Understanding Representation Learning Paradigms with Applications to Low Resource Text Classification

by

Siddhant Garg

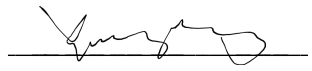
Submitted in partial fulfillment of
the requirements for the degree of

Masters in Computer Sciences

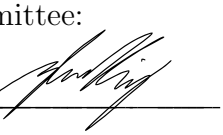
at the

University of Wisconsin-Madison

May 2, 2020

Advisor: Yingyu Liang 

Approval committee:

Prof. Xiaojin Zhu 

Prof. Mohit Gupta 

Acknowledgements

I want to express my deepest gratitude and respect to my thesis and research advisor Yingyu Liang for his guidance and support during my Masters. He has always motivated me and encouraged my ideas and suggestions towards achieving my research goals. I would also like to thank professors Xiaojin Zhu and Mohit Gupta for serving as the committee members on my thesis.

I would like to thank Goutham Ramakrishnan and Arka Sadhu, who have always been willing to help and give their best suggestions on my research. My sincere thanks to Rohit Kumar Sharma, Varun Thumbe, Adarsh Kumar, Vibhor Goel, Mehmet Furkan Demirel and Shengchao Liu for being my collaborators on various projects during my time at UW Madison.

Finally, a special thanks to my family for always supporting and motivating me to deliver my best.

Abstract

A crucial component of modern machine learning systems is learning input representations which can be used for prediction tasks. The expensive cost of labelling and easy availability of unlabelled data has led to the popularity of representation learning techniques on unlabelled data. In thesis we present two ideas in the domain of representation learning. Firstly, we show that self-supervised representation learning approaches like variational auto-encoders and masked self-supervision can be viewed as imposing a regularization on the representation via a learnable function. We present a discriminative theoretical framework for analysing the underlying assumptions and sample complexities of representation learning via such functional regularizations. Our results show that functional regularization on unlabelled data can prune the hypothesis space and reduce the sample complexity of labelled data. We then consider the domain of NLP where fine-tuning pre-trained sentence embedding models like BERT has become the default transfer learning approach. We propose an alternative transfer learning approach called SimpleTran for low resource text classification characterized by small sized datasets. We train a simple sentence embedding model on the target dataset, combine its output embedding with that of the pre-trained model via concatenation or dimension reduction, and finally train a classifier on the combined embedding either by fixing the embedding model weights or training the classifier and the embedding models end-to-end. With end-to-end training, SimpleTran outperforms fine-tuning on small and medium sized datasets with negligible computational overhead. We provide theoretical analysis for our method, identifying conditions under which it has advantages.

Contents

List of Tables

Chapter 1

Introduction

Learning representations forms the backbone of a majority of modern machine learning techniques [1]. These learned representations of the data make it easier to extract useful information from it which can be useful for predictors built over it. Assuming that the input is generated from some underlying data distribution, a good representation often successfully captures the posterior distribution of the underlying factors which explain the corresponding output. While supervised learning tasks implicitly extract the complex representations in the input, the term representation learning has popularly been used for learning representations on unlabelled data.

As mentioned in Bengio et al. [1], one of the key challenges in representation learning has been the difficulty of establishing a clear training objective unlike that in classification tasks. There has been significant research [2, 3, 4, 5, 6, 7] in designing techniques to learn richer representations from the data which can possibly be even generalized to other applications by means of transfer learning.

These learned representations capture the structure in the input which is effective to learn target downstream tasks like regression and classification on labelled data. This paradigm has been shown to achieve superior empirical performance over prediction models trained solely on labelled data, especially when large-scale labelled datasets are unavailable. With increasing model sizes, in terms of the number of

trainable parameters, more labelled data is required for efficient learning. The expensive cost of labelling and easy availability of unlabelled data has led to the popularity of this representation learning approach.

In this thesis we work on two different aspects of representation learning: one is to provide a theoretical framework for analyzing a popular form of representation learning through self-supervision, and the other is to augment the representations learned from pre-trained NLP models with domain specific knowledge to improve transfer learning for low resource text classification applications. We now motivate the two aspects of our work below.

1.1 Motivation for a theoretical framework

Several popular techniques for this representation learning paradigm can be viewed as special cases of self-supervised learning, which poses a supervised learning task over unlabelled data by predicting a subset of information from the remaining. For example, variational auto-encoders [8, 9], that learn a latent variable space for reconstructing the input using unlabelled data, have been extensively used in several computer vision [10] and NLP applications [11]. Masked self-supervision, which learns representations from unlabelled data by hiding or distorting a portion of the input and then reconstructing it, is another popular technique that has proved effective in training powerful language models [12, 13, 14].

In contrast to the popularity and impressive empirical performance of these representation learning methods in practical applications, there is lack of a strong theoretical understanding for them. While one may naturally assume that using unlabelled data for representation learning will lead to a reduction in the size of labelled data required for the target task, theoretically there is ambiguity as to why the representations learned from the unlabelled data should transfer well to the target task.

In this work, we explore some natural questions: *Under what conditions on the data*

distribution is self-supervised representation learning able to reduce the amount of labelled data necessary for the downstream target task? How much unlabelled data is required for learning meaningful representations? Can we estimate the minimum size of labelled data required for the downstream task?. Our aim here is to move towards understanding these representation learning paradigms from a theoretical perspective. We formulate these representation learning approaches as imposing a regularization on the representation via a learnable function. We present a discriminative theoretical framework for analysing the underlying assumptions and sample complexities of representation learning via such functional regularizations.

1.2 Transfer learning for low resource NLP

After gaining some theoretical insights on self-supervised representation learning, we shift our focus to the domain of natural language processing. As previously mentioned, BERT [15] trains a transformer [16] architecture using masked self-supervised learning. Fine-tuning (FT) powerful pre-trained models like BERT has recently become the de facto standard for text classification tasks. This entails learning a classifier on the sentence embedding from a pre-trained model while fine-tuning the pre-trained model at the same time. Typically, FT significantly improves the performance on the target data, since pre-training is done on general domain datasets, while the target data stems from special domains with significant semantic differences.

While FT has been shown to be a simple and effective strategy for large datasets which typically have hundreds of thousands of training points, recent works [17] show that FT in a low resource domain may be unstable having a high variance due to lack of enough data to specialize the general semantics learned by the pre-trained model to the target domain. Other works [18, 19] have tried to improve upon fine-tuning pre-trained models like BERT when the target datasets are small. Further, [20] discuss that fine-tuning all layers of the pre-trained model maybe sub-optimal for small datasets.

Many real world NLP applications have small or medium-sized datasets from special domains, such as finance, political science, and medicine unlike many popularly studied applications like Machine Translation and Question Answering in the academic research community which have large datasets. In this work, we consider the task of text classification in a low resource setting which is characterised by the availability of only a small number of training examples. We propose a simple and efficient method called SimpleTran [21] for transferring pre-trained sentence embedding models for low resource datasets from specific domains. We train a simple sentence embedding model on the target dataset, combine its output embedding with that of the pre-trained model via concatenation or dimension reduction, and finally train a classifier on the combined embedding either by fixing the embedding model weights or training the classifier and the embedding models end-to-end.

We first present our formulation of functional regularization and some sample complexity bounds under our theoretical framework in Chapter 2. We then present our transfer learning approach SimpleTran in Chapter 3 and present theoretical and empirical analysis on it. We finish by concluding our work in Chapter 4 and mentioning avenues of future improvements.

Chapter 2

Functional Regularization for Representation Learning

We provide a unified framework for analyzing these representation learning techniques, by formulating a problem of learning input representations through a learnable regularization function imposed on the representation. We term this problem as *representation learning via functional regularization*. Here, representations are learned jointly on unlabelled data and labelled data, the former being used in an auxiliary task that jointly learns the representation and the regularization function, and the latter being used in the typical supervised learning target task.

We show that several self-supervision based representation learning techniques, with applications to vision, natural language and robotics, can be formulated in our framework, thereby allowing a theoretical analysis of a wide variety of practical applications.

We present a PAC-style [22] discriminative framework to analyse the underlying assumptions in self-supervised representation learning and provide insights on the importance of the structure in the data and the choice of hypothesis classes for learning meaningful representations. To quantify the sizes of unlabelled and labelled data required for learning meaningful representations, we perform sample

complexity analysis under different assumptions on the model and data distributions. In particular, we show that functional regularization using unlabelled data can prune the model hypothesis class for learning representations and reduce the sample complexity of labelled data for the target task. We theoretically quantify the reduction of sample complexity by functional regularization, and give a concrete example showing this significant reduction.

We provide empirical support to our theoretical framework through experiments on synthetically generated data and real data from an application in NLP of sentence-pair classification. The major novel contributions of our work are:

- We present a theoretical framework of representation learning via functional regularization, which unifies several practical representation learning techniques like variational auto-encoders and masked self-supervision under a single formulation.

- We analyse the underlying assumptions on the data distribution and hypothesis classes and provide sample complexity bounds on unlabelled and labelled data under our framework. We show that when the hypothesis classes are properly chosen, functional regularization can prune the representation hypothesis space and reduce sample complexity of labelled data.

- We provide concrete examples showing the benefit of functional regularization and empirical support on synthetic and real-world data for our analysis.

2.1 Problem Definition

Consider data $\{(x_i, y_i)\}$ from a distribution $\mathcal{D}$ over the domains $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^d$ is the input feature space and $\mathcal{Y}$ is the label space. The objective is to learn a function $p : \mathcal{X} \to \mathcal{Y}$ which maps the input x to the output y. This can be done in two steps by first learning a representation function $\phi = h(x) \in \mathbb{R}^p$ over the input and then learning a predictor $y = f(\phi) \in \mathcal{Y}$ on the representation.

Let $\mathcal{H}, \mathcal{F}$ denote the hypothesis classes for h and f respectively. Let ℓ_c denote the loss function for the predictor. For the case of classification and regression problems, ℓ_c models the classification and prediction losses respectively.

We study a setting when the representation function $\phi = h(x)$ is learned by imposing a regularization on the representation through a learnable function g. Let $\mathcal{G}$ denote the hypothesis class for g. The functions g and h are jointly learned through an auxiliary task where the loss (regularization loss) ℓ_r is formulated using $g(h(x))$.

We use this two step learning paradigm to learn the representation h through an auxiliary task which does not require a labelled dataset. This auxiliary task is posed as an unsupervised learning task (or rather self-supervised learning task) depending only on the input x. Thus, h can be learned on labelled data examples without their labels or on a separate set of unlabelled data examples. The labelled data points are then used to learn f. When using unlabelled data in addition to labelled data, the two sets of input data can be from the same or different distributions.

Note that a similar two step learning paradigm is to first learn h via an auxiliary task by imposing a functional regularization g, and then jointly learn f and h by fine-tuning from this starting point. This uses unlabelled data to find a good starting point for h and then searches its neighborhood for the optimal h. For simplicity, we restrict our analysis to the case where obtaining a good starting point in $\mathcal{H}$ and then searching for the optimal in its neighborhood are both done in a single step over the unlabelled data. Insights from our work can be directly applied to this setting of fine-tuning.

2.1.1 Practical Examples of Functional Regularization

Here we show that several self-supervised representation learning strategies can be viewed as imposing a learnable function to regularize the representations being learned.

Auto-encoders use an encoder network e_ϕ to map the input x to a lower dimensional

space z and a decoder network d_θ to reconstruct the input back from z using a MSE loss $\mathcal{L}_{MSE} = \mathbb{E}_{x \in \mathcal{D}} ||x - d_\theta(e_\phi(x))||^2$. One can view d_θ as regularizing the feature representation $z = e_\phi(x)$ through this loss thereby giving an analogy $d_\theta \rightarrow g$, $e_\phi \rightarrow h$ and $\mathcal{L}_{MSE} \rightarrow L_r$ to our formulation.

Variational auto-encoders(VAE) encode the input x as a distribution $q_\phi(z|x)$ over a parametric latent space z instead of a single point and sample from it to reconstruct x using a decoder network $p_\theta(x|z)$. The encoder network $q_\phi(z|x)$ is used to model the underlying mean μ_z and co-variance matrix σ_z of the distribution over z. VAEs are trained by minimising a loss function

$$\mathcal{L}_x(\theta, \phi) = -\mathbb{E}_{z \sim q_\phi(z|x)}[log\ p_\theta(x|z)] + \mathbb{KL}(q_\phi(z|x)||p(z))$$

where p is specified as $\mathcal{N}(0, 1)$, the prior distribution over z. One can view $p_\theta(x|z)$ as a learnable regularization on the distribution of z in the loss $\mathcal{L}_x(\theta, \phi)$ thereby giving an analogy to our formulation of functional regularization.

Masked self-supervision techniques, in abstract terms, cover a portion of the input and then predict the masked input portion. More concretely, say the input $x = [x_1, x_2, \ldots, x_d]$ is masked as $x' = [x_1, \ldots, x_i, 0, \ldots, 0, x_j, \ldots, x_d]$ and a function g is learned to predict the masked input $[x_{i+1}, \ldots, x_{j-1}]$ over an input representation $h(x)$. This function g used to reconstruct x, can be viewed as imposing a regularization on $h(x)$ and naturally fits our formulation.

Techniques imposing explicit regularizations on the representation h being learned, often use an ℓ_p norm penalty on $h(x)$ i.e, $||h(x)||_p^p$ to the prediction loss while jointly training f and h. This can be viewed as a trivial relaxation of our problem formulation using a fixed regularization function $g(h(x)) = ||h(x)||_p^p$.

2.1.2 Structure Properties in Data

We now motivate the importance of the underlying data distribution on learning meaningful representations useful for a downstream through a simple illustrative

example. Consider a regression problem where masked self-supervision is used as the functional regularization by masking the last coordinate of the input. We consider labelled input pairs (x_i, y_i) where $x = [x_1, \ldots, x_d] \in R^d$ and the label is given by $y = \sum_{i=1}^{i=d} x_i$. We consider two special types of data distributions $\mathcal{D}_x$ over the input x:

- Each co-ordinate of x is drawn i.i.d. from a standard Gaussian distribution such that $x_i \sim \mathcal{N}(0, I)\ \forall i \in [1, d]$
- All co-ordinates of x are equal to a scalar drawn from a standard Gaussian distribution ($x_1 = \ldots = x_d = k \sim \mathcal{N}(0, I)$)

Under the first data distribution, learning g to predict the masked x_d will not be able to learn a meaningful representation over x since x_d is independent from all other coordinates of x. While under the second data distribution, learning g to predict the masked x_d will directly learn the the underlying representation as being a single coordinate of x. Thus, functional regularization will not help prune the hypothesis space $\mathcal{H}$ in the first case while in the second case, it is able to prune $\mathcal{H}$ to a single representation function h which is also the optimal function for predicting y.

This example illustrates the two extreme abstractions of inherent structure in the input data distribution $\mathcal{D}$. For a practical application, the structure in $\mathcal{D}$ will influence the benefits of using functional regularization in terms of the fraction of hypotheses $\in \mathcal{H}$ that can be be pruned. We now develop a formal framework to quantify the benefits of using functional regularization and the assumptions on the data distribution $\mathcal{D}$ and the hypotheses classes $\mathcal{H}, \mathcal{F}, \mathcal{G}$ under which it helps.

2.2 A Formal Framework

For clarity of notation, we present and describe our framework using functional regularization through masked self-supervised learning as the auxiliary task to learn the representation. The last coordinate x_d of the input $x = [x_1, x_2, \ldots, x_d]$ is masked $x' = [x_1, x_2, \ldots, x_{d-1}, 0]$ and a regularization function g is used to predict $x_d =$

$g(h(x'))$. For simplicity, suppose $\mathcal{X} = \{0,1\}^d, \mathcal{Y} = \{0,1\}$, and the losses ℓ_c and ℓ_r are 0-1 loss.

We denote the regularization loss for the auxiliary task by $L_r(h, g; x)$. For example, in our case of masked self-supervised learning, $L_r(h, g; x) = l_r(g(h(x')), x^d)$. It can be similarly defined for auto-encoders and other regularization methods on the representation h.

Definition 1. Given the loss $L_r(h, g; x)$ of a representation function h and a regularization function g on an input point x, denote the loss of h and g on a distribution $\mathcal{D}_X$ over $\mathcal{X}$, or on a set of unlabelled data samples U respectively as

$$L_r(h, g; \mathcal{D}_X) = \mathbb{E}_{x \sim \mathcal{D}_X} [L(h, g; x)] \tag{2.1}$$

$$L_r(h, g; U) = \frac{1}{|U|} \sum_{x \in U} L(h, g; x) \tag{2.2}$$

The loss of the representation function h on $\mathcal{D}_X$ and the set of unlabelled data samples U is given by

$$L_r(h; \mathcal{D}_X) = \min_{g \in \mathcal{G}} L(h, g; \mathcal{D}_X) \tag{2.3}$$

$$L_r(h; U) = \min_{g \in \mathcal{G}} L(h, g; U) \tag{2.4}$$

$L_r(h; U)$ can be viewed as a formal notion of incompatibility of a representation function h over the set of unlabelled data points U. Intuitively this shows how well the best regularization function can reconstruct x_d from the set U using a representation function h. We now define a notation for the set of representation functions whose incompatibility is at most some given value τ.

Definition 2. Given a value $\tau \in [0, 1]$, we define

$$\mathcal{H}_{\mathcal{D}_X, L_r}(\tau) = \{h \in \mathcal{H} : L_r(h; \mathcal{D}_X) \leq \tau\} \tag{2.5}$$

$$\mathcal{H}_{U, L_r}(\tau) = \{h \in \mathcal{H} : L_r(h; U) \leq \tau\} \tag{2.6}$$

Given the prediction loss function l_c for the target task, the prediction loss of $f \circ h$ on a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ or the set of labelled data samples S is given by

$$L_c(f, h; \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[l_c(f \circ h(x), y) \right] \tag{2.7}$$

$$L_c(f, h; S) = \frac{1}{|S|} \sum_{(x,y)\in S} l_c(f \circ h(x), y) \tag{2.8}$$

2.3 Sample Complexity Analysis

Here we present sample complexity results under different assumptions of the hypothesis classes and the underlying data distributions. In particular we show how unlabelled data and a suitable representation loss $L_r(h; U)$ can reduce the number of labelled data examples required.

2.3.1 Realizable Case

For simplicity, we begin with the realizable case, where the hypothesis classes contain functions g^*, h^*, f^* with a zero prediction and regularization loss. Here we consider that the unlabelled U and labelled S samples are from the same domain distribution $\mathcal{D}_X$. We derive the following Theorem.

Theorem 1. *Suppose there exist $h^* \in \mathcal{H}, f^* \in \mathcal{F}, g^* \in \mathcal{G}$ such that $L_c(f^*, h^*; \mathcal{D}) = 0$ and $L_r(h^*, g^*; \mathcal{D}_X) = 0$. Then a set U of m_u unlabelled examples and a set S of m_l labelled examples is sufficient to learn the prediction to an error ϵ with probability $1 - \delta$, where*

$$m_u = \frac{1}{\epsilon} \left[\ln |\mathcal{H}| + \ln |\mathcal{G}| + \ln \frac{2}{\delta} \right] \tag{2.9}$$

$$m_l = \frac{1}{\epsilon} \left[\ln |\mathcal{H}_{\mathcal{D}_X, L_r}(\epsilon)| + \ln |\mathcal{F}| + \ln \frac{2}{\delta} \right] \tag{2.10}$$

In particular, with probability at least $1 - \delta$, all hypotheses $h \in \mathcal{H}, f \in \mathcal{F}$ with $L_c(f, h; S) = 0$ and $L_r(h; U) = 0$ will have $L_c(f, h; \mathcal{D}) \leq \epsilon$.

Proof. The probability that given hypotheses g and h with $L_r(h, g; \mathcal{D}_X) \geq \epsilon$ has $L_r(h, g; U) = 0$ is at most $(1-\epsilon)^{m_u} \leq \frac{\delta}{2|\mathcal{H}||\mathcal{G}|}$ for the given value of m_u. Therefore, using the union bound, the number of unlabelled examples is sufficient to guarantee that with probability at least $1 - \delta/2$, only g and h with $L_r(h, g; \mathcal{D}_X) \leq \epsilon$ have $L_r(h, g; U) = 0$. Then only hypotheses $h \in \mathcal{H}_{\mathcal{D}_X, L_r}(\epsilon)$ have $L_r(h; U) = 0$. The number of labelled examples then similarly guarantees that with probability $1-\delta/2$, none of the hypotheses whose true prediction loss is at least ϵ have an empirical loss of 0, proving the theorem. □

Remark. The theorem shows that, if the target function $f^* \circ h^*$ is indeed perfectly correct for prediction and $h^* \circ g^*$ has a zero regularization loss (perfect masked prediction), then optimizing the prediction and the regularization loss to 0 over the given number of data examples is sufficient to learn an accurate predictor (in the PAC learning sense).

This theorem bounds the numbers of unlabelled and labelled examples needed to achieve the PAC guarantee. In particular, it bounds the number of labelled examples needed based on the effect of the functional regularization by the unlabelled data. For our setting, the regularization is helpful for learning the predictor, when the regularization prunes away a large fraction of $\mathcal{H}$ so that $\mathcal{H}_{\mathcal{D}_X, L_r}(\epsilon)$ is small and thus we do not require a large number of labelled examples to find a good representation function among them.

When the hypothesis classes are infinite, we can replace the sizes of the hypothesis classes with other complexity measures, such as their metric entropy, their VC-dimensions, etc. Here we provide a bound using VC-dimension, and prove a similar bound using metric entropy in the Appendix A.1. Let $d(\mathcal{C})$ denote the VC-dimension of the family of functions $\mathcal{C}$. Note that $d(\mathcal{C}_1 \circ \mathcal{C}_2) = O(d(\mathcal{C}_1) + d(\mathcal{C}_2) \ln(d(\mathcal{C}_1) + d(\mathcal{C}_2)))$.

Theorem 2. *Suppose there exist $h^* \in \mathcal{H}, f^* \in \mathcal{F}, g^* \in \mathcal{G}$ such that $L_c(f^*, h^*; \mathcal{D}) = 0$ and $L_r(h^*, g^*; \mathcal{D}_X) = 0$. Then a set U of m_u unlabelled examples and a set S of m_l labelled examples is sufficient to learn the prediction to an error ϵ with probability*

$1-\delta$, *where*

$$m_u = \frac{1}{\epsilon}\left[d(\mathcal{H}\circ\mathcal{G})\ln\frac{16}{\epsilon}+\ln\frac{4}{\delta}\right] \tag{2.11}$$

$$m_l = \frac{1}{\epsilon}\left[d(\mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon)\circ\mathcal{F})\ln\frac{16}{\epsilon}+\ln\frac{4}{\delta}\right] \tag{2.12}$$

In particular, with probability at least $1-\delta$, *all hypotheses* $h\in\mathcal{H}, f\in\mathcal{F}$ *with* $L_c(f,h;S)=0$ *and* $L_r(h;U)=0$ *will have* $L_c(f,h;\mathcal{D})\leq\epsilon$.

Proof. By the VC-dimension theory, we know that the probability that there exist g and h with $L_r(h,g;\mathcal{D}_X)\geq\epsilon$ and $L_r(h,g;U)=0$ is at most $\delta/2$ for the given value of m_u. Therefore, with probability at least $1-\delta/2$, only g and h with $L_r(h,g;\mathcal{D}_X)\leq\epsilon$ have $L_r(h,g;U)=0$. Then only hypotheses $h\in\mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon)$ have $L_r(h;U)=0$. The number of labelled examples then similarly guarantees that with probability $1-\delta/2$, none of the hypotheses whose true prediction loss is at least ϵ have an empirical loss of 0, proving the theorem. □

2.3.2 Unrealizable Case

The assumptions that the regularization and prediction losses are 0 are usually impractical due to noise in the data distribution. Realistically we may assume that there exist ground-truth functions that can make the regularization and prediction losses small. We begin by considering a setting where the prediction loss can be zero while the regularization loss is not.

Theorem 3. *Suppose there exist* $h^*\in\mathcal{H}, f^*\in\mathcal{F}, g^*\in\mathcal{G}$ *such that* $L_c(f^*,h^*;\mathcal{D})=0$ *and* $L_r(h^*,g^*;\mathcal{D}_X)\leq\epsilon_r$. *Then a set* U *of* m_u *unlabelled examples and a set* S *of* m_l *labelled examples is sufficient to learn the prediction to an error* $2\epsilon+\epsilon_c$ *with probability* $1-\delta$, *where*

$$m_u = \mathcal{O}\left(\frac{1}{\epsilon^2}\left[d(\mathcal{H}\circ\mathcal{G})\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right]\right) \tag{2.13}$$

$$m_l = \mathcal{O}\left(\frac{1}{\epsilon}\left[d(\mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon_r+\epsilon)\circ\mathcal{F})\ln\frac{1}{\epsilon}+\ln\frac{1}{\delta}\right]\right) \tag{2.14}$$

In particular, with probability at least $1-\delta$*, the hypotheses* $f \in \mathcal{F}, h \in \mathcal{H}$ *that optimize* $L_c(f,h;S)$ *subject to* $L_r(h,g;U) \leq \epsilon_r + \epsilon$ *for some* $g \in \mathcal{G}$ *satisfy*

$$L_c(f,h;\mathcal{D}) \leq \epsilon$$

Proof. With m_u unlabelled examples, it is guaranteed that with probability $1-\delta/2$, all $h \in \mathcal{H}$ and $g \in \mathcal{G}$ satisfy $|L_r(h,g;U) - L_r(h,g;\mathcal{D}_X)| \leq \epsilon$. In particular, $L_r(h^*,g^*;U) \leq L_r(h^*,g^*;\mathcal{D}_X) + \epsilon \leq \epsilon_r + \epsilon$. So $g^* \in \mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon_r+\epsilon)$, and thus the optimal value $L_c(f,h;S)$ subject to $L_r(h,g;U) \leq \epsilon_r+\epsilon$ for some $g \in \mathcal{G}$ is 0. Then the labelled sample size m_l (using standard VC bounds) implies that with probability at least $1-\delta/2$, for all $h \in \mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon_r+\epsilon)$ and all $f \in \mathcal{F}$, only those with $L_c(f,h;\mathcal{D}) \leq \epsilon$ can have $L_c(f,h;S) = 0$. The theorem statement thus follows. □

We are now ready to present the result for the setting where both the optimal prediction and regularization losses are non-zero.

Theorem 4. *Suppose there exist* $h^* \in \mathcal{H}, f^* \in \mathcal{F}, g^* \in \mathcal{G}$ *such that* $L_c(f^*,h^*;\mathcal{D}) \leq \epsilon_c$ *and* $L_r(h^*,g^*;\mathcal{D}_X) \leq \epsilon_r$*. Then a set* U *of* m_u *unlabelled examples and a set* S *of* m_l *labelled examples is sufficient to learn the prediction to an error* $2\epsilon + \epsilon_c$ *with probability* $1-\delta$*, where*

$$m_u = \mathcal{O}\left(\frac{1}{\epsilon^2}\left[d(\mathcal{H}\circ\mathcal{G})\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right]\right) \tag{2.15}$$

$$m_l = \mathcal{O}\left(\frac{1}{\epsilon^2}\left[d(\mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon_r+2\epsilon)\circ\mathcal{F})\ln\frac{1}{\epsilon} + \ln\frac{1}{\delta}\right]\right) \tag{2.16}$$

In particular, with probability at least $1-\delta$*, the hypotheses* $f \in \mathcal{F}, h \in \mathcal{H}$ *that optimize* $L_c(f,h;S)$ *subject to* $L_r(h,g;U) \leq \epsilon_r + \epsilon$ *for some* $g \in \mathcal{G}$ *satisfy*

$$L_c(f,h;\mathcal{D}) \leq L_c(f^*,h^*;\mathcal{D}) + 2\epsilon$$

Proof. With m_u unlabelled examples, it is guaranteed that with probability $1-\delta/4$, all $h \in \mathcal{H}$ and $g \in \mathcal{G}$ satisfy $|L_r(h,g;U) - L_r(h,g;\mathcal{D}_X)| \leq \epsilon$. In particular,

$L_r(h^*, g^*; U) \leq L_r(h^*, g^*; \mathcal{D}_X) + \epsilon \leq \epsilon_r + \epsilon$. Then the labelled sample size m_l (using standard VC bounds) implies that with probability at least $1 - \delta/2$, all hypotheses $h \in \mathcal{H}_{\mathcal{D}_X, L_r}(\epsilon_r + 2\epsilon)$ and all $f \in \mathcal{F}$ have $L_c(f, h; S) \leq L_c(f, h; \mathcal{D}) + \epsilon$. Finally, by using Hoeffding's bounds, with probability at least $1 - \delta/4$, we have

$$L_c(f^*, h^*; S) \leq L_c(f^*, h^*; \mathcal{D}) + \mathcal{O}\left(\sqrt{\frac{1}{m_l} \ln \frac{1}{\delta}}\right)$$

Therefore, with a probability of at least $1 - \delta$, the hypotheses $f \in \mathcal{F}, h \in \mathcal{H}$ that optimizes $L_c(f, h; S)$ subject to $L_r(h, g; U) \leq \epsilon_r + \epsilon$ for some $g \in \mathcal{G}$ have

$$\begin{aligned} L_c(f, h; \mathcal{D}) &\leq L_c(f, h; S) + \epsilon && (2.17) \\ &\leq L_c(f^*, h^*; S) + \epsilon && (2.18) \\ &\leq L_c(f^*, h^*; \mathcal{D}) + \mathcal{O}\left(\sqrt{\frac{1}{m_l} \ln \frac{1}{\delta}}\right) + \epsilon && (2.19) \\ &\leq L_c(f^*, h^*; \mathcal{D}) + 2\epsilon && (2.20) \end{aligned}$$

This completes the proof of the theorem. □

2.3.3 Different Domains

In practice, it is often the case that the unlabelled data is from a different domain than the labelled data. For example, state-of-the-art NLP systems are often trained on a large general unlabelled corpus (e.g., the entire Wikipedia) and a small specific labelled corpus (e.g., a set of medical records). That is, the unlabelled data U is from a distribution $\tilde{\mathcal{D}}_X$ different from $\mathcal{D}_X$, the marginal distribution of x in the labelled data. In this setting, we show that our previous analysis still holds, with essentially no change from the setting of the same domain distributions.

Theorem 5. *Suppose the unlabelled data U is from a distribution $\tilde{\mathcal{D}}_X$ different from $\mathcal{D}_X$. Suppose there exist $h^* \in \mathcal{H}, f^* \in \mathcal{F}, g^* \in \mathcal{G}$ such that $L_c(f^*, h^*; \mathcal{D}) \leq \epsilon_c$ and $L_r(h^*, g^*; \tilde{\mathcal{D}}_X) \leq \epsilon_r$. Then the same sample complexity bounds as in Theorem 4 hold.*

The complete proof for Theorem 5 is provided in Appendix A.2. The bound implies

that unlabelled data from a different domain can help the learning, as long as there exists a "ground-truth" representation function h^* that is shared across the two domains that leads to a small prediction loss in the labelled data domain and a small regularization loss in the unlabelled data domain. The former (small prediction loss) is typically assumed according to domain knowledge, e.g., for text data, common semantic features are believed to be used across different corpora, and for image data, common visual perception features are believed to be used across different types of images. The latter requires a careful design of the regularization function class $\mathcal{G}$ to ensure h^* can lead to a small regularization loss.

2.3.4 An Illustrative Example

We now provide a concrete example to illustrate the benefits of functional regularization with unlabelled data. Let $\mathcal{H}$ be the class of linear projections from $\mathbb{R}^d$ to $\mathbb{R}^r$ where $r < d/2$, $\mathcal{G}$ be the class of linear projections from $\mathbb{R}^r$ to $\mathbb{R}^d$, and $\mathcal{F}$ be the class of linear functions. That is,

$$h_W(x) = Wx, \text{ where } W \in \mathbb{R}^{r\times d} \tag{2.21}$$

$$g_V(z) = Vz, \text{ where } V \in \mathbb{R}^{d\times r} \tag{2.22}$$

$$f_a(z) = a^\top z, \text{ where } a \in \mathbb{R}^r \tag{2.23}$$

Assume that the rows of W are orthonormal, the columns of V are orthonormal, and $\|a\|_2$=1. In this case, the losses are given by $L_r(h, g; x)$=$\|x - g(h(x))\|_2^2$ and $L_c(f, h; x) = \|y-f(h(x))\|_2^2$. Viewing the data as a composition of knowledge signals and noise, here we assume that the signals are constrained in an r-dimensional subspace. Let $B\in\mathbb{R}^{d\times d}$ be an orthonormal matrix whose columns can be viewed as a set of basis vectors of the input space. The structural assumption implies that the data lies mostly within the subspace spanned by the first r columns $B_{1:r}$ of B, i.e.,

$$\mathbb{E}\|x - B_{1:r}B_{1:r}^\top x\|_2^2 = \epsilon_r$$

for a small $\epsilon_r > 0$. Furthermore, the regression labels are entirely determined by the signal in these dimensions: $y{=}\langle a^*, B_{1:r}^\top x\rangle$ for a ground-truth parameter a^*. If we use $\mathcal{G}$ to perform the functional regularization with $L_r(h, g; \mathcal{D}_X) \leq \epsilon_r$, then the only h's that can satisfy this condition are[1]

$$\mathcal{H}_{\mathcal{D}_X, L_r}(\epsilon_r) = \{OB_{1:r}^\top : O \in \mathbb{R}^{r\times r}, O \text{ is orthonormal}\}.$$

Let $\mathcal{N}_{\mathcal{H}_r}(\epsilon)$ denote the covering number of this set of hypotheses (w.r.t., say the spectral norm). On the other hand,

$$\mathcal{H} \supseteq \{OB_S^\top : O \in \mathbb{R}^{r\times r}, O \text{ is orthonormal}, S \subset [d], |S|{=}r\}$$

where B_S is the sub-matrix of the columns in B with indices in S (Here $[d] = \{1, \ldots, d\}$). We have the relation

$$\mathcal{N}_{\mathcal{H}}(\epsilon) \geq \binom{d}{r}\mathcal{N}_{\mathcal{H}_r}(\epsilon)$$

Using Theorem 6 in the Appendix A.1, this reduction in the size of the hypothesis class leads to reduction a factor of $\Omega\left(\frac{1}{\epsilon}\ln\binom{d}{r}\right) = \Omega\left(\frac{r}{\epsilon}\right)$ in the sample complexity.

2.4 Experimental Support

In this section, we present experiments to lay support to some of our analysis from the theoretical framework. We first present data from a synthetic setting and demonstrate some claims of our theoretical model. Then we present an example using real data to show the benefits of using functional regularization over explicit regularizations on the representation learnt.

[1]Strictly speaking, we need to allow $L_r(h, g; \mathcal{D}_X) \leq \epsilon_r + \epsilon$ for a small $\epsilon > 0$ due to finite unlabelled data. A similar but more complex argument holds for that case. Here we pretend we have infinite unlabelled data to simplify the presentation and better illustrate the intuition, since our focus is on the labelled data.

2.4.1 Controlled Data

Here we choose a regression problem by trying to fit inputs of the form $x = [x_1, x_2 \ldots, x_D] \in R^D$ to their corresponding labels y. We generate input data x to have a structure such that $\sum_{i=1}^{i=n} x_i = 0$ and the output y corresponds to $x_1 + a * x_2 + b * \sum_{i=3}^{i=n} x_i$ where a and b are scalars not equal to 1. We want a representation function to learn a 3-dimensional embedding for the input with the three components corresponding to x_1 , x_2 and $\sum_{i=3}^{i=n} x_i$. The target function f in this case is a simple linear layer which maps the representation generated by h i.e, $h(x)$ to a single output corresponding to y. $f(h(x))$ is trained by first learning h through masked self supervised learning. Specifically, we generate $x \in \mathcal{R}^D$ where we choose $D = 10$. Each co-ordinate $x_i, i \in \{2, \ldots, D\}$ is randomly generated from $[-1, 1]$ and x_1 is chosen such that $\sum_{i=1}^{i=n} x_i = 0$. For our experiments we choose arbitrary values for constants $a = -3, b = 5$.

Effect of Training Data Size We use a 2 hidden layer neural network with 50 hidden units in each hidden layer as h with the final representation being 3 dimensional and a single linear layer regularization function g (can be termed reconstruction in this case) to map the 3-dimensional representation $h(x)$ to a single scalar. We randomly mask either the first or the second component of x by setting it to 0 and use the scalar $g(h(x))$ to predict the masked value. Training g and h together requires only the input without the output y. We then train the weights of h and f together using the output y along with the input x. We use a test data size of 1000, a labelled training data set of size 5000 and an unlabelled data set of size 5000. We vary the number of train data points for our experiments. We use a simple gradient descent algorithm and tune the learning rate $\in [0.0001, 0.01]$ through cross-validation on a held out validation dataset of 500 data points. We use ReLU as the activation function and choose g and f to be linear functions over the representation layer of dimension 3.

Under our framework, this can be viewed as the realizable case where the unlabelled and labelled data is from the same domain with zero prediction and recon-

Train Size	10	100	500	1000	2000	5000
$f(h(x))$	3.24	0.032	81×10^{-4}	40×10^{-4}	9×10^{-4}	7×10^{-4}
$g(h(x)) \rightarrow f(h(x))$	2.56	0.017	25×10^{-4}	8×10^{-4}	5×10^{-4}	4×10^{-4}

Table 2.1: Performance of end-to-end training and masked self-supervision on varying the training dataset size. Metric reported is the MSE loss on the test dataset averaged over 5 training runs.

struction loss. This is because choosing $g(h(x)) = -\sum_{i=1}^{i=3} h(x)_i$ and $f(h(x)) = h(x)_1 + a * h(x)_2 + b * h(x)_3$ can give perfect reconstruction and prediction when $[h(x)_1, h(x)_2, h(x)_3]$ correspond to $[x_1, x_2, \sum_{i=3}^{i=n} x_i]$ and can be learned by the single linear layer g and f networks we use.

We report the Mean Square Error on the test set (1000 data points) as the metric, average the results over 5 runs and present them in Table 2.1.

From the table, we can see that using masked self-supervision to learn the representation h of the input x yields a lower test loss while predicting y. As the size of the labelled data set available for training increases, the performance gap between the end-to-end training and training after masked prediction approach each other. However, it can be observed that when the labelled training data size is small, representation learning via the learnable linear g function helps in the output prediction.

Role of $\mathcal{H}$ and $\mathcal{G}$ in $\mathcal{H}_{S,L_r}(\tau)$ From our bounds of the realizable setting (Theorem 2), the pruning of the hypothesis space $\mathcal{H}$ depends on the quantity $\mathcal{H}_{S,L_r}(\tau)$ denoting the set of hypotheses $h \in \mathcal{H}$ with a maximum reconstruction loss $L_r(h; U)$ of τ. $\mathcal{H}_{S,L_r}(\tau)$ involves the representation function h explicitly, and the reconstruction function g implicitly through the reconstruction loss L_r. Here we want to show that the benefits of functional regularization for representation learning depend on the choices of the hypothesis space $\mathcal{H}$ and $\mathcal{G}$.

We choose $\mathcal{H}$ as the fully connected layer network which maps x to a 3 dimensional representation. This choice is motivated so as to restrict the representations to learn more complicated dependencies on the input x. We experiment with 2 different

Loss	L_r			L_p		
Size	1k	2k	5k	1k	2k	5k
g_1	2.5×10^{-14}	1.8×10^{-14}	1.5×10^{-14}	4.81	2.81	1.98
g_2	3.2×10^{-4}	3.3×10^{-4}	2.9×10^{-4}	19.71	20.42	18.72

Table 2.2: Performance of masked self-supervision on different reconstruction functions on changing the size of the unlabelled data. Metric reported is MSE averaged over 5 training runs. Size refers to the size of unlabelled dataset used for training $h(g(x))$.

reconstruction functions g: g_1, g_2 where $g_1 = w_1 * h_1 + w_2 * h_2 + w_3 * h_3$ and $g_2 = w_1 * {h_1}^3 + w_2 * {h_2}^3 + w_3 * {h_3}^3$ where $w = [w_1, w_2, w_3]$ represents the learnable weights of g. In these experiments, once g and h are trained, we freeze the weights of h and train only f using the output y along with the input x. We experiment using a similar training setting to that of Table 2.1 and present the results in Table 2.2. We use 1000 labelled data points and vary the number of unlabelled data points for learning the representations.

In this case the reconstruction loss from using g_1 on $h(x)$ can be zero while that from using g_2 cannot since h can only learn a linear function over the input. To have a small reconstruction loss, g_2 will force h to learn the representation $h(x) = [\sqrt[3]{x_1}, \sqrt[3]{x_2}, \sqrt[3]{\sum_{i=3}^{i=n} x_i}]$ which is difficult given its shallow capacity. Hence when the weights of h are frozen and f is learnt, the loss on the test data is much higher when using g_2 over g_1. This shows that the choice of the optimal g and h depend on the both $\mathcal{H}, \mathcal{G}$.

2.4.2 Real Data

We consider the application of sentence pair classification in Natural Language Processing. We use the Microsoft Research Paraphrase Corpus [23][2] which has sentence pairs with annotations of whether the two sentences are semantically equivalent. This dataset has approximately 3.7k and 1.7k sentence pairs in the train and test splits respectively. Here we specifically choose the MRPC dataset as it has a smaller

[2] https://www.microsoft.com/en-us/download/details.aspx?id=52398

size of labelled training data in comparison to most NLP datasets. To show the empirical benefits of using unlabelled data in addition to the limited train data available, we use the popular BERT [13] language model. BERT, based on a transformer architecture, has been pre-trained using a masked token self-supervision task which involves masking a portion of the input sentence and using BERT to predict the masked tokens. This pre-training is done over a large corpus of text (approximately 2 billion words) and hence we can view the pre-trained BERT, under our framework, as having already pruned a large fraction of the hypothesis space of $\mathcal{H}$ for learning the representation on the input text.

Training Details We use the 12-layer BERT Base uncased model for our experiments with an Adam optimizer having a learning rate $2e^{-5}$. We perform end to end training over the training data. The number of fine-tuning epochs is a hyper-parameter tuned over the validation dataset. We report the accuracy and F1 scores as the metric on the test data averaged over three runs. When randomly initialising the weights of BERT, we use a standard normal distribution with mean 0 and standard deviation of 0.02 for the layer weights and we set all the biases to zero vectors. We set the layer norms to have weights as a vector of ones with a zero vector as the bias. When adding the l_p penalty on the BERT representations on randomly initialising the weights, we choose an appropriate weighting function λ to make the training loss a sum of the cross entropy classification loss and λ times the l_p norm of the BERT representation. The λ is chosen $\in [0.001, 1000]$ by validation over a set of 300 data points randomly sampled from the training split. We use the huggingface transformers repository [3] for our experiments.

Training Settings To quantify the benefits of using unlabelled data for pre-training as compared to direct end-to-end training using the labelled data, we compare the performance of fine-tuning the pre-trained BERT with training a randomly initialised BERT from scratch on the MRPC train data. For the latter setting, we use three different loss formulations to further study the benefits of regularization

[3] https://github.com/huggingface/transformers

Train Data Size	200	500	1000	2000	3668
BERT-FT	**68.1 / 80.6**	**71.0 / 80.6**	**72.7 / 81.8**	**74.9 / 82.4**	**80.3 / 85.7**
Random	64.1 / 74.8	64.7 / 75.66	67.0 / 80.1	68.9 / 79.0	68.9 / 79.3
Random-ℓ_1	54.7 / 65.1	62.6 / 75.5	63.6 / 76.7	63.4 / 76.6	66.3 / 79.6
Random-ℓ_2	65.3 / 78.6	66.4 / 79.7	65.3 / 78.6	65.0 / 78.4	66.5 / 79.9

Table 2.3: Performance of fine-tuning pre-trained BERT (BERT-FT) and end-to-end training of a randomly initialised BERT on varying the training dataset size. Metrics reported are the Accuracy/F1 scores on the test dataset. The training data size is 3668 sentence pairs.

on the text representation being learnt: (i) the Cross-Entropy loss $\mathcal{L}_{CE}$ on the predicted output (ii) $\mathcal{L}_{CE}$ along with a ℓ_1 norm penalty on the representation (i.e, the 768-dimensional representation from BERT corresponding to the [CLS] token) (iii) $\mathcal{L}_{CE}$ along with a ℓ_2 norm penalty on the representation. We refer to these three different loss formulations as Random, Random-ℓ_1 and Random-ℓ_2 respectively for notational simplicity. We are also interested in studying how decreasing the labelled data size can impact the training performance. We present the results in Table 2.3.

Results From the table, we can infer that the performance of training BERT from pre-trained weights is better than the performance of training the BERT architecture from randomly initialised weights. When viewed under our framework, this empirically shows the benefits of using a learnable regularization function over fixed functions like the ℓ_1 or ℓ_2 norms of the representation. Note that for the explicit norm regularized settings we do not pre-train BERT again with those loss functions over the unlabelled data as BERT pre-training is very expensive.

On increasing the training data size, we can observe that the performance of all the four training modes increases. However, we can see that the performance improvement of Random, Random-ℓ_1 and Random-ℓ_2 is marginal when compared to the improvement in BERT Fine-tuning. The latter can be attributed to the fact that the pre-trained weights of BERT are adjusted by specialising them towards the target data domain. Under our framework, we can view this as searching for the best representation function h in the neighborhood of the pre-trained weights which have already pruned away a large fraction of the hypothesis space $\mathcal{H}$.

Apart from the results reported in Table 2.3, we also experimented by keeping the BERT weights fixed and only training the classifier. We observe that under such a setting, when we use a small training set, the model is unable to converge to a different minima than the initialisation as similarly observed by [24]. With very small train data sizes, we can also see that the performance of fine-tuned BERT is superior to the other cases of randomly initialising the BERT weights. Thus, we can conclude that even a small labelled data set can align the BERT weights correctly for the target domain as is evidenced by the high F1 scores.

2.5 Related Work

Representation learning techniques, which can be viewed as imposing a functional regularization, have been used in various applications across computer vision tasks. Self-supervision has been used to learn representations from images through variational auto-encoders [10, 25] and auxiliary tasks such as masked image patch prediction [26], image rotations [27], pixel colorization [28], context prediction of image patches [29, 30, 31], etc. Videos have also been used to provide self-supervision through visual tracking for learning representations in vision tasks [32].

Representation learning via self-supervision finds practical use in several robotics applications [33, 34, 35, 36, 37]. Masked self-supervised learning has lead to learning powerful language models like BERT [13] and RoBERTa [14] in NLP. Recent work [38, 39] has used self-supervision for other tasks like extractive summarization and learning dialogues in natural language processing.

[40] present a discriminative theoretical framework for analysing semi-supervised learning which uses both unlabelled and labelled data either through transductive learning (where the goal is to label the unlabelled data) or through inductive learning (where the goal is to find a mapping function from the input to output space using both the splits of the data). [41] present a theoretical framework to analyse unsupervised representation learning techniques which can be posed as a contrastive

learning problem using semantically similar data points sampled from the same latent class. The generalisation bounds on this contrastive unsupervised representation learning approach were improved by [42]. [43] provide a theoretical analysis of unsupervised learning through a discriminative framework with applications to dictionary learning and spectral auto-encoders.

Chapter 3

SimpleTran: Transfer Learning for Low Resource Text Classification

A popular trend in NLP recently [44] has been combating the issue of resource scarceness and developing algorithms for low resource settings. We consider the task of text classification in a low resource setting which is characterised by the availability of only a small number of training examples. Fine-tuning (FT) powerful pre-trained models like BERT has recently become the de facto standard for text classification tasks. FT has been shown to be a simple and effective strategy for several datasets like GLUE [45], DBpedia [46], Sogou News [47], etc. which typically have hundreds of thousands of training points.

However, some recent works [17] show that FT in a low resource domain may be unstable having a high variance due to lack of enough data to specialize the general semantics learned by the pre-trained model to the target domain. Other works [18, 19] have tried to improve upon fine-tuning pre-trained models like BERT when the target datasets are small. The popularity of FT and lack of work on transfer learning methods beyond FT leads us to some natural questions: *Is there an alternative simple efficient method to enhance the specialization of pre-trained models to the target special domains for small-sized datasets? When will it have significant advantages*

over fine-tuning?

We propose a simple and efficient method called SimpleTran for transferring pre-trained sentence embedding models for low resource datasets from specific domains. First, we train a simple sentence embedding model on the target dataset (which we refer to as the *domain specific model*). We combine the embeddings from this model with those from a pre-trained model using one of three different combination techniques: Concatenation, Canonical Correlation Analysis (CCA), and Kernel Canonical Correlation Analysis (KCCA). Once we have the combined representation, we train a linear classifier on top of it in two different ways: 1) by training only the classifier while fixing the embedding models, or 2) by training the whole network (the classifier plus the two embedding models) end-to-end. The former is simple and efficient. The latter gives more flexibility for transferring at the expense of a marginal computational overhead compared to FT.

We perform experiments on seven small to medium-sized text classification datasets over tasks like sentiment classification, question type classification and subjectivity classification, where we combine domain specific models like Text-CNN with pre-trained models like BERT. Results show that our simple and straightforward method is applicable to different datasets and tasks with several advantages over FT. First, our method with fixed embedding models using concatenation and KCCA can achieve significantly better prediction performance on small datasets and comparable performance on medium-sized datasets. It reduces the run time by $30\%-50\%$, without the large memory GPUs required for FT. Second, our method with end-to-end training outperforms FT, with less than 10% increase in the run time and about 1% increase in memory. We provide theoretical analysis for our combination methods, identifying conditions under which they work.

3.1 Problem Definition and Methodology

Let s denote a sentence, and assume that we have a sentence embedding model $v_{1s} = f_1(s)$ which is pre-trained in a source domain and maps s to a vector $v_{1s} \in \mathbb{R}^{d_1}$.

Here f_1 is assumed to be a large and powerful model such as BERT. Given a set of labeled training sentences $\mathcal{S} = \{(s_i, y_i)\}_{i=1}^m$ from a target domain, our goal is to use f_1 and $\mathcal{S}$ to learn a classifier that works well on the target domain. [1]

SimpleTran first trains a sentence embedding model f_2 different from f_1 on $\mathcal{S}$, which is typically much smaller than f_1 and thus can be trained on the small dataset. f_2 can be learned through unsupervised (such as Bag-of-Words) or supervised (such as a text CNN [48] where the last layer is regarded as the sentence embedding) learning techniques. Let $v_{2s} = f_2(s) \in \mathbb{R}^{d_2}$ denote the embedding from this model. Our method combines v_{1s} and v_{2s} to get an adaptive sentence representation $\bar{v}_s$ using one of 3 approaches:

Concatenation: We use $\bar{v}_s^\top = [v_{1s}^\top, v_{2s}^\top]$. We also consider $\bar{v}_s^\top = [v_{1s}^\top, \alpha v_{2s}^\top]$ for some hyper-parameter $\alpha > 0$, to have different emphasis on the two embeddings. Note that previous work (e.g., ELMo [12]) uses concatenation of multiple embeddings, but not for transferring pre-trained representations.

CCA: Canonical Correlation Analysis [49] learns linear projections Φ_1 and Φ_2 into dimension d to maximize the correlations between the projections $\{\Phi_1 v_{1s_i}\}$ and $\{\Phi_2 v_{2s_i}\}$. Formally, we compute

$$(\Phi_1, \Phi_2) = \max_{P_1, P_2} \frac{\mathbb{E}_i[\langle P_1 v_{1s_i}, P_2 v_{2s_i} \rangle]}{\sqrt{\mathbb{E}_i[\|P_1 v_{1s_i}\|_2^2]} \sqrt{\mathbb{E}_i[\|P_2 v_{2s_i}\|_2^2]}}$$

where $\mathbb{E}_i$ is the average over the training data, d is a parameter satisfying $d \leq \min\{d_1, d_2\}$. To maximize the representation power, we use $d = \min\{d_1, d_2\}$. Then we set $\bar{v}_s^\top = \frac{1}{2}\Phi_1 v_{1s_i} + \frac{1}{2}\Phi_2 v_{2s_i}$.

KCCA: Kernel Canonical Correlation Analysis [50] first applies nonlinear projections g_1 and g_2 and then CCA on $\{g_1(v_{1s_i})\}_{i=1}^m$ and $\{g_2(v_{2s_i})\}_{i=1}^m$. The technical details can be found in [50] or [51]. Again, we set $d = \min\{d_1, d_2\}$ and $\bar{v}_s^\top = \frac{1}{2} g_1(v_{1s_i}) + \frac{1}{2} g_2(v_{2s_i})$.

[1] Our method is general enough for longer texts, and easily applicable to multiple pre-trained models

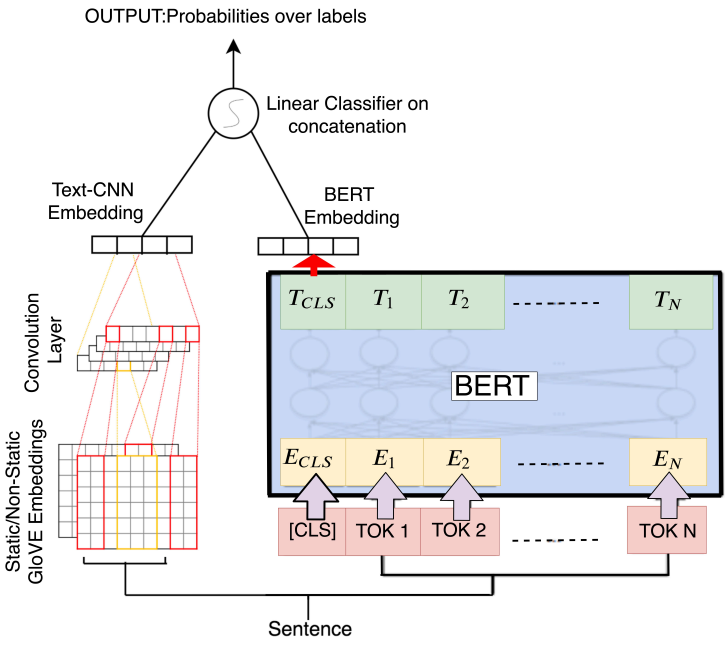


Figure 3.1: ConcatFT of Text-CNN and BERT

Finally, our method trains a linear classifier $c(\bar{v}_s)$ on $\bar{v}_s$ using the target dataset $\mathcal{S}$ in two different ways: (i) Training only the classifier c while fixing the weights of the underlying embedding models f_1 and f_2, (ii) End-to-end training the classifier c as well as f_1 and f_2.

Since CCA and KCCA have computationally expensive projections and concatenation is observed to have strong performance in our experiments, we use the end-to-end training method only for concatenation, which we refer to as **ConcatFT** (See Figure 3.1 for an illustration). Therefore, we have 4 variants of SimpleTran: 3 on fixed embedding models (Concat, CCA, and KCCA), and one with end-to-end training (ConcatFT).

3.2 Theoretical Analysis

For insights on how our methods affect the information contained in the representations for classification, we analyze them under a theoretical model of the data.

Theoretical model Assume there exists a "ground-truth" embedding vector v_s^* for

each sentence s with label y_s, and a linear classifier $f^*(s) = \langle w^*, v_s^* \rangle$ with a small loss $L(f^*) = \mathbb{E}_s[\ell(f^*(s), y_s)]$ w.r.t. some loss function ℓ (such as cross-entropy), where $\mathbb{E}_s$ denotes the expectation over the true data distribution. The good performance of the concatenation method (see Section 3.3) suggests that there exists a linear relationship between the embeddings v_{1s}, v_{2s} and v^*. So our theoretical model assumes: $v_{1s} = P_1 v_s^* + \epsilon_1$, and $v_{2s} = P_2 v_s^* + \epsilon_2$ where ϵ_i's are noises independent of v_s^* with variances σ_i^2's. If $P^\top = [P_1^\top, P_2^\top]$ and $\epsilon^\top = [\epsilon_1^\top, \epsilon_2^\top]$, then the concatenation is $\bar{v}_s = [v_{1s}^\top, v_{2s}^\top]^\top = P v_s^* + \epsilon$. Let $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$.

3.2.1 Concatenation

We have the following theorem about the prediction power of concatenation.

Theorem 6. *Suppose the loss function is λ-Lipschitz for the first parameter, and P has full column rank. Then there exists a linear classifier $\bar{f}$ over $\bar{v}_s$ such that $L(\bar{f}) \le L(f^*) + \lambda\sigma\|(P^\dagger)^\top w^*\|_2$ where $P^\dagger$ is the pseudo-inverse of P.*

Proof. Let $\bar{f}$ have weight $\bar{w} = (P^\dagger)^\top w^*$. Then

$$\begin{aligned}
\langle \bar{w}, \bar{v}_s \rangle &= \langle (P^\dagger)^\top w^*, P v_s^* + \epsilon \rangle \\
&= \langle w^*, P^\dagger P v_s^* \rangle + \langle (P^\dagger)^\top w^*, \epsilon \rangle \\
&= \langle w^*, v_s^* \rangle + \langle (P^\dagger)^\top w^*, \epsilon \rangle.
\end{aligned}$$

Then the difference in the losses is

$$\begin{aligned}
L(\bar{f}) - L(f^*) &= \mathbb{E}_s[\ell(\bar{f}(s), y_s) - \ell(f^*(s), y_s)] \\
&\le \lambda \mathbb{E}_s |\bar{f}(s) - f^*(s)| \\
&= \lambda \mathbb{E}_s |\langle (P^\dagger)^\top w^*, \epsilon \rangle| \\
&\le \lambda \sqrt{\mathbb{E}_s \langle (P^\dagger)^\top w^*, \epsilon \rangle^2} \\
&\le \lambda \sqrt{\mathbb{E}_s \|(P^\dagger)^\top w^*\|_2^2 \|\epsilon\|_2^2} \\
&= \lambda\sigma \|(P^\dagger)^\top w^*\|_2
\end{aligned}$$

where we use the Lipschitz-ness of the loss in the second step, Jensen's inequality in the fourth, and Cauchy-Schwarz inequality in the fifth. □

The assumption of Lipschitz-ness of the loss implies that the loss changes smoothly with the prediction. The assumption on P having full column rank implies that v_{1s}, v_{2s} contain the information of v_s^* and ensures that $P^\dagger$ exists.[2]

Explanation: Suppose P has singular vector decomposition $P = U\Sigma V^\top$, then $\|(P^\dagger)^\top w^*\|_2 = \|(\Sigma^\dagger)^\top V^\top w^*\|_2$. So if the top right singular vectors in V align with w^*, then $\|(P^\dagger)^\top w^*\|_2$ will be small. This means that if P_1 and P_2 together cover the direction w^*, they can capture information important for classification, and then there will be a good linear classifier on the concatenated embeddings (assuming the noise level σ is not too large).

Consider a simple example where v_s^* has 4 dimensions, and $w^* = [1, 1, 0, 0]^\top$, i.e., only the first two dimensions are useful for classification. Suppose $P_1 = \mathrm{diag}(c, 0, 1, 0)$ is a diagonal matrix, so that v_{1s} captures the first dimension with scaling factor $c > 0$ and the third dimension with factor 1, and $P_2 = \mathrm{diag}(0, c, 0, 1)$ so that v_{2s} captures the other two dimensions. Hence we have $(P^\dagger)^\top w^* = [1/c, 1/c, 0, 0]^\top$, and thus $L(\bar{f}) \leq L(f^*) + \sqrt{2}\lambda\frac{\sigma}{c}$. Thus the quality of the classifier is determined by the noise-signal ratio σ/c. If c is small, implying that v_{1s} and v_{2s} contain a large amount of noise, then the loss is large. If c is large, implying that v_{1s} and v_{2s} contain useful information for classification along and very low noise, then the loss is close to that of f^*. Note that $\bar{f}$ can be much better than any classifier that uses only v_{1s} or v_{2s} since the latter only has a part of the features determining the class labels.

3.2.2 Dimension Reduction

A significant observation in our experiments (see Section 3.3) is that CCA on the embeddings leads to worse performance (sometimes much worse) than concatenation. To explain this, the following theorem constructs an example, which shows that

[2]Dropping the full-rank assumption leads to a more involved and non-intuitive analysis

even when v_{1s} and v_{2s} have good information for classification, CCA can eliminate it and lead to bad performance.

Theorem 7. *Let $\bar{v}_s$ denote the embedding for sentence s obtained by concatenation, and $\tilde{v}_s$ denote that obtained by CCA. There exists a setting of the data and w^*, P, ϵ such that there exists a linear classifier $\bar{f}$ on $\bar{v}_s$ with the same loss as f^*, while CCA achieves the maximum correlation but any classifier on $\tilde{v}_s$ is at best random guessing.*

Proof. Suppose we do CCA to d dimensions. Suppose v_s^* has $d+2$ dimensions, each being an independent Gaussian. Suppose $w^* = [1, 1, 0, \ldots, 0]^\top$, and the label is $y_s = 1$ if $\langle w^*, v_s^* \rangle \geq 0$ and 0 otherwise. Suppose $\epsilon = 0$, $P_1 = \text{diag}(1, 0, 1, \ldots, 1)$, and $P_2 = \text{diag}(0, 1, 1, \ldots, 1)$.

Let the linear classifier $\bar{f}$ have weight $[1, 0, \mathbf{0}, 0, 1, \mathbf{0}]^\top$ where $\mathbf{0}$ is the zero vector of d dimensions. Clearly, $\bar{f}(s) = f^*(s)$ for any s, so it has the same loss as f^*.

For CCA, since the coordinates of v_s^* are independent Gaussians, v_{1s} and v_{2s} only have correlation in the last d dimensions. Solving the CCA optimization, the projection matrices for both embeddings are the same $\phi = \text{diag}(0, 0, 1, \ldots, 1)$ which achieves the maximum correlation. Then the CCA embedding is $\tilde{v}_s = [0, 0, (v_s^*)_{3:(d+2)}]$ where $(v_s^*)_{3:(d+2)}$ are the last d dimensions of v_s^*, which contains no information about the label. Therefore, any classifier on $\tilde{v}_s$ is at best random guessing. □

Explanation: Intuitively, v_{1s} and v_{2s} have some common information and each has a set of special information about the correct class labels. If the two sets of special information are uncorrelated, then they will be eliminated by CCA. Now, if the common information is irrelevant in determining the labels, then the best any classifier can do on the CCA embeddings is just random guessing. This is a fundamental drawback of this unsupervised technique, clearly demonstrated by the extreme example in the theorem. In practice, the common information can contain some relevant information for the classification task, thus making CCA

embeddings worse than concatenation but better than random guessing. KCCA can be viewed as CCA on a nonlinear transformation of v_{1s} and v_{2s} where the special information gets mixed non-linearly and cannot be separated out and eliminated by CCA. This explains why the poor performance of CCA is not observed for KCCA in our experiments.

Empirical Verification of Theorem 2 An important insight from the analysis is that when the two sets of embeddings have special information that is not shared with each other but is important for classification, then CCA will eliminate such information and have bad prediction performance. Let $r_{2s} = v_{2s} - \Phi_2^\top \Phi_2 v_{2s}$ be the residue vector for the projection Φ_2 learned by CCA for the special domain, and similarly define r_{1s}. Then the analysis suggests that the residues r_{1s} and r_{2s} contain information important for prediction. We conduct experiments for BERT+CNN-non-static on Amazon reviews, and find that a classifier on the concatenation of r_{1s} and r_{2s} has accuracy 96.3%. This is much better than 81.3% on the combined embeddings via CCA. These observations provide positive support for our analysis.

3.3 Experiments

3.3.1 Datasets

We evaluate our method on different text classification tasks including sentiment classification, question type classification, subjectivity classification, etc. We consider 3 small datasets: derived from Amazon, IMDB and Yelp reviews; and 4 medium-sized datasets: movie reviews (MR), opinion polarity (MPQA), question type classification (TREC) and subjectivity classification (SUBJ). The Amazon, Yelp and IMDB review datasets capture sentiment information from different target domains which is very different from the general text corpora of the pre-trained models and have been used in recent related work [52]. The dataset statistics are summarized in Table 3.1.

- *Amazon*: A Amazon product reviews dataset with 'Positive'/'Negative' re-

views [3].

- *IMDB*: A dataset of movie reviews on IMDB with 'Positive'/'Negative' reviews [3].

- *Yelp*: A dataset of restaurant reviews from Yelp with 'Positive'/'Negative' reviews[3].

- *MR*: A dataset of movie reviews based on sentiment polarity and subjective rating [53][4].

- *MPQA*: An unbalanced dataset (70% negative examples) for opinion polarity detection [54][5].

- *TREC*: A question type classification dataset with 6 classes for questions about a person, location, numeric information, etc. [55][6].

- *SUBJ*: A dataset for classifying a sentence as subjective or objective [56].

Dataset	c	N	Test
Amazon [52]	2	1000	100
IMDB [52]	2	1000	100
Yelp [52]	2	1000	100
MR [53]	2	10662	1067
MPQA [54]	2	10606	1060
TREC [55]	6	5952	500
SUBJ [56]	2	10000	1000

Table 3.1: Dataset statistics. c: Number of classes, N: Dataset size, *Test*: Test set size (if no standard test set, we use a random train/dev/test split of 80/10/10 %)

3.3.2 Models for Evaluation

We choose 2 domain specific models: A Bag-of-Words model that averages word vectors in the sentence to get its embedding; and a Text-CNN [48] with 3 approaches to initialize the word embeddings: (i) randomly initialized which we refer to as CNN-rand (ii) initialized with GloVe vectors and made non-trainable which we refer to as

[3] https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences
[4] https://www.cs.cornell.edu/people/pabo/movie-review-data/
[5] http://mpqa.cs.pitt.edu/
[6] http://cogcomp.org/Data/QA/QC/

CNN-static and (iii) initialized with GloVe vectors and trainable (CNN-non-static). We use 12 layer BERT [15] base uncased model as the pre-trained model. We also experiment with other pre-trained models like GenSen and InferSent and present these results in Appendix B. The complete training details and hyper-parameters are presented in Appendix C.

3.3.3 Results on Small Datasets

On the 3 small datasets, we evaluate variants of SimpleTran in Table 3.2. The key observations from these results are as follows:

- ConcatFT always gets the best performance. Furthermore, even Concat and KCCA always improves over the used baselines (the domain specific and fine-tuned BERT). This demonstrates its effectiveness in transferring the knowledge from the general domain while exploiting the target domain. The CCA variant shows inferior performance, worse than the baselines. Our analysis in Section 3.2 shows that this is because CCA can potentially remove useful information and capture nuisance or noise. This is also further verified empirically at the end of this section.

- Concat is simpler and computationally cheaper than KCCA and achieves better results, hence is the recommended method over KCCA.

- Our method gets better results using better domain specific models (using CNN instead of BOW). This is because better specific models capture more domain specific information useful for classification.

	Amazon				**Yelp**				**IMDB**			
	BERT-FT: 94.00		**Adapter:** 94.25		**BERT-FT:** 91.67		**Adapter:** 93.50		**BERT-FT:** 92.33		**Adapter:** 90.50	
	BOW	**CNN-R**	**CNN-S**	**CNN-NS**	**BOW**	**CNN-R**	**CNN-S**	**CNN-NS**	**BOW**	**CNN-R**	**CNN-S**	**CNN-NS**
Default	79.2	91.1	94.7	95.9	81.3	92.7	95.2	95.8	89.3	93.2	96.6	96.8
ConcatFT	-	**94.0**	**95.7**	**96.8**	-	**96.2**	**97.2**	**98.3**	-	**97.0**	**98.3**	**98.4**
Concat	89.6	93.2	95.3	96.4	89.0	96.5	97.1	98.3	89.3	96.2	98.1	98.3
KCCA	89.1	91.5	94.3	95.8	88.5	91.5	91.9	96.2	88.3	94.1	97.9	97.2
CCA	50.9	79.1	83.6	81.3	50.3	71.5	67.8	69.4	51.0	80.8	83.3	85.0

Table 3.2: Test accuracy for Amazon, Yelp and IMDB datasets. BOW, CNN-R, CNN-S and CNN-NS refers to Bag of Words, text CNN with random initialised, static and non-static word embeddings. Best results in boldface.

The Concat, CCA and KCCA variants of our method require less computational resources than BERT-FT. The total time of our method is the time taken to train the text-CNN, extract BERT embeddings, apply a combination (concatenation, CCA, or KCCA), and train a classifier on the combined embedding. For the Amazon dataset, Concat requires about 125 seconds, reducing around 30% of the 180 seconds for FT BERT. Additionally, our approach has small memory requirements as it can be computed on a CPU in contrast to BERT-FT which requires, at minimum, a 12GB memory GPU. The total time of ConcatFT is 195 seconds, which is less than a 9% increase over FT. It also has a negligible 1.04% increase in memory (the number of parameters increases from 109,483,778 to 110,630,332 due to the text-CNN).

3.3.4 Results on Medium-sized Datasets

We use the CNN-non-static model and omit KCCA due to its inefficient non-linear computations and summarize the results in Table 3.3. Again, ConcatFT achieves the best performance on all the datasets improving the performance of BERT-FT. This improvement comes at small computational overhead. On the MR dataset, ConcatFT requires 610 seconds which is about 9% increase over the 560 seconds of BERT-FT, and recall that it only has about 1% increase in memory. Concat can achieve comparable test accuracy on all the tasks while being much more computationally efficient. On the MR dataset, it requires 290 seconds, reducing about 50% of the 560 seconds for BERT-FT.

	MR	**MPQA**	**SUBJ**	**TREC**
CNN-non-static	80.93	88.38	89.25	92.98
BERT No-FT	83.26	87.44	95.96	88.06
Adapter BERT	85.55	90.40	97.40	96.55
BERT FT	86.22	90.47	96.95	96.40
Concat	85.60	90.06	95.92	96.64
CCA	85.41	77.22	94.55	84.28
ConcatFT	**87.15**	**91.19**	**97.60**	**97.06**

Table 3.3: Test accuracy for medium-sized datasets. Best results on the datasets are highlighted in boldface.

The Adapter approach [20] injects new adapter modules into the pre-trained BERT model, freezes the weights of BERT and trains the adapter module on the tar-

Size	**CNN**	**No-FT**	**FT**	**Concat**	**CCA**	**ConcatFT**
100	76.23	75.69	76.85	73.28	55.78	**78.76**
200	74.95	77.31	79.41	77.74	61.73	**80.02**
500	78.39	78.75	80.38	80.10	58.87	**80.96**
2000	78.30	80.78	82.75	80.89	79.53	**83.94**
4000	79.29	81.96	83.88	83.85	83.37	**84.82**
6000	80.14	82.80	84.72	85.08	85.28	**86.06**
8528	80.93	83.26	86.22	85.60	85.41	**87.15**

Table 3.4: Test accuracy for the MR dataset with different training dataset sizes. FT and No-FT refers to BERT fine-tuned and not fine-tuned respectively.

get data. Therefore, our method with fixed embedding models (Concat, CCA and KCCA) can be directly compared with this since neither fine-tunes the BERT parameters. Interestingly, the Concat variant of our method can outperform the Adapter approach for small datasets and perform comparably on medium sized datasets having 2 clear advantages over the latter:

- We do not need to open the BERT model and access its parameters to introduce intermediate layers and hence our method is modular for a large range of pre-trained models.
- Concat introduces roughly only 1% extra parameters as compared to the $3-4\%$ of the latter thereby being more parameter efficient. Concat-FT which performs end-to-end fine-tuning over the BERT model beats the performance of the Adapter approach for all the datasets.

3.3.5 Effect of Dataset Size

To study the effect of the dataset size on the performance, we vary the training data size in MR dataset via random sub-sampling and then use our method. From Table 3.4, we observe that ConcatFT gets the best results across all training data sizes, significantly improving over BERT-FT. Concat gets performance comparable to BERT-FT on a wide range of dataset sizes, from 500 points on. The performance of CCA improves with more training data as more data leads to less noise and thus less nuisance information in the obtained embeddings (Ref Sec 3.2).

We present a qualitative analysis through examples which SimpleTran is able to correctly classify but the pre-trained and domain specific model fail to classify correctly independently in Appendix B.2.

3.4 Related Work

Here we focus on the recent progress in pre-trained models that can provide sentence embeddings (explicitly or implicitly) [12, 57]. Among these, InferSent [58] is trained on natural language inference data via supervised learning and generalizes well to many different tasks. GenSen [59] aims at learning a general purpose, fixed-length representation of sentences via multi-task learning, which is useful for transfer and low-resource learning. BERT [15] learns language representations via unsupervised learning using a deep transformer architecture thereby exploiting bidirectional contexts.

The standard practice for transferring these pre-trained models is FT the pre-trained model by doing end-to-end training on both the classifier and the sentence embedding model [60, 57, 61, 62]. There exist other more sophisticated transferring methods, but they are typically much more expensive or complicated. For example, [63] does "post-training" the pre-trained model on the target dataset, [20] injects specifically designed new adapter layers, and [64] first trains a deep network classifier on the fixed pre-trained embedding and then fine-tunes it. Our focus is to propose alternatives to FT for the low resource setting with similar simplicity and computational efficiency, and study conditions where it has significant advantages.

Several prior works for transferring machine learning models exist including that by [65] which proposes to use $[x, x, 0]$ and $[x, 0, x]$ as the features for a source and target data point respectively where x is a representation of the point, and then train a classifier on top of the union of the source and target data. There are subsequent variants to this like [66, 67], [68] using auxiliary tasks, [69, 70] using adversarial training, and [71] using semi-supervision to align the source and target

representations and then train on the source labels. A recent work [19] uses phrasal paraphrase relations to improve over BERT FT on small datasets. This however only applies to language understanding tasks which involve para-phrasal relations. [18] show that within-task pre-training can harm the performance of pre-trained models for small datasets. This provides motivation for a transfer learning strategy not involving additional pre-training.

Chapter 4

Conclusion

In this work, we have presented a unified discriminative framework for representation learning under which learning representations can be analysed as imposing a regularization on the representation via a learnable function. We have derived sample complexity results under various assumptions on the hypothesis classes and shown that unlabelled data can be used to prune the hypothesis class and reduce the sample complexity of the labelled data. An interesting future work direction is to explore the algorithmic implications of our theoretical framework. Other ideas may include formulating representation learning algorithms which can match the sample complexity bounds that we have presented.

We also proposed a simple method for transferring a pre-trained sentence embedding model for text classification tasks in a low resource setting. We experimentally demonstrated that our method can transfer knowledge from the pre-trained model and leverage it in the target domain, leading to substantial improvement over baselines on small and medium-sized datasets. We also provided theoretical analysis identifying the success conditions of the method and explaining the experimental results.

Bibliography

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, p. 1798–1828, Aug 2013.

[2] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, p. 1527–1554, July 2006.

[3] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19* (B. Schölkopf, J. C. Platt, and T. Hoffman, eds.), pp. 153–160, MIT Press, 2007.

[4] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, "Efficient learning of sparse representations with an energy-based model," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, (Cambridge, MA, USA), p. 1137–1144, MIT Press, 2006.

[5] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *Journal of Machine Learning Research*, vol. 10, no. 1, pp. 1–40, 2009.

[6] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?," *J. Mach. Learn. Res.*, vol. 11, p. 625–660, Mar. 2010.

[7] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics* (D. van Dyk and M. Welling, eds.), vol. 5 of *Proceedings of Machine Learning Research*, (Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA), pp. 448–455, PMLR, 16–18 Apr 2009.

[8] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[9] M. Tschannen, O. Bachem, and M. Lucic, "Recent advances in autoencoder-based representation learning," *ArXiv*, vol. abs/1812.05069, 2018.

[10] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, (New York, NY, USA), p. 1096–1103, Association for Computing Machinery, 2008.

[11] Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick, "Improved variational autoencoders for text modeling using dilated convolutions," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, (International Convention Centre, Sydney, Australia), pp. 3881–3890, PMLR, 06–11 Aug 2017.

[12] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of NAACL-HLT*, pp. 2227–2237, 2018.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long*

and Short Papers), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[14] 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 pre-training approach," *CoRR*, vol. abs/1907.11692, 2019.

[15] 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.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, pp. 5998–6008, Curran Associates, Inc., 2017.

[17] S. Garg, T. Vu, and A. Moschitti, "Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection," *AAAI*, 2019.

[18] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune BERT for text classification?," *CoRR*, vol. abs/1905.05583, 2019.

[19] Y. Arase and J. Tsujii, "Transfer fine-tuning: A BERT case study," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, (Hong Kong, China), pp. 5393–5404, Association for Computational Linguistics, Nov. 2019.

[20] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*, pp. 2790–2799, 2019.

[21] S. Garg, R. K. Sharma, and Y. Liang, "Simpletran: Transferring pre-trained sentence embeddings for low resource text classification," 2020.

[22] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, 1998.

[23] W. B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases," in *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[24] O. Kovaleva, A. Romanov, A. Rogers, and A. Rumshisky, "Revealing the dark secrets of BERT," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, (Hong Kong, China), pp. 4365–4374, Association for Computational Linguistics, Nov. 2019.

[25] R. Zhang, P. Isola, and A. Efros, "Split-brain autoencoders: Unsupervised learning by cross-channel prediction," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, (United States), pp. 645–654, Institute of Electrical and Electronics Engineers Inc., 11 2017.

[26] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 766–774, Curran Associates, Inc., 2014.

[27] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, 2018.

[28] R. Zhang, P. Isola, and A. Efros, "Colorful image colorization," in *European Conference on Computer Vision*, vol. 9907, pp. 649–666, 10 2016.

[29] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, (USA), p. 1422–1430, IEEE Computer Society, 2015.

[30] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2536–2544, 2016.

[31] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, 2016.

[32] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, (USA), p. 2794–2802, IEEE Computer Society, 2015.

[33] B. Sofman, E. Lin, J. Bagnell, J. Cole, N. Vandapel, and A. Stentz, "Improving robot navigation through self-supervised online learning," *J. Field Robotics*, vol. 23, pp. 1059–1075, 11 2006.

[34] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3406–3413, 2015.

[35] P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," *Conference on Neural Information Processing Systems*, 06 2016.

[36] F. Ebert, C. Finn, A. X. Lee, and S. Levine, "Self-supervised visual planning with temporal skip connections," in *CoRL*, 2017.

[37] E. Jang, C. Devin, V. Vanhoucke, and S. Levine, "Grasp2vec: Learning object representations from self-supervised grasping," in *Proceedings of The 2nd Conference on Robot Learning* (A. Billard, A. Dragan, J. Peters, and J. Morimoto, eds.), vol. 87 of *Proceedings of Machine Learning Research*, pp. 99–112, PMLR, 29–31 Oct 2018.

[38] H. Wang, X. Wang, W. Xiong, M. Yu, X. Guo, S. Chang, and W. Y. Wang, "Self-supervised learning for contextualized extractive summarization," in *Pro-*

ceedings of the 57th Annual Meeting of the Association for Computational Linguistics, (Florence, Italy), pp. 2221–2227, Association for Computational Linguistics, July 2019.

[39] J. Wu, X. Wang, and W. Y. Wang, "Self-supervised dialogue learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3857–3867, Association for Computational Linguistics, July 2019.

[40] M.-F. Balcan and A. Blum, "A discriminative model for semi-supervised learning," *J. ACM*, vol. 57, Mar. 2010.

[41] N. Saunshi, O. Plevrakis, S. Arora, M. Khodak, and H. Khandeparkar, "A theoretical analysis of contrastive unsupervised representation learning," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, (Long Beach, California, USA), pp. 5628–5637, PMLR, 09–15 Jun 2019.

[42] K. Nozawa, P. Germain, and B. Guedj, "Pac-bayesian contrastive unsupervised representation learning," 2019.

[43] E. Hazan and T. Ma, "A non-generative framework and convex relaxations for unsupervised learning," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, (Red Hook, NY, USA), p. 3314–3322, Curran Associates Inc., 2016.

[44] C. Cherry, G. Durrett, G. Foster, R. Haffari, S. Khadivi, N. Peng, X. Ren, and S. Swayamdipta, eds., *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, (Hong Kong, China), Association for Computational Linguistics, Nov. 2019.

[45] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understand-

ing," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*, (Brussels, Belgium), Association for Computational Linguistics, Nov. 2018.

[46] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web Journal*, vol. 6, no. 2, 2015.

[47] C. Wang, M. Zhang, S. Ma, and L. Ru, "Automatic online news issue construction in web environment," in *Proceedings of the 17th WWW*, (New York, NY, USA), p. 457–466, Association for Computing Machinery, 2008.

[48] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1746–1751, 2014.

[49] H. Hotelling, "Relations between two sets of variates.," *Biometrika*, 1936.

[50] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[51] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[52] P. K. Sarma, Y. Liang, and B. Sethares, "Domain adapted word embeddings for improved sentiment classification," in *Proceedings of the 56th ACL*, pp. 37–42, 2018.

[53] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of ACL*, pp. 115–124, 2005.

[54] J. Wiebe and T. Wilson, "Annotating expressions of opinions and emotions in language," *Language Resources and Evaluation*, vol. 39, pp. 165–210, May 2005.

[55] X. Li and D. Roth, "Learning question classifiers," in *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, (Stroudsburg, PA, USA), pp. 1–7, Association for Computational Linguistics, 2002.

[56] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity," in *Proceedings of ACL*, pp. 271–278, 2004.

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

[58] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 670–680, 2017.

[59] S. Subramanian, A. Trischler, Y. Bengio, and C. J. Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," *arXiv preprint arXiv:1804.00079*, 2018.

[60] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339, 2018.

[61] M. Peters, S. Ruder, and N. A. Smith, "To tune or not to tune? adapting pretrained representations to diverse tasks," *arXiv preprint arXiv:1903.05987*, 2019.

[62] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?," *arXiv preprint arXiv:1905.05583*, 2019.

[63] H. Xu, B. Liu, L. Shu, and S. Y. Philip, "Bert post-training for review reading comprehension and aspect-based sentiment analysis," in *Proceedings of the 2019 NAACL: HLT*, pp. 2324–2335, 2019.

[64] R. Wang, H. Su, C. Wang, K. Ji, and J. Ding, "To tune or not to tune? how about the best of both worlds?," *ArXiv*, 2019.

[65] H. Daume III, "Frustratingly easy domain adaptation," in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 256–263, 2007.

[66] Y.-B. Kim, K. Stratos, and R. Sarikaya, "Frustratingly easy neural domain adaptation," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 387–396, 2016.

[67] Y.-B. Kim, K. Stratos, and D. Kim, "Domain attention with an ensemble of experts," in *Proceedings of the 55th ACL (Volume 1: Long Papers)*, pp. 643–653, 2017.

[68] J. Yu and J. Jiang, "Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification," in *Proceedings of EMNLP 2016*, 2016.

[69] Z. Li, Y. Zhang, Y. Wei, Y. Wu, and Q. Yang, "End-to-end adversarial memory network for cross-domain sentiment classification.," in *IJCAI*, pp. 2237–2243, 2017.

[70] X. Chen, Y. Sun, B. Athiwaratkun, C. Cardie, and K. Weinberger, "Adversarial deep averaging networks for cross-lingual sentiment classification," *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 557–570, 2018.

[71] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "Adaptive semi-supervised learning for cross-domain sentiment classification," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3467–3476, 2018.

Appendix A

More Sample Complexity Bounds

A.1 Same Domain, Realizable, Using Metric Entropy

Here we give a bound based on the covering number of the hypothesis classes. Suppose $\mathcal{H}$ is indexed by parameter set Θ_H with norm $\|\cdot\|_H$, $\mathcal{G}$ by Θ_G with norm $\|\cdot\|_G$, and $\mathcal{F}$ by Θ_F with norm $\|\cdot\|_F$. Assume that the losses are L-Lipschitz with respect to the parameters. That is,

$$
\begin{aligned}
|L_r(h_\theta, g; x) - L_r(h_{\theta'}, g; x)| &\leq L\|\theta - \theta'\|_H, \forall g \in \mathcal{G}, x \in \mathcal{X}, \\
|L_r(h, g_\theta; x) - L_r(h, g_{\theta'}; x)| &\leq L\|\theta - \theta'\|_G, \forall h \in \mathcal{H}, x \in \mathcal{X}, \\
|L_c(h_\theta, g; x) - L_r(h_{\theta'}, g; x)| &\leq L\|\theta - \theta'\|_H, \forall g \in \mathcal{G}, x \in \mathcal{X}, \\
|L_c(h, g_\theta; x) - L_r(h, g_{\theta'}; x)| &\leq L\|\theta - \theta'\|_G, \forall h \in \mathcal{H}, x \in \mathcal{X}.
\end{aligned}
$$

Let $\mathcal{N}_{\mathcal{G}}(\epsilon)$ be the ϵ-covering number of $\mathcal{G}$ w.r.t. the associated norm. This is similarly defined for the other function classes.

Theorem 6. *Suppose there exist $h^* \in \mathcal{H}, f^* \in \mathcal{F}, g^* \in \mathcal{G}$ such that $L_c(f^*, h^*; \mathcal{D}) = 0$ and $L_r(h^*, g^*; \mathcal{D}_X) = 0$. Then a set U of m_u unlabeled examples and a set S of m_l labeled examples are sufficient to learn a classifier to an error ϵ with probability $1-\delta$,*

where

$$m_u = \mathcal{O}\left(\frac{1}{\epsilon}\ln\frac{1}{\delta}\left[\ln\mathcal{N}_{\mathcal{H}}\left(\frac{\epsilon}{2L^2}\right)+\ln\mathcal{N}_{\mathcal{G}}\left(\frac{\epsilon}{2L^2}\right)\right]\right) \tag{A.1}$$

$$m_l = \mathcal{O}\left(\frac{1}{\epsilon}\ln\frac{1}{\delta}\left[\ln\mathcal{N}_{\mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon)}\left(\frac{\epsilon}{2L^2}\right)+\ln\mathcal{N}_{\mathcal{F}}\left(\frac{\epsilon}{2L^2}\right)\right]\right). \tag{A.2}$$

In particular, with probability at least $1-\delta$*, all* $h \in \mathcal{H}, f \in \mathcal{F}$ *with* $L_c(f,h;S)=0$ *and* $L_r(h;U)=0$ *have* $L_c(f,h;\mathcal{D}) \leq \epsilon$*.*

Proof. By the basic ϵ-net argument, we know that the probability that there exist g and h with $L_r(h,g;\mathcal{D}_X) \geq \epsilon$ and $L_r(h,g;U)=0$ is at most $\delta/2$ for the given value of m_u. Therefore, with probability at least $1-\delta/2$, only g and h with $L_r(h,g;\mathcal{D}_X) \leq \epsilon$ have $L_r(h,g;U)=0$. Then only the hypotheses $h \in \mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon)$, have $L_r(h;U)=0$. The number of labeled examples m_l similarly ensures that with probability $1-\delta/2$, none of the hypotheses whose true prediction loss is at least ϵ have an empirical loss of 0, yielding the theorem. □

A.2 Different Domains, Unrealizable

Theorem 5. *Suppose the unlabelled data* U *is from a distribution* $\tilde{\mathcal{D}}_X$ *different from* $\mathcal{D}_X$*. Suppose there exist* $h^* \in \mathcal{H}, f^* \in \mathcal{F}, g^* \in \mathcal{G}$ *such that* $L_c(f^*,h^*;\mathcal{D}) \leq \epsilon_c$ *and* $L_r(h^*,g^*;\tilde{\mathcal{D}}_X) \leq \epsilon_r$*. Then the same sample complexity bounds as in Theorem 4 hold.*

Proof. With m_u unlabeled examples, it is guaranteed that with probability $1-\delta/4$, all $h \in \mathcal{H}$ and $g \in \mathcal{G}$ satisfy $|L_r(h,g;U) - L_r(h,g;\tilde{\mathcal{D}}_X)| \leq \epsilon$. In particular, $L_r(h^*,g^*;U) \leq L_r(h^*,g^*;\tilde{\mathcal{D}}_X)+\epsilon \leq \epsilon_r+\epsilon$. Then the labeled sample size m_l (using standard VC bounds) implies that with probability at least $1-\delta/2$, all $h \in \mathcal{H}_{\mathcal{D}_X,L_r}(\epsilon_r+2\epsilon)$ and all $f \in \mathcal{F}$ have $L_c(f,h;S) \leq L_c(f,h;\mathcal{D})+\epsilon$. Finally, by Hoeffding bounds, with probability at least $1-\delta/4$, we have

$$L_c(f^*,h^*;S) \leq L_c(f^*,h^*;\mathcal{D}) + \mathcal{O}\left(\sqrt{\frac{1}{m_l}\ln\frac{1}{\delta}}\right).$$

Therefore, with probability at least $1-\delta$, the $f \in \mathcal{F}, h \in \mathcal{H}$ that optimize $L_c(f, h; S)$ subject to $L_r(h, g; U) \leq \epsilon_r + \epsilon$ for some $g \in \mathcal{G}$ have

$$L_c(f, h; \mathcal{D}) \leq L_c(f, h; S) + \epsilon \tag{A.3}$$

$$\leq L_c(f^*, h^*; S) + \epsilon \tag{A.4}$$

$$\leq L_c(f^*, h^*; \mathcal{D}) + \mathcal{O}\left(\sqrt{\frac{1}{m_l} \ln \frac{1}{\delta}}\right) + \epsilon \tag{A.5}$$

$$\leq L_c(f^*, h^*; \mathcal{D}) + 2\epsilon. \tag{A.6}$$

□

Appendix B

Additional Results of SimpleTran

B.1 Error Bounds

We present comprehensive results along with error bounds on small datasets (Amazon, IMDB and Yelp reviews) in Table B.2, where we evaluate **SimpleTran** using three popularly used pre-trained sentence embedding models, namely BERT, GenSen and InferSent. We present the error bounds on the results for medium sized datasets in Table B.1.

	MR	**MPQA**	**SUBJ**	**TREC**
CNN-non-static	80.93 ± 0.16	88.38 ± 0.28	89.25 ± 0.08	92.98 ± 0.89
BERT No Fine-tuning	83.26 ± 0.67	87.44 ± 1.37	95.96 ± 0.27	88.06 ± 1.90
BERT Fine-tuning	86.22 ± 0.85	90.47 ± 1.04	96.95 ± 0.14	96.40 ± 0.67
Concat	85.60 ± 0.95	90.06 ± 0.48	95.92± 0.26	96.64 ± 1.07
CCA	85.41 ± 1.18	77.22 ± 1.82	94.55 ± 0.44	84.28 ± 2.96
ConcatFT	**87.15 ± 0.70**	**91.19 ± 0.84**	**97.60 ± 0.23**	**97.06 ± 0.48**

Table B.1: Test accuracy (± std dev) for four medium-sized datasets. Best results on the datasets are highlighted in boldface. The domain specific embedding model used is CNN-non-static, and the pre-trained model used is BERT.

B.2 Qualitative Analysis

We present some qualitative examples from the Amazon, IMDB and Yelp datasets on which BERT and CNN-non-static are unable to provide the correct class predictions, while Concat or KCCA can successfully provide the correct class predictions in Table B.3. We observe that these are either short sentences or ones where the content is

				BOW	CNN-rand	CNN-static	CNN-non-static
Amazon	**Default**			79.20 ± 2.31	91.10 ± 1.64	94.70 ± 0.64	95.90 ± 0.70
	BERT	94.00 ± 0.02	ConcatFT	-	94.05 ± 0.23	95.70 ± 0.50	***96.75 ± 0.76***
			Concat	89.59 ± 1.22	93.20 ± 0.98	95.30 ± 0.46	**96.40 ± 1.11**
			KCCA	89.12 ± 0.47	91.50 ± 1.63	94.30 ± 0.46	95.80 ± 0.40
			CCA	50.91 ± 1.12	79.10 ± 2.51	83.60 ± 1.69	81.30 ± 3.16
	GenSen	82.55 ± 0.82	Concat	82.82 ± 0.97	92.80 ± 1.25	94.10 ± 0.70	95.00 ± 1.0
			KCCA	79.21 ± 2.28	91.30 ± 1.42	94.80 ± 0.75	**95.90 ± 0.30**
			CCA	52.80 ± 0.74	80.60 ± 4.87	83.00 ± 2.45	84.95 ± 1.45
	InferSent	85.29 ± 1.61	Concat	51.89 ± 0.62	90.30 ± 1.48	94.70 ± 1.10	95.90 ± 0.70
			KCCA	52.29 ± 0.74	91.70 ± 1.49	95.00 ± 0.00	**96.00 ± 0.00**
			CCA	53.10 ± 0.82	61.10 ± 3.47	65.50 ± 3.69	71.40 ± 3.04
Yelp	**Default**			81.3 ± 2.72	92.71± 0.46	95.25 ± 0.39	95.83 ± 0.14
	BERT	91.67 ± 0.00	ConcatFT	-	96.23 ± 1.04	97.23 ± 0.70	***98.34 ± 0.62***
			Concat	89.03 ± 0.70	96.50 ± 1.33	97.10 ± 0.70	**98.30 ± 0.78**
			KCCA	88.51 ± 1.22	91.54 ± 4.63	91.91 ±1.13	96.2 ± 0.87
			CCA	50.27 ± 1.33	71.53 ± 2.46	67.83 ± 3.07	69.4 ± 3.35
	GenSen	86.75 ± 0.79	Concat	85.94 ± 1.04	94.24 ± 0.53	95.77 ± 0.36	**96.03 ± 0.23**
			KCCA	83.35 ± 1.79	92.58 ± 0.31	95.41 ± 0.45	95.06 ± 0.56
			CCA	57.14 ± 0.84	84.27 ± 1.68	86.94 ± 1.62	87.27± 1.81
	InferSent	85.7 ± 1.12	Concat	50.83 ± 0.42	91.94 ± 0.46	96.10 ± 1.30	**97.00 ± 0.77**
			KCCA	50.80 ± 0.65	91.13 ± 1.63	95.45 ± 0.23	95.57 ± 0.55
			CCA	55.91 ± 1.23	60.80 ± 2.22	54.70 ± 1.34	59.50 ± 1.85
IMDB	**Default**			89.30 ± 1.00	93.25 ± 0.38	96.62 ± 0.46	96.76 ± 0.26
	BERT	92.33 ± 0.00	ConcatFT	-	97.07 ± 0.95	98.31 ± 0.83	***98.42 ± 0.78***
			Concat	89.27 ± 0.97	96.20 ± 2.18	98.10 ± 0.94	**98.30 ± 1.35**
			KCCA	88.29 ± 0.65	94.10 ± 1.87	97.90 ± 0.30	97.20 ± 0.40
			CCA	51.03 ± 1.20	80.80 ± 2.75	83.30 ± 4.47	84.97 ± 1.44
	GenSen	86.41 ± 0.66	Concat	86.86 ± 0.62	95.63 ± 0.47	97.22 ± 0.27	**97.42 ± 0.31**
			KCCA	84.72 ± 0.93	93.23 ± 0.38	96.19 ± 0.21	96.60 ± 0.37
			CCA	51.48 ± 1.02	86.28 ± 1.76	87.30 ± 2.12	87.47 ± 2.17
	InferSent	84.3 ± 0.63	Concat	50.36 ± 0.62	92.30 ± 1.26	97.90 ± 1.37	97.10 ± 1.22
			KCCA	50.09 ± 0.68	92.40 ± 1.11	97.62 ±0.48	**98.20 ± 1.40**
			CCA	52.56 ± 1.15	54.50 ± 4.92	54.20 ± 5.15	61.00 ± 4.64

Table B.2: Test accuracy (± std dev) for small datasets. Default values are performance of the domain specific models. Concat-FT refers to Concat Fine-Tuning and gets the best results (in boldface and italic). Concat, CCA, KCCA correspond to classifier trained on the combined embeddings, with fixed embedding model weights and the best results from these are in boldface.

tied to the specific reviewing context as well as the involved structure to be parsed with general knowledge. Such input sentences thus require combining both the general semantics of BERT and the domain specific semantics of CNN-non-static to predict the correct class labels.

B.3 Comparison with Adapter Modules

We compare our method with the Adapter approach of [20] in Table B.4. Recall that the Adapter approach injects new adapter modules into the pre-trained BERT model, freezes the weights of BERT and trains the weights of the adapter module

on the target data for transferring. The Concat-FT variant of our method which performs end-to-end fine-tuning over the BERT model beats the performance of the Adapter approach for all the datasets.

Correctly classified by KCCA
However-the ringtones are not the best, and neither are the games.
This is cool because most cases are just open there allowing the screen to get all scratched up.
Correctly classified by Concatenation
TNot nearly as good looking as the picture makes it look .
Magical Help .

(a) Amazon

Correctly classified by KCCA
I would have casted her in that role after ready the script .
Predictable , but not a bad watch .
Correctly classified by Concatenation
I would have casted her in that role after ready the script .
Predictable , but not a bad watch .

(b) IMDB

Correctly classified by KCCA
The lighting is just dark enough to set the mood .
I went to Bachi Burger on a friend's recommendation and was not disappointed .
dont go here .
I found this place by accident and I could not be happier .
Correctly classified by Concatenation
The lighting is just dark enough to set the mood .
I went to Bachi Burger on a friend's recommendation and was not disappointed .
I found this place by accident and I could not be happier .

(c) Yelp

Table B.3: Sentences from Amazon, IMDB, Yelp datasets where KCCA and concatenation of BERT and CNN-non-static embeddings succeeds while they individually give wrong predictions.

	Amazon	**Yelp**	**IMDB**	**MR**	**MPQA**	**SUBJ**	**TREC**
BERT-FT	94.00 ± 0.02	91.67 ± 0.00	92.33 ± 0.00	86.22 ± 0.95	90.47 ± 1.04	96.95 ± 0.14	96.40 ± 0.67
Adapter	94.25 ± 0.96	93.50 ± 1.00	90.50 ± 0.58	85.55 ± 0.38	90.40 ± 0.14	97.40 ± 0.26	96.55 ± 0.30
Concat	96.40 ± 1.11	98.30 ± 0.78	98.30 ± 1.35	85.60 ± 0.95	90.06 ± 0.48	95.92 ± 0.26	96.64 ± 1.07
ConcatFT	**96.75 ± 0.76**	**98.34 ± 0.62**	**98.42 ± 0.78**	**87.15 ± 0.70**	**91.19 ± 0.84**	**97.60 ± 0.23**	**97.06 ± 0.48**

Table B.4: Comparing SimpleTran's test accuracy (± std dev) with Adapter on small and medium-sized datasets. Best results are shown in boldface. SimpleTran's ConcatFT results outperform Adapter results for all datasets.

Appendix C

Training Details for SimpleTran

We train domain specific embeddings on the training data and extract the embeddings. We combine these with the embeddings from the pre-trained models and train a regularized logistic regression classifier on top. The classifier is learned on the training data, while using the dev data for hyper-parameter tuning of the regression weights. The classifier can be trained by fixing the weights of the underlying embedding models or training the whole network end-to-end. The performance is tested on the test set. For concatenation, we also tune the hyper-parameter α over the validation data on medium-sized datasets while fixing it to 1 on small datasets. We use test accuracy as the performance metric and report all results averaged over 10 experiments unless mentioned otherwise. The experiments are performed on an NVIDIA Tesla V100 16 GB GPU.

Text-CNN For all datasets, we use filter windows of sizes 3, 4, 5 with 100 feature maps each and a dropout rate of 0.5 trained with $L2$ constraint loss to obtain 384 dimensional sentence embeddings.

BERT We use BERT[1] in two ways: (i) Fine-tuned end-to-end on the train and validation splits of the datasets. BERT fine-tuning results were reported after fine-tuning over the dataset for 20 epochs with early stopping by choosing the best performing model on the validation data. (ii) A classifier learned naïvely on pre-trained BERT model embeddings of 768 dimensions.

[1] https://github.com/google-research/bert

InferSent We use the pre-trained InferSent model to obtain 4096 dimensional sentence embeddings using the implementation provided in the SentEval[2] repository. For all the experiments, InferSent v1 was used.

GenSen Similar to the InferSent model, we use the GenSen model implemented in the SentEval repository to obtain 4096 dimensional sentence embeddings.

We build upon code of the SentEval toolkit and use the sentence embeddings obtained from each of the models: BERT, InferSent and GenSen to perform text classification. A batch size of 128 is used and the classifier is trained for 2 epochs on the training data. Further experimental details are presented below:

- **Concat:** The hyper-parameter α determines the weight corresponding to the domain specific embeddings. We tune the value of α via grid search in the range [0.002, 500] in multiplicative steps of 10 over the validation data.
- **CCA:** The regularization parameter for canonical correlation analysis is tuned via grid search in [0.00001, 10] in multiplicative steps of 10 over the validation data.
- **KCCA**: We use a Gaussian kernel with a regularized KCCA implementation where the Gaussian sigma and the regularization parameter are tuned via grid search in $[0.05, 10]$ and $[0.00001, 10]$ respectively in multiplicative steps of 10 over the validation data.
- **ConcatFT**: We first train the domain specific model independently on the training data of the target dataset. We combine these embeddings with those obtained from BERT by concatenation. We train a simple linear classifier along with both the Text-CNN and BERT models end-to-end. The end-to-end training is done over the training data for 20 epochs with early stopping by choosing the best performing model on the validation data. As the models can update the embeddings while learning the classifier, we do not use the α parameter for the concatenation here.

[2]https://github.com/facebookresearch/SentEval