing the teacher prediction to be used later in the distillation loss function. By calculating the cross entropy loss between the teacher prediction and ground truth label, different weights are assigned which represent the sample-wise confidence. The less cross-entropy loss between a teacher and the student would provide more weight to the teacher. In this way, this work has tried to prevent the student from getting misguided by low quality teachers.

2.3. State of the Art BERT Models made using Knowledge Distillation

Below are some state-of-the-art BERT models which have been built by distilling large BERT-base models. DistilBERT [20] is one of the most popular and compact forms of BERT known to have performed outstandingly in the GLUE benchmark dataset while retaining 97 percent performance of BERT-base. However, though, there have been other interesting efforts to implement knowledge distillation to build compact forms of BERT-base and most of them vary from each other in the kind of knowledge they take from the teacher. While DistilBERT takes the soft target probabilities and embedding outputs from BERT-base, Tiny BERT takes hidden outputs and self-attention distribution from the intermediate layers. TinyBERT also performs a task-specific distillation on augmented data to achieve a greater student accuracy [13] and succeeds in retaining 96 percent performance of BERT(base). Efforts like MiniLM and MobileBERT take a task-agnostic knowledge distillation approach on Knowledge Distillation where the former does not consider the soft target probabilities while distilling [23] while the latter considers [22].

Table 2.1. Comparison of different student BERT models distilled from large BERT models

BERT variant	**Teacher Model**	**Distilled Knowledge**	**Number of Parameters**	**Details**
DistilBERT	BERT-base	Soft target probabilities and embedding outputs	52.2M, 6 layers	Number of layers reduced by factor of 2, Distilled on MLM, Retains 97 percent of BERT(base)
Tiny BERT	BERT-Base	Embedding outputs, Hidden States, Self-attention distributions	14.5M, 4 layers, 66M for 12 layers	Learns behavior of intermediate layer through general distillation; Task specific distillation on augmented data; Retains 96 percent performance of BERT(base), 9.4X faster and 7.5X smaller
MiniLM	IB-BERT-large	Distills the attention layer output of the last transformer layer besides Hidden states and self attention distribution	66M	Does not consider soft target probabilities while distilling;
MobileBERT	BERT-base	Self attention distribution and Self attention value relation	15.1M	Considers soft target probabilities while distilling

3. THEORETICAL BACKGROUND

To understand the experimental design and setups of our research work, it is worth understanding the underlying concepts related to **BERT** and **Knowledge Distillation**. We will start with how **Encoder** and ultimately try **Understanding the Architecture of BERT**. We will also discuss **What BERT fine-tuning actually means?**. In the latter half of this section, we will discuss **Knowledge Distillation, Different types of Knowledge Distillation and conclude with how Multi Teacher Knowledge Distillation works**.

Before describing how BERT works, it is crucial to understand how Encoder works since BERT is nothing but a stack of Encoders working together.

3.1. Encoder

An encoder is a combination of a self-attention layer and a feed-forward neural network. The self-attention layer is a layer that helps look at a word and then at the other words to help the neural net understand what other words are more relevant to the current word it is encoding. The output of the self-attention layer is fed to the feed-forward neural network.

Now that we have discussed a little about the basic architecture, let's try to tie it down with how various vectors/tensors flow between these components to turn input sentences of a trained model into a usable output. In the case of NLP applications in general, each word is converted to a vector using embedding algorithms. All the embedding happens in the bottom encoder and the other encoders receive the output from the encoder layers below them. Each embedding is of size 512 or the size of the longest sentence. Referring to Figure 3.1 again, the embedding vectors then go through the self-attention layer and the softmax is fed to the Feed Forward network. The word in each position has its path to flow through. It is only the self-attention path where there is an interdependence between different words(vectors/embedding/tensors). After the self-attention layer, there are no dependencies. So, an encoder receives a vector as input, passes it through self-attention and the feed-forward neural network, and then sends the output to the next encoder.

3.2. What is Self- Attention Layer?

" The animal didn't cross the street because it was too tired". In the aforementioned sentence, what does "it" in the sentence refer to? It refers to "Animal". When the model is processing this word, it should be able to associate "it" with "animal" and NOT "street". The self-attention layer takes care of this and it is very core to why an encoder-driven model like BERT is actually can establish a very good understanding of a language while being pre-trained. When the model processes a word, this layer allows it to scan through other words at other positions in the input sequence and make a better encoding of the word. That's why in this layer of the encoder there is a dependency between the other words.

3.3. Understanding the Architecture of BERT

Now that we have briefly discussed about Encoder, BERT is nothing but a trained encoder stack. Encoder is the fundamental unit of a BERT architecture. There are two sizes of the BERT variant - BERT-base and BERT large. BERT base has 12 layers in the Encoder stack while BERT large has 24 layers in the Encoder stack. The way these encoder stacks are trained is what makes BERT an inevitable choice for the researchers working in NLP. The training of BERT is called BERT-pretraining and is sort of an unsupervised training. Therefore, the training needs to have some tasks it requires to accomplish and the tasks are - **Mask Language Modelling(MLM)** and **Next Sentence Prediction(NSP)**. In MLM, before feeding the architecture with input words, some the words are masked and the model is trained on the masked dataset where it tries to predict the masked word with the right word. In NSP, the model tries to predict if a certain sentence is the immediate sentence of the previous sentence or not. It is during going through these pre-training that the BERT model establishes a very solid understanding of the language it is being trained upon. It is these training technique that makes BERT unique than any other previous language models.

3.4. What is BERT fine tuning

BERT-fine tuning is a technique of adding one or more layers to the existing architecture by using one of the following techniques:

- **Training the entire architecture:** In this type of fine-tuning, it is more like back propagating through the entire architecture after calculating the cross entropy loss from the output

softmax layer. While doing so, it updates the weights and biases along the entire architecture. This type of fine-tuning is more prone to over-fitting.

- **Training some layers while freezing the rest:** Train a pre-trained model partially as an additional application. What we can do is freeze the weights of the model's lower layers while retraining only the upper layers. How many layers should be frozen and trained can be experimented with.

- **Freezing the entire architecture:** We can even freeze all of the model's layers before adding a few of our own neural network layers and training the new model. Keep in mind that during model training, just the associated layers' weights will be updated.

What is ideally done in fine-tuning a BERT model is an additional or multiple layers are added to the existing pre-trained model. The layer forms a classifier for the domain-specific task the fine-tuning is done for.

Since fine-tuning is essentially a training process, so the pre-trained model has to be fed with an input sentence sequence. This is where BERT gets interesting because BERT expects the input to be in a certain type and a certain order-

- First of all, the input sequence is tokenized, and [CLS] token is added at the start of the sequence, and [SEP] token is added to address that a particular sentence has reached is termination.

- In addition to the input tokens, two other embeddings are also added - segment embedding and position embedding. Segment embedding tells the model which sentence the token belongs to. The position embedding tells the model the position of a token in the input sentence.

- The size of each vector is 768 in BERT-base and 1024 in BERT-large. These embeddings flow along the encoder stack and the output of the model is entailed in [CLS] token of the last encoder layer which is fed to the custom linear classifier layer during fine-tuning.

3.5. Knowledge Distillation

The ability of deep learning models to encode large datasets and maneuver billions of parameters has made them a successful contender in the field of computer vision and NLP [6]. While the importance of having a large pre-trained model cannot be denied, it is also a fact that the usefulness of these large models is still restricted to research communities or corporations with high-performance computing architecture that can deploy it for commercial use. To solve this problem, a variety of model compression and techniques have been developed. One of the most popular techniques is Knowledge Distillation which has received increasing attention from the research community. In knowledge distillation, a small student model tries to mimic a large model to achieve a similar or better accuracy on the same training and test data. Knowledge Distillation consists of three components - Knowledge, Distillation Algorithm and Teacher-Student Architecture [6].

3.5.1. What is Knowledge?

In KD, it is basically the logits produced by the last layer in the teacher model that forms the source of knowledge for the student to look up to.However, there are three kinds of knowledge in KD-

- **Response Based Knowledge:** It is one of the most common types of knowledge used for knowledge distillation. Here, the knowledge is the final output of the teacher model. The student simply tries to mimic the prediction of teacher. It tries to do so by calculating the distillation loss which is the cross entropy loss between the teacher and student logits. It is this distillation loss that the student tries to minimize to achieve an accuracy close to the teacher by just mimicking the way the teacher is predicting on a training sample.

- **Feature Based Knowledge:** In pre-trained models like BERT or ResNet, there are useful pieces of information embedded in the intermediate layers besides the final output layer. The distillation loss in this type of KD tries to minimize the feature activation of the teacher model and the student model by comparing the output of the intermediate layers.

- **Relation Based Knowledge:** Knowledge that captures the relationship between feature maps can be utilized to train a student model in addition to knowledge stored in the output layers and the intermediate levels of a neural network. The correlation between feature maps,

graphs, similarity matrices, feature embeddings, or probabilistic distributions based on feature representations can be used to model this relationship.

In a traditional neural networking training iteration, it is only the cross entropy loss between the ground truth and the predicted truth that forms the loss function. In knowledge distillation, the loss function of the student training is modified. Below are the steps that explain what the distillation loss function means in Knowledge Distillation:

- For a given training sample, the teacher and the student produces logits from the final output layer.
- The output is then fed to the softmax function to convert the logits into meaningful probability distributions suitable for the classifier.
- The cross-entropy loss between the student softmax **p**, and teacher softmax **q** is then calculated and we call it the distillation loss.
- The cross-entropy loss between the student softmax **p** and ground truth **g** is then calculated. We call this loss student loss.
- We incorporate a weight called alpha to smooth the loss between the teacher and the student.
- Finally, the loss function is formulated as **Loss = alpha* student loss + (1-alpha) * distillation loss**.

It is this loss function that the student wants to minimize during the distillation training.

3.5.2. Modes of Distillation

There are two modes of Distillation which is widely used while implementing KD to teach smaller models from large models:

- **Offline Distillation**: This is the most common form of distillation where the teacher model is first pre-trained on a large dataset. The student model then learns from the pre-trained model.
- **Online Distillation**: The problem with offline distillation is - sometimes, a pre-trained model won't be always available for the distillation. To solve this issue, the larger model is trained parallel to the student model.

3.5.3. Multi Teacher Knowledge Distillation

Participating in multi-instructor distillation allows students to learn from a variety of different teacher models. Using a group of teachers can provide the student model with a variety of special forms of knowledge, as opposed to knowledge learned via a single-teacher model. With the help of the expertise of several professors, the average answer for all models can be combined. The foundations for the knowledge that is typically conveyed by teachers are matrices and feature representations. Many different teachers can impart various expertise.

3.5.4. Task Based Knowledge Distillation

Another two key concepts that are crucial to understanding our research study is two types of knowledge distillation concerning the task the models are working on:

- **Task Agnostic Knowledge Distillation:** The figure above shows the major flow of the distillation process for task agnostic KD. Task refers to fine-tuning a pre-trained model with some domain-specific data. In Task Agnostic KD, the student is at first distilled from the pre-trained teacher model and then fine-tuned with the domain-specific data.

- **Task Specific Knowledge Distillation:** In Task Specific KD:

  - The teacher model is at first distilled to form a student model.
  - The student and teacher are both fine-tuned on the domain-specific data.
  - General distillation training is then implemented using the teacher and the student.

  In our research study, we used the technique of task-specific distillation for both QA and NLI tasks. The only exception is, we did not necessarily initialize the student model by distilling the teacher models. We fine-tuned lighter BERT models(DistilBERT and DistilBERT-SQuAD) as students which we fine-tuned with our dataset and then pushed into the distillation training pipeline.

4. METHODOLOGY

4.1. Experiment Pipeline

Figure 4.1 shows the general overview of the pipeline we used in our research study to get the result of our proposed Multi-Teacher Knowledge Distillation approach. **The prime task of our research study is to study the potential of Multi-Teacher Knowledge Distillation to train a student and see if it can outperform the teacher in terms of overall accuracy and per-class prediction accuracy**. So, the main experiments revolve around NLI tasks in which *the model has to predict if a sentence belongs to a certain class or not which is a* ***multi-class classification problem***.In this chapter, we will discuss our methodologies by dividing them into the following sections-

- Datasets
- Fine-tuning the teacher and student BERT model
- Finding teachers best in predicting each class
- Perform distillation training based on the analysis from the previous step and
- Comparing the performance of student and teacher model in predicting each class

4.2. Datasets

Table 4.1. Datasets used in our study

Dataset	**train size**	**validation size**	**test size**
MIND	24428	10000	5000
EMOTION	16000	2000	2000
Twitter-Eval Emotion	3257	374	1421
Twitter-Eval Sentiment	15000	2000	3000

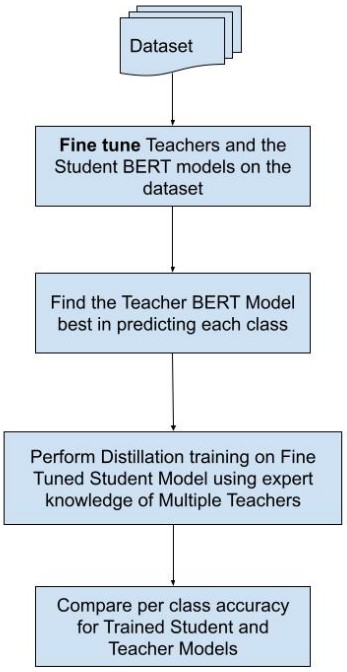

Figure 4.1. A general overview of the pipeline we have used in our study for Natural Language Inference task

4.2.1. MIND

This dataset[1] is a Microsoft News Dataset and we have used the snapshots from the hugging face dataset library. As training data, we have used the official training, validation, and test set from hugging face. We removed all the null values and made sure the data is not imbalanced by taking the sentences from the classes whose distribution is close, if not the same, to each other. Therefore, we picked only 5 different classes for our NLP task. The NLP task using the dataset is to predict whether a sentence belongs to either of the five news topics or not: Video, Finance, Food/Drink, Travel, and Lifestyle. We used 24428 samples for training and used 10000 samples for validation. Later, we compared the performance of the student and the teachers on the same 5000 test samples.

[1] `https://huggingface.co/datasets/linxinyuan/mind`

4.2.2. EMOTION

We used another dataset[2] we found suitable for our multi-class classification language inference task which is called *Emotion* dataset. As training data, we have used the official training, validation and test set from huggingface dataset library. The NLP task using this dataset is to predict what emotion among the five different emotion class does each sentence describe to: Love, Anger, Fear, Joy, Surprise, Sadness. We used 16000 samples for training, 2000 for validation and later compared the performance of student the teachers on the same 2000 test samples. The training, validation and test data we used were exactly how it was given in the huggingface dataset library.

Additionally, we also used two different datasets from huggingface dataset library *tweet eval*[3]

4.2.3. Twitter-Eval Emotion

The NLP task that we had to perform using this dataset[4] is to read a tweet and predict which of the following emotion the tweet refer to Anger, Optimism, Sadness, and Joy. We used 3257 samples for training, and 374 for validation and later compared the performance of students the teachers on the same 1421 test samples. The training, validation, and test data we used were exactly how it was given in the huggingface dataset library.

4.2.4. Twitter-Eval Sentiment

The NLP task that we had to perform using this dataset[5] is a basic sentiment analysis on the data they have provided. We used 15000 samples for training, 2000 for validation and later compared the performance of student the teachers on the same 3000 test samples. The training, validation and test data we used were exactly how it was given in the huggingface dataset library.

4.3. Fine tuning the teacher and student BERT model

Before discussing the next step, it is worth mentioning that we have used three BERT models as teachers: BERT-base-uncased, DeBERTa, and RoBERTa. We have not used any ensemble of these models and rather used the vanilla variant of them before fine-tuning them for our task-specific purpose. As students, we picked an already good student which is known to be good on NLI tasks

[2] https://huggingface.co/datasets/emotion
[3] https://huggingface.co/datasets/tweet_eval
[4] https://huggingface.co/datasets/tweet_eval/viewer/emotion/train
[5] https://huggingface.co/datasets/tweet_eval/viewer/sentiment/train

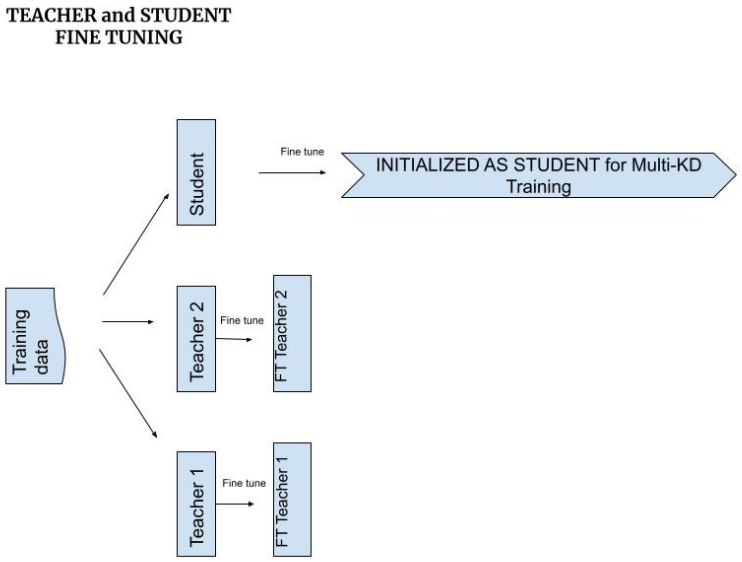

Figure 4.2. Fine Tuning Student and the Teacher for Offline distillation

and is lighter than all the teachers. Table 4.2 provides the contrast between the teacher and the student model.

Table 4.2. Number of Layers and Trainable Parameters for different BERT models

Model	**Number of Encoder Layers**	**Number of Parameters**
BERT-base	12	110M
DeBERTa	12	110M
RoBERTa	12	110M
DistilBERT	6	66M

As shown in Figure 4.2, we used task-specific Knowledge Distillation in our study, and hence the teachers and the students needed to be fine-tuned on the specific task. Fine-tuning on a specific task essentially means fine-tuning the BERT models on a specific dataset we are running the pipeline for. Since we fine-tuned the teacher and the student before pushing them towards Knowledge Distillation, so it means that we performed Offline Distillation in our study.

4.3.1. What is happening to the student and the teacher due to fine-tuning?

A very simple reason to fine-tune the teacher and the student is to make it adaptive to the specific dataset. Basically, fine-tuning does two important things -

- It adds a final classifier layer the output of which is to provide the softmax for each target class of the dataset.
- Updating some weights and biases of the original BERT model during the adaption.

After the fine-tuning process, the teacher and the student are ready for the distillation training since the teacher has the classifier layer now which, if given the input tokens, can provide softmax output that can be used as a reference for the student during the distillation. The student is directly ready for the distillation training after this step but the teacher undergoes a little more analysis before passing them to the distillation training step.

Below are the training hyper-parameters we used while fine-tuning. Since our research was not to make each model very much accurate in the prediction task, so we fixed ourselves to the defaults provided by Huggingface trainer API we used for our Machine Learning pipeline -

- learning rate: 5e-05
- train batch size: 8
- eval batch size: 8
- optimizer: Adam with betas=(0.9,0.999) and epsilon=1e-08
- epochs: 3
- Temperature: 1

4.4. Setting the Temperature

Temperature is an important factor in knowledge distillation because it is used for calculating the soft predictions from the logits of both the student and the teachers. Usually, temperature is set to 1 as it provides the standard softmax. If temperature is increased, the probability becomes softer and it provides more information as to which target class the teacher found its prediction more closer to. In our experiment setting, we went with a temperature setting of 1 and 2 and

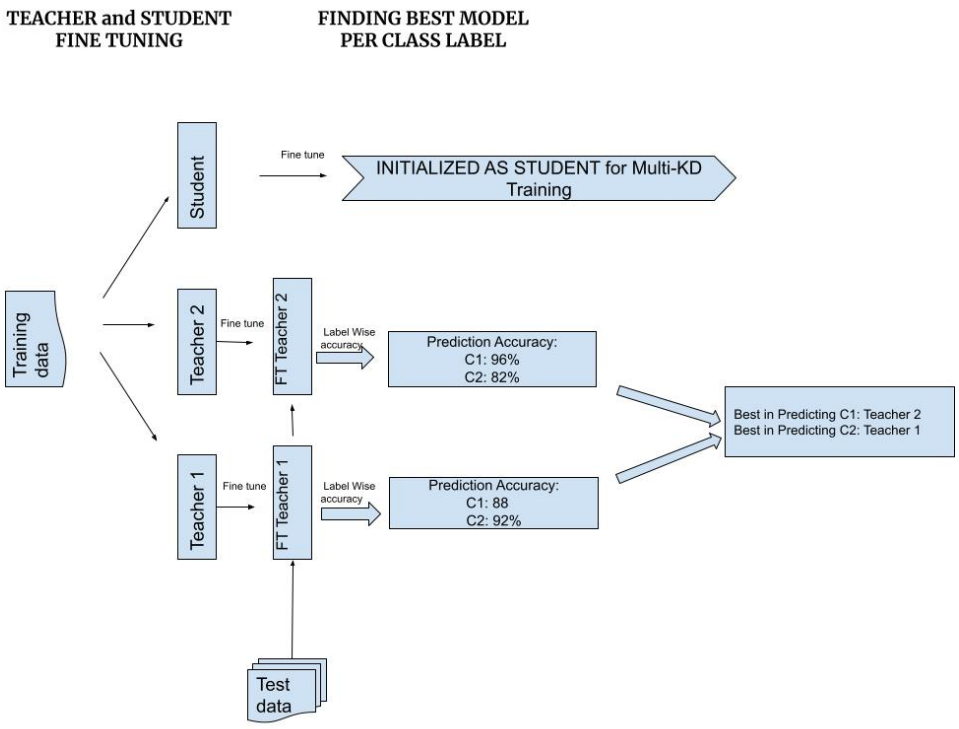

Figure 4.3. Figure explaining how we listed the best model per target class

observed little to no deviation in the reported accuracy and hence we took T=1 since it accounts for a standard softmax. Our future work concerns with properly fine tuning this hyper-parameter so that the distillation training allows the student to go deeper into the understanding of a teacher.

4.5. Find the Teacher BERT Model best in predicting each target class

This specific step takes a trivial yet effective take on the multi-teacher knowledge distillation(MT-KD). Oftentimes, for MT-KD the choice of logits among multiple teachers ends up giving more weight to the teacher whose soft predictions are closer to the ground truth than the other teachers. For the multi-class classification problem we are trying to solve, we decided to not confuse the student at all with a combined output. Combined weighted output is a promising way since it takes a bit of best from all the teachers but we took a more naive approach and see if we get exciting results in a more simplistic experimental setup. So, with the fine-tuned teachers, we calculated the prediction accuracy per class for each teacher model. Referring to Figure 4.3, which is an extension of Figure 4.2, where we see that we fed the test data to the teacher models and have them do the prediction for us. Say, the test set has two target classes C1 and C2 and we are trying to find the answer to this question - ***Is one teacher good at predicting C1 while the other one is good***

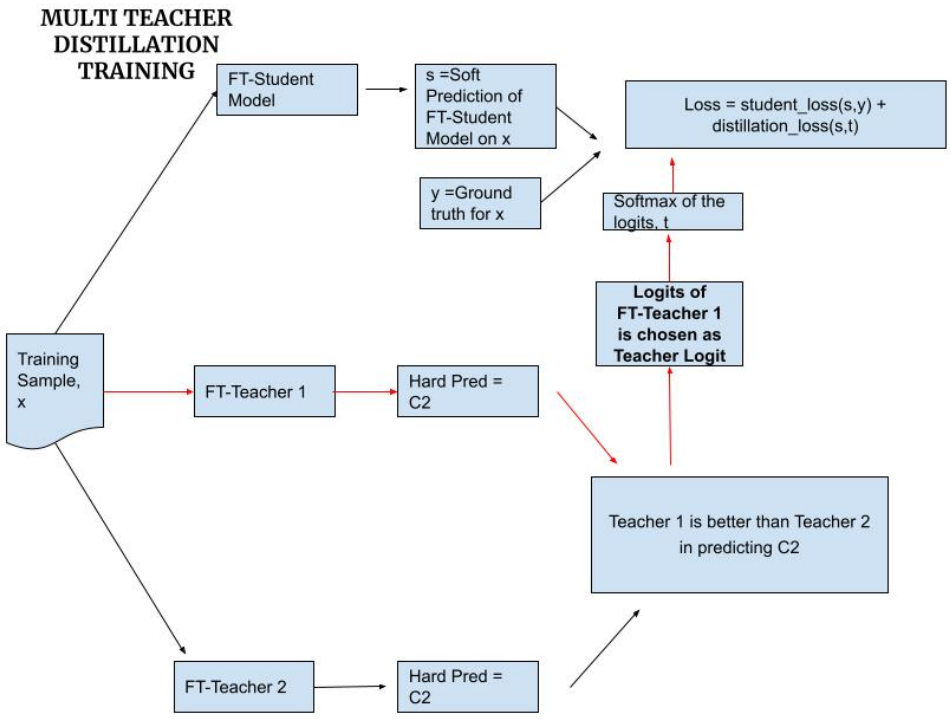

Figure 4.4. Our proposed Distillation Training Flow

at predicting the other class?. The figure shows a hypothetical situation where Teacher 1 is good at predicting C2 while Teacher 2 is good at predicting C1. The output of this step is to have the best model per class. One model can be good in predicting two or more classes and there were cases where we encountered one model that was good in predicting 3/4 classes. If two models have similar accuracy in predicting a certain class, then we took any one of them randomly. The output of the step will be discussed in the chapter *Results and Discussions*.

4.6. Multi Teacher Distillation Training

In this step, the input is the finely tuned student teacher and the set of teacher models which qualified as the best model in predicting all the classes. We used the knowledge we gathered about the teacher's expertise in the previous step and we used that knowledge in the multi-teacher distillation training step. Our multi-teacher distillation approach is based on this notion - *For a training sample, if we find one teacher who comes up with a hard prediction it is best known for,*

then we go blindly with this teacher soft predictions to calculate the distillation loss. Referring to Figure 4.4 and the knowledge we have about the prediction accuracy of the teacher model, below are the steps that explains what happens during the training process -

- At each training step, the fine-tuned student model calculates the soft prediction,s for the training sample x.
- The training sample goes through Teacher 1 and Teacher 2 producing their hard prediction from the logits.
- Both teachers came up with C2 as a hard prediction but Teacher 1 is known to be better than any other teacher in predicting C2.
- The logits of FT-Teacher 1 us chosen as the Teacher Logit and the softmax of the logits is being calculated to prepare the soft prediction, t, of the teacher.
- At this point, we also have the ground truth y for x.
- We used s and t to calculate the distillation function.
- Using s and y we calculated the student loss and used the standard loss function as shown in Figure 4.4 to calculate the distillation loss.

Therefore, our training approach is concerned with hard knowledge while choosing a teacher but takes soft knowledge while calculating the distillation loss function.

It is this loss we tried to minimize while training the neural network for several epochs which was 7 in our case. However, the scenario we showed above is a very best-case scenario and the situation won't be this straightforward all the time. For example, we know if the same situation happens but just with both the teacher predicting C1, then we would choose Teacher 2 for the same reason we chose teacher 1 in the first scenario. Now, let's try to understand what would happen in a scenario where, for a training sample x, **both the teacher ends up predicting what they are best known for predicting?**

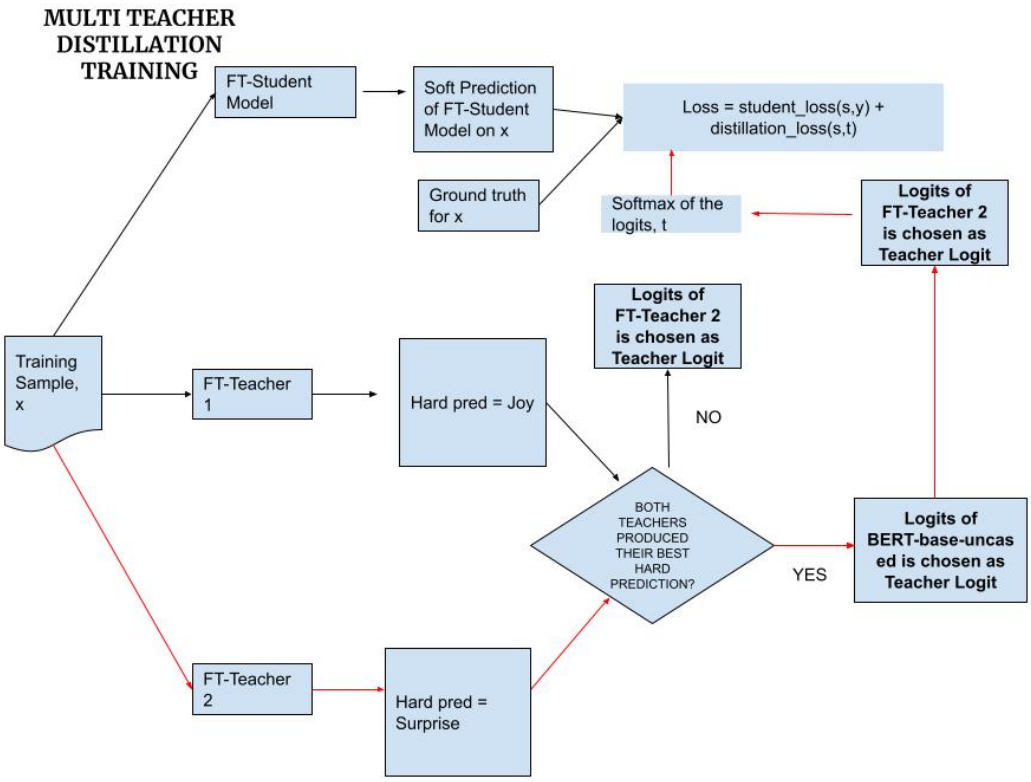

Figure 4.5. Modification to the previous distillation training approach

4.6.1. What happens if both the teacher ends up predicting what they are best known for?

Such a scenario will just pose confusion for the student and we saw that our distillation training approach required a modification so that it provides a generic solution across all scenarios. Figure 4.5 shows how we modified our previous flow to have it adapt to such confusing scenarios. Suppose, FT-Teacher 1 is known as best in predicting if a sentence represents the emotion of *Joy* while FT-Teacher 2 is known for *Happy* or *Surprise*. For a training sample x, FT-Teacher 1 comes up with a hard prediction of *Joy*, and FT-Teacher 2 predicts *Surprise*. To tackle this situation, we also made sure that the training process knows which Fine Tuned Teacher model achieved the highest overall accuracy for this dataset. Therefore, in case of confusion, the student ended up picking the logits produced by that teacher model.

4.7. A general solution for N-teachers with K class labels

However though, what happens if there are more than two teachers and only two of them produced their best hard prediction and the other one did not? We have tried to solve this problem by generalizing our training approach and helping a student find the answer to the question - ***What happens if a student has N teachers to choose from and the total number of target class for the dataset is k?***

- For a training sample, x, $\text{HP}_i = T_i(x)$ where i $\in 1, 2, 3, ..., n$ and n is the total number of teachers and HP denotes the hard prediction produced by the teacher i for the training sample. At each step, we have a set of Teacher-Hard Prediction Pair, $S_{T-HP} = \{T_1HP_1, T_2HP_2..., T_nHP_n\}$.

- From section 4.4, we have domain knowledge about the performance the fine-tuned teachers have on each class label. The output of this step is a teacher model per class label and hence is nothing but a set of Teacher-Hard Prediction pair, $S_{BT-CL} = \{BT_1CL1, BT_2CL_2..., BT_kCL_k\}$ where BT_k qualifies as the best teacher for predicting CL_k.

- This trainer now knows what each Teacher produced as Hard prediction, S_{T-HP}, and best model per class label, S_{BT-CL}. If there is only one pair in $S_{T_HP} \cap S_{BT-CL}$, it means no other teacher produced a hard prediction they are best known for predicting and we proceed with the logit of the teacher we found in $S_{T_HP} \cap S_{BT-CL}$.

- If there is more than one item in the intersection operation, then we know that there are multiple teachers who ended up giving their best hard predictions. This scenario would confuse the student and hence we went with the most accurate fine-tuned teacher BERT model for the dataset .

4.7.1. Example

Let's try to do a small case study to understand the whole workflow-

4.7.1.1. Scenario: Straight Forward

- $S_{T-HP} = \{T_1Apple, T_2Apple, T_3Orange\}$. It means Teacher 1 has a hard prediction of Apple and so on.

- $S_{BT-CL} = \{T_1Apple, T_2Orange, T_3Grass\}$. It means Apple is best predicted by Teacher 1 and so on.

- So, we can see that none of the other teachers except for T1 has predicted anything they are best known for predicting. It means the $S_{T_HP} \cap S_{BT-CL} = \{T_1 Apple\}$.
- The student proceeds with the logits of T1 to produce the soft predictions out of it.

4.7.1.2. Scenario: Confusing

- $S_{T-HP} = \{T_1 Apple, T_2 Orange, T_3 Orange\}$. It means Teacher 1 has a hard prediction of Apple and so on.
- $S_{BT-CL} = \{T_1 Apple, T_2 Orange, T_3 Grass\}$. It means Apple is best predicted by Teacher 1 and so on.
- Let's say M is the most accurate teacher in terms of overall accuracy for the dataset the student is being trained on.
- So, we can see that two of the other teachers have predicted something they are best known for predicting. It means the $S_{T_HP} \cap S_{BT-CL} = \{T_1 Apple, T_2 Orange\}$.
- The student gets confused because the student would only proceed with a corresponding teacher logit if a total item from the resultant set from the intersection operation is 1. Therefore, the Student proceeds with the logits from M.

In this way, our training approach was generalized across n teachers for k class labels and was able to avoid any confusing scenarios for the student.

4.8. Multi Teacher Distillation Using Average of Teacher Logits

In addition to the approach we took for multi-teacher distillation, we set up another experiment to compare if doing an average of teacher logit would create a student that would perform close or better than the teachers in predicting the class labels. Figure 4.6 shows the overview of the distillation training approach. Here, every step is similar except it has no end cases like the previous one and is a straightforward average of the logits. We performed this approach on the same datasets which we used in the previous experiment. The results of these two experiments will be discussed in chapter 5.

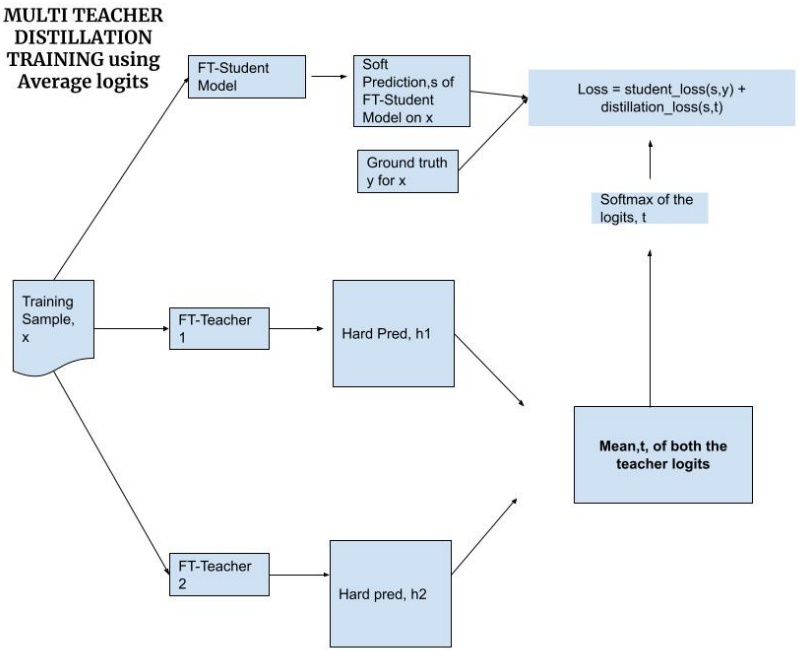

Figure 4.6. Distillation Training using Average of Teacher logits

4.9. Baseline: Random Teacher Selection

For each training sample, the student chooses a random teacher among the set of all the teachers we have used in our study which are fine tuned version of - DeBERTa base, RoBERTa base and BERT base uncased.

4.10. Comparing performance of student and teacher model in predicting each class

At the end of distillation training for each dataset, we have the student model, distilled from the teachers, and the fine-tuned teacher models. To evaluate the result, we decided to make a fair comparison by using the same test data on the teacher and student model and find the per-class accuracy. This is the step that produces all the results of the experiments above which we will shortly discuss in Chapter 5.

4.11. Additional Experiment with a complex Clinical Dataset

As an additional standalone experiment, we tried to explore the potential of Multi-Teacher Knowledge Distillation on another type of Natural Language Processing task which is analogous to Named Entity Recognition and Question Answering.

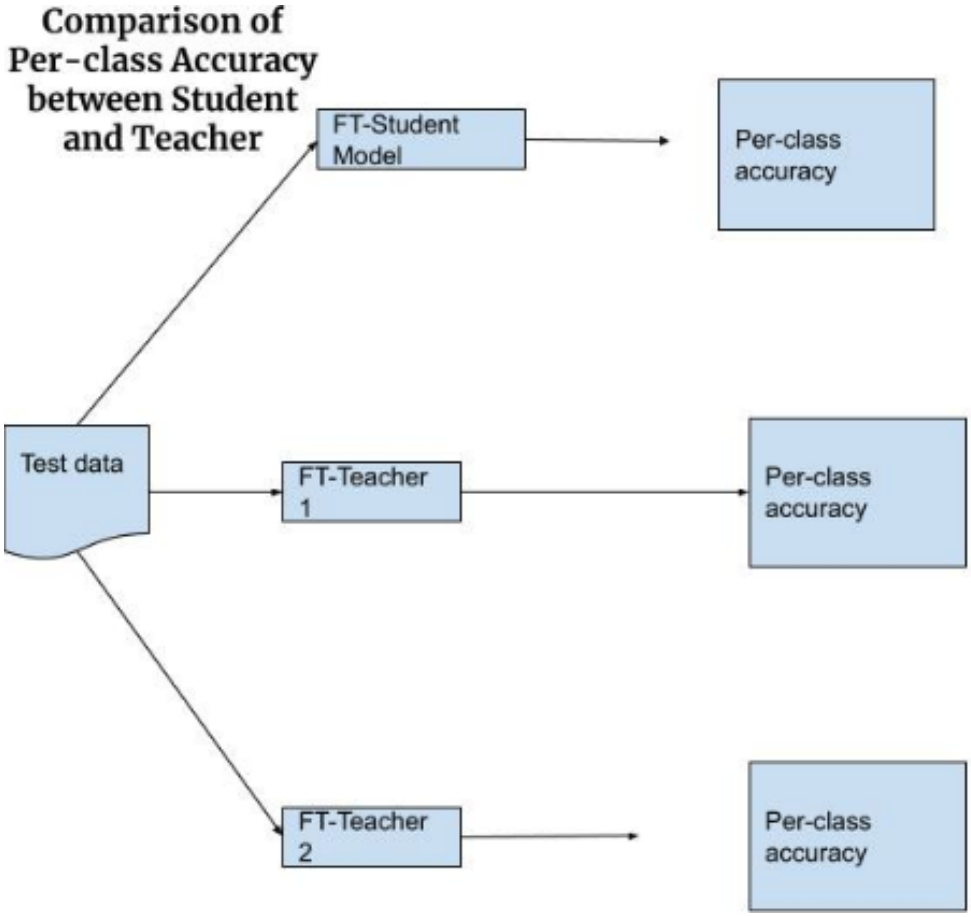

Figure 4.7. Workflow that compares the performance of student and teacher

Skills examination is one component of the United States Medical Licensing Examination® (USMLE®). It is conducted by the National Board of Medical Examiners (NBME).

Problem Statement: Can we identify key clinical phrases in patient notes and automate the Step 2 Clinical Skill Examination Scoring method?

4.11.1. What is this exam?

- Exam to test how good a clinical note a physician is taking a while "talking directly" to a patient.
- "Good" - More the number of clinical phrases, the better the clinical note.
- Test takers talk to standardized patients and take notes.
- Experienced and licensed physicians score the clinical notes. More clinical terms (given that they are meaningful) mean a better score.
- Scoring is a time-intensive and expensive process in terms of money. The goal of the final model would be to **automate the clinical note-scoring process by identifying key phrases in the patient notes.**

However, though, the main research question we tried to answer with this dataset is - **Can we combine a general BERT and a clinical BERT variant and use their knowledge to build a smaller student model that performs close or better than the teacher models?**

4.11.2. Sample Input and Output

Input would consist the set of following items -

- **HPI**: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest." 2 days ago during a soccer game had an episode, but this time had chest pressure and felt as if he were going to pass out (did not lose conciousness). Of note patient endorses abusing adderall, primarily to study (1-3 times per week). Before recent soccer game, took adderrall night before and morning of game. Denies shortness of breath, diaphoresis, fevers, chills, headache, fatigue, changes in sleep, changes in vision/hearing, abdominal paun, changes in bowel or urinary habits.
- **PMHx**: none

- **Rx**: uses friends adderrall
- **FHx**: mom with "thyroid disease," dad with recent heart attcak
- **All**: none
- **Immunizations**: up to date
- **SHx**: Freshmen in college. Endorses 3-4 drinks 3 nights / week (on weekends), denies tabacco, endorses trying marijuana. Sexually active with girlfriend x 1 year, uses condoms.

Output for the aforementioned input would be the phrases which are related to terms like **myocardial diseases and thyroid** : mom **with "thyroid disease," dad with recent heart attcak**

4.11.3. Experiment Design

In this task, we also applied a task-specific knowledge distillation approach. We fine-tuned DeBERTa large (24 layers), BioMedNLP BERT, and Roberta large(24 layers) with the dataset. We observed, as we will see more thoroughly in the Result chapter of this report, that DeBERTa performed best among all the variants in identifying the clinical phrases correctly. Hence, we took DeBERTa as a teacher model and in addition, we incorporated a clinical BERT called BiomedNLP-PubmedBERT. As a student, we used a BERT-base-uncased we found in the huggingface model library. We performed three experiments-

- *Experiment 1*: In distillation training, we took the average of logits produced by both the teachers and proceeded with the standard distillation loss function.
- *Experiment 2*: We repeated the distillation training but used the confidence aware weighted loss as distillation loss function [27]. It is important to note that we have not used the output from intermediate layers of the teachers and it is only the *MT-Dist* we have used from the aforementioned work.
- *Experiment 3*: We repeated the distillation training but this time we added teacher loss in addition to student loss and distillation loss. For distillation loss, we used the average logits approach.

- We recorded the F1 score of the student and teachers.

We repeated all the experiments but with a different student, the DistilBERT base cased which is already fine-tuned with the SQuAD dataset. We performed this experiment to see if involving a student which is already fine-tuned on a Question Answer dataset improves the F1 score of the student since our prediction task is analogous to the Question Answering task.

In chapter 5, we will extend this discussion in terms of results we have found from all the experiments we have discussed in this chapter.

5. RESULTS AND DISCUSSION

Table 5.1. Accuracy (in percentage) of the Student and Teacher Models on the validation set

Dataset	**FT Teachers**			**Student**		
	BERT-base-uncased-12	**RoBERT base 12**	**DeBERT base 12**	**Our Approach**	**Average Logits**	**Baseline**
Emotion	94.3	93.6	94.4	94.2	93.2	91
MIND	92.3	88.8	90.6	91.6	81.9	85.5
Twitter-Emotion	79.1	74.3	76.5	75.4	73.4	72.7
Twitter-Sentiment	69.2	66.4	45.1	67.3	67.5	64.9

For reference,

- ***FT Teachers*** refers to Fine Tuned Teachers
- ***Average Logits*** refers to student trained taking average of all teacher logits.
- ***Our Approach*** refers to student trained using our proposed approach.
- ***Baseline*** refers to student which have trained by using a random teacher at each training step. We used this as a baseline for our study.

Table 5.1 shows the accuracy of the student and the teacher models on the validation set.It was recorded while fine-tuning the teacher models on the dataset and the student models were being trained using multi-teacher knowledge distillation. The reported accuracy comes from the same validation set for each of the models mentioned above. We could make the following observations -

- For students trained using our approach for the EMOTION dataset, it was 0.1% less accurate than its teacher BERT-base-uncased and 0.2% less accurate than DeBERTa. Interestingly, our student ended up being 0.6% more accurate than RoBERTa. The student trained using KD and average logits of multiple teachers, it was 1.1%, 0.4%, and 1.2% less accurate than BERT-base-uncased, RoBERTa, and DeBERTa respectively.

- For the MIND dataset, a student trained using our approach turned out to be 2.8% more accurate than RoBERTa and 1% more accurate than DeBERTa while it was 0.7% less accurate than BERT-base-uncased. The student trained using KD and average logits of multiple teachers performed very poorly in this scenario. It was 10.4% less accurate than BERT-base-uncased, 6.9% less accurate than RoBERTa, and 8.7% less accurate than DeBERTa.

- For the Twitter-Emotion dataset, our student performed 1.1% better than RoBERTa while it was 3.7% and 1.1% less accurate than BERT-base-uncased and DeBERTa respectively. The student trained using KD and average logits of multiple teachers under-performed in this scenario as well. It was 5.7% less accurate than BERT-base-uncased while it was 0.9% and 3.1% less accurate than RoBERTa and DeBERTa respectively.

- For the Twitter-Sentiment dataset, we excluded DeBERTa from the discussion because it was not performing at all even during the training improving very little per epoch. Therefore, we excluded DeBERTa as a teacher in both the multi-teacher distillation experiment. Our student performed 0.9% more accurately than RoBERTa while being 1.9% less accurate than BERT-base-uncased. The other student trained using the average of logits of the teachers performed 1.1% more accurately than RoBERTa and 1.7% less accurately than BERT-base-uncased.

- One interesting observation was, the student could not surpass BERT-base-uncased in any of the instances under discussion. Also, BERT-base-uncased was the best teacher model in almost all the cases except for the EMOTION dataset which gave us the confidence to pick it as a go-to model to pick logit from when the student is confused.

- Students trained using MT-KD and average logit rarely outperformed any teacher.

- Our baseline, RT, was not able to outperform any of its teachers for any dataset. For sentiment analysis, we did not use DeBERTa in distillation training because the accuracy of fine tuned teacher was overwhelmingly under-performing which could be seen during per class accuracy.

Now, for the major part of the rest of the chapter, we will discuss the per-class accuracy analysis we did. It is not fair to weigh the capability of the student model just by using overall accuracy. For this reason, we tried to explore the potential of the student and answer the research

question by dissecting how the student compares to its teacher models in terms of predicting each label. The accuracy reported above is on the validation set during the training process. For per-class accuracy, we picked the best checkpoint of each model and ran it through the test set. This analysis is what the output of the last step of our methodology is comprised of which we discussed in the previous chapter. For the rest part of this chapter, we will refer to the student which was trained to take the average of teacher logits as *Student-Avg*. For each dataset, we will try to report what our research study found for the student and the teacher models. The discussion for each dataset will be divided into two sections-

- Comparison of Our Student vs Student-Average in predicting each target class.
- Quantifying the overall performance of our students with the teacher.
- Comparison of Our Student and Student-Average with each teacher model.

Before proceeding, it is worth mentioning FT-MODEL-12 refers to the fine-tuned teacher which has 12 encoder layers. Also, the students we trained are with 6 encoder layers.

5.1. EMOTION

Table 5.2. Per class accuracy for the Teacher and Student Models on EMOTION Dataset

FT Teachers				**Student**		
Class	**BERT-base-uncased-12**	**Roberta-base-12**	**Deberta-base-12**	**Our approach**	**Average Logits**	**Baseline**
Sadness	96.7	95.3	96	97.4	94.4	95.2
Joy	95.6	97.6	98.5	95.4	98.1	96.2
Love	79.8	74.2	69.1	84.3	66.7	64.2
Anger	94.1	98.5	96.3	94.9	96.4	95.6
Fear	84.3	80.3	88.8	84.9	92.4	94.2
Surprise	71.2	78.7	65.2	71.2	53	21.2

In our methodology, we had a crucial step where we formalized the best teacher model in predicting each class label as S_{BT-CL}. While training the model with this dataset, the student model has access to this set which tells the student which teacher to consult with for soft prediction if it encounters a hard prediction. **Comparison of Our Student and Student-Average with**

Table 5.3. Median of difference in per class accuracy from the teachers

	Teachers		
Student	**BERT-base-uncased-12**	**Roberta-base-12**	**Deberta-base-12**
Our Approach	+0.65	-0.05	0
Average Logits	0	-1.5	-1
Baseline	-0.45	-2.15	-1.55

each teacher model: Table 5.2 shows the per-class accuracy for the Teacher and Student Models over the test set of the EMOTION dataset. The teacher models seen in the table are the set of best models that we found from the analysis. To have a general sense of the result, we can see in Figures 5.1, 5.2, and 5.3 that Our Student outperformed all the three models on *Sadness* by achieving accuracy which is 0.7%, 2.1% and 1.4% better than BERT-base, RoBERTa, and DeBERTa respectively. It also did the same on *Love* by achieving accuracy which is 4.5% better than BERT-base, 10.1% better than RoBERTa, and 15.2% better than DeBERTa. It was able to outperform BERT-base and RoBERTa on *Fear* while it outperformed DeBERTa on *Surprise* by achieving 6% better accuracy. In the meantime, except for *Fear*, the Student-Average could not outperform the teacher for any instance. It fluctuated throughout predicting on the test set by reaching as high as 12.1% better than RoBERTa for *Fear* while going down as bad as 25.7% worse than RoBERTa on *Surprise*.

Looking at the same figures, we can see the fluctuation of the Student-Average from all its teachers, unlike our student. Taking average logits for multi-classification problems like this rather confused the student. Figure 5.2 shows that our Student was able to score better or equal accuracy than BERT in predicting the target class. The average Student was not able to do so. Figure 5.2 shows the accuracy of our students was very close to the DeBERTa for most of the target classes. In none of the cases, our student performed worse than the Student-Average. In figure 5.3, we can see that our Student was very close to the teacher RoBERTa but in the case of Surprise, both the students failed to be close enough to the teacher it can be seen that in none of the cases our student was worse than the Student Average.

Similarly, when student picked a teacher randomly(baseline), it was still not able to outperform any teacher for most of the target classes except for Fear. It performed overwhelmingly bad for the class *Surprise*.

Quantifying the overall performance of our student with the teacher : We know how if our student was performing better than the teacher in predicting each class or not. However, it is important to come up with a quantity that compares how accurate a student is to the teacher in predicting all the class labels. In figure 5.1, we can see the difference in accuracy between the student and teacher BERT-base uncased in predicting each class label. For each student, we will have k differences where k is the number of class labels. We then calculated the median of the differences in accuracy so that we can have a quantifiable value that characterizes how our students are performing across k class labels in general. A positive value of X% would mean that the student has in general achieved an accuracy that is X% better than the teacher. Table 5.3 shows the median values which say that Our Student is 0.65% more accurate than its teacher BERT-base-uncased across all class labels while it is 0.05% worse than FT-RoBERTa and no difference from DeBERTa. On the other hand, the Student-Average was not able to be better than the teacher in any of the cases. Similarly, it could be seen that baseline student was not able to perform better than any of its teacher in terms of median of difference in per class accuracy.

5.2. MIND

Table 5.4. Per class accuracy for the Teacher and Student Models on EMOTION Dataset

FT Teachers			**Student**		
Class	**BERT-base-uncased-12**	**Roberta-base-12**	**Our approach**	**Average Logits**	**Baseline**
Food & Drink	96.8	94.7	97.2	92.7	93.7
Lifestyle	94.2	91.4	95.5	90.5	90.2
Video	95.7	93.8	96.4	69.6	90.7
Travel	92.8	91.6	95.1	85.7	88.1
Finance	94.9	95.3	96.7	91.3	93.4

Table 5.5. Median of difference in per class accuracy from the teachers

	FT Teachers	
Student	**BERT-base-uncased-12**	**Roberta-base-12**
Our Approach	+1.3	+2.6
Average Logits	-4.1	-4
Baseline	-4	-1.9

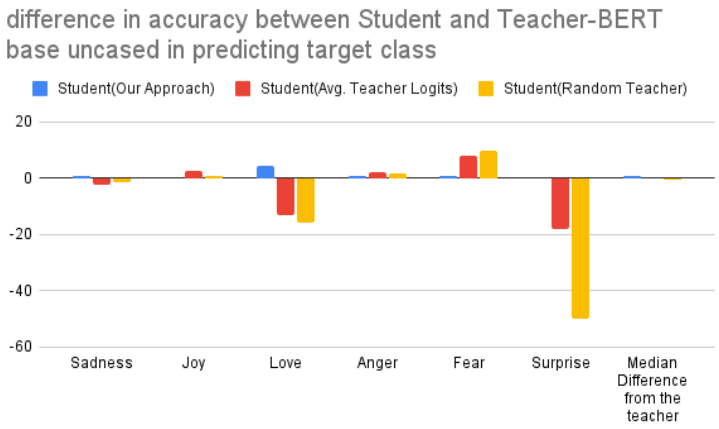

Figure 5.1. Comparison on Students with BERT-base-uncased

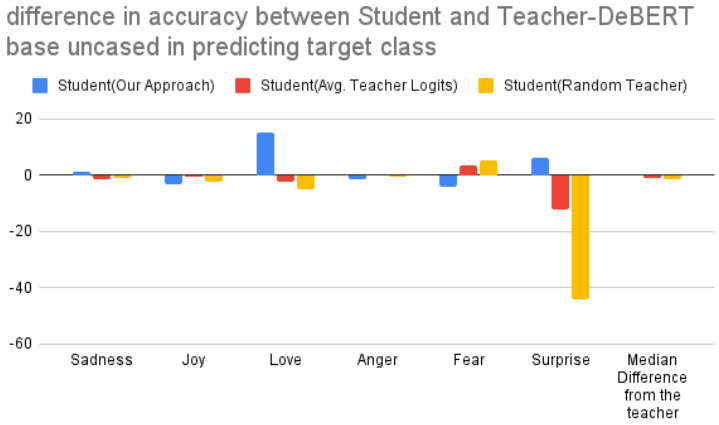

Figure 5.2. Comparison on Students with DeBERTa

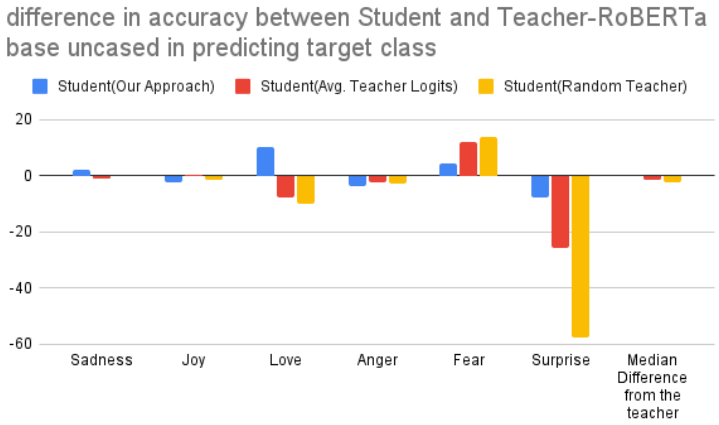

Figure 5.3. Comparison on Students with RoBERTa

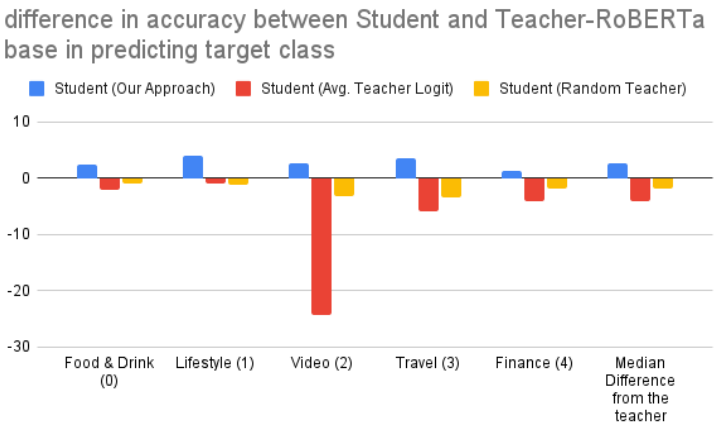

Figure 5.4. Comparison on Students with BERT-base uncased

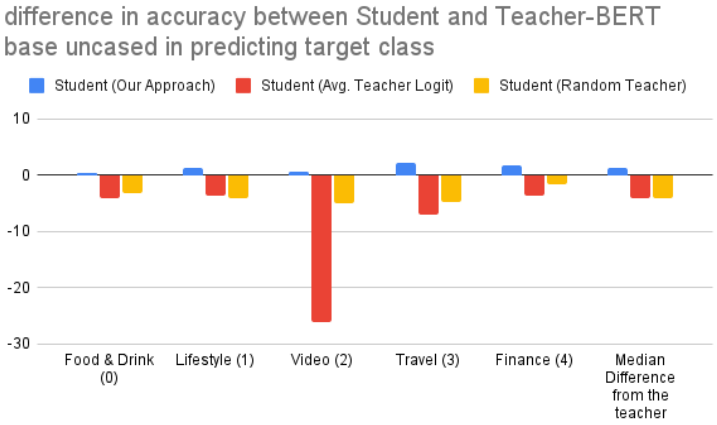

Figure 5.5. Comparison on Students with RoBERTa

In Table 5.2, Figures 5.4 and 5.5 we can see that our Student outperforms all the teachers in predicting all the class labels. Student-Average could not outperform any of the teachers in any instance. Student-Average and Baseline Student were not able to outperform our Student in any of the class labels.

Referring to Figure 5.4 and 5.5, we can see that our student is +1.3% better than with BERT-base-uncased in predicting all the class labels while it is 2.6% better than RoBERTa. On the other hand, Student-Average is 4% worse than BERT-base-uncased while it is 4.1% worse than RoBERTa. Student which picked teacher at random (baseline) was also worse than all of its teacher in terms of median of difference in per class accuracy.

5.3. Twitter-Emotion

Table 5.6. Per class accuracy for the Teacher and Student Models on Twitter-Emotion Dataset

FT Teachers				**Student**		
Class	**Bert-base-uncased-12**	**Roberta-base-12**	**Deberta-base-12**	**Our approach**	**Average**	**Baseline**
Sadness	92.1	91.2	85.5	90.9	87.8	87.8
Joy	76.8	75.1	79.3	77.9	81.2	79.6
Anger	52	44.7	44.7	47.1	17.1	20.3
Optimism	58.9	75.1	75.7	63.4	63.1	63

Table 5.7. Median of difference in per class accuracy from the teachers

	FT Teachers		
Student	**Bert-base-uncased-12**	**Roberta-base-12**	**Deberta-base-12**
Our Approach	-0.05	+1.05	+0.5
Average Logits	-0.05	-7.7	-5.35
Baseline	-0.75	-7.75	-6.2

As we can see in Table 5.6 accompanied by Figures 5.6, 5.7, and 5.9, Our student was 5.4% better than DeBERTa for *Sadness*, 1.1% and 2.8% better than BERT-base and RoBERTa respectively for *Joy*, 2.4% better than both the DeBERTa and RoBERTa for *Anger* and 4.5% more accurate than BERT-base for *Optimism*. Our Student performed significantly worse than RoBERTa and DeBERTa in predicting *Optimism*. On the other hand, Student-Average outperformed all the teachers in predicting *Joy* but could not surpass in any other instance except for *Sadness*, 2.3% better than DeBERTa, and *Optimism*, 4.2 % better than BERT-base-uncased. The Student-Average was showing little potential in this dataset but it performed underwhelmingly badly in predicting Anger which reduced its overall performance. The baseline, on the other hand, performed better than all the teachers in predicting *Joy* while it was underwhelming in predicting *Anger*.

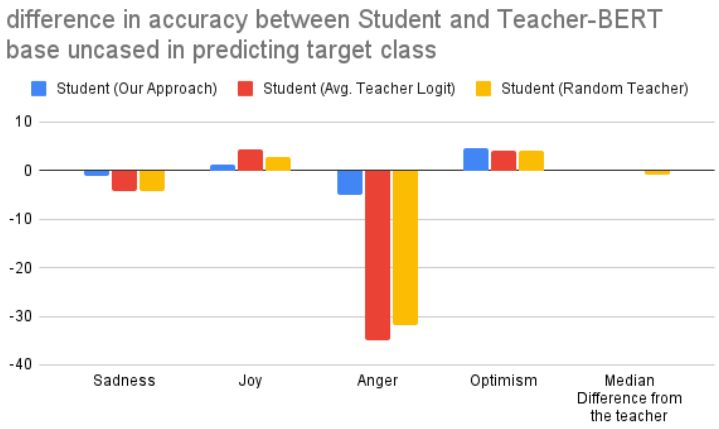

Figure 5.6. Comparison on Students with BERT-base-uncased

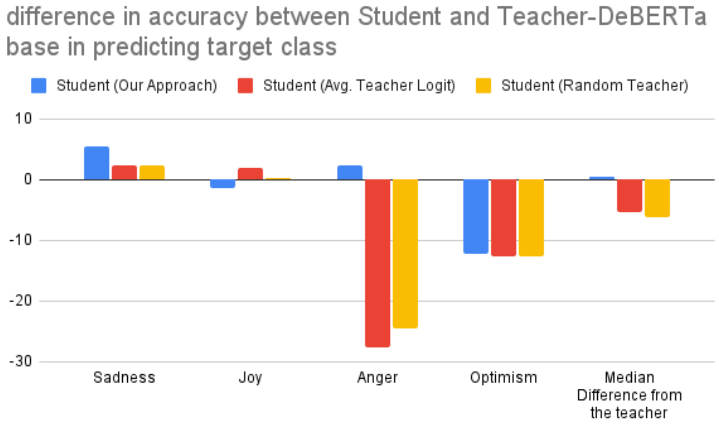

Figure 5.7. Comparison on Students with DeBERTa

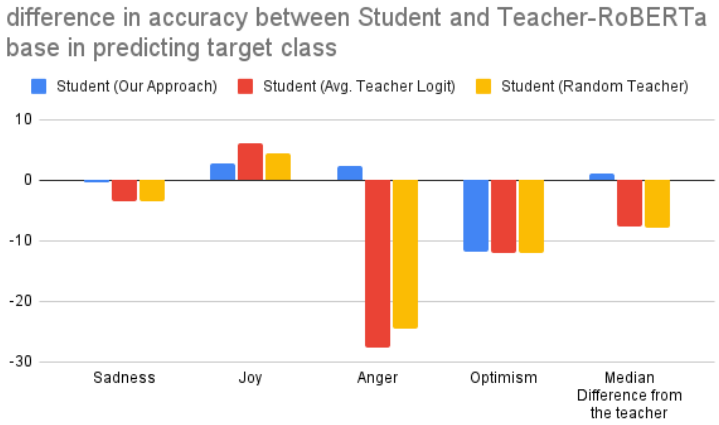

Figure 5.8. Comparison on Students with RoBERTa

Quantifying the overall performance of our student with the teacher : Referring to Table 5.7- we see that our student is 0.05% worse than BERT-base-uncased while it is 1.05% better than RoBERTa and 0.5% better than DeBERTa. Student Average struggled in this experiment and its median accuracy difference is less than all its teachers. For baseline, it could be observed that the behaviour of this student is similar to the average logits and it deviated and ended up becoming significantly under-performing than RoBERTa and DeBERTa. Our student, on the other hand, was very close to bert-base-uncased while outperforming the rest of the two teachers.

Comparison of Our Student and Student-Average on test data : Except for *Joy*, Our Student performed better than the Student-Average in predicting the class label accurately.

5.4. Twitter-Sentiment

Table 5.8. Per class accuracy for the Teacher and Student Models on Twitter-Sentiment Dataset

FT Teachers			**Student**		
Class	**Bert-base-uncased-12**	**Roberta-base-12**	**Our approach**	**Average**	**Baseline**
Positive	76.3	57.7	68.1	39.4	14.5
Negative	51.5	66.3	55.5	73.9	86.3
Neutral	65.4	68.22	65.7	67	57.7

Table 5.9. Median of difference in per class accuracy from the teachers

	FT Teachers	
Student	**Bert-base-uncased-12**	**Roberta-base-12**
Our Approach	+0.3	-2.52
Average Logit	+1.6	-1.22
Baseline	-7.7	-10.52

Interestingly enough, there is no scenario where our Student outperformed all the teachers. Student-Average on the other hand outperforms all the teachers in predicting Negative tweets. It outperforms BERT-base by 22.4%. However, it performs worse than all the teachers in predicting positive tweets where our Student performs better than RoBERTa by 10.4%. Our Student achieves an accuracy that is 4% better than BERT-base on *Negative* Tweets and 0.3% better than BERT-base for the neutral tweet. Basline, on the other hand, did not outperform its teachers except

in predicting *Negative*. Its performance was worse than its teachers and also other students in predicting the other target classes.

If we refer to Table 5.9, Student-Average has a median accuracy of 1.6% better than BERT-base and 1.22% worse than RoBERTa where Our Student has a median accuracy of 0.3% better than BERT-base and 2.52% worse than RoBERTa making the Student-Average performing generally better than student trained using our approach. One of the important takeaways is- our student encountered no scenario where it performed worse than all the teachers. Baseline under-performed than all its teacher and the other type of students were able to perform better than the baseline.

5.5. In Multi-Teacher Distillation, can a student outperform its teachers given it knows exactly which teacher to follow depending on the training sample?

In most of the cases, Students trained using our approach were achieving accuracy which was in the positive direction that is - better than its teachers. After trying to generalize **the difference between the teacher and student accuracy for each label** into one median value, we could see that the student trained using our approach performed close or better than their teacher for most of the cases. Hence it shows the promise of the approach we took for training a lightweight student by distilling multiple teachers and their domain expertise. One important thing to notice is - our student was able to be better than **Baseline** and **Average Logits** and in terms of median of difference in accuracy from all its teachers for all of the dataset. It shows that it is always handy for a student to know which teacher to incline towards depending on the specific problem it is exposed with rather than randomly selecting a teacher or taking mean capability from all the teachers.

5.6. Results for Additional Experiment on NBME Clinical Patient Notes

As mentioned in the previous chapter, we call this an additional effort because this problem has been dealt with with a different approach and is not a multi-class classification problem. The idea is to combine a general BERT with a Clinical BERT variant and see if we can build a lighter student which performs at least close to the teacher model.

Table 5.6 shows the results we got from fine-tuning various BERT models on the NBME dataset. Among the two clinical variants of BERTs we used, biomednlp-pubmedbert-base-uncased, 12 layers with 110M parameters, scored an F1 of 0.82 whereas a general language model deberta-v3-large, 24 encoder layers with 330M parameters, scored a F1 score of 0.88 which is the best among

Table 5.10. Results of fine tuning BERT models on NBME Dataset

BERT Variant Fine Tuned	**F1**
BERT-Base-Uncased	0.74
Bioclinical BERT	0.80
biomednlp-pubmedbert-base-uncased	0.82
roberta-large	0.83
debarta-v3-large	0.88

all the experiments we did on the dataset. Hence, for the multi-teacher distillation, we picked these two models as teachers. As students, we picked BERT-base and DistilBERT-base-uncased-SQuAD.

Table 5.11. Comparison of F1 for student BERT models distilled from mix of general BERT model and Clinical Model

Teacher	**Student**	**Teacher Knowledge**	**Loss Function**	**F1**
Deberta-v3-large-finetuned + biomednlp-finetuned	Bert Base	Average Logit	Student loss + Distillation loss	0.78
Deberta-v3-large-finetuned + biomednlp-finetuned	Bert Base	Average Logit	Student Loss + CA-MKD	0.74
Deberta-v3-large-finetuned + biomednlp-finetuned	Distillbert fine tuned with SQUAD data	Average Logit	Student loss + Distillation loss	0.82
Deberta-v3-large-finetuned + biomednlp-finetuned	Distillbert fine tuned with SQUAD data	Average Logit	Student Loss + CA-MKD	0.81

Table 5.11 shows the result from the four experiments we did with Multi Teacher Knowledge Distillation on this dataset. For all the cases, we took the Average of all teacher logits as the teacher knowledge. With each student, we just changed the definition of the loss function and recorded the f1 score. When we used BERT base as the student, using the basic loss function yielded a better f1 score, 0.78, than using CA-MKD which produced 0.74. Interestingly enough, when we changed the student to Distillbert fine-tuned with SQUAD data, the overall f1 improved. When we used the basic loss function, the f1 was 0.82. However, even with this student, the f1 score with CA-MKD is 0.81. DistilBERT when fine-tuned with SQuAD dataset and used as a student yielded an f1 score which is better than what BERT-base could yield as a student. It shows that it is important to strategize the task-specific knowledge distillation by using a proper student and fine-tuning a dataset that suits the new task. Also, the new smaller model performs similarly to the fine-tuned *biomednlp-pubmedbert-base-uncased* but with a lighter architecture.

6. CONCLUSION

In the areas of natural language processing and computer vision, deep neural networks have become increasingly popular. To make neural network models as effective as possible, research was done to develop sophisticated architectural designs. The only option to take advantage of the intricate architecture and make the model as accurate as possible was to create it from scratch, but this had certain drawbacks. If people had access to powerful computing devices, such as high GPU machines, they might be able to help with good models. Pre-trained models became more popular as a result, both in the NLP and computer vision fields. Pretrained learning models are even more well-liked because they are open-sourced for the research community. Like other inventions, it comes with a drawback, namely that it couldn't be used in systems with low levels of computing and storage capacity. Huge models might be compressed using ideas like knowledge distillation while maintaining performance levels that were very near to the large model. In this study, we attempted to investigate Multi-Teacher Information Distillation in the area of NLP utilizing Large BERT models. Knowledge distillation is a teacher-student knowledge exchange strategy. We used four different datasets for NLP tasks like knowledge inference. We used a task-specific knowledge distillation technique using DeBERTa, RoBERTa, and BERT-base uncased teacher models, which required us to fine-tune the teacher models using the datasets. Then, we suggested a distillation training strategy that -

- Lists the best model suited to predict each class label. We saw that we now had access to teachers' expertise on a deeper level.
- We saw that the distillation training allowed the student to pick the soft prediction from the teacher who is best suited to provide the prediction on the training sample.
- We also tried to extend our solution so that it works for N teachers on a dataset with K labels.
- We also fed the student and the teachers to do predictions on the same test set and calculated the per-class prediction accuracy.

We saw that our approach showed high promise in multiple datasets like MIND and EMOTION. In MIND, a student trained using our approach outperformed all the teachers in predicting all the class labels. In terms of median accuracy difference, Our Student achieved an accuracy of 1.3% better than its teacher BERT and 2.6% better than its teacher RoBERTa. For the EMOTION dataset, the median accuracy difference of Our Student is seen to be 0.64% better than its teacher BERT-base uncased while 0% for DeBERTa which means the student performed similarly to how DeBERTa performed. We also observed that for the Twitter Emotion dataset, our student's accuracy was 1.05% better than its teacher RoBERTa and 0.5% better than its teacher DeBERTa.

The overall performance of the student trained using our approach shows that it is possible to incorporate multiple large teachers with 12 encoder layers and 110M paratmeters to distill to a model with 6 layers and 66M parameters and still be close to the teacher models in terms of overall accuracy and outperform the teachers while looking at per-class accuracy.

In our additional experiment with clinical dataset, we achieved some promising results. We fine tuned the a very large variant of DeBERTa, 24 layers of encoder and 303M parameters, and achieved f1 score of 0.88 while fine tuning biomednlp-pubmedbert-base-uncased, 12 layers of encoder with 110M parameters, gave us a f1 score of 0.82. We tried to compress these huge models by taking average of logits of these two teacher models. Using BERT-base as a student gave us f1 of 0.78 while using even a lighter student, DistilBERT which is 6 layers of encoders, 66M parameters and is fine tuned with SQuAD dataset, gave us an impressive f1 score of 0.82 which is similar to the performance of a fine tuned biomednlp-pubmedbert-base-uncased.

6.1. Limitations

Our approach is not without limitations-

- The scope of our research study is heavily directed toward multi class classification problem because the expert models are bounded by some finite number of target class they are known to predict. If we extend this approach to solve QA based task/NER tasks which is more like token based prediction and hence would fit inappropriate to our approach since a model now has to have a huge domain knowledge.

- There might be scenarios where same teacher ends up being the best teacher for predicting all the k labels and N-1 teachers are not used in the distillation training method at all. It will then be nothing but a single teacher distillation. It is expected to always look for diverse teachers and have at least two teachers in the set S_{BT-CL}.

6.2. Future Works

The development of a pipeline that automatically adjusts the temperature and alpha constant parameters of the loss function is one path in which we would like to take this research project. Adding a new step to our training would include changing these parameters in order to get the ideal alpha and temperature values for the total loss function. Utilizing different ensemble models might be a worthwhile additional experiment to conduct. In other words, the student would use the combined expertise rather than relying solely on Teacher A's ability to forecast class C. For example - *For class c, take a% from Teacher A and b% from Teacher B would be the the piece of information the student will seek from* S_{BT-CL}. Learning from a more seasoned student instead of constantly relying on the large instructor model is another intriguing feature that might be worthwhile to attempt. This, in our opinion, would cut the training time even more. Incorporating the results of the instructor encoders' intermediary layers while computing cross entropy is another broad area of research. The reason for this is that output from the intermediary layers often contains useful information on how the model is appropriately interpreting the context of each word in a phrase. Since this information is more akin to understanding how a teacher actually learns what it learns, it may be helpful to the student. A developing area of study in both computer vision and natural language processing is knowledge distillation. By observing how the instructor reacts to the same training sample and adjusting the weights and biases by measuring the loss, it enables taking advantage of big pre-trained models and training a new, smaller model from scratch or pre-trained models. Our strategy proved to be effective since it allowed us to attain results that were previously only possible with models with enormous 110M or 303M trainable parameters. The promise of knowledge distillation is frequently insufficiently supported by examining only F1 scores or total validation accuracy. We took note of the overall accuracy, probed further to look at per-class accuracy, and discovered that the lighter models are operating more effectively than it meets the eye.

REFERENCES

[1] Defang Chen, Jian-Ping Mei, Yuan Zhang, Can Wang, Zhe Wang, Yan Feng, and Chun Chen. Cross-layer distillation with semantic calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7028–7036, 2021.

[2] Andrew M. Dai and Quoc V. Le. Semi-supervised sequence learning. In *NIPS*, 2015.

[3] Xiang Deng and Zhongfei Zhang. Can students outperform teachers in knowledge distillation based model compression? 2020.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] Takashi Fukuda, Masayuki Suzuki, Gakuto Kurata, Samuel Thomas, Jia Cui, and Bhuvana Ramabhadran. Efficient knowledge distillation from an ensemble of teachers. In *Interspeech*, pages 3697–3701, 2017.

[6] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[7] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23, 2021.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

[10] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[11] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[12] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*, 2019.

[13] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[14] Kisoo Kwon, Hwidong Na, Hoshik Lee, and Nam Soo Kim. Adaptive knowledge distillation based on entropy. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7409–7413. IEEE, 2020.

[15] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[16] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[17] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. 2018.

[18] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

[19] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[20] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[21] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[22] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*, 2020.

[23] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

[24] Meng-Chieh Wu, Ching-Te Chiu, and Kun-Hsuan Wu. Multi-teacher knowledge distillation for compressed video action recognition on deep neural networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2202–2206. IEEE, 2019.

[25] Shan You, Chang Xu, Chao Xu, and Dacheng Tao. Learning from multiple teacher networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1285–1294, 2017.

[26] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[27] Hailin Zhang, Defang Chen, and Can Wang. Confidence-aware multi-teacher knowledge distillation. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4498–4502. IEEE, 2022.