Exploration of Knowledge Distillation Methods on Transformer Language Models for Sentiment Analysis

Haonan Liu

School of Science

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 04.11.2022

Supervisor

Professor Saikat Chateerje
Senior University Lecturer
Wilhelmiina Hämäläinen

Advisors

Anubhab Ghosh

Michael Troung

Aalto University
School of Science

Copyright © 2022 Haonan Liu

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Haonan Liu

Title Exploration of Knowledge Distillation Methods on Transformer Language Models for Sentiment Analysis

Degree programme ICT Innovation

Major Data Science **Code of major** SCI3095

Supervisor Professor Saikat Chateerje
Senior University Lecturer Wilhelmiina Hämäläinen

Advisors Anubhab Ghosh, Michael Troung

Date 04.11.2022 **Number of pages** 40 **Language** English

Abstract

Despite the outstanding performances of the large Transformer-based language models, it proposes a challenge to compress the models and put them into the industrial environment. This degree project explores model compression methods called knowledge distillation in the sentiment classification task on Transformer models. Transformers are neural models having stacks of identical layers. In knowledge distillation for Transformer, a student model with fewer layers will learn to mimic intermediate layer vectors from a teacher model with more layers by designing and minimizing loss. We implement a framework to compare three knowledge distillation methods: MiniLM, TinyBERT, and Patient-KD. Student models produced by the three methods are evaluated by accuracy score on the SST-2 and SemEval sentiment classification dataset. The student models' attention matrices are also compared with the teacher model to find the best student model for capturing dependencies in the input sentences. The comparison results show that the distillation method focusing on the Attention mechanism can produce student models with better performances and less variance. We also discover the over-fitting issue in Knowledge Distillation and propose a Two-Step Knowledge Distillation with Transformer Layer and Prediction Layer distillation to alleviate the problem. The experiment results prove that our method can produce robust, effective, and compact student models without introducing extra data. In the future, we would like to extend our framework to support more distillation methods on Transformer models and compare performances in tasks other than sentiment classification.

Keywords Natural Language Processing, Sentiment Analysis, Language Model, Transformers, Knowledge Distillation

Preface

I would like to thank my KTH supervisor, Anubhab Ghosh, for giving me lots of advice during our meetings. His advice inspires me every time to reflect on my current work and helps me explore my thesis on the principles of scientific research. The thesis project will not be possible without the help of Michael Truong and Pieter Buteneers from Sinch. They offer continuous guidance and resources to support my thesis work. At last, I would like to express my sincere gratitude to my family and friends for their unconditional support in every decision I have made.

Otaniemi, 04.11.2022

Haonan Liu

Contents

Abbreviations

BERT	Bidirectional Encoder Representations from Transformers
FFN	Feed-Forward Network
GPT	Generative Pre-trained Transformer
LSTM	Long Short-Term Memory
MLM	Masked Language Model
NLM	Neural Language Model
NLP	Natural Language Processing
NLU	Natural Language Understanding
SDG	Sustainable Development Goal
SemEval	Semantic Evaluation
SST-2	Standford Sentiment Treebank Binary
UN	United Nations

1 Introduction

Transformer-based language models have proven performances in various fields but are resource intensive. In this thesis, we will like to explore the knowledge distillation method to compress models. In specific, we will compare three popular knowledge distillation methods on Transformer-based language models for sentiment analysis and we will explore approaches to alleviate over-fitting in knowledge distillation.

1.1 Background

The proposal of transformer architecture makes training larger neural models possible and starts a new generation of studies on machine learning models. The design of transformer architecture can process the entire input sequence all at once and consider the dependency of items in the sequence. It allows for more parallelism so large models can be developed for various tasks. The transformer-based NLM dominated NLP and NLU tasks, and recently the Transformer has proven its success in computer vision [1], time-series [2] and many other fields [3].

NLM is a machine learning model that can produce text representation. The model can transform the text into meaningful vectors to feed into machine learning models as input for downstream NLP applications. Previous NLM usually produces context-free text representations, which means the same word in different text contexts is represented as the same vector although the word might have different meanings (e.g., word2vec[4], FastText[5]). ELMo[6] is the first attempt to train a NLM to produce dynamic word representation but the use of LSTM [7] limit its efficiency. After the emergence of Transformer architecture, recent studies replace LSTM with Transformer to efficiently produce dynamic text representation (e.g., BERT[8], GPT[9]). Those studies also propose a new approach to adapt NLM in downstream applications called fine-tuning. Instead of only using the produced text representation, fine-tuning means that an extra prediction layer is added on top of the NLM then both the model and the layer are trained with task-specific labeled data and a supervised objective. The training of NLM and adaption procedures are known as pre-training and fine-tuning respectively [10] and it is inspired from the transfer learning[11][12]. It has been proven that the Transformer model learns linguistic knowledge in the pre-training stage and can transfer the knowledge in the fine-tuning stage[13].

1.2 Problem

Despite the outstanding performances of the Transformer-based NLM in various fields, putting them into production has many challenges. The first reason is the low inference speed of the models due to a large number of parameters. The base version of the BERT model has 12 transformer blocks with 103 million parameters, making it impossible to put it into an online service. The second reason is a large consumption of energy and resources using models, which leads to economic and environmental costs. It restricts organizations from effectively using them for research or applications.

To develop efficient transformer models for downstream tasks, recent studies focus on 1) reducing the computational complexity of the transformer architecture [14] and 2) compressing the models while keeping the most of their performances [15]. The compression can be performed by parameter pruning[16], quantization[17], and knowledge distillation[18]. The focus of this project is knowledge distillation, which has been shown as a promising method in transformer compression for supervised task[19]. Knowledge distillation aims at compressing a large teacher model into a compact student model by training the student model to imitate the behaviors of a teacher. The vanilla knowledge distillation suggests that a student can learn from the logits before the Softmax function from the teacher's prediction layer[20]. Recent studies have proposed many intermediate behaviors[21] specified in the Transformer architecture that students should learn.

Sinch AB, a communication platform, deploys a real-time sentiment analysis model with millions of inference calls weekly. They identified a need to apply an efficient and effective transformer-based classifier to improve their current performance.

Research Question

Knowledge distillation is a method compressing Transformer models while keeping most of the performances. Many distillation methods have been proposed and this thesis would like to discover a good practice for Knowledge Distillation on Transformer models. Specifically, we propose two research questions. The first question is we would like to explore the differences on performances between different distillation methods. The second is to discover methods to alleviate over-fitting in knowledge distillation on Transformers.

Hypothesis

The hypothesis of the performance variation is that the distillation method involving more fine-grained imitation will have better performances. We assume that we can use approaches like increasing training data and reducing model complexity to alleviate over-fitting issue in knowledge distillation on Transformers as in other machine learning problems.

Purpose

The primary purpose of the thesis is to compare different knowledge distillation methods for the sentiment classification task. The project also contributes to discovering approaches to alleviate over-fitting in knowledge distillation. The resulting report provides Sinch, the end-user company, with suggestions on the choices of Knowledge Distillation methods and strategies to deal with the over-fitting issue.

1.3 Goals

This project aims to explore knowledge distillation methods on transformer language models for sentiment analysis. It is divided into the following three sub-goals:

- Develop a framework for knowledge distillation on Transformer-based models supporting different methods.
- Compare different distillation methods and evaluate their performances on the sentiment classification task.
- Exploration on approaches to alleviate the over-fitting issue in Knowledge Distillation.

1.4 Delimitations

The degree project is limited to evaluating the student models only on the sentiment classification task. That is because Sinch has a specific interest in the task and wishes to investigate methods to improve it. Due to the time limit, the project is also limited to experimenting with only a part of the current state-of-the-art distillation methods. The conclusion from the experiment's result should be general for other similar distillation methods.

1.5 Structure of the thesis

The structure of the thesis is as follows:

- Chapter 2 presents relevant background information about Transformer architecture, Knowledge Distillation, and Sentiment Analysis.
- Chapter 3 presents the method of comparing distillation methods and proposed approach to alleviate over-fitting.
- Chapter 4 presents the developed distillation framework and experiment settings, then presents the results and analysis.
- Chapter 5 discusses our proposed method to alleviate over-fitting in Knowledge Distillation.
- Chapter 5 concludes the degree project and points out potential future work.

2 Background

After the success of Transformer architecture in machine translation[22], studies propose to apply it to train NLM for NLP and NLU tasks. The first attempt is the GPT model that use a multi-layer Transformer decoder architecture[9]. However, the GPT model is a unidirectional language model, meaning it can only process input sequences from left to the right. To have a bidirectional context for each item in the input, the study of BERT proposes the MLM inspired by the Cloze task[23]. BERT model is a multi-layer Transformer encoder trained with the objective of the MLM task and has proven to outperform the unidirectional models on downstream tasks. Due to its state-of-the-art performance and wide application in the industry, this project focuses only on the BERT model.

This chapter will present the Transformer architecture and the BERT model. Then followed by an introduction to knowledge distillation and current state-of-the-art methods for distillation on Transformer-based language models. It goes on to introduce the downstream sentiment analysis task.

2.1 Transformer

The Transformer is an encoder-decoder architecture with a specially designed attention mechanism proposed for neural machine translation in 2017[24]. Compared to previous state-of-the-art neural models at that time, the Transformer can process input sequences in parallel using Attention instead of step by step and achieves better results. Apart from the parallelism benefiting from Attention, the Transformer applies Layer Normalization[25] and Residual Connection[26], making it possible to train deep Transformer models.

Encoder and Decoder

The Transformer encoder has a stack of N identical encoder blocks. In each block, the input will first be fed to a multi-head attention layer and then a position-wise fully connected feed-forward layer. The dimension of the input and output features are the same in d_{model} to apply residual connection. Residual connection and layer normalization are employed in each layer to facilitate training of the N-block deep model. It means that the output of each layer is added to the original input and then normalized.

The decoder has a similar design as the encoder, except that the multi-head attention layer is masked. By masking and shifting the target sequences by one position, the decoder ensures that the next predicted item is only based on previously seen items. That explains the unidirectional feature of the GPT model. In the original paper, a Transformer architecture for machine translation has six encoder blocks and six decoder blocks. The output from the last encoder block is fed into the first decoder block.

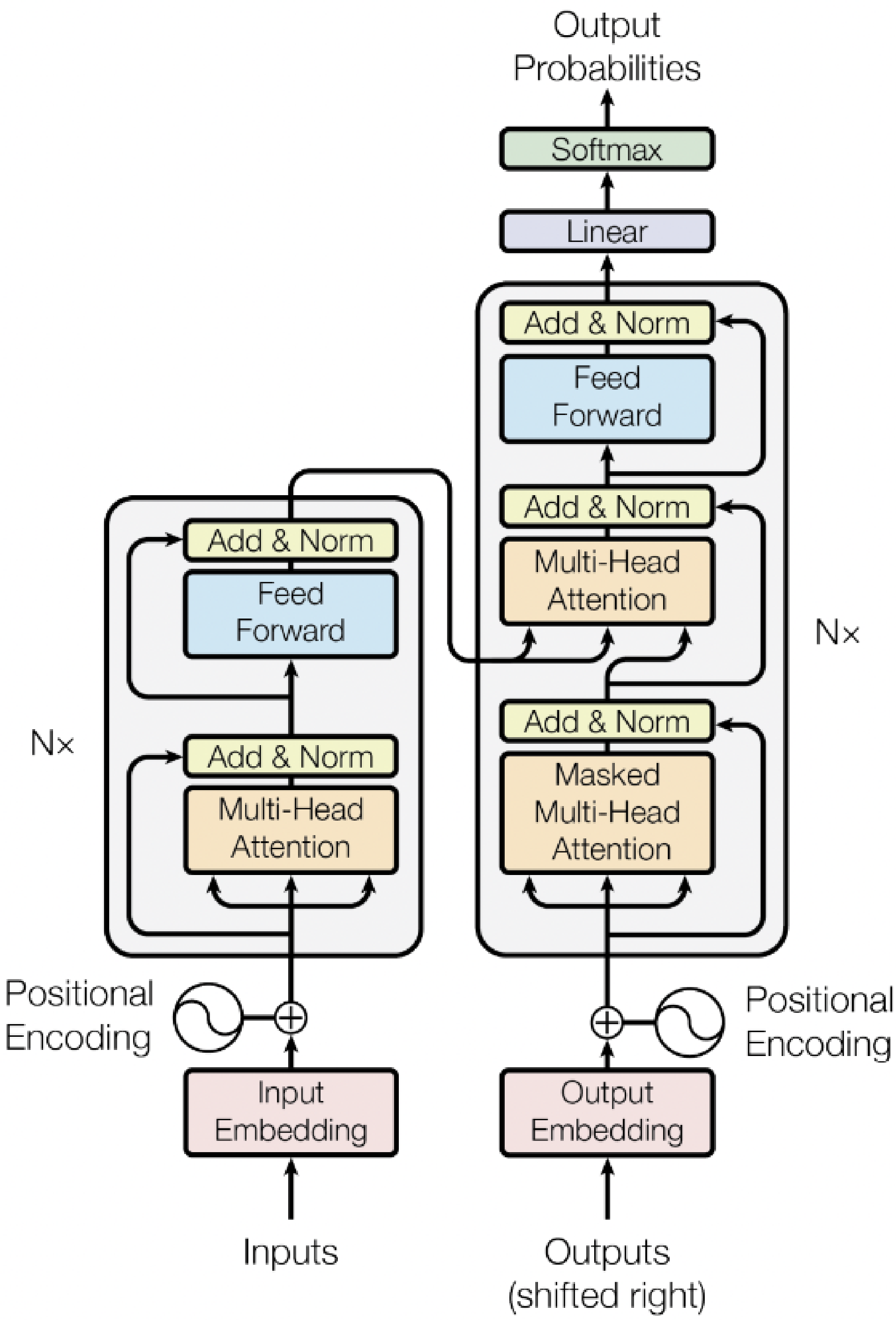

Figure 1: Transformer architecture from the original paper[22]

Attention

The Attention mechanism used in the Transformer is based on a alignment function proposed for machine translation [24]. The mechanism takes a query vector and key-value vector pairs as input and output a vector. The output vector is a weighted sum of value vectors where the weight is calculated by a correlation function between query and key. In Transformer architecture, scaled dot-product attention shown in the Equation 1 is used.

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax(\frac{\mathbf{QK}}{\sqrt{d_k}})\mathbf{V} \tag{1}$$

Q, K, V are matrices having the same dimension $n \times d_k$. To understand relations between each item in a sequence and encode the relations into each item, Transformer proposes a self-attention. For an input matrix X containing a sequence of word vectors, self-attention projects the matrix into query, key, and value matrices using learn-able matrices W^Q, W^K, W^V and perform scaled dot-product attention, which is calculated in by Equations from 2 to 5.

$$\mathbf{Q} = \mathbf{XW}^q \tag{2}$$

$$\mathbf{K} = \mathbf{XW}^k \tag{3}$$

$$\mathbf{V} = \mathbf{XW}^v \tag{4}$$

$$SelfAttention(\mathbf{X}) = Attention(\mathbf{XW}^q, \mathbf{XW}^k, \mathbf{XW}^v) \tag{5}$$

To capture different relations in a sequence, Transformer proposes multi-head attention, which projects input matrix X into h linear spaces in parallel using h sets of $(\mathbf{W}_i^q, \mathbf{W}_i^k, \mathbf{W}_i^v)$. The h attention results are concatenated and projected back to the input dimension. It can be expressed by:

For the i_{th} head:

$$head(\mathbf{X}, i) = Attention(\mathbf{XW}_i^q, \mathbf{XW}_i^k, \mathbf{XW}_i^v) \tag{6}$$

the concatenation and output is archived by:

$$MultiHeadAtten(\mathbf{X}) = Concat(head(\mathbf{X}, 1), ..., head(\mathbf{X}, n))\mathbf{W}^o \tag{7}$$

Position-wise Feed-Forward Networks

The position-wise FFN combines two linear transformations with a ReLU activation in between applying to each input position. The module can be computed by the Equation 8.

$$FFN(\mathbf{x}) = max(0, \mathbf{xW}_1 + b_1)\mathbf{W}_2 + b_2 \tag{8}$$

$\mathbf{W}_1$, $\mathbf{W}_2$ are transformation matrices and b_1, b_2 are the biases. The first transformation will project dimension from d_{model} into a higher dimension d_{inner} and the second transformation will project it back to the dimension d_{model} for residual connection.

Positional Encoding

The Transformer is highly parallelizable due to the design of the attention mechanism but at the expense of losing the order information of input sequences. Positional encoding is introduced to inject information about the relative or absolute position of the input sequences[27]. The Transformer uses a fixed positional representation of dimension d_{model}, the same as the input features. For different position pos of an item in a sequence and index of the representation vector i. The value can be calculated using the sine and cosine functions as the Equations 9, 10 below.

For singular index:

$$PE(pos, 2i+1) = cos(\frac{pos}{10000^{2i/d_{model}}}) \tag{9}$$

For plural index:

$$PE(pos, 2i) = sin(\frac{pos}{10000^{2i/d_{model}}}) \tag{10}$$

BERT

BERT[8], Bidirectional Encoder Representations from Transformers is a language model proposed in 2018. Unlike other language models at that time, BERT uses stacks of transformer encoder blocks to train a deep language model conditioning on both left and right contexts using a massive amount of unlabeled text. The training process is known as pre-training. An extra linear layer needs to be added on top of the trained model o apply the model to downstream tasks, and then the new network needs to be trained with in-domain task data and corresponding objectives, called fine-tuning. The model, along with the pre-train then fine-tune paradigm, are powerful, and the recent state-of-the-art performances in NLP and NLU tasks are still backboned on the idea of BERT[28].

2.2 Knowledge distillation on Transformer-based Language Model

The Transformer model has shown to be influential in many downstream tasks. However, deploying them in the real industrial scenario is hard due to the high computational requirement. Knowledge distillation is a model compression method achieved by training a student model under the supervision of a teacher model on a task. In Knowledge Distillation for Transformer-based NLM, the student is trained with two objectives. The first objective is to train the student NLM by imitating of teacher NLM internal behaviors. The second objective is to adapt the student NLM to the task supervised by the teacher. Thus, the objective of Knowledge Distillation can be formulated as minimizing the Equation 11.

$$Loss = \sum_{i}^{n} D(f_i^s(x), f_{m(i)}^t(x)) + L(f_n^s(x), y) \tag{11}$$

where n is the number of the student blocks, $f_i^s()$ and $f_i^t()$ represent behavior of student and teacher model at block i respectively. $m(i)$ is a mapping function that decides which teacher block the i_{th} student block should learn from. $D(\cdot,\cdot)$ is a loss function calculating the difference between the inputs. $L(\cdot,\cdot)$ is a loss function measuring loss between the student's prediction and the real labels. Both teacher and student have several Transformer blocks and an extra Linear Layer block for adaptation to a task. Since behaviors in Transformer layer blocks ($i < n$) and the Prediction layer block ($i = n$) are different, we refer to the losses in the two kinds of blocks as Prediction layer loss and Transformer layer loss, respectively. Recent Knowledge Distillation methods on NLM have proposed losses consisting of fine-grained Transformer Layer Losses and Prediction Layer Losses. The following subsections will introduce the two types of losses.

Prediction Layer Loss

In vanilla knowledge distillation[20], it is suggested that the student should mimic the logits vector of the final Softmax function from the teacher. The logit vector with a dimension of the number of labels provides a roughly estimated confidence for each predicted label, and the largest logit corresponds with the correct label. To help the student have a better understanding of the relation between labels in knowledge distillation, Softmax with temperature is proposed to soften the logits vector shown in the Equation 12:

$$z_T = softmax(T, z) = \frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_i}{T}}} \tag{12}$$

The larger the T is, the softer the probability distribution over classes. The loss is then calculated by the Cross-Entropy of the two distributions, which can be formulated as:

$$L_{ce} = -(z_T^s * log(z_T^t)) \tag{13}$$

z_T^s is logits from a student with temperature T while z_T^t means the same but from a teacher. The Cross-Entropy loss is equivalent to the Mean Squared Error loss (shown in Equation 14) in the high temperature limit proven in [20].

$$L_{mse} = \frac{1}{2}(z_T^s - z_T^t)^2 \tag{14}$$

Transformer Layer Loss

Knowledge Distillation considering only the Prediction Layer behaviors can achieve a comparable result on the training set, but it is shown to have a poor generalization ability [29]. Taking the Transformer Layer Loss into account is a method to alleviate the issue.

TinyBERT[30] proposes distillation on three types of intermediate behaviors: 1) the input of the first block, which is the text embedding vectors $\mathbf{E}$; 2) the hidden vectors $\mathbf{H}$ output from each Transformer block; 3) the attention matrix $\mathbf{A}_i$ in the i_{th} head from the Transformer block. To fix the dimension d_{model} mismatching problem in features between teacher and student, TinyBERT uses projection matrices to map them into the same dimension. The loss functions for the behaviors between one student block and one teacher block are defined as:

$$L_{embd} = MSE(\mathbf{E}^S\mathbf{W}_e, \mathbf{E}^T) \tag{15}$$

$$L_{hidn} = MSE(\mathbf{H}^S\mathbf{W}_h, \mathbf{H}^T) \tag{16}$$

$$L_{attn:mse} = \frac{1}{h}\sum_{i=1}^{h} MSE(\mathbf{A}_i^S, \mathbf{A}_i^T) \tag{17}$$

MiniLM[21] proposes a deep self-attention distillation loss. The method focus on the attention matrix $\mathbf{A}_i$ and its associated value vector $\mathbf{V}_i$ in the i_{th} head from a transformer block. In an Attention matrix of $n \times n$, each row is a distribution revealing the correlation scores of an item to other items in a sequence. In MiniLM, the loss functions for $\mathbf{A}_i$ is the sum of KL-divergence between rows j in student and teacher's attention matrices:

$$L_{attn:kld} = \frac{1}{h}\sum_{i=1}^{h}\sum_{j=1}^{n} KL(\mathbf{A}_{ij}^T || \mathbf{A}_{ij}^S) \tag{18}$$

For value vector $\mathbf{V}_i$, since d_k in teacher and student models are different, MiniLM proposes to learn from the value relation instead of the value vectors. The relation $\mathbf{VR}_i$of value $\mathbf{V}_i$ and the loss are calculated by:

$$\mathbf{VR}_i = softmax(\frac{\mathbf{V}_i\mathbf{V}_i^T}{\sqrt{d_k}}) \tag{19}$$

$$L_{VR} = \frac{1}{h}\sum_{i=1}^{h} KL(\mathbf{VR}_i^T || \mathbf{VR}_i^S) \tag{20}$$

The Transformer losses proposed by TinyBERT and MiniLM involves heavy computations because it calculates losses between features at all positions. Since the BERT model only uses the representation of the first special token [CLS] to make a prediction on a input sentence, Patient Knowledge Distillation[29] proposes an internal loss only considering the normalized [CLS] token features f. In a student-teacher block pair, the loss is defined by:

$$L_{pkd} = MSE(\frac{\mathbf{f}^T}{||\mathbf{f}^T||_2}, \frac{\mathbf{f}^S}{||\mathbf{f}^S||_2}) \tag{21}$$

Summary

Training a Transformer student model under the supervision of a Transformer teacher model for a task has two objectives. The first is to mimic internal teacher behaviors mainly achieved by Transformer Layer Loss. The second is to adapt the student model to the task, mainly achieved by Prediction Layer Loss and corresponding task loss. The overview of knowledge distillation for a Transformer student model is illustrated in Figure 2.

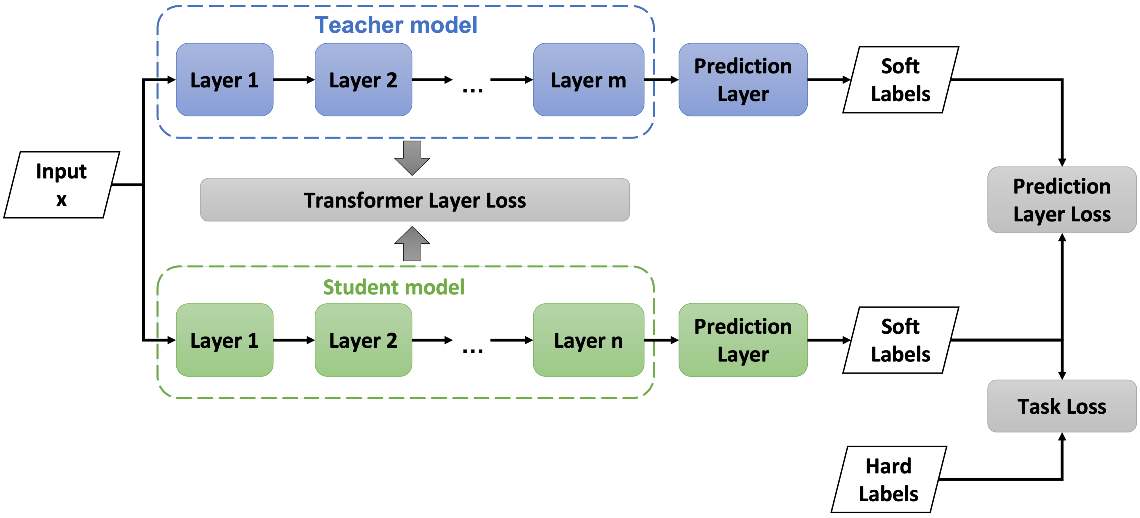

Figure 2: Overview of Knowledge Distillation on Transformer models

2.3 Sentiment Analysis

Sentiment analysis is a crucial downstream task in NLP. The task helps the analysis of people's opinions on various topics, events, and products. Government, companies, and individuals can benefit from insights produced by sentiment analysis and make wise decisions.

We can perform sentiment analysis at several levels: Document Level, Sentence Level, and Phrase Level. Document level sentiment classification regards the whole document as input and classifies it as expressing positive or negative sentiment. Sentence-level sentiment classification considers a sentence in a document and classifies its opinion. Phrase-level sentiment classification splits a sentence into phrases and predicts its sentiment.

Traditional methods for sentiment analysis are mainly lexicon-based approaches [31]. A lexicon is a collection of vocabularies, and each word is given a sentiment polarity score. To classify sentiment for a sentence first is to split it into words and then aggregate their scores to make a prediction. The method does not rely on training data but requires much manual work to develop the lexicon.

Recent approaches for sentiment classification are mostly data-driven supervised machine learning method[32]. Words in sentences or documents are projected into meaningful real vectors and then are concatenated to represent the whole text. Machine learning models can be trained to take the text representation as input and predict the sentiment labels. The emergence of Transformer Language models dominates sentiment classification tasks at each level [8]. Figure 3 shows an overview of applying BERT in sentence sentiment prediction. The BERT model takes a sentence as input and will output confidence scores for being positive or negative.

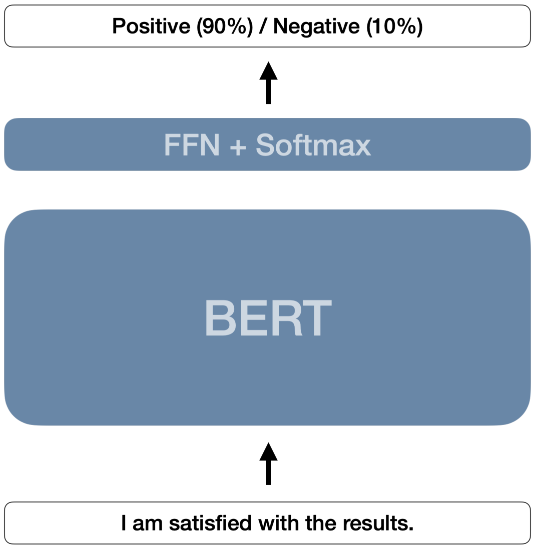

Figure 3: Overview of using BERT for sentiment classification

3 Methodology

This chapter introduces the method for exploring reducing the resource consumption of Transformer models using Knowledge Distillation. In detail, it will present motivation and method used to compare different distillation methods and explain the approaches used to solve the over-fitting issue. It follows by an introduction to the evaluation method on the produced student models.

3.1 Comparison

Many studies have proposed Knowledge Distillation methods on Transformer-based models. Comparisons on different methods are usually with bias. In the comparison experiments performed by the TinyBERT paper[30], they introduce a task-specific and task-agnostic distillation along with a data augmentation process only for the TinyBERT model. And in MiniLM[21], they use an in-house teacher model to perform comparison. In this project, we would like to discover that the difference on performances of different distillation methods under the same condition. The results will help us better applying Transformer-based model into industrial scenario.

Teacher and Student Models

The teacher model in distillation experiments is a large Transformer NLM model. The model has many blocks, and each block should output its internal behaviors to perform Transformer Layer Loss. Since the comparison is on the sentiment analysis task, the teacher model should be fine-tuned to adapt to the task before Knowledge Distillation.

The goal of Knowledge Distillation is to train a faster student model under the supervision of a large teacher model. Student Transformer models can be faster by taking efficient calculations[14], modifying architecture[33], or reducing the number of blocks and hidden dimensions. This project uses student models with fewer blocks and hidden dimensions to perform Transformer Layer Loss. The weights in the models are initialized by pre-training. Knowledge Distillation will adapt the student models to the sentiment analysis task with the help of teacher models.

Distillation

In the comparison of distillation methods, we mainly focus on comparing the performance difference of Transformer Layer losses defined in MiniLM, TinyBERT, and Patient-KD under the same condition. The teacher model and student model used in the distillation experiments will have the same initialization. The loss functions of all experiments will be a combination of Transformer Layer Loss, Prediction Layer Loss, and the corresponding task loss shown in below:

$$L = L_{trn} + L_{pred} + L_{task} \tag{22}$$

In the TinyBERT experiment, the Transformer Layer Loss has embedding loss, hidden loss, and attention loss stated in Equations 15, 16, 17:

$$L_{trn} = L_{embd} + L_{hidn} + L_{attn:mse} \tag{23}$$

In the experiment for MiniLM, the Transformer Layer Loss has attention loss and value relation loss stated in Equations 18, 19:

$$L_{trn} = L_{VR} + L_{attn:kld} \tag{24}$$

For Patient-KD, the Transformer Layer Loss is equivalent to the [CLS] token feature loss shown in Equation 21:

$$L_{trn} = L_{pkd} \tag{25}$$

The result of each experiment will be an average result from three experiments using different random seeds.

3.2 Over-fitting Solutions

Over-fitting is a common issue in a machine learning problem. In this project, over-fitting may happen to student models on the sentiment analysis task. This project explores two directions to alleviate the case: increase data size and reduce model complexity.

Increase data size

If a model is over-fitted on the existing training set, increasing training data can make it harder to over-fit on all the samples and force it to generalize. It is not always easy to expand the labeled dataset, but in Knowledge Distillation, we can also use unlabeled text data to help students mimic teacher's behaviors[34]. The loss function for distillation with unlabeled data only consists of the Transformer Layer Loss and Prediction Layer Loss. The corresponding task loss is not applicable as the data is unlabeled. To alleviate over-fitting, we expand the original training set with unlabeled data from a similar domain. To find a dataset from a similar domain, we mainly consider the text length of samples and scenarios of the dataset. In this project, we expand the movie review data with amazon product review data as samples in both of the datasets have similar text lengths and are reviews on items.

Reduce model complexity

A natural idea to reduce model complexity is to reduce the number of blocks and dimensions in the student models. Chapter **??** will present the results and discussion of the method. It turns out to be not effective for solving over-fitting.

In this project, we propose a Two-step Knowledge Distillation method to reduce model complexity to solve the over-fitting issue. The method has two sequential procedures.

Transformer Layer Distillation

The first step is the Transformer Layer Distillation which only considers Transformer Layer Losses. In this step, we only consider parameters in the student model while the teacher model is frozen. To update the student model, input is fed to both the teacher and student model and only the Transformer Layer Loss is calculated. The goal in this step is to update the student model using the Transformer Layer Loss to help the student better mimic the teacher behaviors. Figure 4 illustrates the first step and the loss function is calculated by $L_{first} = L_{trn}$.

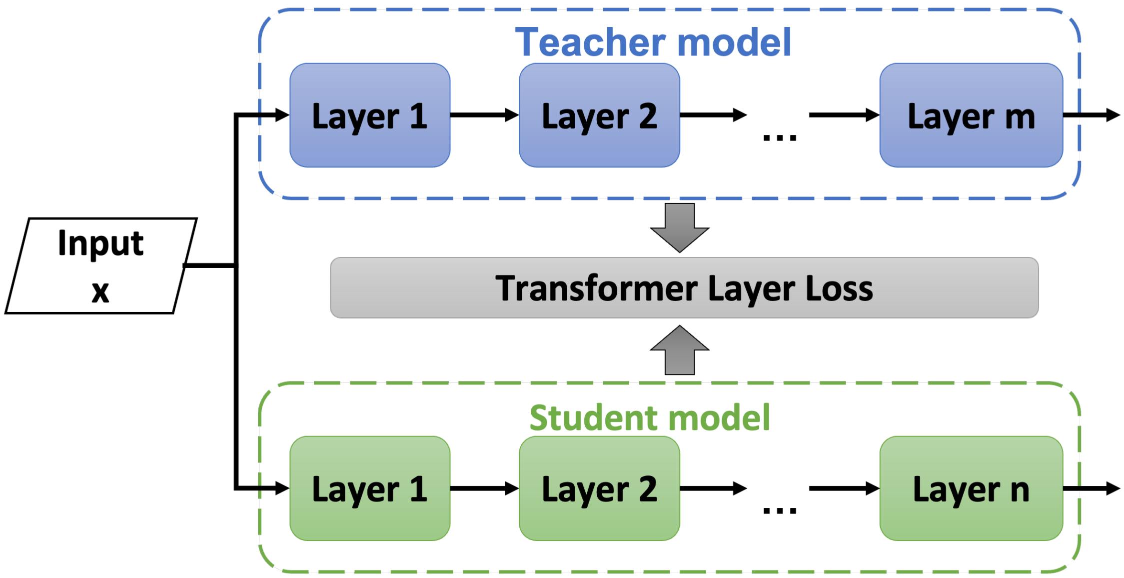

Figure 4: The first step of the Two-Step distillation method

Prediction Layer Distillation

The second step is the Prediction Layer distillation which only uses the Prediction Layer Losses and corresponding task loss. The teacher models and associated prediction layer are frozen as before. In this step, the weights in the Student model are also fixed. We consider only the prediction layer associated with the student model and update it with prediction layer loss and task loss. Figure 5 illustrates the second step and loss function of the step is $L_{second} = L_{pred} + L_{task}$.

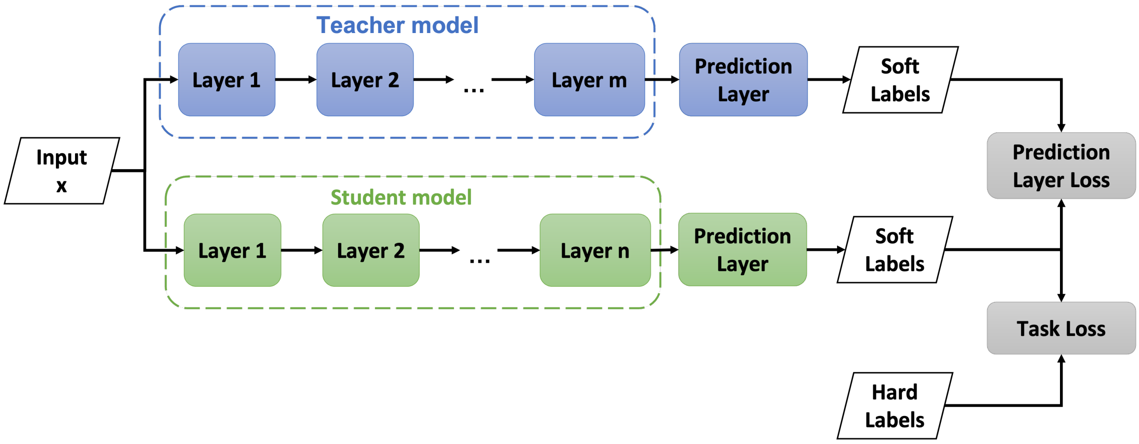

Figure 5: The second step of the Two-Step distillation method

3.3 Evaluation

Assessment focus on the student models produced by the Knowledge Distillation experiments. The student models are evaluated on the validation and test data of sentiment classification tasks using F1-score and accuracy. The curves of training loss, validation loss, training metrics, and validation metrics will be monitored to evaluate the generalization ability of models. The metrics are averaged over three experiments with different random seeds to ensure reliability.

Platform	CPU	GPU
Google Colaborary	Intel(R) Xeon(R)	Tesla P100
Server at Sinch	AMD Ryzen 3960X	RTX 3090

Table 1: Computing resources in each platform

4 Experiments

This chapter introduces implementations of comparison between different distillation losses and solutions for over-fitting. It follows by presenting the experiment results and analysis.

4.1 Settings

Environments

The experiments are performed on a Google Colaborary platform and a server at Sinch. Most investigations are run using the GPU resources, and the available resources in the platforms are listed in Table 1

The thesis project is implemented in Python and its libraries. The libraries used mostly were Pytorch[35], Transformers[36], Pytorch-Lightening[37], Wandb[38] and Github was used for version control.

Data

The SST-2 dataset and 2017 SemEval Task 4 dataset are used for knowledge distillation and evaluation. The SST-2 dataset is a corpus consisting of 70,042 single sentences extracted from movie reviews. Three human judges annotate all the reviews as either negative or positive. The SemEval 2017 dataset consists of 59,899 tweets in English with either positive, neutral, or negative sentiment labels. The ratios of the training, validation, and test split follow the setting of the original description to compare with the public results.

Both datasets have two fields called text and label, respectively. The text field will first split into subwords and then be mapped into number ids using a WordPieces tokenizer[39] so that the Transformer models can read and process the text. The label is a sentiment class that will be mapped into a corresponding number id.

4.2 Implementation

Teacher and Student Models

The teacher models and student models are based on BERT implemented by Transformers library. The source code is modified to produce intermediate behaviors, including the attention scores matrices, value vectors from each attention layer, and hidden vectors from each block. The teacher model is the base version of BERT having 12 layers and 768 hidden dimensions. There is a large version but we leave it for future work as it requires more computational resources. For student models, this project will use BERT models with 12 layers and 384 hidden dimensions, six layers, and 384 dimensions, and four layers and 312 hidden dimensions. The weights of all these BERT models are loaded from Huggingface Model Hub and pre-trained on a large corpus of English data. To perform Knowledge Distillation for sentiment analysis, we prepare two teacher models, and each is fine-tuned on SST-2 and SemEval data, respectively. The student models are adapted to two tasks by Knowledge Distillation supervised by the corresponding teacher models.

Distillation Framework for Comparison

An open-source framework integrating different distillation losses on Transformer is implemented for easy comparison in the thesis project.

The key idea of the framework is to break losses defined in recent Distillation methods on Transformers into small loss units, namely adaptor modules. Each adaptor helps the student adapt (learn) behavior from the teacher. Below are the currently available adaptors in the framework, and it is easy to extend for additional customized adaptors.

- **LogitMSE Adaptor:** Calculate Mean Squared Error between teacher's and student's logits before Softmax.
- **AttnMSE Adaptor:** Calculate Mean Squared Error between teacher's and student's attention matrices.
- **AttnKLD Adaptor:** Calculate KL-Divergence between teacher's and student's attention matrices.
- **ValKLD Adaptor:** Calculates KL-Divergence between teacher's and student's value relations.
- **HidnMSE Adaptor:** Calculates Mean Squared Error between teacher's and student's hidden vectors.
- **CLSMSE Adaptor:** Calculates Mean Squared Error loss between teacher's and student's hidden vectors of $[CLS]$ token.

Using the framework for Knowledge Distillation on Transformer involves three steps: 1) prepare distillation dataset, teacher model and student model; 2) define adaptors and configure distillation; 3) perform distillation and produce the trained student model. The framework also implements helper modules, including a fine-tuner and a pre-trainer for model training.

This project uses the framework for comparing three distillation losses: TinyBERT, MiniLM, and Patient-KD. Adaptors used for each distillation loss are introduced in Table 2. For all distillation experiments, we use a batch size of 32, a learning rate of 0.00003, and a maximum input sequence length is 128. The weights are updated using AdamW[40] optimizer with a learning rate linear warm-up scheduler.

Method	Adaptors
TinyBERT	AttnMSE, HidnMSE, LogitMSE
MiniLM	AttnKLD, ValKLD, LogitMSE
Patient-KD	CLSMSE, LogitMSE

Table 2: Adaptor settings for each distillation method

Data / Methods	Acc on SST2		Acc on SemEval	
	Validation	Test	Validation	Test
Teacher	93.0	92.8	74.3	74.5
MiniLM	**92.0±0.1**	**91.6±0.2**	**73.8±0.4**	72.5±0.4
TinyBERT	91.8±0.1	91.5±0.1	72.8±0.5	**73.0±0.5**
Patient-KD	91.9±0.5	91.3±0.3	71.5±1.5	72.0±1.0

Table 3: Comparison results on different distillation losses

Distillation Framework for Over-fitting Solutions

Increasing data size and reducing model complexity are two directions to solve over-fitting in Knowledge Distillation.

To alleviate over-fitting by increasing data, we introduce review data from Amazon since this thesis project is interested in sentiment analysis. The Amazon reviews dataset consists of English reviews from amazon. The data contains about 35 million reviews from 2013. Each sample includes a review content and a binary sentiment label, which is about 100 times larger than SST-2 and SemEval. For distillation on each task, we combine the Amazon data with the task data and then perform Knowledge Distillation.

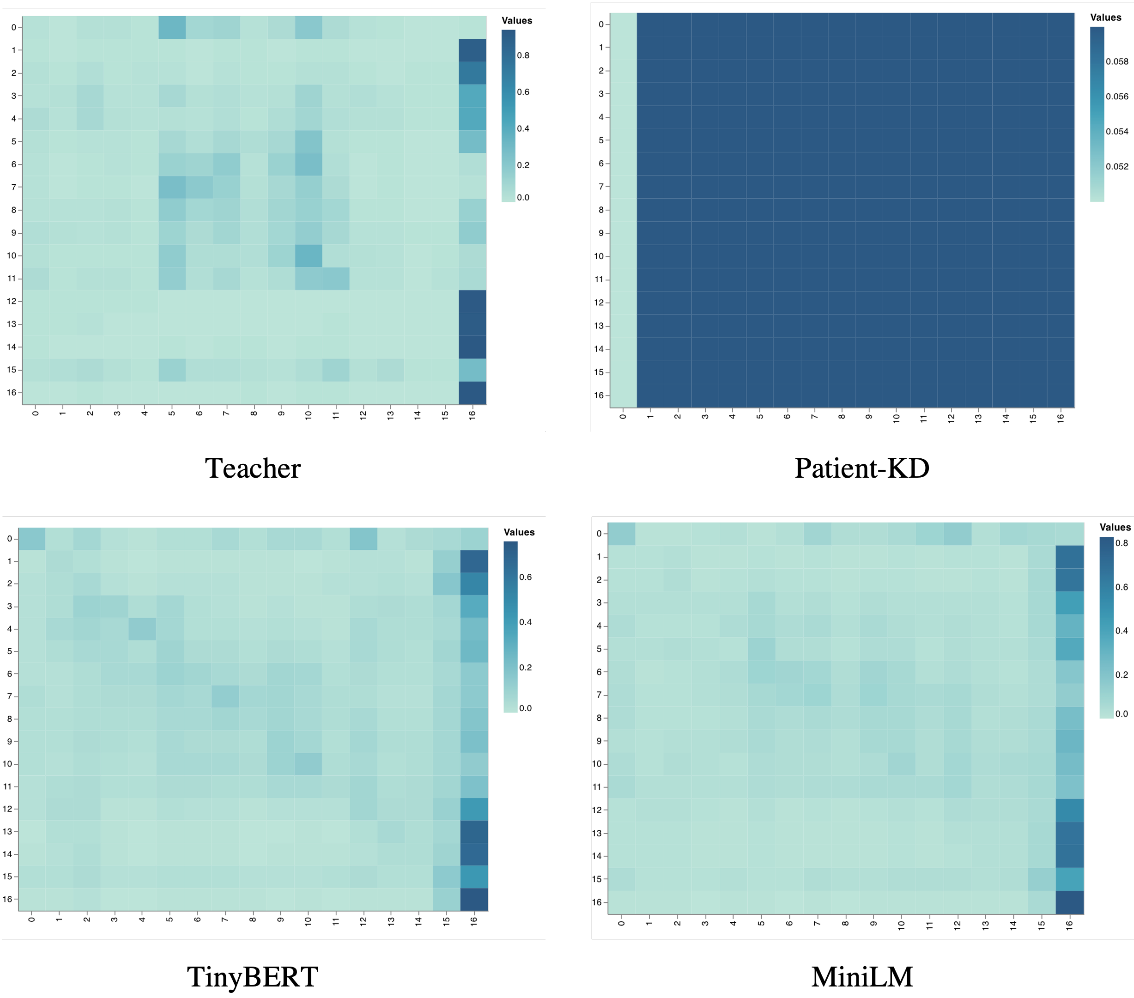

Figure 6: Attention matrices of the last block in teacher and students

We use the proposed two-step Knowledge Distillation method to reduce model complexity to alleviate over-fitting. The distillation process is split into two stages. The first stage lasts for ten epochs, only the Transformer Layer adaptors are used, and weights only in the Transformer Layers are updated. The second stage of 10 epochs freezes the Transformer layers and only uses Prediction Layer adaptors with hard label loss. The data for distillation is the same for the two phases.

4.3 Results

This section presents the experiment results.

Comparison

Students' performances on the holdout test set from the SST2 and Tweet dataset are summarised in Table 3. The curve of metrics and loss on the validation dataset is shown in Figure 8. It is demonstrated that MiniLM and TinyBERT perform slightly better than Patient-KD. And Patient-KD shows a more significant variance in the results compared with the other two losses.

Figure 6 shows Attention matrices of the last layers in student models and the teacher model in the last epoch. It proves that adaptors associated with Attention help the students understand correlations between words in sentences. That might be a reason for producing a more robust and effective student model using MiniLM and TinyBERT than Patient-KD.

Data / Methods	Acc on SemEval	
	Validation	Test
Teacher	93.0	92.8
MiniLM	73.8±0.4	72.5±0.4
MiniLM+data	73.4	72.6
TinyBERT	72.8±0.5	73.0±0.5
TinyBERT+data	73.3	73.3
Patient-KD	71.5±1.5	72.0±1.0
Patient-KD+data	73.3	73.1

Table 4: Results of distillation using extra data

However, the training loss, validation loss, and validation metrics curves reveal the over-fitting problem in these distillation experiments. We've tried with different configurations and repeated experiments several times, but the issue remains. Thus we conclude that it is due to insufficient data and complicated models. The following section presents solving the over-fitting with the two previous approaches.

Over-fitting Solutions

Increasing the data size with Amazon data helps with the over-fitting issue. The results are shown in Figure 7 and Table 4. Training with data at this size requires more computing resources and time. Thus we only perform experiments on SemEval data and only repeat once. The resulting student models fit better on the sentiment analysis task.

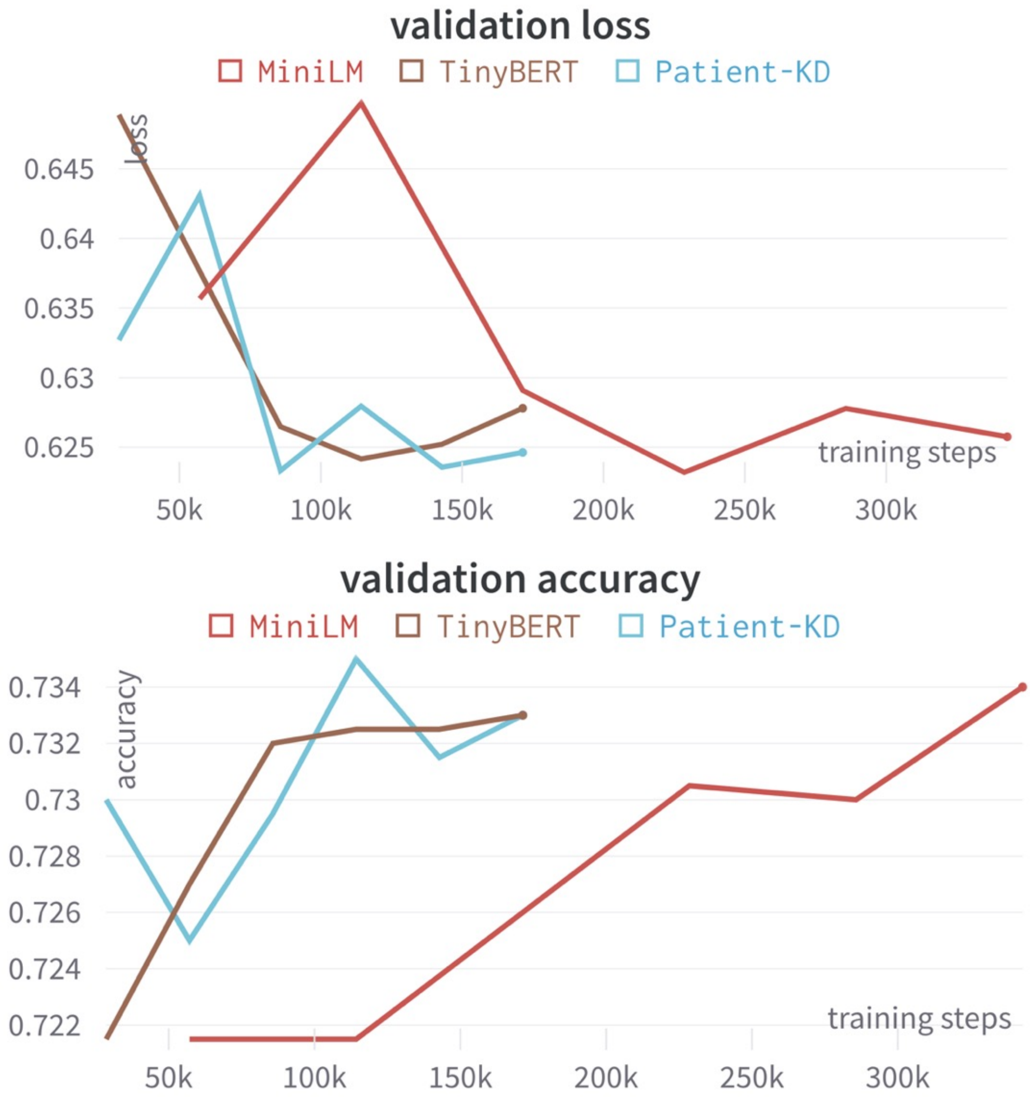

Figure 7: Solution for over-fitting by increasing data size

Another approach we experimented with is the proposed Two-Step Distillation method. We compare the results between Two-Step and Mix-Step method in Figure 9, 8, and Table 5. The Mix-Step (MS) method stands for the original distillation method, and Two-Step (TS) method is the proposed method. No extra data is introduced in this method, but the method alleviates the over-fitting problem.

The produced student models show a convergence trend and have more robust performances in every validation step after each epoch than the Mix-Step method. The model results variances are slight and can sometimes outperform results produced by the Mix-Step method. It reveals that our method can be an alternative method to alleviate over-fitting in knowledge distillation.

Data / Methods	Acc on SST2		Acc on SemEval	
	Validation	Test	Validation	Test
Teacher	93.0	92.8	74.3	74.5
MiniLM (MS)	**92.0±0.1**	**91.6**±0.2	**73.8**±0.4	**72.5**±0.4
MiniLM (TS)	90.3±0.1	90.0±**0.0**	71.8±**0.0**	70.8±**0.1**
TinyBERT (MS)	91.8±0.1	91.5±0.1	72.8±0.5	73.0±0.5
TinyBERT (TS)	**91.9±0.1**	**91.7±0.1**	**73.0±0.0**	**73.5±0.0**
Patient-KD (MS)	**91.9**±0.5	**91.3**±0.3	71.5±1.5	72.0±1.0
Patient-KD (TS)	88.0±**0.2**	89.1±**0.1**	**73.0±0.0**	**72.5±0.0**

Table 5: Comparison results on different distillation methods

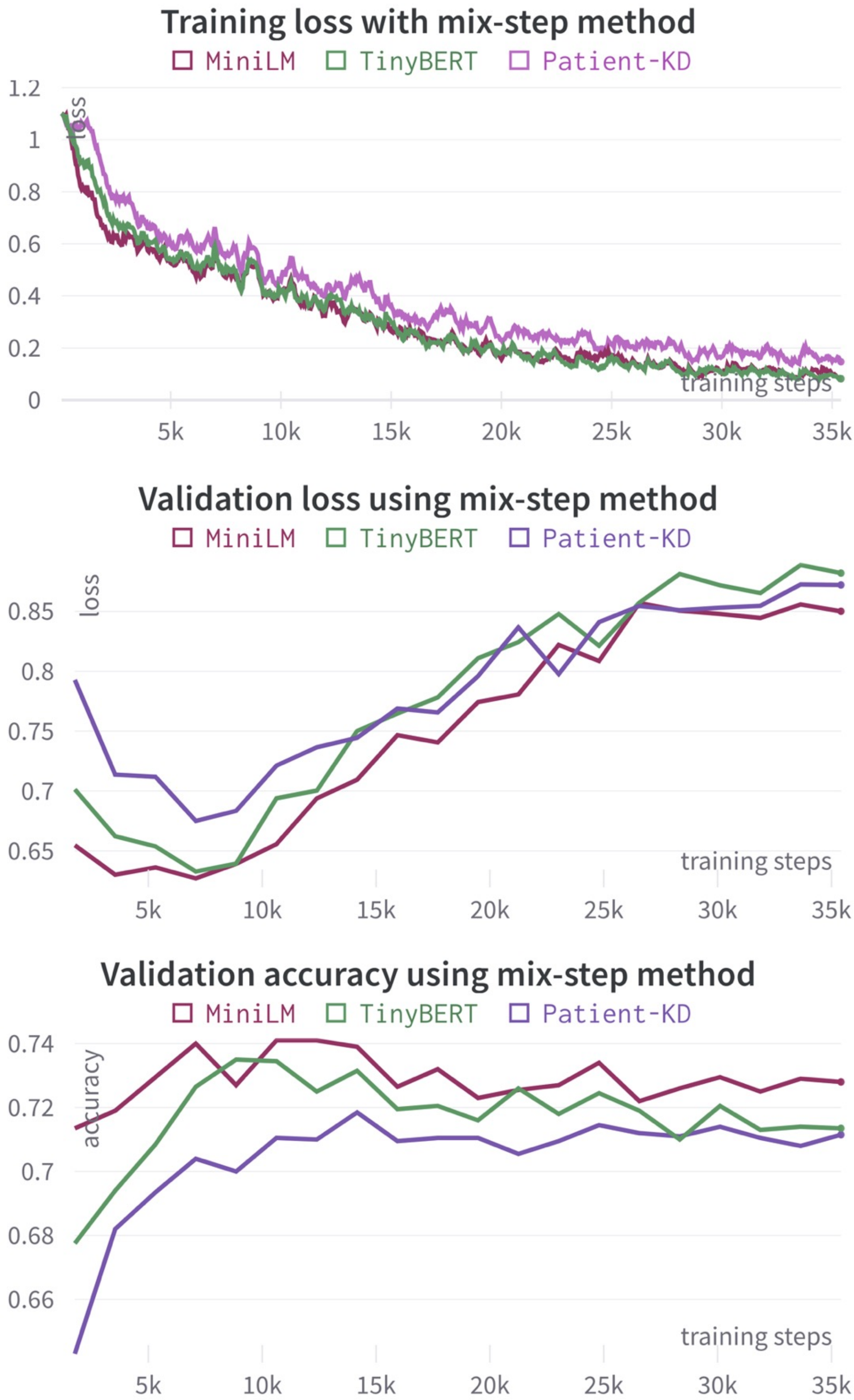

Figure 8: Comparison between different distillation loss using mix-step method

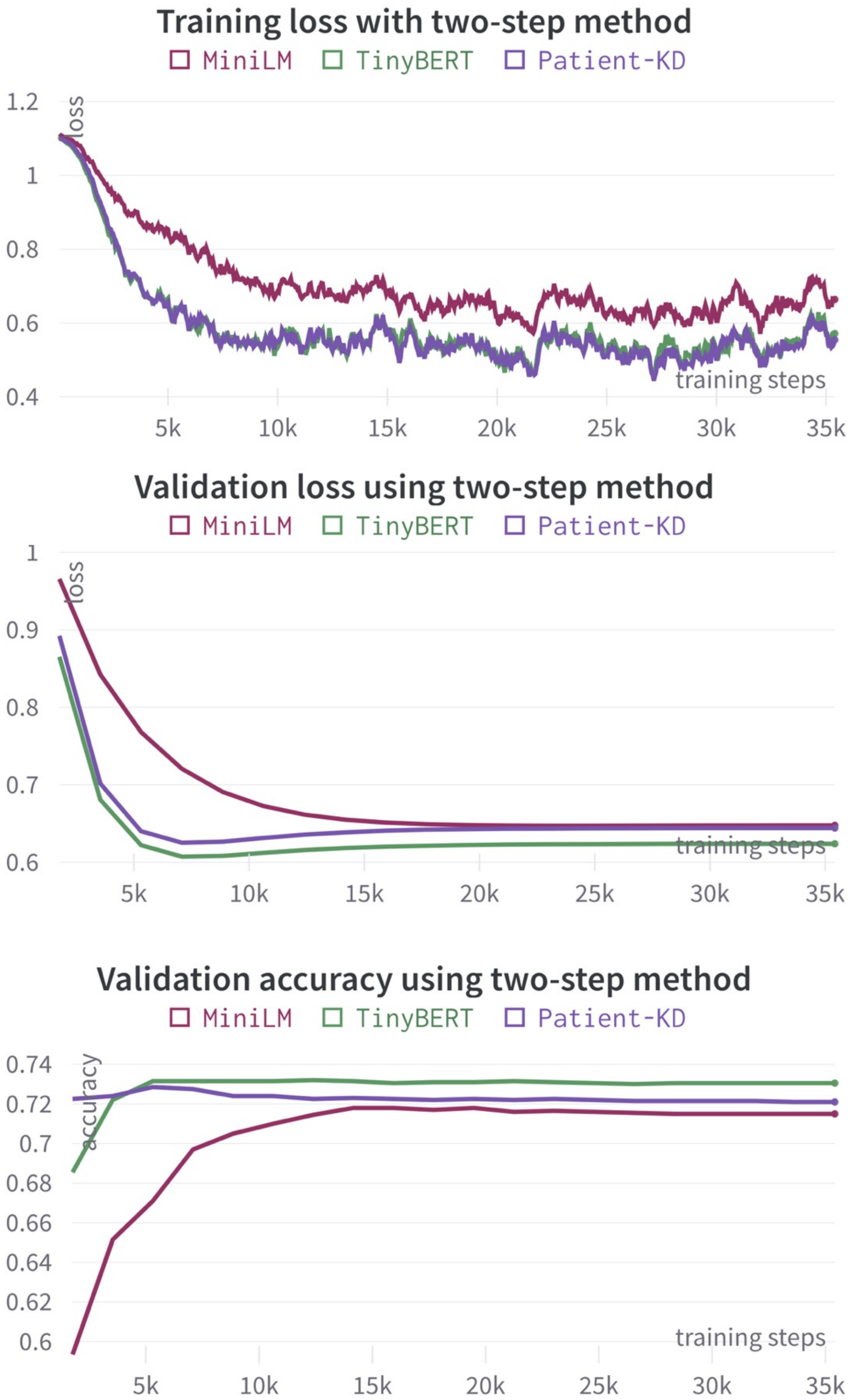

Figure 9: Comparison between different distillation loss using two-step method

5 Discussion

An intuitive idea to reduce student Transformer model complexity is using a Transformer with fewer blocks and dimensions. This chapter will discuss results of this idea and analyze our proposed Two-Step method.

5.1 Reduce complexity using small models

The student model used in previous experiments is a BERT model with 12 layers and 384 hidden dimensions. To reduce model complexity to alleviate over-fitting, we tried models with six layers and 384 hidden dimensions and models with four layers and 314 hidden dimensions. The results in Figure 12 to 14 show that the over-fitting problem remains even smaller student models are used for Knowledge Distillation.

The results show that reducing student model size cannot help with the over-fitting issue, so we start to consider an alternative way to reduce model complexity in Knowledge Distillation.

5.2 Analysis of Two-Step method

Notice that there are two objectives in Knowledge Distillation on Transformer Distillation. The first objective is to train the student model to mimic the teacher model's internal behaviors. The second objective is to adapt the student model to the task supervised by the teacher. The student models in our experiments are over-fitted on the second objective. Thus we would like to think about an approach to reduce the model complexity for the second objective while keeping the advantage of learning from the first objective. That is the motivation of the Two-Step method.

It turns out that the proposed method can alleviate the over-fitting issue and sometimes lead to better performances. We analyze the reason for the proposed method's effects from two possible perspectives. The first is that the two objectives in Knowledge Distillation may have different difficulties. The first objective requires student models to mimic the teacher's intermediate features. Predicting the specific task with the teacher's help is the second objective. The first objective seems more complex than the second one, and it makes sense to use a more complex model for a more complicated problem. The second perspective is that the Two-Step method can help students mimic the teacher's internal behaviors better. It comes from Attention matrices and internal behaviors difference between students and teachers. The attention matrices in the last layer of the same input from a teacher model and student models produced by two methods using TinyBERT loss are illustrated in Figure 10. The student model created by the Two-Step method has a more similar attention matrix to the teacher compared to the other student model. Figure 11 compare mean squared error between attention matrices in a student and a teacher using Two-Step and Mix-Step methods. It reveals that the Two-Step method can produce student models with more similar attention behaviors than the Mix-Step method.

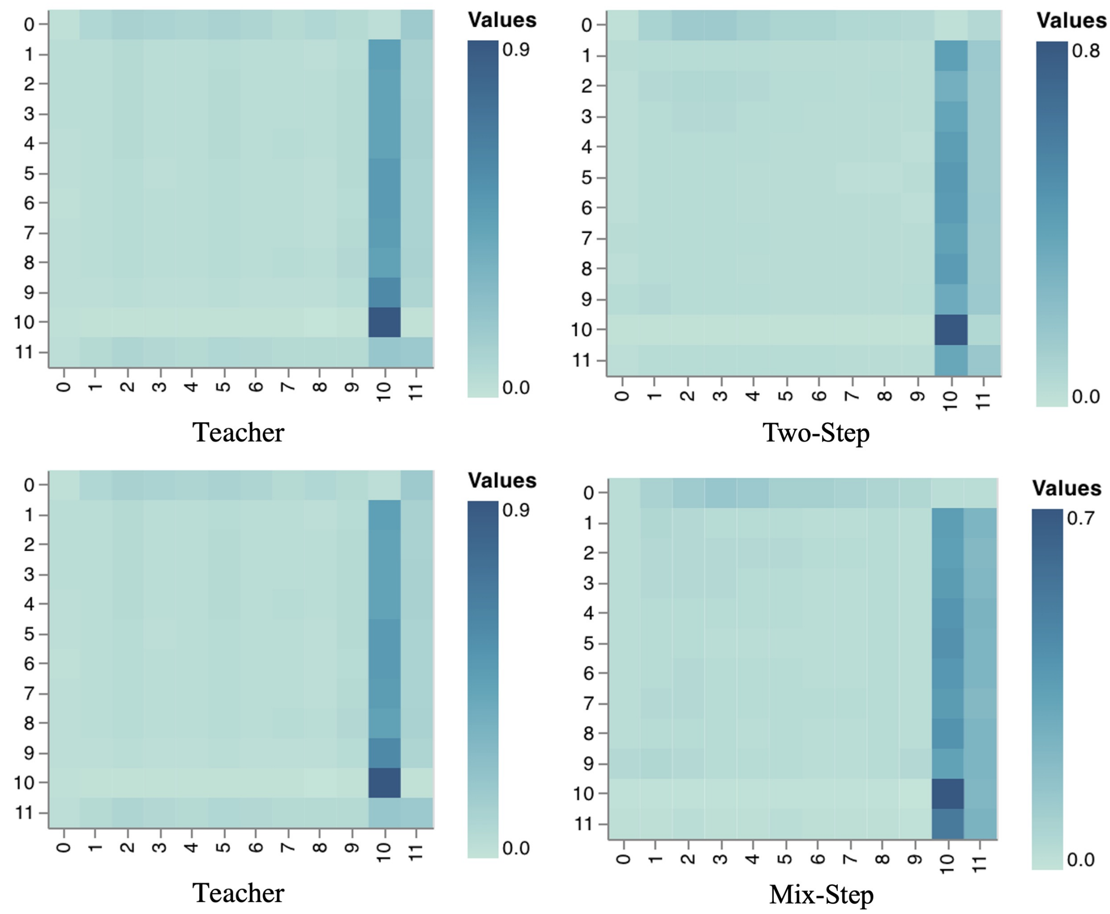

Figure 10: Attention matrices from models using TS and MS methods

From Figure 12 to 14, results of comparison on Two-Step and Mix-Step distillation using small student models and different losses are shown. It has been proven that the benefits of applying the Two-Step method remain when we reduce the student model sizes. The experiments and analysis in this chapter show that the proposed Two-Step method effectively alleviates the over-fitting issue in Knowledge Distillation by reducing student model complexity.

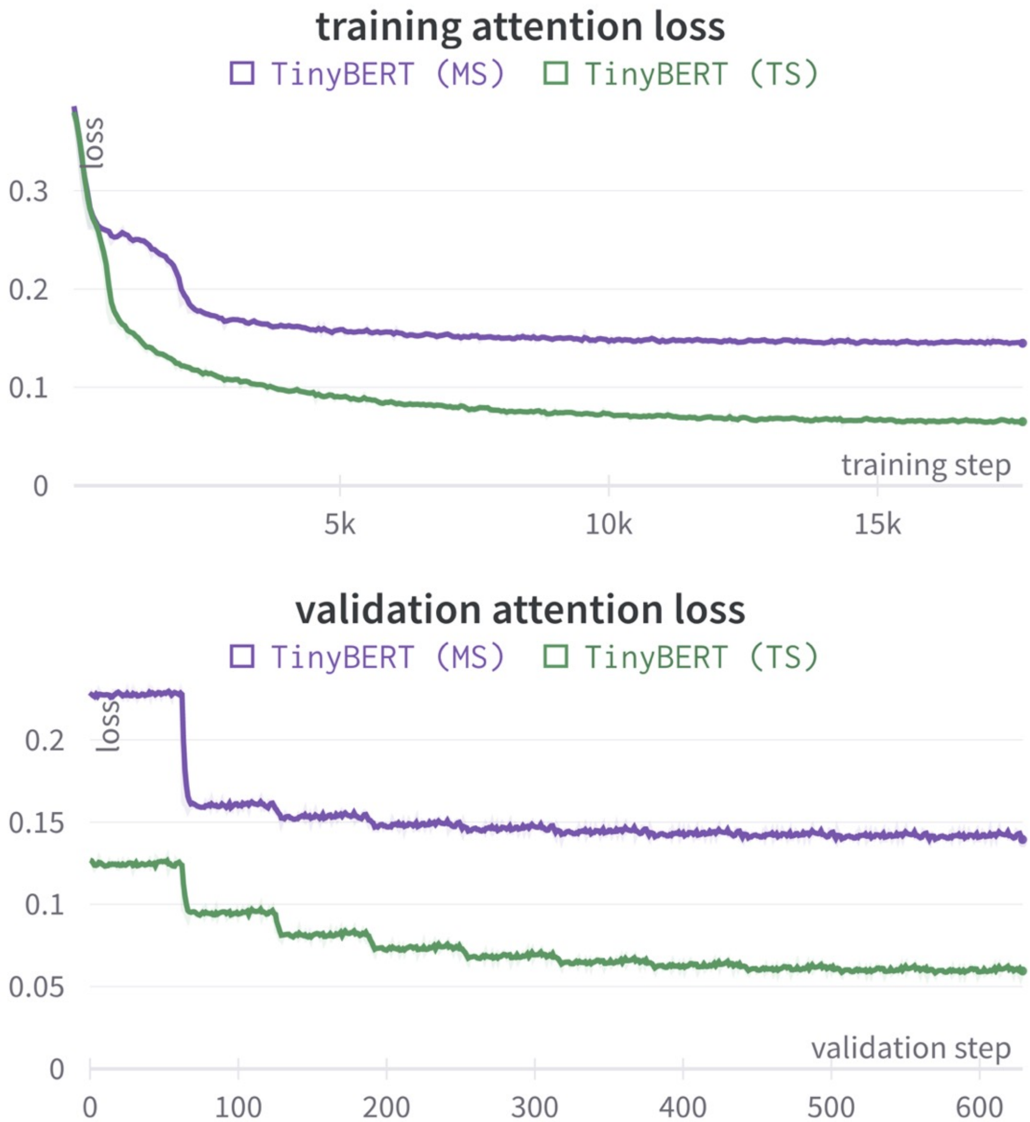

Figure 11: Attention loss between teacher and student using different distillation methods

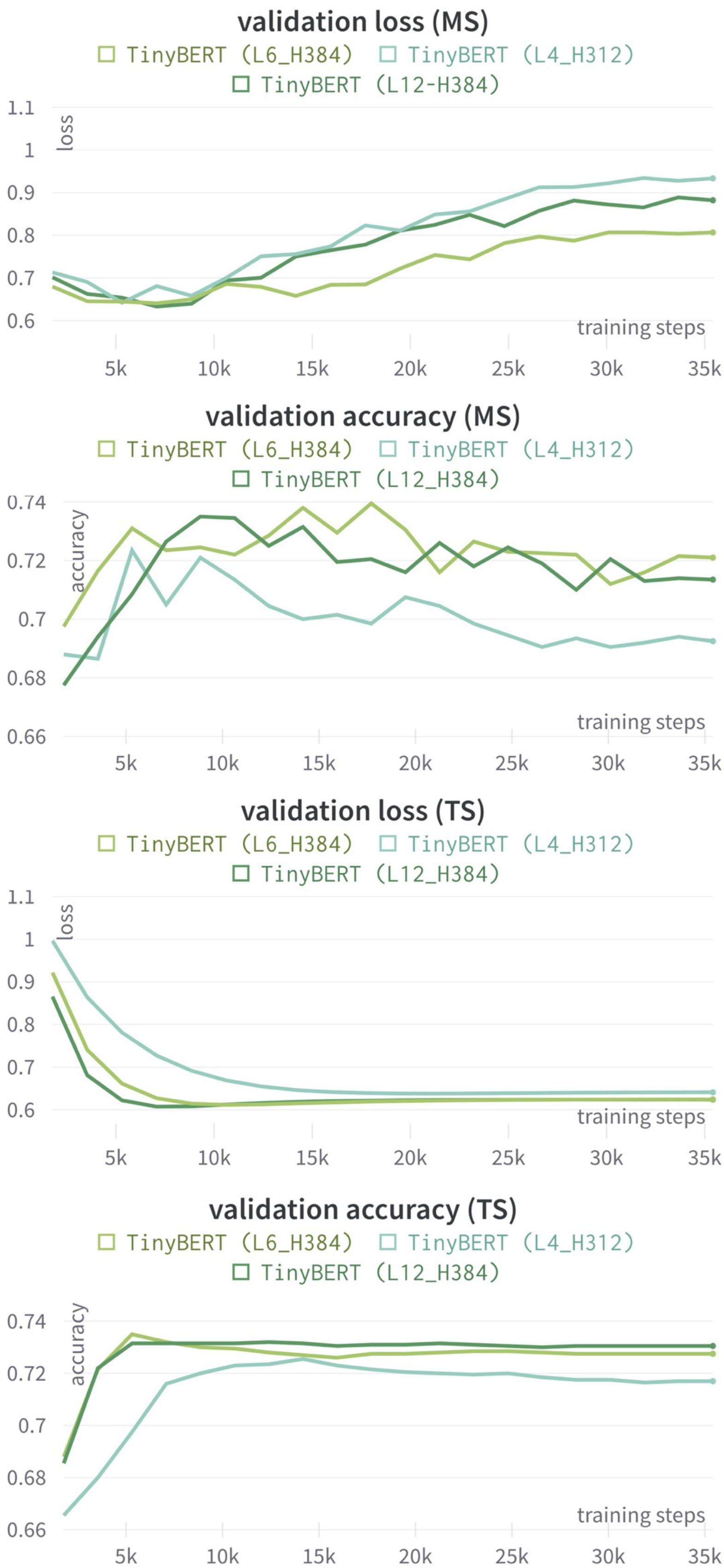

Figure 12: Comparison results using small student models and TinyBERT

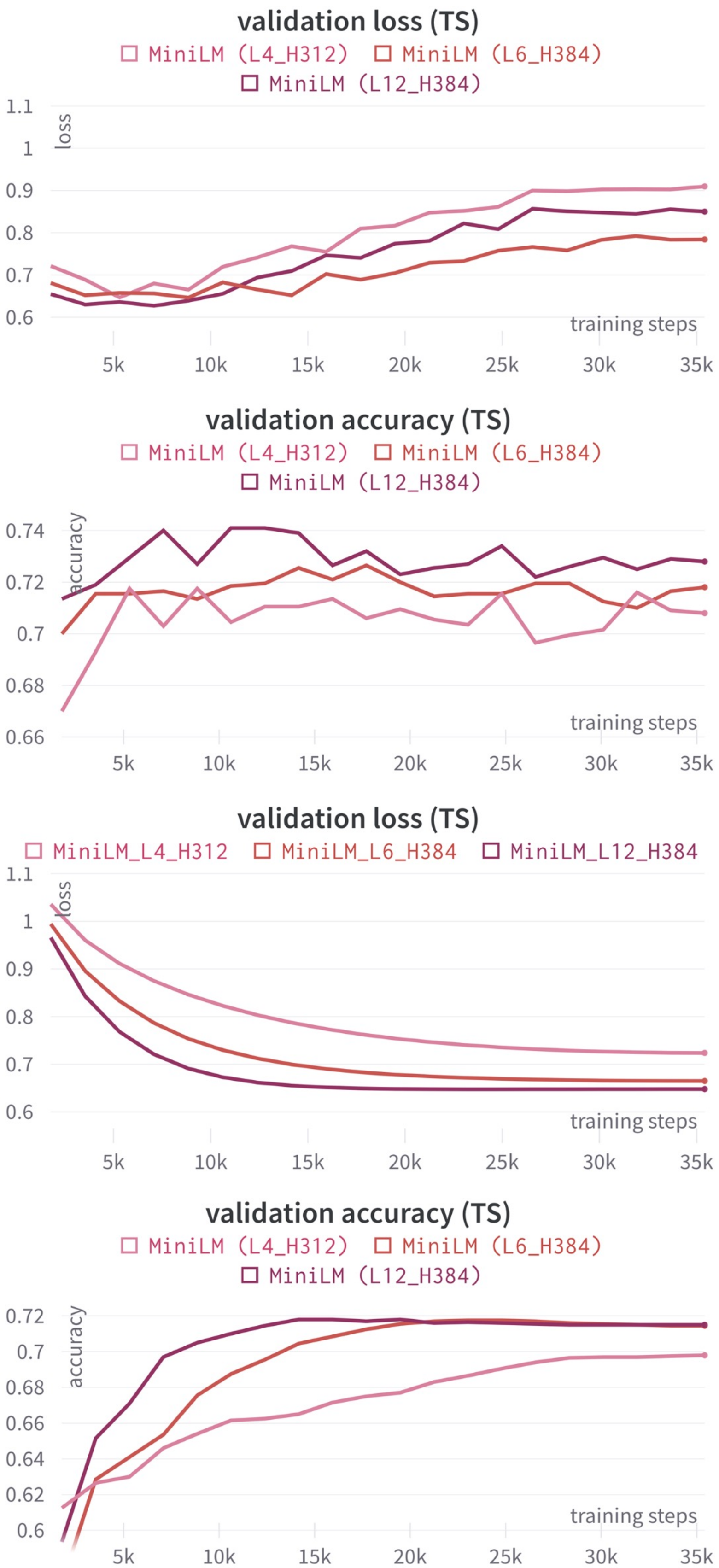

Figure 13: Comparison results using small student models and MiniLM

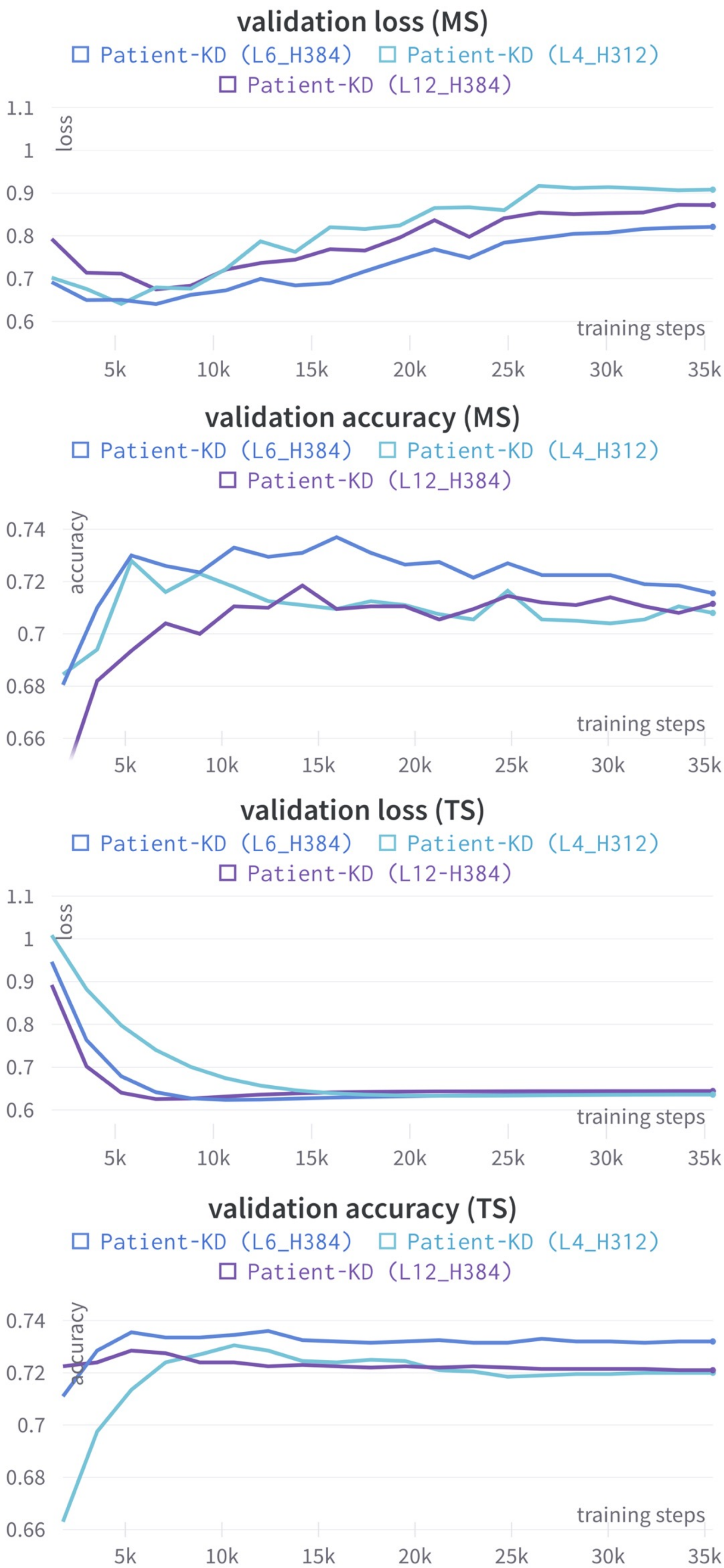

Figure 14: Comparison results using small student models and Patient-KD

6 Conclusion and Future Work

This chapter provides conclusions, discusses limitations of the current work, suggests future work and reflects on the sustainability aspects.

6.1 Conclusions

This degree project develops a framework for Knowledge Distillation on Transformer models. This project uses the framework to compare TinyBERT, MiniLM, and Patient-KD. The results suggest that distillation methods involving more internal behaviors like TinyBERT, and MiniLM usually perform better. It provides Sinch with suggestions on choosing the distillation methods to produce compact student Transformer models. They can use the framework for other exploration and comparison of Knowledge Distillation on Transformer. The project also discovers solutions for over-fitting in Knowledge Distillation. We find out that increasing training data size can alleviate the over-fitting issue but reducing the sizes of student models cannot help. Thus we propose a Two-Step distillation method to solve the problem. The results reveal that the method is effective for solving over-fitting. In Chapter **??**, we analyze the proposed method and show that it can help students better mimic the teacher's internal behaviors, thus sometimes leading to better evaluation performances.

6.2 Limitations

A limitation of this work is that it only performs distillation and evaluates student models on sentiment analysis tasks. That is due to our interest in the task. And we also assume that performances of different distillation methods are similar in other tasks. Another limitation is the size of the teacher model due to computing resources limitation. This work does not consider using the large version of BERT due to the high time consumption and limited computing resources.

6.3 Future work

Although this thesis project has presented encouraging results, there are still good directions for future work.

Evaluation on other downstream tasks

Current conclusions are retrieved from experiments on the sentiment analysis task. Although we try our best to make the results reliable and general, there still might be risks when we apply the conclusions to other tasks. Thus, future work should verify the findings on other tasks.

Implementation of other distillation methods

The current framework supports three methods, and it is desirable to implement other distillation methods for the framework. With more adaptor modules, the framework offers flexibility to combine and compare different adaptors and makes it possible to discover a new combination to produce better student models.

Discovery on over-fitting in distillation

Although over-fitting is a common problem in machine learning, there are limited studies on the over-fitting issue in Knowledge Distillation. It is not clear about the amount of data required for distillation without an over-fitting problem. The performances of the proposed Two-Step method in experiments with different data sizes are also worth exploring. In this work, the data size is increased by 100 times, but maybe fewer data can serve the same effect.

6.4 Reflections

One of the most important results of this work is reducing the resources required for the Transformer NLM. A company can apply these compact models with similar performances in the business at a lower environmental and economic cost than before. Thus this work contributes to the UN SDG numbers 9 and 13 by producing efficient models and the awareness of the energy consumption of models.

References

[1] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu et al., "A survey on vision transformer," IEEE transactions on pattern analysis and machine intelligence, 2022.

[2] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," arXiv preprint arXiv:2202.07125, 2022.

[3] R. M. Rao, J. Liu, R. Verkuil, J. Meier, J. Canny, P. Abbeel, T. Sercu, and A. Rives, "Msa transformer," in International Conference on Machine Learning. PMLR, 2021, pp. 8844–8856.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," Advances in neural information processing systems, vol. 26, 2013.

[5] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," Transactions of the association for computational linguistics, vol. 5, pp. 135–146, 2017.

[6] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," CoRR, vol. abs/1802.05365, 2018.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[9] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[10] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," arXiv preprint arXiv:1801.06146, 2018.

[11] S. Thrun and L. Pratt, Learning to Learn: Introduction and Overview. Boston, MA: Springer US, 1998, pp. 3–17. ISBN 978-1-4615-5529-2.

[12] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on knowledge and data engineering, vol. 22, no. 10, pp. 1345–1359, 2009.

[13] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does BERT look at? an analysis of bert's attention," CoRR, vol. abs/1906.04341, 2019.

[14] W. Hua, Z. Dai, H. Liu, and Q. Le, "Transformer quality in linear time," in International Conference on Machine Learning. PMLR, 2022, pp. 9099–9117.

[15] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, “A survey of model compression and acceleration for deep neural networks,” arXiv preprint arXiv:1710.09282, 2017.

[16] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, “Pruning and quantization for deep neural network acceleration: A survey,” Neurocomputing, vol. 461, pp. 370–403, 2021.

[17] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, “A survey of quantization methods for efficient neural network inference,” arXiv preprint arXiv:2103.13630, 2021.

[18] J. Gou, B. Yu, S. J. Maybank, and D. Tao, “Knowledge distillation: A survey,” International Journal of Computer Vision, vol. 129, no. 6, p. 1789–1819, Mar 2021. doi: 10.1007/s11263-021-01453-z.

[19] Y. J. Kim and H. H. Awadalla, “Fastformers: Highly efficient transformer models for natural language understanding,” CoRR, vol. abs/2010.13382, 2020.

[20] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015.

[21] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers,” CoRR, vol. abs/2002.10957, 2020.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” CoRR, vol. abs/1706.03762, 2017.

[23] W. L. Taylor, ““cloze procedure”: A new tool for measuring readability,” Journalism quarterly, vol. 30, no. 4, pp. 415–433, 1953.

[24] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” arXiv preprint arXiv:1409.0473, 2014.

[25] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” arXiv preprint arXiv:1607.06450, 2016.

[26] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[27] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, “Convolutional sequence to sequence learning,” in International conference on machine learning. PMLR, 2017, pp. 1243–1252.

[28] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” arXiv preprint arXiv:1907.11692, 2019.

[29] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for bert model compression," arXiv preprint arXiv:1908.09355, 2019.

[30] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," 2020.

[31] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," Computational linguistics, vol. 37, no. 2, pp. 267–307, 2011.

[32] X. Zhang, S. Huang, J. Zhao, X. Du, and F. He, "Exploring deep recurrent convolution neural networks for subjectivity classification," IEEE Access, vol. 7, pp. 347–357, 2018.

[33] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, "Mobilebert: a compact task-agnostic bert for resource-limited devices," arXiv preprint arXiv:2004.02984, 2020.

[34] R. Tang, Y. Lu, L. Liu, L. Mou, O. Vechtomova, and J. Lin, "Distilling task-specific knowledge from bert into simple neural networks," arXiv preprint arXiv:1903.12136, 2019.

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., "Pytorch: An imperative style, high-performance deep learning library," Advances in neural information processing systems, vol. 32, 2019.

[36] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz et al., "Transformers: State-of-the-art natural language processing," in Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations, 2020, pp. 38–45.

[37] W. Falcon and The PyTorch Lightning team, "PyTorch Lightning," 3 2019.

[38] L. Biewald, "Experiment tracking with weights and biases," 2020, software available from wandb.com.

[39] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," arXiv preprint arXiv:1609.08144, 2016.

[40] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," arXiv preprint arXiv:1711.05101, 2017.