Learning Transferable Features for Unsupervised Domain Adaptation in Natural Language Processing

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Xia Cui

April 2020

Abstract

Machine Learning is concerned with the problem of learning a model to perform a task. Supervised machine learning is a successful approach to many natural language processing (NLP) tasks and typically requires a large amount of labelled data for training an accurate model. However, it is infeasible to manually annotate sufficient data for training. Training and test data are usually independent. Even we have a well-trained model through training, this supervised model may still face major challenges when the training and test data are distributed differently (i.e., from different domains). Domain Adaptation (DA) considers the problem of coping with the changes in the data distributions: a model is learned to adapt from a domain (i.e., the source domain) to a different domain (i.e., the target domain). In many NLP tasks, we may have labelled data in a well-studied domain, but there might not be any available data from the target domain. This setting refers to unsupervised DA (UDA). In the thesis, we focus on learning transferable features for UDA in NLP.

UDA methods can be classified into two main categories: feature-based and instance-based UDA methods. In the thesis, we have studied both categories and made contributions to the applications in NLP. Our research started with feature-based UDA and first considered pivot selection strategies. Pivots are features common to the source and target domains. Many feature-based approaches rely on the selection of pivots to introduce a common space between two domains. In Chapter 3, we conduct a comparative study on previously proposed pivot selection strategies. we show the limitations of the pivot selection strategies that use heuristics. In Chapter 4, we propose two pivot selection methods. One aims at learning pivots that are common and task-specific, the other aims at learning pivots for imbalanced data. In Chapter 5, we propose a novel UDA method that expands the feature vectors using core-periphery decomposition. In Chapter 6, we consider instance-based UDA methods, especially self-training approaches. We propose a novel UDA method that combines projection learning and self-training. In Chapter 7, we study the problem of negative transfer for multi-source UDA and propose a novel UDA method using domain attention and self-training. In Chapter 8, we conclude the thesis and give an overview of several possible future directions.

Acknowledgements

As a PhD candidate for the last four years, I would like to thank many people who have supported my research. This thesis could not be completed without them.

Firstly, I would like to thank my primary supervisor Prof. Danushka Bollegala. I am grateful and fortunate to have him as my supervisor, I enjoyed working with him. He is talented and shared his expertise in the area with me. He has been very supportive in every aspect of my PhD study.

Secondly, I would like to thank my secondary supervisor Prof. Frans Coenen. He is a very experienced supervisor and provided much useful advice on my PhD study.

Thirdly, I would like to thank Dr. Naoki Masuda and Dr. Sadamori Kojaku for their guidance and support on core-periphery structures. It was good experience to work with them. I would like to thank Dr. Frans Oliehoek and Dr. Rahul Savani for giving me the opportunity to participate in AI research in education and attend my first academic conference.

Then, I would like to thank my IPAP members Prof. Yannis Goulermas and Dr. Vitaliy Kurlin for their advice and encouragement during my PhD study. I also would like to thank other advisors Prof. Karl Tuyls and Dr. Tingting Mu for their engagement.

Futhermore, I would like to thank again Prof. Danushka Bollegala and all Natural Language Processing (NLP@UoL) research group members: Dr. Angrosh Mandya, Huda Hakami, James O'Neill, Micheal Abaho, Dr. Mohammed Alsuhaibani, Dr. Pavithra Rajendran and Yi Zhou. The weekly group meetings have been very helpful in my research as well as good opportunities to practise presentation skills. I also would like to thank Phil Jimmieson for his guidance on demonstrating of the students, and Prof. Katie Atkinson for the opportunity to gain experience on teaching in a different department. The publications mentioned in this thesis were submitted to various venues, and I have received comments from anonymous reviewers. I would like to take this opportunity to thank them for the advice given to me to improve my research.

Finally, I would like to thank my parents Zhongdan Gu and Jianzhong Cui, for their continuous support for my study in the UK.

Publications

1. X. Cui, F. Coenen, and D. Bollegala. Effect of data imbalance on unsupervised domain adaptation of part-of-speech tagging and pivot selection strategies. In L. Torgo, B. Krawczyk, P. Branco, and N. Moniz, editors, *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, volume 74 of *Proceedings of Machine Learning Research*, pages 103–115, ECML-PKDD, Skopje, Macedonia, 22 Sep 2017. PMLR.

2. X. Cui, F. Coenen, and D. Bollegala. Tsp: Learning task-specific pivots for unsupervised domain adaptation. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pages 754–771, 2017.

3. X. Cui, N. Al-Bazzaz, D. Bollegala, and F. Coenen. A comparative study of pivot selection strategies for unsupervised cross-domain sentiment classification. *The Knowledge Engineering Review*, 33(5):1–12, 2018.

4. X. Cui, S. Kojaku, N. Masuda, and D. Bollegala. Solving feature sparseness in text classification using core-periphery decomposition. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 255–264. Association for Computational Linguistics, 2018.

5. X. Cui and D. Bollegala. Self-adaptation for unsupervised domain adaptation. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 213–222, Varna, Bulgaria, 2019.

Publications under review:

6. X. Cui and D. Bollegala. Multi-source attention for unsupervised domain adaptation. 2020.

Publications not covered in this thesis:

7. F. A. Oliehoek, R. Savani, E. Adderton, X. Cui, D. Jackson, P. Jimmieson, J. C. Jones, K. Kennedy, B. Mason, A. Plumbley, and L. Dawson. Liftupp: Support to develop learner performance. In E. André, R. Baker, X. Hu, M. M. T. Rodrigo, and B. du Boulay, editors,

Proceedings of the International Conference on Artificial Intelligence in Education (AIED), pages 553–556, Cham, 2017. Springer International Publishing. ISBN 978-3-319-61425-0.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

A remarkable ability of humans is to learn and *adapt* from past experiences to cope with novel situations. For example, a human can easily recognise various types of cats after seeing a few photos of cats. Machine intelligence refers to the ability for a computer to learn to perform a task [91, 146]. As early as in 1950s, Alan Turing developed the Turing Test which was designed to test whether a computer is capable to perform a task as a human would do. Language is one example of human intelligence. Human languages are referred to as natural languages, such as English, Chinese, Japanese etc. In the Turing Test, a computer and a human are required to answer written questions in English provided by a human questioner, and the questioner was then asked to distinguish the authors of the answers [74, 146]. As seen, the Turing Test was based on natural languages. One of the main reasons why we want computers to be able to process natural languages is that we want computers to acquire information available in the form of natural languages [146]. Early approaches in Natural Language Processing (NLP) were based on the rules written by humans [32, 142]. However, these rule-based methods were designed for specific domains and were unable to generalise to the information in unseen out-of-domain inputs [36].

With the increasing amounts of digitised textual data in natural languages [57], machine learning has been extensively used to automatically extract the useful knowledge from large amounts of data and apply to tasks that were once used to be done manually. Instead of writing rules, a statistical model is learned to perform a task by *training* on available data [14, 91]. The model automatically learns the information (i.e., patterns) contained in the input data [14, 142]. This setting is known as supervised learning.

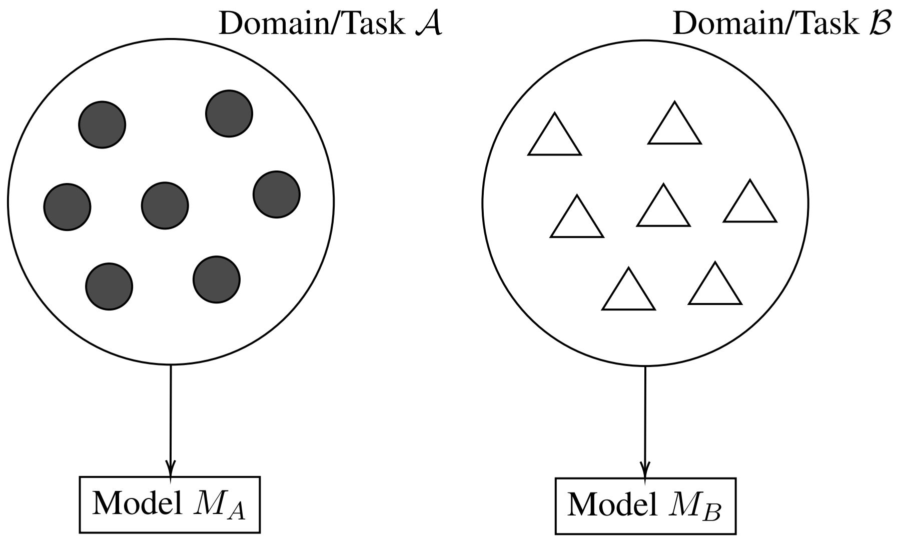


Figure 1.1: Supervised Learning on two different domains/tasks [78, 142].

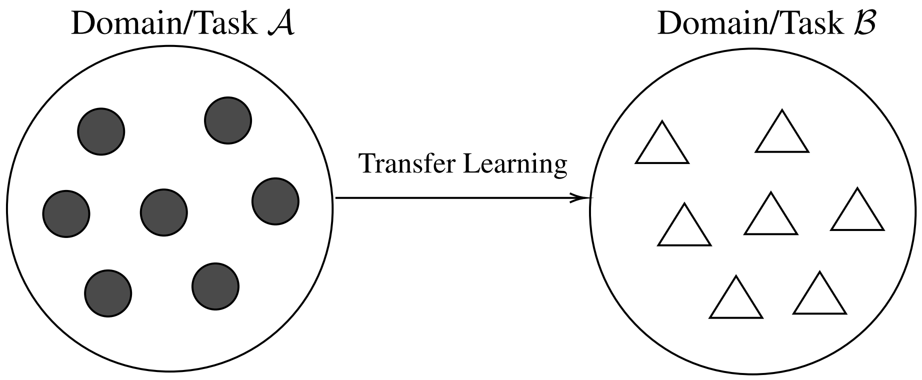


Figure 1.2: Transfer Learning on two different domains/tasks.

A typical supervised machine learner would require a large amount of labelled data to obtain an accurate model. However, it is time-consuming and infeasible to manually annotate large amounts of training data. In addition, there is no guarantee that the annotated data is not biased towards a particular subset of the data [91, 92]. For example, the data collected from a local market in Liverpool can be biased with respect to global markets due to the differences in the cost of living. It might not be necessary to *generalise* to the entire global market, instead the data from UK market could be used to help a model *generalise* to the French market. "Generalisation" is the ability of a model to perform well on test data after learning from the training data [92]. In supervised learning, the available data is typically split into a training set for learning a model and a test set for evaluating the performance of the learned model. During training, the model attempts to minimise the training error. However, we are actually interested in minimising the generalisation error on the test set, the target set which has not been observed during training. Training and test sets are typically assumed to be *independent* and *identically distributed* (*i.i.d*): a collection of samples (i.e., training or test sets) is *independent* and *identically distributed* if each sample in the collection is mutually *independent* and is sampled from the same probability distribution [14, 34, 142]. A *domain* consists of a feature space and its probability distribution. Differently, a *task* consists of a label space and a decision function (i.e., a model) to learn the desired probability distribution in the test set [125]. Traditional supervised learning methods learn an independent model on the training set and test on the data in each target domain for each target task. In other words, in Figure 1.1, if we intend to train a model M_A for some task in a domain $\mathcal{A}$, we have to provide the labelled data for the same task and domain. When the given data is from a different task or a new domain $\mathcal{B}$, we expect to learn a model M_B to perform well on the data from the domain $\mathcal{B}$. These supervised learning methods fail to learn an accurate model when sufficient labelled data for the desired task or domain $\mathcal{B}$ is unavailable.

However, Transfer Learning (TL) is different from traditional supervised learning. Figure 1.2 TL transfers the knowledge to obtain a well-trained model for a new task or a new domain based on a related task or domain [125, 142]. There are two subfields in transfer learning : Multi-task Learning (i.e., transfer across tasks) and Domain Adaptation (i.e., transfer across domains). Multi-task Learning (MTL) aims to learn a model for a task from learning multiple related tasks simultaneously [4], and it is also known as learning with auxiliary tasks [142]. In this thesis, we are interested in learning across domains. Domain Adaptation (DA) considers the problem of training a model using the data from one domain (i.e., *the source domain*) and then applying the trained model on a different domain (i.e., *the target domain*). In DA, the task remains the same in the source and target domains. DA aims to find a common space representation for the two domains to make them *closer* so as to make it easier to adapt from the source domain to the target domain. DA has been applied to various NLP tasks such as sentiment classification [16, 23, 101, 126, 187], Part-of-Speech (POS) tagging [15, 95, 107, 119, 154], machine translation [67, 88], machine reading comprehension [172] and person-job fit [12].

Figure 1.3 outlines the process of DA. Depending on the availability of the training data in the target domain, DA methods can be categorised into Supervised Domain Adaptation (SDA) [45], Semi-Supervised Domain Adaptation (SSDA) [46] or Unsupervised Domain Adaptation (UDA) [15]. SDA uses only the labelled training data in the target domain, whereas UDA uses only the unlabelled training data in the target domain. Moreover, SSDA uses both labelled and unlabelled training data in the target domain.

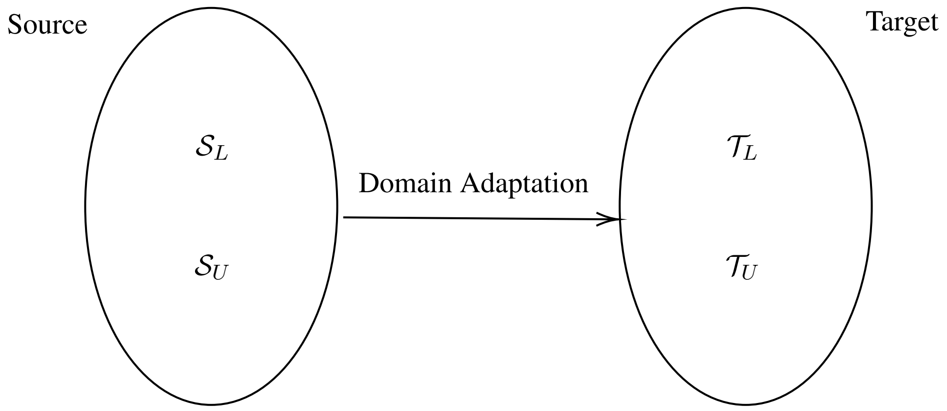


Figure 1.3: Availability of data in DA for a source domain $\mathcal{S}$, and a target domain $\mathcal{T}$. DA is used to learn a mapping from the source domain feature space to the target domain feature space. $\mathcal{S}_L$ is the source domain labelled data and $\mathcal{S}_U$ is the source domain unlabelled data. Similarly, $\mathcal{T}_L$ is the target domain labelled data and $\mathcal{T}_U$ is the target domain unlabelled data. Typically, SDA uses training instances from $\{\mathcal{S}_L, \mathcal{S}_U, \mathcal{T}_L\}$, SSDA uses $\{\mathcal{S}_L, \mathcal{S}_U, \mathcal{T}_L, \mathcal{T}_U\}$ and UDA uses $\{\mathcal{S}_L, \mathcal{S}_U, \mathcal{T}_U\}$.

Using the popular Amazon Product Reviews dataset proposed by Blitzer et al. [16] as an example, let us consider reviews written on *books* as the source domain and reviews written on *electronics* as the target domain. Under the UDA setting, we only have labelled data for *books*. It is possible to train a logistic regression classifier using labelled reviews for *books* with a classification accuracy of 83.25% [39]. However, if we directly apply this model to *electronics*, the classification accuracy drops to 64% [39]. The main reason behind this phenomenon is the "feature mismatch" [156], which refers to the changes in the data distribution between the source domain and the target domain. For example, a negative review for a book might describe the book as *boring* or *dull*, whereas a negative review for an electronic device is unlikely to use those words. Considering the reviews from a product as a domain, by learning these indicative words for *books* will not help much in learning an accurate classifier for *electronics*. As mentioned previously, manually labelling data is an expensive and a time-consuming process. In real-world scenarios, large amounts of labelled data may be available only for certain well-studied domains, whereas little or no labelled data is available for the desired target domain in which we would like to apply the trained model.

In this thesis, we focus on UDA that is technically more challenging than SDA and SSDA because no labelled data is available for the target domain. UDA is more attractive in real-world DA tasks because it obviates the need to label target domain data. To answer the problem of learning representations in DA, we propose novel UDA methods across multiple sub-topics: feature-based and instance-based methods. Moreover, we compare the performance of the proposed methods against existing UDA methods.

1.2 Research Objectives

This thesis studies the problem of automatically learning transferable features across domains for UDA in NLP. The main research question is the following: *How can we represent data using features that will enable us to adapt a machine learning model trained using data from one domain to a different domain?*

This research question can be further divided into sub-problems which will be discussed in the subsequent chapters:

1. **Initial Input of Training Data:** Based on the initial input, UDA methods can be categorised into feature-based UDA [15, 16, 30, 61, 104, 126] and instance-based UDA [55, 56, 108, 167, 168, 175, 180]. In feature-based UDA, a successful approach is to project the feature spaces of the two domains to a shared common space to reduce the dissimilarities between the two domains. Then an adaptive model that can be used in the two domains is learned in this common space. Differently, instance-based UDA methods reduce the discrepancy by reweighting the instances from the source domain and then an adaptive model is trained on the weighted instances [173]. In this thesis, we study feature-based UDA methods in Chapters 3, 4 and 5. Furthermore, we study instance-based UDA methods in Chapters 6 and 7.

2. **Pivot Selection:** In DA, *pivots* are a subset of the features (e.g. words), that are common to the source and target domains [15]. In feature-based DA, pivot selection is conducted as a separate step from other steps in UDA. Different pivot selection strategies have been proposed in the literature. However, no comparative study has been conducted for evaluating these strategies. In Chapter 3, we compare the heuristic methods such as the frequency of a pivot in a domain, mutual information (MI) or pointwise mutual information (PMI) between a pivot and a domain (§3.1). Two research issues are covered: (a) *how does the sets of pivots selected by two strategies differ in practice?*, and (b) *what is the relative gain/loss in performance in a UDA task when we use pivots selected using a particular selection*

strategy?. To answer (a) and (b), we evaluate pivot selection strategies based on the pivots' overlap and rank similarity, and their performance when applied to the feature-based UDA methods such as Structural Correspondence Learning (SCL) [15] and Spectral Feature Alignment (SFA) [126] (§3.2).

Existing heuristic pivot selection strategies focus on either: (a) selecting a subset of common features to source and target domains as pivots, or (b) selecting a subset of task-specific features (features selected based on source domain's labelled training instances) as pivots. In our experiments (§4.1.2), we find the pivots must be both common as well as task-specific. In addition, it is non-trivial as to how we can combine the two requirements (a) and (b) in a consistent manner to select pivots. In Chapter 4, we propose Task-Specific Pivot (TSP) selection for UDA to overcome the above-mentioned limitations of existing pivot selection strategies. We evaluate the performance on the proposed method applied to the feature-based methods against heuristic methods.

3. **Data Imbalance:** Data imbalance issue arises in many NLP tasks such as sentiment classification [102], POS tagging [137] and Name Entity Recognition (NER) [13]. Data imbalance results in discrepancy in terms of the amount of training data for the different target classes we would like to learn [69, 135, 189]. For example, in sentiment classification, we might have a disproportionately large amount of positively labelled data to that of negatively labelled data. If we simply mix all available data and train a classifier, then the trained classifier might be biased towards predicting the positive label by default. In UDA, the problem is more serious because we have only a limited amount of labelled training data. Typical solutions such as oversampling [28] or undersampling [72] might not always be effective. Another example is cross-domain POS tagging. The POS distribution of words is highly uneven. Some categories such as nouns and adjectives are highly frequent whereas adverbs are much less frequent. Several heuristic methods have been proposed in prior work on cross-domain POS tagging [15]. However, these methods have ignored this data imbalance issue. We study the effect of data imbalance in UDA applied to cross-domain POS tagging in Chapter 4. We propose a pivot selection method using the F-score on the source domain labelled data for imbalanced training data. We evaluate the proposed method on a benchmark dataset for cross-domain POS tagging. Moreover, we present the results for other combinations of pivot selection strategies.

4. **Feature Sparseness:** In UDA, training and test data are selected from different domains with small intersection of feature spaces. The number of features presented in the intersection will

be a small fraction of the set of all features, which makes it difficult to use any frequency-based methods. This problem is known as *feature sparseness*.

To address the feature sparseness problem encountered in cross-domain classification tasks, two complementary approaches have been proposed in the literature: (a) expanding the instances by predicting the missing features, and (b) projecting instances to a dense space and performing the task in the projected space. The method proposed in Chapter 5 can be categorised into the first group of methods. Specifically, we decompose a feature-relatedness graph into core-periphery structures, where a core feature is linked to a set of peripheries, indicating the connectivity of the graph. So that the related features can be appended to the feature vectors to reduce the sparsity. We evaluate the proposed method with the state-of-art feature-based UDA methods. We additionally compare the performance on the short-text classification with document-level embedding methods, such as Skip-thought [87].

5. **Projection and Self-training:** Two main approaches for UDA can be identified from prior work: projection-based and self-training. Projection-based methods for UDA [15, 16, 126] learn an embedding space where the source and target domains become closer to each other than they were in the original feature spaces. Self-training [2, 181] is a technique to provide pseudo-labels using a learner trained on available labelled instances. This technique has been used in UDA for predicting pseudo-labels for unlabelled instances in the target domain [2, 144]. These two approaches have been explored separately in prior work. To overcome the problem of the lack of labelled data in the target domain for UDA, we propose an instance-based method that combines projection and self-training based approaches in Chapter 6. Using the labelled data in the source domain, the proposed method first learns a projection that maximises the distance among the nearest neighbours with opposite labels. Next, it trains a learner for the target domain using this source projection. Then, the same idea is applied to use the pseudo labelled target domain data to learn a target projection. Finally, it trains a model for the target domain using the target projection. We present the results of projection learning and pseudo-labelling and evaluate the performance on a benchmark dataset for cross-domain sentiment classification. We compare the classification accuracy against self-training based methods and neural methods.

6. **Negative Transfer:** Most of the DA methods consider a single domain as the source for adaptation [143]. However, in practice, training data can come from multiple source domains. One of the sub-problems in UDA is to to select a suitable source domain from a given set of domains to a given target domain. However, not all sources are suitable for

learning transferable features to a target domain. Often, it is difficult to select a suitable single source domain to adapt from. Using an unrelated source can result in poor performance on the given target, which is known as the *negative transfer* problem [141]. Furthermore, feature-based methods fail in multi-source settings, because it is challenging to find pivots across all sources such that a shared projection can be learnt. In Chapter 7, to overcome this problem, we propose an attention-based solution at instance-level for multi-source UDA. We evaluate the performance of the proposed method against feature-based and instance-based approaches. Moreover, the proposed method is able to provide evidence for its predictions using attention weights.

1.3 Contributions

In this thesis, we focus on learning transferable features across domains through two types of UDA methods: feature-based and instance-based methods. Our contributions can be summarised as follows:

- We conduct a comparative study on heuristic pivot selection strategies for UDA (§3).
- We propose task-specific pivot selection for UDA that selects the pivots are common and task-specific (§4.1).
- We propose a novel pivot selection strategy to overcome the issue of data imbalance in UDA (§4.2).
- We propose a novel feature-based UDA method that uses core-periphery decomposition to find the candidates for feature expansion to reduce the sparsity in UDA (§5).
- We propose a novel instance-based UDA method combining projection-based and self-training UDA methods (§6).
- We propose a novel instance-based UDA method that uses self-training based pseudo-labelling to learn an attention model for multi-source domain adaptation to overcome the issue of negative transfer in UDA (§7).

Open-source code is provided for all published work [1].

[1]The code is available at: https://github.com/summer1278

1.4 Structure of the Thesis

Figure 1.4 shows the structure of thesis and the accompanying publications. In Chapter 2, we review the prior work on UDA methods related to this thesis. In this thesis, UDA methods are categorised into two parts based on the initial inputs: feature-based and instance-based. For each part, we review various types of approaches based on the research issues we focus in this thesis.

In Chapters 3, 4 and 5, we focus on feature-based UDA methods. We describe pivot selection strategies for UDA in Chapters 3 and 4. In Chapters 6 and 7, we focus on instance-based UDA methods.

In Chapter 3, we conduct a comparative study on heuristic pivot selection strategies. Specifically, we describe different pivot selection methods based on frequency, mutual information and pointwise mutual information. Moreover, we evaluate their performance, applied to feature-based UDA methods: SCL and SFA.

In Chapter 4, we present supervised pivot selection methods to select pivots for different tasks. First, we propose task-specific pivot selection that selects pivots to model the combination of two criteria: similarity between the source and target domains and the related information captured by the source domain. We compare the proposed method against heuristic pivot selection methods using SCL and SFA. Next, we study the problem of data imbalance encountered in domain adaptation. I compare the effect of heuristic pivot selection strategies for selecting pivots for UDA of POS tagging under data imbalance. we propose a pivot selection method using the F-score for UDA of POS tagging.

In Chapter 5, we present a graph-based feature expansion UDA method to address the feature sparseness problem in text classification. We use core-periphery decomposition to compute the related features can be expanded to the feature vectors to reduce the sparsity. In the experiments, we evaluate the proposed method in short-text classification tasks against embedding-based methods. We additionally compare the performance of the proposed method (i.e., non-overlapping and overlapping versions) in cross-domain classification against previously proposed feature expansion methods and feature-based UDA methods.

In Chapter 6, we study previously proposed self-training methods in NLP tasks. By combining self-training with projection, we propose an instance-based UDA method that does not require splitting the feature space into pivots and non-pivots. We learn two separate projections for each of the source and target domain. We evaluate the effect of projection learning and pseudo-labelling in the proposed method. We compare our proposed method against classical self-training based methods. We additionally compare against neural UDA methods that use a larger feature space

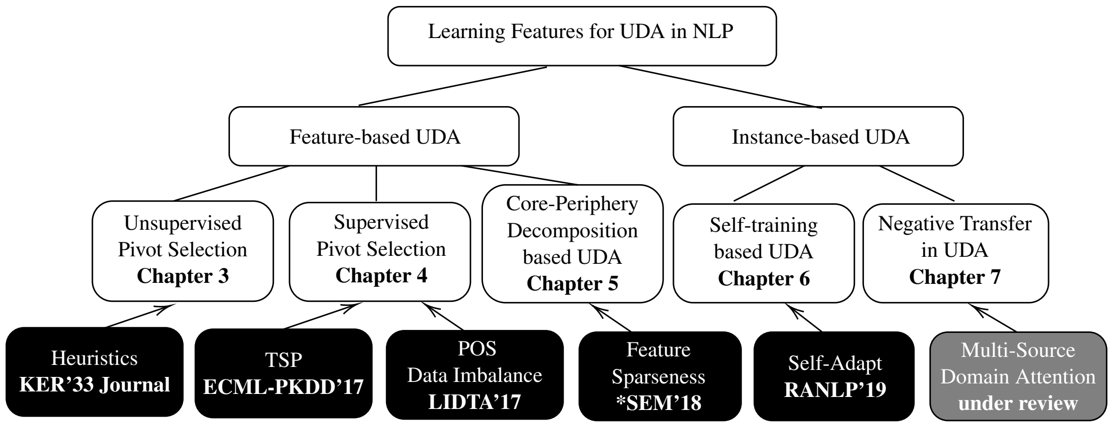

Learning Features for UDA in NLP
Feature-based UDA
Instance-based UDA
Unsupervised Pivot Selection **Chapter 3**
Supervised Pivot Selection **Chapter 4**
Core-Periphery Decomposition based UDA **Chapter 5**
Self-training based UDA **Chapter 6**
Negative Transfer in UDA **Chapter 7**
Heuristics **KER'33 Journal**
TSP **ECML-PKDD'17**
POS Data Imbalance **LIDTA'17**
Feature Sparseness ***SEM'18**
Self-Adapt **RANLP'19**
Multi-Source Domain Attention **under review**

Figure 1.4: The overview of the research conducted in this thesis.

than our proposed method.

In Chapter 7, we tackle the problem of negative transfer encountered in multi-source domain adaptation. We present our proposed method: multi-source domain attention. We compare the proposed method against the state-of-the-art methods under two metrics. We analyse the effect of each component in the proposed method: self-training, pseudo-labelling, relatedness graph and attention.

In Chapter 8, we conclude this thesis, summarising the main findings and provide an overview of potential future work for UDA.

Chapter 2

Literature Review

Unsupervised Domain Adaptation (UDA) assumes that we have some labelled data from the source domain and a large number of unlabelled data from the source and target domains. Before reviewing the related work, we will define the problem and introduce some notations used in this chapter. First, let us assume that our problem is a one-to-one cross domain task that aims to learn an adaptive model f' from a source domain $\mathcal{S} = \{\mathcal{X}_S, P_S(X_S)\}$ to a target domain $\mathcal{T} = \{\mathcal{X}_T, P(X_T)\}$. $\mathcal{X}_S$ and $\mathcal{X}_T$ denote the feature spaces of the source and target domain. $P(X_S)$ and $P(X_T)$ denote the distributions in the source and target domain, respectively. Then, we denote the source domain labelled training instances as $\mathcal{S}_L = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{NS}$ and unlabelled training instances as $\mathcal{S}_U = \{\boldsymbol{x}_j\}_{j=1}^{M^S}$, where $\mathcal{S} = \mathcal{S}_L \cup \mathcal{S}_U$ and y is the label associated with an instance x. we do not assume the availability of any labelled instances from $\mathcal{T}$. We further denote the target unlabelled training instances as $\mathcal{T}_U = \{\boldsymbol{x}_j\}_{j=1}^{MT}$, where $\mathcal{T} = \mathcal{T}_U$ (i.e., no labelled data available in the target domain). The source task T is the same as the target task T', therefore the label space for y in the source domain $\mathcal{Y}_S$ will not change when adapting to the target domain (i.e., $\mathcal{Y}_S = \mathcal{Y}_T$). If we train a model f on $\mathcal{S}_L$, f will not perform well when testing on the target domain. This is caused by *covariate shift* [156], informally referred to as the *feature mismatch* [15, 16], i.e., $P(X_S) \neq P(X_T)$. In this thesis, we are interested in feature representations for UDA. *Pivots* are the features that are common and behave in the same manner in both domains, usually represented as unigrams or bigrams [15]. Based on the initial input of training data, there are two categories of UDA methods: feature-based (§2.1) and instance-based UDA (§2.2). In this chapter, we will give a literature review of existing DA methods that are related to this thesis.

Most of the UDA methods consider adapting to a target domain from a single domain. We will also cover the prior work that consider the adaptation using multiple source domains. In multi-

Category	Projection Learning (§2.1.2)	Feature Expansion (§2.1.3)	Neural Approaches (§2.1.4)
Description	Learn an embedding space to minimise the difference between source and target domains	Extend instances by predicting missing features	Use NNs such as autoencoders to reconstruct the inputs, then minimise the loss between the original inputs and their reconstructions
Strength	Easy to scale, learnt embedding space typically has lower dimensionality	Overcome feature sparseness and increase the feature space of training instances	End-to-end training, do not require manual pivot selection
Weakness	Heavily rely on pivot selection	Heavily rely on expansion candidates	High computational cost, lack of scalability
Methods	SCL [15], SFA [126], SDAMS [174]	FTS [111], SST [23]	SDA [61], mSDA [30], WDP [21], AE-SCL [192], AMN [104]

Table 2.1: Overview of feature-based UDA methods.

source domain adaptation, we assume N source domains, $\mathcal{S}_L^{(i)}$ and $\mathcal{S}_U^{(i)}$ respectively denoting the labelled and unlabelled instances for the i-th source domain. $\mathcal{S}_L = \cup_{i=1}^{N} \mathcal{S}_L^{(i)}$ and $\mathcal{S}_U = \cup_{i=1}^{N} \mathcal{S}_U^{(i)}$ respectively denote the union of the labelled and unlabelled instances for the N source domains. The goal is to learn a model f' from the source domains $\{\mathcal{S}^{(1)}...\mathcal{S}^{(i)}\}$ to adapt to a target domain $\mathcal{T}$.

2.1 Feature-based UDA

One of the main approaches for feature-based UDA methods is based on projection learning. Projection[1]-based methods for UDA learn an embedding space, where the difference between the distribution of features in the source and the target domains are minimised, thereby learning a common feature space [15, 16]. Pivot selection is an important first step in learning the common feature space for projection-based DA. We review the prior work on pivot selection in Section 2.1.1 and projection-based UDA methods in Section 2.1.2. Moreover, feature augmentation has been applied in DA such as Easy Adapt (EA) [45] and its semi-supervised DA extension EA++ [47]. In contrast to UDA, these approaches assume the availability of the labelled data in the target domain. Consequently, they are not included in this thesis.

Feature expansion is proposed for overcoming the feature sparseness problem in text classification [111], which expands the instances by predicting the missing features. Feature sparseness

[1]We consider the terms *project* and *embed* as synonymous in this thesis

is also encountered in the DA setting where the data distribution of the source and target domains has a small intersection of feature spaces. In Section 2.1.3, we review the prior work on feature expansion applied to UDA.

Another line of approaches use neural networks such as denoising autoencoders [30, 61] that do not rely on manual selection of pivots. In Section 2.1.4, we review the prior work on feature-based UDA that use neural networks.

2.1.1 Pivot Selection

In projection-based domain adaptation, pivot selection is conducted as a separate step from the other steps. Different strategies for selecting pivots such as the frequency of a pivot in a domain (FREQ), mutual information (MI), or pointwise mutual information (PMI) between a pivot and a domain label have been proposed in the literature.

Blitzer et al. [15] defined pivots as features that behave in the same way for discriminative learning in both source and target domains. They selected features that occur frequently in both source and target domains as pivots. This pivot selection strategy does not require any labelled data, and was shown to perform well for sequence labelling tasks such as POS tagging. However, for discriminative classification tasks such as sentiment classification, Blitzer et al. [16] showed that MI to be a better pivot selection strategy than frequency. In this strategy, MI between a feature and source domain positive and negative sentiment labelled reviews are computed. Next, features that have high MI with either positive or negative labelled reviews are considered as pivots. The expectation here is that features that are discriminative of sentiment in the source domain will also be discriminative for the sentiment expressed in the target domain. This approach requires source domain labelled data for selecting pivots.

Pan et al. [126] proposed an alternative definition of pivots where they select features that are common to both source and target domains as pivots. They refer to such features as *domain-independent* features, whereas the remainder is considered as *domain-specific*. They proposed the use of MI between a feature and unlabelled training instances in each domain as a pivot selection strategy. If a particular feature has low mutual information with both the source and the target domains, then it is likely to be a domain-independent feature. Considering that the amount of unlabelled data is much larger than that of source domain labelled data in UDA settings, we can make better estimates of MI using unlabelled data. However, we cannot select pivots that discriminate the classes related to the target prediction task (e.g. sentiment classification) using only unlabelled data.

Bollegala et al. [23] proposed PMI [33] as a pivot selection strategy for UDA. PMI has established as an accurate word association measure and have been applied in numerous NLP tasks such as collocation detection [112], word similarity measurement [166], relational similarity measurement [165] etc. Furthermore, Positive Pointwise Mutual Information (PPMI) [105] is a variation of PMI, in which all negative values of PMI are replaced with zero. PPMI is also proposed to be a pivot selection strategy for UDA [21].

These measures have been computed using either unlabelled data (available for both source as well as target domain), or labelled data (available only for the source domain). In Chapter 3, the conducted comparative study considers all possible combinations between types of data (labelled vs. unlabelled), and pivot selection strategies (FREQ, MI, PMI). Moreover, we evaluate their performance applied to UDA. In Chapter 4, we propose a novel pivot selection strategy to select features that are task-specific and common to the two domains.

2.1.2 Projection Learning

The main issue when adapting from one domain to another is that the two domains are represented in domain-specific feature spaces [92]. Learning a projection from the two domains to a domain-invariant space is an ideal solution for learning transferable features across domains.

Structural Correspondence Learning (SCL) [15] is a method to automatically identify the correspondence between a *source* domain and a *target* domain to reduce the mismatch of the features between the two domains. This method was initially introduced for POS tagging and later extended to sentiment classification [16]. First, SCL selects k features by a certain selection method. Next, k binary predictors are trained to model the correlation of pivot features and non-pivot features. Then, Singular Value Decomposition (SVD) is performed on the weight matrix to learn a lower-dimensional projection for the pivot predictors. Finally, a logistic regression classifier is trained on the labelled data represented as the concatenation of (a) the original features and (b) the predicted pivot features.

Spectral Feature Alignment (SFA) [126] is a method designed for cross-domain sentiment classification. First, all features are divided into two mutually exclusive groups : domain- independent and domain-specific. Next, SFA constructs a bipartite graph between domain-independent and domain-specific features based on their total number of co-occurrences in the same document across two domains. Our experiments in Section 3.2.2 show that SFA works better than SCL when selecting graph candidates from a larger number of input data. Then, SFA adapts spectral clustering to create a lower dimensional representation by top eigenvectors for projecting domain-specific features. Similar to SCL, the final step is to learn a logistic regression model using labelled data from the source domain by (a) the original features and (b) the projected domain-specific features.

Sentiment Domain Adaptation method from Multiple Sources (SDAMS) [174] is a multi-source UDA method designed for cross-domain sentiment classification. It introduces two components: a sentiment graph and a domain similarity measure. The sentiment graph is extracted from unlabelled data using sentiment polarity relations between word-pairs. The domain similarity measure is based on the word-pairs of domains in the sentiment graph. The projection is learnt from the sentiment graph in each domain.

2.1.3 Feature Expansion

Another line of feature-based UDA methods is based on feature expansion. To overcome the feature sparseness in classification tasks, the training data is expanded by predicting the missing features.

Frequent Term Sets (FTS) [111] is a method used to expand feature vectors. This method first computes the co-occurrence among the features and then the expansion candidates are selected by a pre-defined threshold on the frequency. Finally, the features in the original feature vectors are expanded using these frequently co-occurring features. Ma et al. [110] proposed an improvement of FTS by introducing the support and confidence to the co-occurrence relationship when they create the frequent term sets for expansion.

Sentiment Sensitive Thesaurus (SST) [23] is a method to automatically create a thesaurus to group different features that express the same sentiments for cross-domain sentiment classification. First, each feature is represented as a feature vector by a set of features that co-occur with the feature and a set of sentiment features by the source labelled instances that the feature occurs. Next, SST measures the relatedness to other features and groups them in the descending order of relatedness score to create a thesaurus. Additionally SST creates sentiment features for a thesaurus by using the labelled information in the source instances that the feature occurs. After that, the instance vector of the document is expanded by inducting top k related features from the thesaurus created in the previous step. Finally, a classifier is learnt using expanded document vectors.

In Chapter 5, we propose a UDA method using feature expansion. We use core-periphery decomposition to find the candidates for expansion instead of applying heuristic measures such as MI or PMI. The related work on core-periphery decomposition will be described in Section 5.1.

2.1.4 Neural Approaches to Feature-based UDA

Deep learning has been widely applied to UDA. Stacked Denoising Autoencoders (SDA) [61] is one of the earliest works using deep learning for domain adaptation. In autoencoder-based methods, the model is trained to minimise the loss between the original inputs and their reconstructions. First, SDA uses stacked denoising autoencoders to find an invariant feature space on the source and target domain data. Then, a Support Vector Machine (SVM) classifier is trained on the invariant features extracted by the encoder and the original features in the source domain. Liu et al. [106] proposed two variations of SDA: SDA with Domain Supervision (SDA-DS) and SDA with Sentiment Supervision (SDA-SS). SDA-DS adds an additional layer to predict the domain distribution during reconstruction of the input. SDA-SS adds an additional layer to incorporate sentiment labels.

Marginalized Denoising Autoencoders (mSDA) [30] is an extension to SDA. In mSDA, the reconstructive mapping can be computed in close-form without corrupting a single instance [78]. The training is performed through entire data layer by layer and each layer tries to reconstruct the previous layer's output. Finally, an SVM classifier is trained on the concatenation of the inputs and all hidden layers. mSDA improves the problem of high computational cost and lack of scalability of high-dimensional features compared to SDA by only reconstructing the domain independent features [145].

Word Distribution Prediction (WDP) [21] is based on popular word embedding methods such as skip-gram [118], which aims to predict the distribution of a word changes from one domain to another domain. First, WDP selects pivots and uses SVD to create two latent feature spaces separately for the source and target domains. Second, it learns a mapping from the source latent feature space to the target latent feature space using Partial Least Square Regression (PLSR) [1]. Yang and Eisenstein [180] proposed a similar approach Feature Embeddings for Domain Adaptation (FEMA). Instead of using pivots, FEMA uses neural language model to obtain low-dimensional embeddings.

Neural Structural Correspondence Learning (AE-SCL) [192] is a method that combines autoencoder based approaches [30, 61, 179] and SCL [15]. The model structure is similar to an autoencoder, unlike other autoencoder-based methods, AE-SCL does not reconstruct all the inputs. AE-SCL aims to learn a prediction function from the non-pivot features to the pivot features. Then, the pivot features can be reconstructed from the learnt representation.

Adversarial Memory Network (AMN) [104] consists of two classifiers for predicting labels and domains inspired by Domain Adversarial Neural Networks (DANN) [56] (§2.2.3), and two

Category	Instance Selection and Reweighting (§2.2.1)	Self-labelling (§2.2.2)	Neural Approaches (§2.2.3)
Description	Select or reweight source domain's training instances based on their importance to the target domain	Assign pseudo labels to unlabelled instances	Use NNs such as adversarial training to minimise the discrepancy between the source and target domains
Strength	No need to label target domain's unlabelled instances	A direct solution to increase the feature space of training data in the target domain	End-to-end training, require less effort on pre-processing
Weakness	Assume there are some useful instances in the source domain's training data	Pseudo-labels may not be accurate	Sensitive to the hyperparameters and require problem-specific techniques
Methods	MMD [65], KMM [76], TCA [127], PU Learning [175]	Self-training [181], Co-training [17], Tri-training [190], MT-Tri [144]	DDCN [167], DAN [108], DANN [56], ADDA [168], MDAN [188], MoE [70]

Table 2.2: Overview of Instance-based UDA methods.

corresponding shared-parameter memory networks [160] . AMN extracts pivots that have two attributes using two memory networks: (a) important sentiment words for sentiment classification (MN-sentiment); (b) shared in both domains (MN-domain). Unlike the instance-based approaches with neural networks, AMN automatically captures the pivots using attention without manual pivot selection and provides a direct visualisation of selected pivots.

2.2 Instance-based UDA

Instance-based UDA is a popular solution to tasks in pattern recognition and computer vision [58, 63, 147], because these visual tasks are different from NLP tasks that are not suitable for using pivot features as the input. Unlike the projection-based approaches that transform the source domain and the target domain into a shared subspace, one line of the solutions is instance selection and reweighting. Instance selection methods aim to select the most appropriate training instances to reduce the dissimilarities between the source and target domains [175]. Rather than dropping the other training instances, instance reweighting methods assign a new weight to each training instance to approximate the target domain distribution [76, 156]. These methods are reviewed in Section 2.2.1.

Another line of solutions is based on self-labelling. In classification tasks, labelled training instances are often difficult to retrieve. In UDA, we have a significantly limited number of labelled instances in the source domain and large amounts of unlabelled instances in the source and the target domains. Self-labelling provides a direct solution to this problem by using the available unlabelled data [191]. In Section 2.2.2, we review the related self-labelling methods applied to NLP tasks.

Neural methods have gained popularity in recent instance-based approaches such as using adversarial training to reduce the discrepancy between the domains [49]. In Section 2.2.3, we review instance-based approaches using neural networks.

2.2.1 Instance Selection and Reweighting

Selecting or weighting instances based on their importance to the target domain is mostly used for solving the covariate shift problem [91]. Instance selection or weighting methods aim to learn an approximate model on the target domain using the source domain training instances. In other words, some helpful instances must be retrained in order to learn a correct approximate target domain model.

Maximum Mean Discrepancy (MMD) [65] is a weight estimation method to determine whether two data distributions are from different domains using the distance between the means of their mapped representations in a reproducing kernel Hilbert space (RKHS). However it does not generalise the samples and learns a latent space by solving a relatively expensive semi-definite program (SDP).

Kernel Mean Matching (KMM) [76] is a non-parametric method that reweights the training instances such that the means of the training and test features in the RKHS are close. It is an instance weighting method that weights the loss of the source instances to solve DA.

Transfer Component Analysis (TCA) [127] introduces a shared latent subspace by minimising the dissimilarities across domains using MMD. In the learnt subspace, the properties of the original data are preserved. Later, Grubinger et al. [66] applied TCA in multi-source domain adaptation by introducing a universal kernel for all source domains. However, in TCA, all source domains are equally weighted in the joint matrix.

Positive and Unlabelled Learning (PU Learning) [175] was proposed for cross-domain sentiment classification to identify the most useful training instances for DA. PU Learning is used to build a binary in-target-domain selector and assign in-target-domain probabilities to source instances. Two models are proposed in [175]: (a) select the instances with higher in-target-domain probabilities, and (b) reweight the instances with in-target-domain probability and train an instance-weighted naïve Bayes classifier.

Most instance selection methods select data using similarity measures [134, 170]. Ruder and Plank [143] used a collection of similarity and diversity measures to weight training instances from multiple source domains based on Bayesian Optimization [26]. They defined a data selection model using existing similarity and diversity measures. Bayesian Optimization is used to learn the weights in order to optimize the objective function for the respective task using a small set of validation data. Although this work does not outperform the state-of-the-art methods in UDA, it demonstrates the diversity measures are also important in DA and shows the cross-model, cross-domain and cross-task results for the learnt measures.

Several task-specific instance selection UDA methods have been proposed in the literature. Axelrod et al. [8] proposed an instance selection method for machine translation. It ranks the instances in the source domain with respect to the target instances using the sum of cross-entropy differences over sentences of the corpus. Plank and van Noord [134] proposed to use similarity measures based on topic representation to select training data for dependency parsing. Specially, each training instance (i.e., article in parsing task) is represented by a topic distribution at word-level and the similarity between instances is then measured by the topic distributions.

2.2.2 Self-labelling

Self-labelling is a category of semi-supervised learning methods that train a model on the labelled instances and then use the learnt model to assign pseudo labels to the unlabelled instances [142, 190]. In UDA, unlabelled data can be pseudo-labelled by self-labelling methods and used to increase the feature space of training data.

Self-training [181] has been adapted to various cross-domain NLP tasks such as document classification [136], POS tagging [115, 138] and sentiment classification [51]. Although different variants of self-training algorithms have been proposed [2, 184] a common recipe can be recognised involving the following three steps: (a) Initialise the training dataset, $\mathcal{L} = \mathcal{S}_L$, to the labelled data in the source domain, and train a classifier for the target task using $\mathcal{L}$, (b) apply the classifier trained in step (a) to the unlabelled data in the target domain $\mathcal{T}_U$, and append the most confident predictions as identified by the classifier (e.g. higher than a pre-defined confidence threshold τ) to the labelled dataset $\mathcal{L}$, (c) repeat the two steps (a) and (b) until we cannot append additional high confidence predictions to $\mathcal{L}$.

Co-training [17] is another popular approach for inferring labels for the target domain, where the availability of multiple views of the feature space is assumed. In the simplest case where there are two views available for the instances, a separate classifier is trained using the source domain's labelled instances that involve features only from a particular view. Next, the two classifiers are used to predict pseudo labels for the target domain unlabelled instances. If the two classifiers agree on the label for a particular unlabelled instances, then that label is assigned to that instance. Co-training has been applied to UDA [29, 184], where the feature spaces in the source and target domains were considered as the multiple views. The performance of co-training will depend on the complementarity of the information captured by the different feature spaces. Therefore, it is important to carefully design multiple feature spaces when performing UDA.

Tri-training [190] relaxes the requirement of co-training for the feature spaces to be sufficient and redundant views. Specifically, in tri-training, as the name implies three separate classifiers are trained using bootstrapped subsets of instances sampled from the labelled instances. If at least two out of the three classifiers agree upon a label for an unlabelled instance, that label is then assigned to the unlabelled instance. Søgaard [158] proposed a variation of tri-training (i.e. tri-training with diversification) that diversifies the sampling process and reduces the number of additional instances, where they require exactly two out of the three classifiers to agree upon a label and the third classifier to disagree. Saito et al. [148] proposed a deep tri-training method with three neural networks, two for pseudo labelling the target unlabelled data and another one for learning discriminator using the inferred pseudo labels for the target domain. Ruder and Plank [144] proposed Multi-task Tri-training (MT-Tri) based on tri-training and Bi-LSTM. They show that tri-training is a competitive baseline and rivals more complex neural adaptation methods. Although MT-Tri does not outperform state-of-the-art on cross-domain sentiment classification tasks, their proposal reduces the time and space complexity required by the classical tri-training.

In Chapter 6, we propose Self-Adapt, which differs from the prior work discussed above in that it (a) does not require pivots, (b) does not require multiple feature views, (c) learns two different projections for the source and target domains, and (d) combines a projection and a self-training step in a non-iterative manner to improve the performance of UDA.

2.2.3 Neural Approaches to Instance-based UDA

As in feature-based UDA methods, recent approaches have used neural networks in instance-based UDA methods as well.

Deep Domain Confusion Network (DDCN) [167] is a discrepancy-based domain adaptation method using convolutional neural networks (CNN). This method aims to learn a representation that minimises the distribution distance between the source and the target domains. It computes the distribution distance using MMD. Then, the learnt representation is used in the loss function for classification. A regularisation hyperparameter is introduced to adjust the *confusion* between the source and the target domains. To avoid overfitting on the source domain, a lower-dimensional *bottleneck* layer is introduced. The MMD loss is added on top of this layer to the standard AlexNet [94].

Deep Adaptation Network (DAN) [108] also uses a modified AlexNet [94] architecture. The modified AlexNet has 8 layers: the first three layers of convolution network for learning general transferable feature from the input domain, two layers for learning domain specific features and the last three fully connected layers for learning domain invariant features. The discrepancy loss happens at the last fully connected layer. A multiple kernel variant of MMD (MK-MMD) is used to compute the RKHS distance between the mean embeddings of two distributions.

Domain Adversarial Neural Networks (DANN) [56] was proposed to regularise the immediate layers to perform feature learning, domain adaptation and classifier learning jointly in the model architecture using backpropagation for UDA. Ganin et al. [56] focus on learning features with both discriminateness and domain-invariance. This approach seeks three separate parameters to maximise the loss of the domain classifier (discriminates between the source and the target domain), minimise the loss of the label predictor (predicts the class labels) during feature mapping and minimise the loss of domain classifier. Multiple Source Domain Adaptation with Adversarial Learning (MDAN) [188] is proposed as a generalisation of DANN that aims to learn domain independent features while being relevant to the target task. Task learning is also combined in the training model. Zhao et al. [188] proposed two versions: hard and smooth. The source domain that achieves the minimum domain classification error is backpropagated with gradient reversal in the hard version. All the domain classification errors are combined and backpropagated adaptively in the smooth version.

Adversarial Discriminative Domain Adaptation (ADDA) [168] is based on a Generative Adversarial Network (GAN) [64]. In DA, a discriminator is tied to distinguish between training (source) and test (target) domain examples. ADDA first learns discriminative representations using the labels in the source domain, and then an encoder maps the target data to the same space through domain-adversarial loss. The main aim is to minimise the distance between source and target mapping distributions so that the source domain model can be used directly on the target domain.

Mixture of Experts (MoE) [70] is a multiple-source domain adaptation method using a mixture of experts for each domain. Guo et al. [70] model the domain relations using a *point-to-set* distance metric to the encoded training matrix for source domains. Next, they perform joint training over all domain-pairs to update the parameters in the model by *meta-training* [162, 164]. Kim et al. [85] models domain relations using *example-to-domain* based on an attention mechanism. However, the attention weights are learnt using source domain training data in a supervised manner.

In Chapter 7, we propose a multi-source UDA method that uses self-training (§2.2.2) to assign pseudo-labels for unlabelled target domain instances and learns an embedding for each domain using an attention mechanism.

Chapter 3

Unsupervised Methods for Pivot Selection

Selecting pivot features that connect a source domain to a target domain is an important first step in projection-based UDA methods. Although different strategies such as the frequency of a feature in a domain [15], mutual (or pointwise mutual) information [16, 22, 126] have been proposed in prior work in DA for selecting pivots, a comparative study into (a) how the pivots selected using existing strategies differ, and (b) how the pivot selection strategy affects the performance of a target DA task remain unknown. In this chapter, we perform a comparative study covering different strategies that use both labelled (available for the source domain only) as well as unlabelled (available for both the source and target domains) data for selecting pivots for UDA. We call these strategies as *unsupervised* methods for pivot selection because they are based on heuristics, whereas Chapter 4 will cover the work on supervised methods for pivot selection. In Section 3.2, we show that in most cases, pivot selection strategies that use labelled data outperform their unlabelled counterparts, emphasising the importance of the source domain labelled data for UDA. Moreover, PMI, and frequency-based pivot selection strategies obtain the best performances in two state-of-the-art UDA methods.

One of the fundamental challenges in UDA is the mismatch of features between the source and the target domains. Because in UDA labelled data is available only for the source domain, even if we learn a highly accurate predictor using the source domain's labelled data, the learnt model is often useless for making predictions in the target domain. The features seen by the predictor in the source domain's labelled training instances might not occur at all in the target domain test instances. Even in cases where there is some overlap between the source and the target domain feature spaces, the discriminative power of those common features might vary across the two domains. For example, the word *lightweight* often expresses a positive sentiment for *mobile electronic devices* such as mobile phones, laptop computers, or handheld cameras, whereas the same word has a negative sentiment associated in *movie* reviews, because a movie without any dramatic or adventurous storyline is often perceived as boring and lightweight. Consequently, a classifier learnt from reviews on mobile electronic devices is likely to predict a movie review that contains the word *lightweight* to be positive in sentiment.

To overcome the above-mentioned feature mismatch problem in UDA, a popular solution is to learn a *projection* (possibly lower-dimensional) between the source and the target domain feature spaces [15, 16, 126]. To learn such a projection, first, we must identify a subset of the features that are common to the two domains. Such *domain-independent* features that can be used to learn a projection are often called *pivots*. For example, in SFA [126], a bipartite graph is constructed between the domain-independent (pivots) and domain-specific features. Next, spectral methods are used to learn a lower-dimensional projection from the domain-specific feature space to the domain-independent feature space. Using the learnt projection, we can transform a linear classifier trained using source domain's labelled training instances to classify test instances in the target domain. Conversely, SCL [15, 16] first learns linear binary classifiers to predict the presence (or absence) of a pivot in a review. Next, the learnt pivot predictors are projected to a lower-dimensional space using SVD. As seen from SCL and SFA examples, pivots play an important role in many UDA methods [18, 21, 23, 183]. A detailed description of SCL and SFA can be found in Section 2.1.2.

Different strategies for selecting pivots such as the the frequency of a pivot in a domain (FREQ), mutual information (MI), or pointwise mutual information (PMI) between a pivot and a domain label have been proposed in the literature. Despite the significant role played by pivots in UDA, to the best of our knowledge, no comparative study has been conducted for evaluating the different pivot selection strategies. In particular, it remains unclear as to (a) *how the sets of pivots selected using two selection strategies differ in practice?*, and (b) *what is the relative gain/loss in performance in a UDA task when we use pivots selected using a particular selection strategy?* In this chapter, we answer both questions (a) and (b) by conducting a comparative study covering three previously proposed pivot selection strategies (i.e. FREQ, MI, and PMI) using cross-domain sentiment classification as a concrete UDA task. Specifically, to answer (a), we conduct an experiment where we compare two lists of pivots ranked according to two different pivot selection strategies using the Jaccard coefficient. High Jaccard coefficients indicate that the pivots selected by the two methods are similar. To answer (b), we set up an experiment where we use pivots selected using different strategies to train a cross-domain sentiment classifier using two UDA methods, namely

SCL and SFA. Although we limit our evaluation to cross-domain sentiment classification because it is the most frequently used benchmark task for unsupervised domain adaptation methods in the NLP community, pivot selection is not limited to sentiment classification and appears in other domain adaptation tasks such as cross-domain part-of-speech tagging [15] and cross-domain named entity recognition [77]. Moreover, we evaluate the effectiveness of using labelled vs. unlabelled data for pivot selection.

Our experimental results reveal several interesting facts about the pivot selection strategies for UDA.

- For a particular pivot selection strategy, it turns out that it is better to select pivots using the labelled data from the source domain as opposed to unlabelled data from both domains. This result indicates that source domain labelled data play two distinctive roles in UDA. First, with more labelled data for the source domain we will be able to learn a more accurate predictor for a DA task. Second, and a less obvious effect is that we can identify better pivots using source domain labelled data. Indeed, prior work on multi-domain UDA [18, 113] show that the performance of a UDA method on a single target domain is improved by simply combining multiple source domains.

- Although there are a moderate level (i.e. Jaccard coefficients in the range $[0.6, 0.8]$) of overlap and a low level of rank similarity (i.e. Kendall coefficients in the range $[0.1, 0.3]$) among the top-ranked pivots selected using different strategies, the overlap quickly decreases when we select more pivots whereas the rank similarity increases. This result shows that different pivot selection strategies compared in this chapter are indeed selecting different sets of features, and pivot selection strategy is an important component in a UDA method. Considering that in existing UDA methods pivots are selected in a pre-processing step that happens prior to the actual domain adaptation, we believe that our findings will influence future work on UDA to more carefully consider the pivot selection strategy.

- In contrast to prior proposals to use mutual information as a pivot selection strategy [16, 126], in our experiments pointwise mutual information [23] turns out be a better alternative. However, there is no clear single best pivot selection strategy for all source-target domain-pairs when applied to two state-of-the-art UDA methods.

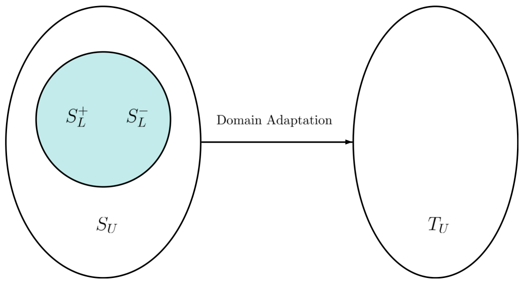


Figure 3.1: UDA from $\mathcal{S}$ to $\mathcal{T}$.

3.1 Pivot Selection Strategies for Unsupervised Domain Adaptation

Let us consider the UDA setting shown in Figure 3.1 where we would like to adapt a model trained using labelled data from a source domain $\mathcal{S}$ to a different target domain $\mathcal{T}$. We consider binary classification tasks involving a single source-target domain pair for simplicity. However, the pivot selection strategies we discuss here can be easily extended to multi-domain adaptation settings and other types of prediction tasks, not limiting to binary classification. In UDA, we assume the availability of labelled training data from the source domain for a particular task. For the binary classification setting we consider here, let us assume that we are given some positively and negatively labelled data for the task, denoted respectively by $\mathcal{S}_L^+$ and $\mathcal{S}_L^-$. In addition to these labelled datasets, in UDA we have access to unlabelled datasets $\mathcal{S}_U$ and $\mathcal{T}_U$, respectively from the source and the target domains.

Next, we discuss three popular pivot selection strategies proposed in prior work in UDA.

3.1.1 Frequency (FREQ)

If a feature x occurs a lot in both the source and the target domain unlabelled training instances ($\mathcal{S}_U$ and $\mathcal{T}_U$), then it is likely that x is not specific to the source or the target domain. Therefore, we might be able to adapt a model trained using source domain's labelled data to the target domain using such features as pivots. This approach was first proposed by Blitzer et al. [15], and was shown to be an effective strategy for selecting pivots for cross-domain POS tagging and dependency parsing tasks. The frequency of a feature x in a set of training instances $\mathcal{D}$ is computed as

$$\begin{aligned} h(x,d) &= \begin{cases} 1, & \text{if } x \in d, \\ 0, & \text{otherwise}, \end{cases} \\ \text{FREQ}(x,\mathcal{D}) &= \sum_{d \in \mathcal{D}} h(x,d). \end{aligned} \tag{3.1}$$

d denotes a document in $\mathcal{D}$. Then we can compute the *pivothood* (the degree to which a feature is likely to become a pivot) of x as

$$\text{FREQ}_U(x) = \min\left(\text{FREQ}(x, \mathcal{S}_U), \text{FREQ}(x, \mathcal{T}_U)\right). \tag{3.2}$$

We sort features x in the descending order of their pivothood given by (3.2), and select the top-ranked features as pivots to define a pivot selection strategy based on frequency and unlabelled data.

One drawback of selecting pivots using (3.2) for discriminative classification tasks such as cross-domain sentiment classification is that the pivots with high FREQ$(x, \mathcal{S}_U)$ could be specific to the sentiment in the source domain, therefore not sufficiently discriminative of the target domain's sentiment. To overcome this issue, Blitzer et al. [16] proposed the use of source domain labelled data, which leads to the pivothood score [1] given by

$$\text{FREQ}_L(x) = \left|\text{FREQ}(x, \mathcal{S}_L^+) - \text{FREQ}(x, \mathcal{S}_L^-)\right|. \tag{3.3}$$

Here, if a x is biased towards either one of $\mathcal{S}_L^+$ or $\mathcal{S}_L^-$, then it will be a good indicator of sentiment in the source domain. The expectation in this proposal is that such sentiment-sensitive features will be useful for discriminating the sentiment in the target domain as well. We sort features in the descending order of their pivothood scores given by (3.3), and select the top-ranked features as pivots to define a pivot selection strategy based on frequency and labelled data.

3.1.2 Mutual Information (MI)

Using raw frequency to measure the strength of association between a feature and a set of instances is problematic because it is biased towards frequent features, irrespective of their association to the set of instances. MI overcomes this bias by normalising the feature occurrences, and has been used as a pivot selection strategy in prior work on UDA [16, 126]. MI between a feature x and a set of instances $\mathcal{D}$ is given by,

$$\text{MI}(x, \mathcal{D}) = p(x, \mathcal{D}) \log\left(\frac{p(x, \mathcal{D})}{p(x)p(\mathcal{D})}\right). \tag{3.4}$$

[1] Note that the original proposal by [16] was to use mutual information with source domain labelled data as we discuss later in Section 3.1.2. However, for comparison purposes we define a pivothood score based on frequency and source domain labelled data here.

We compute the probabilities in (3.4) using frequency counts as

$$\begin{aligned} p(x, \mathcal{D}) &= \text{FREQ}(x, \mathcal{D})/\text{FREQ}(*, *), \\ p(x) &= \text{FREQ}(x, *)/\text{FREQ}(*, *), \\ p(\mathcal{D}) &= \text{FREQ}(*, \mathcal{D})/\text{FREQ}(*, *). \end{aligned}$$

We use "*" to denote the summation over the set of values (e.g. set of features, or sets of instances for all domains) a particular variable can take.

Blitzer et al. [16] consider features that are associated with one of the two classes (e.g. positive or negative sentiment) to be more appropriate as pivots. Based on their proposal, we define the pivothood score as

$$\text{MI}_L(x) = \left| \text{MI}(x, \mathcal{S}_L^+) - \text{MI}(x, \mathcal{S}_L^-) \right|. \tag{3.5}$$

We rank features x in the descending order of their $\text{MI}_L(x)$ scores, and select the top-ranked features as pivots to define a pivot selection strategy based on MI and labelled data.

Pan et al. [126] used MI with unlabelled data to select pivots. They argue that features that have low MI with both source and the target domains are likely to be domain-independent features, thus more appropriate as intermediate representations for DA. Their proposal can be formalised to define a pivothood score as

$$\text{MI}_U(x) = \min \left(\text{MI}(x, \mathcal{S}_U), \text{MI}(x, \mathcal{T}_U) \right). \tag{3.6}$$

Here, we sort features x in the ascending order of their $\text{MI}_U(x)$ scores, and select the top-ranked features as pivots to define a pivot selection strategy based on MI and unlabelled data.

3.1.3 Pointwise Mutual Information (PMI)

PMI between a feature x and a set of training instances $\mathcal{D}$ is given by,

$$\text{PMI}(x, \mathcal{D}) = \log \left(\frac{p(x, \mathcal{D})}{p(x)p(\mathcal{D})} \right), \tag{3.7}$$

where the probabilities are computed in the same manner as described in Section 3.1.2. Unlike MI or PMI does not weight the amount of information obtained about one random event by observing another by the joint probability of the two events. PMI has been used extensively in NLP for measuring the association between words [33]. Because MI takes into account all the joint possibilities (i.e. by multiplying with joint probabilities) its value can become too small and unreliable when the feature space is large and sparse. To overcome this disfluency, Bollegala et al. [23] proposed PMI as a pivot selection strategy for UDA.

Analogous to MI_L and MI_U defined respectively by (3.5) and (3.6), we define two PMI-based pivothood scores PMI_L and MI_U as

$$\text{PMI}_L(x) = \left|\text{PMI}(x, \mathcal{S}_L^+) - \text{PMI}(x, \mathcal{S}_L^-)\right|, \tag{3.8}$$

$$\text{PMI}_U(x) = \min\left(\text{PMI}(x, \mathcal{S}_U), \text{PMI}(x, \mathcal{T}_U)\right). \tag{3.9}$$

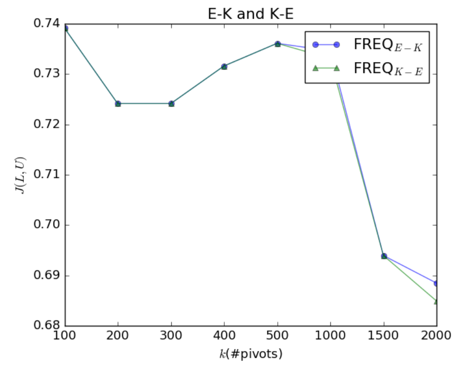
We sort the features x separately in the descending order of $\text{PMI}_L(x)$, and in the ascending order of $\text{PMI}_U(x)$ scores, and select the top-ranked features to define two pivot selection strategies based on PMI.

3.2 Experiments

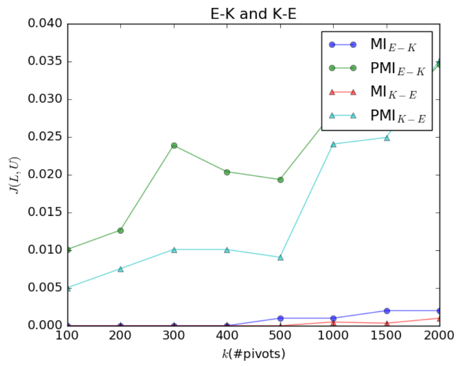
Pivot selection strategies do not concern a specific DA task, hence can be applied in any UDA method that requires a set of pivots. As a concrete evaluation task, in this chapter we use cross-domain sentiment classification, where the goal is to learn a binary sentiment classifier for a target domain using the labelled data from a source domain. This problem has been frequently used in much prior work on UDA as an evaluation task. Therefore, by using cross-domain sentiment classification as an evaluation task, we will be able to perform a fair comparison among the different pivot selection strategies described in Section 3.1.

In our experiments, we use the multi-domain sentiment dataset[2] produced by Blitzer et al. [16]. This dataset consists of Amazon product reviews for four different product types: books (**B**), DVDs (**D**), electronics (**E**), and kitchen appliances (**K**). Each review is assigned with a rating (1-5 stars), a reviewer name and location, a product name, a review title and date, and the review text. Reviews with rating > 3 are labelled as positive, whereas those with rating < 3 are labelled as negative. For each domain, there are 1000 positive and 1000 negative examples, the same balanced composition as the polarity dataset constructed by Pang et al. [130]. The dataset also contains unlabelled reviews (the number of unlabelled review for each domain shown within brackets) for the four domains **K** (16,746), **D** (34,377), **E** (13,116), and **B** (5947). Following previous work, we randomly select 800 positive and 800 negative labelled reviews from each domain as training instances (total number of training instances are $1600 \times 4 = 6400$), and the remainder is used for testing (total number of

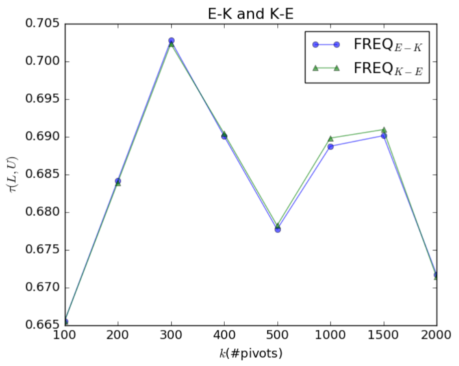
[2]`http://www.cs.jhu.edu/~mdredze/datasets/sentiment/`



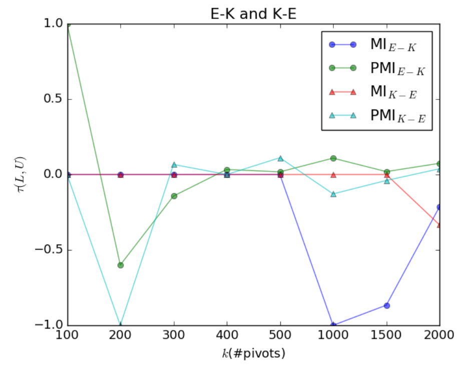
(a)



(b)



(c)



(d)

Figure 3.2: Jaccard coefficient $J(L,U)$ and Kendall coefficient $\tau(L,U)$ for the E-K and K-E adaptation tasks are shown against k of top-k ranked pivots selected using FREQ (left), MI and PMI (right) strategies. For each strategy, we compare the Jaccard coefficient and Kendall coefficient between the sets of pivots selected using the labelled data and the unlabelled data.

test instances are $400 \times 4 = 1600$). With the four domains in the dataset, we generate $\binom{4}{2} = 12$ UDA tasks, which we denote by the source-target domain labels. We select pivots for each pair of source-target domains using $3 \times 2 = 6$ strategies (FREQ, MI, PMI with $\mathcal{L}$ or $\mathcal{U}$) .

3.2.1 Pivot Overlap and Rank Similarity

The degree of overlap between the top-k ranked pivots selected by two strategies is an indicator of the similarity between those strategies. To measure the overlap between the top-k ranked pivot sets $\phi_k(M_1)$ and $\phi_k(M_2)$ selected respectively by two strategies M_1 and M_2, we compute their

Jaccard coefficient, $J(M_1, M_2)$, as follows:

$$J(M_1, M_2) = \frac{|\phi_k(M_1) \cap \phi_k(M_2)|}{|\phi_k(M_1) \cup \phi_k(M_2)|} \tag{3.10}$$

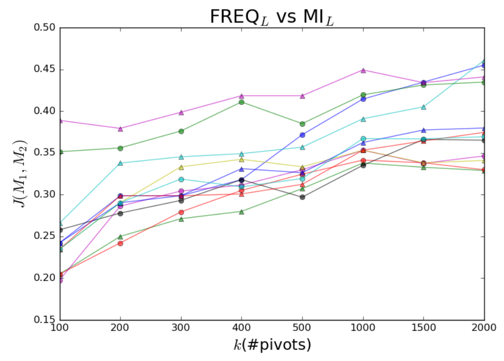
A pivot selection strategy must ideally rank pivots that are useful for DA at the top. However, Jaccard coefficient is insensitive to the ranking among pivots selected by different strategies. To compare the ranks assigned to the common set of pivots selected by two different pivot selection strategies M_1, and M_2, we compute their Kendall's rank correlation coefficient, $\tau(M_1, M_2)$. In practice, pivots selected by M_1 and M_2 will be different. To overcome this issue when comparing ranks of missing elements, we limit the computation of $\tau(M_1, M_2)$ to the intersection $\phi_k(M_1) \cap \phi_k(M_2)$.

For each pivot selection strategy we compute the overlap between the top-k pivots selected using labelled data and unlabelled data. Figure 3.2 shows the results for adapting between **E** and **K** domains. From Figures 3.2a and 3.2b we observe that there is a high overlap between the sets of pivots selected from labelled and unlabelled data using FREQ, compared to that by MI or PMI. This shows that FREQ is relatively insensitive to the label information in the training instances. However, when we increase the number of pivots k selected by a strategy, the overlap gradually drops with FREQ whereas it increases with MI and PMI. This shows that despite the overlap of pivots at top ranks is smaller, it increases when we select more pivots. Because existing UDA methods typically use a smaller (less than 500) set of pivots, the differences between MI and PMI methods will be important.

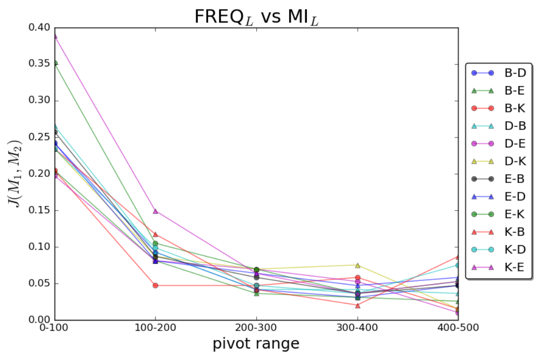
Figures 3.2c and 3.2d show that there is a high correlation between the top ranked pivots, which drops steadily when we select more pivots with a strategy. Because we limit the computation of $\tau(M_1, M_2)$ to the common pivots, the Kendall coefficients obtained for smaller overlapping sets (corresponding to smaller Jaccard coefficients) contain a smaller number of pairwise comparisons, hence insignificant.

Similar trends were observed for all 12 domain pairs. This shows that PMI and MI rank very different sets of pivots at the top ranks. This result supports the proposal by Blitzer et al. [16] to use MI, and Bollegala et al. [23] to use PMI, instead of FREQ for selecting pivots for discriminative DA tasks such as sentiment classification.

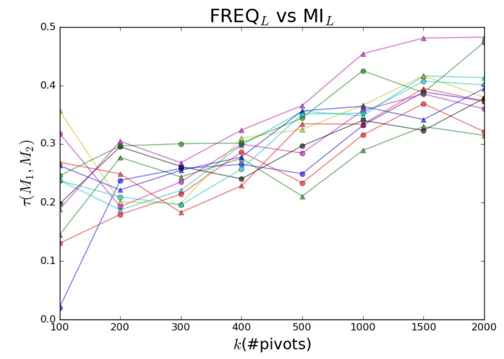
We compare FREQ, MI, and PMI strategies among each other for the same type (labelled or unlabelled) of data. We show results for the comparisons between FREQ$_L$ and MI$_L$ in Figure 3.3. From Figure 3.3a and Figure 3.3c we observe that the overlap and the correlation between the rankings for the sets of pivots selected by those two strategies increase with the number of pivots



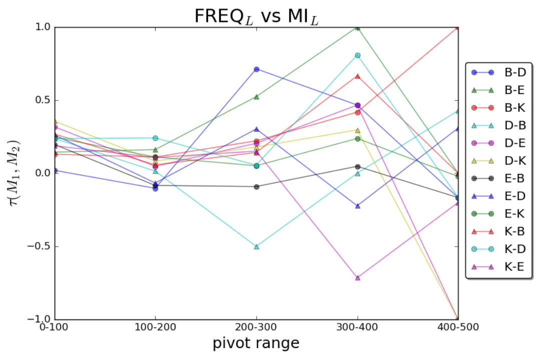
(a) Cumulative comparison of Jaccard coefficient.



(b) Marginal comparison of Jaccard coefficient.



(c) Cumulative comparison of Kendall coefficient.



(d) Marginal comparison of Kendall coefficient.

Figure 3.3: Cumulative and marginal comparisons of pivots selected by FREQ$_L$ and MI$_L$.

selected. However, as seen from Figure 3.3b, the amount of overlap decreases with the number of pivots selected. The overlap between pivots sets is too small to derive any meaningful comparisons using Kendall coefficient beyond 100-200 range. This result implies that although there is some overlap among the top-ranked pivots selected by FREQ and MI from labelled data, the resemblance decreases when we consider lower ranks. Similar trends could be observed for FREQ$_U$ vs. MI$_U$, FREQ$_L$ vs. PMI$_L$, and FREQ$_U$ vs. PMI$_U$. PMI vs. MI show a high degree of overlap (Jaccard coefficients in the range $[0.7, 0.9]$) compared to FREQ vs. PMI and FREQ vs. MI, which can be explained by the close relationship between the definitions of PMI and MI.

Table 3.1 shows the top 5 pivots selected by different strategies for the UDA setting **K**-**E**. We observe a high overlap between the sets of pivots selected by FREQ$_L$ and FREQ$_U$, indicating the insensitivity of FREQ to the label information. Moreover, we observe that with MI- and PMI-based strategies retrieve bigrams as pivots, which would not be ranked at the top by FREQ because the

FREQ$_L$	FREQ$_U$	MI$_L$	MI$_U$	PMI$_L$	PMI$_U$
not	not	not	got+to	waste	got+to
great	great	great	of+room	your+money	of+room
very	good	love	ok+i	waste+your	even+though
good	very	easy	sliding	great+product	using+my
get	no	easy+to	especially+like	worst	ok+i

Table 3.1: Top 5 pivots selected from the six strategies for **K**-**E**. Bigrams are denoted by "+".

S-T	NA	SCL						SFA					
		FREQ$_L$	FREQ$_U$	MI$_L$	MI$_U$	PMI$_L$	PMI$_U$	FREQ$_L$	FREQ$_U$	MI$_L$	MI$_U$	PMI$_L$	PMI$_U$
B-E	52.03	69.75	68.25	68.75	65.75	69.50	**75.75***	70.50	**74.00**	73.25	66.00	**74.00**	71.00
B-D	53.51	70.25	73.25	74.25	59.75	**76.50**	72.00	71.50	**78.00**	69.50	60.00	72.75	74.75
B-K	51.63	76.25	74.25	78.25	63.50	**80.00**	79.50	72.75	74.25	73.00	66.50	**78.50**	75.75
E-B	51.02	60.50	65.25	**66.25**	55.75	64.75	63.00	64.75	64.50	64.00	57.25	**65.75**	59.00
E-D	50.94	68.00	67.75	68.00	66.25	**70.50**	67.00	67.50	**74.50**	63.25	60.75	71.50	65.00
E-K	56.00	81.00	80.50	82.50	80.50	**86.25**	77.50	81.00	82.50	78.25	71.75	**85.50**	79.00
D-B	52.50	72.00	69.25	72.00	56.25	**74.75**	68.50	74.25	**79.00**	69.50	62.00	73.50	73.00
D-E	53.25	71.75	70.50	**74.25**	66.00	**74.25**	65.25	72.50	**75.50**	71.75	65.75	69.00	68.75
D-K	54.39	70.75	75.25	74.00	57.25	**80.50**	77.25	73.75	76.75	74.75	56.50	**81.00**	79.75
K-B	51.29	66.75	67.75	68.50	56.00	**74.00**	70.00	67.75	70.00	69.00	58.00	66.50	**71.25**
K-E	54.86	74.00	74.25	75.50	78.00	**80.00**	72.25	80.50	**84.50**	79.25	70.25	77.25	71.75
K-D	50.94	67.00	65.75	68.00	60.00	**71.50**	67.50	67.25	**77.75***	67.75	60.50	68.00	71.00

Table 3.2: Accuracy of SCL and SFA under different pivot selection strategies. For a domain pair, best results are bolded, whereas statistically significant improvements over the second best (according to Clopper-Pearson confidence intervals $\alpha = 0.05$) are indicated by "*".

frequency of bigrams are typically smaller than that of the constituent unigrams.

3.2.2 Cross-domain Sentiment Classification

To compare the different pivot selection strategies under a UDA setting, we use the selected pivots in two state-of-the-art cross-domain sentiment classification methods: Structural Correspondence Learning (SCL) [16], and Spectral Feature Alignment (SFA) [126]. In both methods, we keep the same hyperparameters and train a binary logistic regression model as the sentiment classifier using unigram and bigram features extracted from Amazon product reviews. The performance of UDA is measured by the classification accuracy – the percentage of correctly classified target domain test reviews. All parameters in SCL and SFA are tuned using validation data selected from extra domains in the multi-domain sentiment dataset. Target domain classification accuracies for all 12 adaptation tasks are shown for SCL and SFA (Table 3.2). We choose top-500 pivots for every pivot selection strategy, and project to 50 dimensional spaces, as recommended for SCL and

SFA [16, 126]. NoAdapt (NA) baseline applies a binary classifier trained on the source domain's labelled instances directly on the target domain's test instances without performing any DA. NA baseline shows the level of performance we would obtain if I did not perform DA. The almost random level accuracy of the NA baseline emphasises the difficulty of the task and the importance of performing DA.

Overall for both SFA and SCL, we observe that for MI and PMI the labelled version performs equally or better than the corresponding unlabelled version. This indicates that source labelled data are important in UDA not only because it is the only source of data that can be used to train a supervised classifier for the target task, but also it enables us to select task specific pivots. For SCL, PMI$_L$ is the single best (10 out of 12 pairs) pivot selection strategy, whereas for SFA it is FREQ$_U$ (7 out of 12 pairs). SCL uses pivot predictors as extra features for learning an adaptive classifier. By using PMI$_L$, the selected pivots consider both mutual association and overcoming the problem of small values if the feature space is large and sparse (§3.1.3). SFA builds a bi-partite graph between pivots and non-pivots based on a co-occurrence matrix. FREQ$_U$ selects pivots from a larger set of instances ($\mathcal{S}_U + \mathcal{T}_U > \mathcal{S}_L^+ + \mathcal{S}_L^-$) that the effect of unlabelled version is more obvious than the labelled version. The results can be explained by the statement from Pan et al. [126] that SFA performs better than SCL when the feature space is large and sparse. Overall MI$_U$ turns out to be the worst strategy. In addition, FREQ$_U$ performs better than the labelled version in SFA because it was computed using a larger number of unlabelled instances from both domains.

B-**E** pair is exceptional in the sense that for SCL it is the only domain-pair where PMI reports better results for the unlabelled strategy than the labelled strategy. This can be explained by the fact that **B** has the smallest number of unlabelled instances for a single domain, and **B**-**E** pair collectively has the smallest number of unlabelled instances for any domain pair. Consequently, the selected pivots occur in the target domain more than in the source domain, making the projection biased towards the target domain's feature distribution. Considering the fact that such unbalanced unlabelled datasets are likely to exist in real-world UDA settings, we believe that it is important to develop robust UDA methods that take into account such biases.

3.3 Summary

In this chapter, we studied the effect of pivot selection strategies on UDA when computed using the labelled data from the source domain, and the unlabelled data from both the source and the target domains. We measured the overlap and the rank similarity among the top-ranked pivots selected using different strategies. These differences among strategies indicate their different performance in using UDA method doing a NLP task. Using cross-domain sentiment classification as an evaluation task, we empirically evaluated the different pivot selection strategies in conjunction with SCL and SFA. The results from different strategies vary on different domain pairs. Overall for SCL, PMI using labelled data turns out to be the best, for SFA is FREQ using unlabelled data. Our experiments reveal useful insights into the role played by pivots in UDA, the label information helps the performance in most of the cases and there is no single best pivot selection strategy to feature-based UDA methods.

Chapter 4

Supervised Methods for Pivot Selection

Considering the importance of pivot selection strategies in feature-based UDA methods, we studied unsupervised methods for pivot selection in Chapter 3. We evaluated the previously proposed pivot selection strategies extensively by similarity measures and classification accuracies when they are applied to UDA methods.

These strategies use unsupervised methods such as frequency [15]. One of the drawbacks of using unsupervised methods is that they select pivots either as common features in the two domains (i.e., using unlabelled data from two domains) or as task-specific features (i.e., using labelled data in the source domain). There is no single best pivot selection strategy for feature-based UDA methods (§3.2). In Section 4.1, we propose a novel pivot selection method to select pivots that are common and task-specific by exploiting the available labelled and unlabelled data. Specifically, we propose a method to find common features in the two domains using MMD [24]. In Section 3.2.2, we found label information helped the performance of pivot selection applied in UDA. The proposed method in this chapter also incorporates label information.

In Section 4.2, we consider the problem of data imbalance encountered in UDA. This problem arises when we are learning a model with unequal number of training instances for different target classes [25, 28, 93]. Using Movie Review Data provided by Pang et al. [130] as an example, there are 752 negative and 1301 positive reviews. If we want to train a sentiment classifier using the mixture of these reviews, it will be difficult to learn a classifier that is not biased to positive reviews. Previously proposed strategies for pivot selection have ignored this problem. We explore the effect of data imbalance on cross-domain POS tagging and propose the use of F-score to solve this problem.

	Books	*Laptops*
+	I think that this is an **excellent** book.	This is a **powerful**, yet **compact** laptop. An **excellent** choice for a travelling businessman!
-	This book is a **disappointment**, definitely **not recommended**.	It's the **worst** laptop I have ever had. It is **slow** and forever **crashing**.
-	Found myself skipping most of it found it **boring**.	**Pricy** laptop and does not deliver. A big **disappointment**. **Loud** fan, **noisy** hard drive. Never buy again.

Table 4.1: Positive (+) and negative (-) sentiment reviews on *Books* and *Laptops*. Sentiment-bearing features are shown in bold, whereas selected pivots are underlined.

4.1 Task-Specific Pivot Selection

Feature-based methods for UDA first learn an embedding between the source and target domain feature spaces, and then learn a classifier for the target task (e.g. sentiment classification) in this embedded space [15, 23, 126]. In order to learn this embedding, these methods must select a subset of common features (here onwards referred to as *pivots*) to the two domains. For example, consider the user reviews shown in Table 4.1 for *Books* and *Laptops* selected from Amazon[1]. Sentiment-bearing features, such as *powerful*, *loud*, *crashing*, *noisy* are specific to *laptops*, whereas *boring* would often be associated with a *book*. Conversely, features such as *disappointment* and *excellent* are likely to be domain-independent, hence suitable as pivots for UDA.

All existing strategies for selecting pivots are based on unsupervised methods, such as selecting the top-frequent features that are common to both source and target domains [15]. We have discussed FREQ, MI and PMI in Chapter 3. In addition to FREQ, MI [16] and PMI [23], Positive Pointwise Mutual Information (PPMI) [21] has been proposed in prior work on UDA as pivot selection strategies.

There are two fundamental drawbacks associated with all existing unsupervised pivot selection strategies. First, existing pivot selection strategies focus on either: (a) selecting a subset of the common features to source and target domains as pivots, or (b) selecting a subset of task-specific features (eg. sentiment-bearing features selected based on source domain's labelled training instances) as pivots. However, as we observe later in our experiments, to successfully adapt to a new domain, the pivots must be both domain-independent (thereby capturing sufficient information for the knowledge transferring from the source to the target), as well as task-specific (thereby ensuring the selected pivots are accurately related to the labels). Second, it is non-trivial as to how we can combine the two requirements (a) and (b) in a consistent manner to select a set of pivots. Pivot selection can be seen as an optimal subset selection problem in which we must select a subset of

[1]`www.amazon.com`

the features from the intersection of the feature spaces of the two domains. Optimal subset selection is an NP-complete problem [48], and subset enumeration methods are practically infeasible considering that the number of subsets of a feature set of cardinality n is $2^n - 1$, where $n > 10^4$ in typical NLP tasks.

To overcome the above-mentioned limitations of existing pivot selection strategies, we propose **TSP** – Task-Specific Pivot selection for UDA. Specifically, we define two criteria for selecting pivots: one based on the *similarity between the source and target domains under a selected subset of features* (§4.1.1.1), and another based on *how well the selected subset of features capture the information related to the labelled instances in the source domain* (§4.1.1.2). We show that we can model the combination of the two criteria as a single constrained quadratic programming problem that can be efficiently solved for large feature spaces (§4.1.1.3). The reduction of pivot selection from a subset selection problem to a feature ranking problem, based on *pivothood* (α) scores learnt from the data, enables us to make this computation feasible. Moreover, the salience of each criteria can be adjusted via a *mixing parameter* (λ) enabling us to gradually increase the level of task-specificity in the selected pivots, which is not possible with existing heuristic-based pivot selection methods.

We compare the proposed TSP selection method against existing pivot selection strategies using two UDA methods: SFA [126], and SCL [16] on a benchmark dataset for cross-domain sentiment classification. We observe that TSP, when initialised with pre-trained word embeddings trained using Continuous Bag-of-Words (CBOW) [118] and Global Vector Prediction (GloVe) [131] significantly outperforms previously proposed pivot selection strategies for UDA with respect to most domain pairs. Moreover, analysing the top-ranked pivots selected by TSP for their sentiment polarity, we observe that more sentiment-bearing pivots are selected by TSP when we increase the mixing parameter. More importantly, our results show that we must consider both criteria discussed above when selecting pivots to obtain an optimal performance in UDA, which cannot be done using existing pivot selection strategies.

4.1.1 Methods

Let us consider a source domain $\mathcal{S}$ consisting of a set of labelled instances $\mathcal{D}_L^{(S)}$ and unlabelled instances $\mathcal{D}_U^{(S)}$. Without loss of generality we assume the target adaptation task is binary classification, where we have a set of positively labelled instances $\mathcal{D}_+^{(S)}$ and a set of negatively labelled instances $\mathcal{D}_-^{(S)}$. Although we limit our discussion to the binary classification setting for simplicity, we note that the proposed method can be easily extended to other types of DA setting, such as multi-class classification and regression. For the target domain, denoted by $\mathcal{T}$, under UDA we assume the availability of only a set of unlabelled instances $\mathcal{D}_U^{(T)}$.

Given a pair of source and target domains, we propose a novel pivot selection method for UDA named **TSP** – Task Specific Pivot Selection. TSP considers two different criteria when selecting pivots.

First, we require that the *selected set of pivots must minimise the distance between the source and target domains*. The exact formulation of distance between domains and the corresponding optimisation problem are detailed in Section 4.1.1.1.

Second, we require that the *selected set of pivots be task-specific*. For example, if the target task is sentiment classification, then the selected set of pivots must contain sentiment bearing features. Given labelled data (annotated for the target task) from the source domain, we describe an objective function in Section 4.1.1.2 for this purpose.

Although the above-mentioned two objectives could be optimised separately, by jointly optimising for both objectives simultaneously we can obtain task-specific pivots that are also transferable to the target domain. In Section 4.1.1.3, we formalise this joint optimisation problem and provide a quadratic programming-based solution. For simplicity, we present TSP for the pairwise UDA case, where we attempt to adapt from a single source domain to a single target domain. However, TSP can be easily extended to multi-source and multi-target UDA settings following the same optimisation procedure.

4.1.1.1 Pivots are Common to the Two Domains

Theoretical studies in UDA show that the upper bound on the classification accuracy on a target domain depends on the similarity between the source and the target domains [11]. Therefore, if we can somehow make the source and target domains similar by selecting a subset of the features from the intersection of their feature spaces, then we can learn better UDA models.

To formalise this idea into an objective we can optimise for, let us denote the possibility of a feature w_k getting selected as a pivot by $\alpha_k \in [0, 1]$. We refer to α_k as the *pivothood* of a feature $w_k \in \mathcal{W}$ ($|\mathcal{W}| = K$). Higher pivothood values indicate that those features are more appropriate as pivots. The features could be, for example in sentiment classification, n-grams of words, POS tags, dependencies or any of their combinations. For simplicity of the disposition, let us assume that w_k are either unigrams or bigrams of words, and that we have pre-trained word embeddings $\boldsymbol{w}_k \in \mathbb{R}^D$ obtained using some word embedding learning algorithm. The proposed method does not assume any specific properties of the word embeddings used; any fixed-dimensional representation of the features is sufficient, not limited to word embeddings.

Given source domain unlabelled data, we can compute, $\boldsymbol{c}^{(S)}$, the centroid for the source domain as

$$\boldsymbol{c}^{(S)} = \frac{1}{\left|\mathcal{D}_U^{(S)}\right|} \sum_{d \in \mathcal{D}_U^{(S)}} \sum_{w_k \in d} \alpha_k \phi(w_k, d) \boldsymbol{w}_k. \tag{4.1}$$

Here, $\phi(w_k, d)$ indicates the salience of w_k as a feature representing an instance d, such as the tf-idf measure popularly used in text classification. Likewise, we can compute the $\boldsymbol{c}^{(T)}$ centroid for the target domain using the unlabelled data from the target domain as

$$\boldsymbol{c}^{(T)} = \frac{1}{\left|\mathcal{D}_U^{(T)}\right|} \sum_{d \in \mathcal{D}_U^{(T)}} \sum_{w_k \in d} \alpha_k \phi(w_k, d) \boldsymbol{w}_k. \tag{4.2}$$

The centroid can be seen as a representation for the domain consisting all of the instances (reviews in the case of sentiment classification). If two domains are similar under some representation, then it will be easier to adapt from one domain to the other. Different distance measures can be used to compute the distance (or alternatively the similarity) between two domains under a particular representation such as ℓ_1 distance (Manhattan distance) or ℓ_2 distance (Euclidean distance). In this work, we use Euclidean distance for this purpose.

The problem of selecting a set of pivots can then be formulated as minimising the squared Euclidean distance between $\boldsymbol{c}^{(S)}$ and $\boldsymbol{c}^{(T)}$ given by

$$\min_{\boldsymbol{\alpha}} \left|\left| \boldsymbol{c}^{(S)} - \boldsymbol{c}^{(T)} \right|\right|_2^2. \tag{4.3}$$

Here, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)^\top$ is the pivothood indicator vector.

Assuming $\phi(w, d) = 0$ for $w \notin d$, and by substituting (4.1) and (4.2) in (4.3), we can re-write the objective as

$$\min_{\boldsymbol{\alpha}} \left|\left| \sum_{k=1}^{K} \alpha_k f(w_k) \boldsymbol{w}_k \right|\right|_2^2, \tag{4.4}$$

$f(w_k)$ can be pre-computed and is given by,

$$f(w_k) = \frac{1}{\left|\mathcal{D}_U^{(S)}\right|} \sum_{d \in \mathcal{D}_U^{(S)}} \phi(w_k, d) - \frac{1}{\left|\mathcal{D}_U^{(T)}\right|} \sum_{d \in \mathcal{D}_U^{(T)}} \phi(w_k, d). \tag{4.5}$$

Minimisation of (4.4) can be trivially achieved by setting $\alpha_k = 0, \forall k = 1, \ldots, K$. To avoid this trivial solution and to make the objective scale invariant, we introduce the constraint $\sum_k \alpha_k = 1$ where $\alpha_k \geqslant 0$.

Further, if we define $\boldsymbol{u}_k = f(w_k)\boldsymbol{w}_k$, then (4.4) reduces to the constrained least square regression problem given by

$$\begin{aligned} \min_{\boldsymbol{\alpha}} & \left\| \sum_{k=1}^{K} \alpha_k \boldsymbol{u}_k \right\|_2^2, \\ \text{s.t.} & \sum_{k=1}^{K} \alpha_k = 1, \\ & \alpha_k \geqslant 0. \end{aligned} \tag{4.6}$$

4.1.1.2 Pivots are Task Specific

The objective defined in the previous section is computed using only unlabelled data, hence agnostic to the target task. However, prior work on UDA [16, 23] has shown that we must select pivots that are specific to the target task in order to be accurately adapted to cross-domain prediction tasks. Labelled data can be used to measure the task specificity of a feature. However, in UDA, the only labelled data we have is from the source domain. Therefore, we define the task specificity γ_k of a feature w_k as

$$\gamma_k = \left(h(w_k, \mathcal{D}_+^{(S)}) - h(w_k, \mathcal{D}_-^{(S)}) \right)^2. \tag{4.7}$$

Here, $h(w_k, \mathcal{D}_+^{(S)})$ denotes the association between the feature w_k and the source domain positive labelled instances. We use squared value to increase the difference among features. A wide range of association measures such as MI, PMI, PPMI, χ^2, log-likelihood ratio can be used as h [112]. We use PMI to demonstrate the proposed method according to its performance applied to feature-based UDA methods (§3.2). PMI [33] between a feature w_k and a set of training instances $\mathcal{D}$ is given by

$$\text{PMI}(w_k, \mathcal{D}) = \log \left(\frac{p(w_k, \mathcal{D})}{p(w_k)p(\mathcal{D})} \right), \tag{4.8}$$

we compute the probabilities p in (4.8) using frequency counts. PPMI [105] is a variation of PMI and follows PMI$(w_k, \mathcal{D})$ defined by (4.8), PPMI can be given by,

$$\text{PPMI}(w_k, \mathcal{D}) = \max \left(\text{PMI}(w_k, \mathcal{D}), 0 \right), \tag{4.9}$$

we use PPMI as h in our experiments to reduce the noise caused by negative PMI values.

Given the task specificity of features, we can formulate the problem of pivot selection as

$$-\max_{\boldsymbol{\alpha}} \frac{\sum_k \alpha_k \gamma_k}{\sum_k \gamma_k}. \tag{4.10}$$

4.1.1.3 Joint Optimisation

Ideally, we would like to select pivots that: (a) make source and target domain closer, as well as (b) are specific to the target task. A natural way to enforce both of those constraints is to linearly combine the two objectives given by (4.6) and (4.10) into the joint optimisation problem (4.11).

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \left|\left| \sum_{k=1}^{K} \alpha_k \boldsymbol{u}_k \right|\right|_2^2 &- \lambda \frac{\sum_k \alpha_k \gamma_k}{\sum_k \gamma_k}, \\ \text{s.t.} \sum_{k=1}^{K} \alpha_k &= 1, \\ \alpha_k &\geqslant 0, \end{aligned} \tag{4.11}$$

the mixing parameter $\lambda \geq 0$ is a hyperparameter that controls the level of task specificity of the selected pivots. For example, by setting $\lambda = 0$ we can select pivots purely using unlabelled data. When we gradually increase λ, the source domain labelled data influences the pivot select process, making the selected pivots more task specific.

To further simplify the optimisation problem given by (4.11), let us define $\mathbf{U} \in \mathbb{R}^{D \times K}$ to be a matrix where the K columns correspond to $\boldsymbol{u}_k \in \mathbb{R}^D$, and $\boldsymbol{c}$ to be a vector whose k-th element is set to

$$c_k = -\lambda \gamma_k / \sum_k \gamma_k.$$

Then, (4.11) can be written as the following quadratic programming problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T \mathbf{U}^\top \mathbf{U} \boldsymbol{\alpha} &+ \mathrm{c}^T \boldsymbol{\alpha}, \\ \text{s.t. } \boldsymbol{\alpha}^T \mathbf{1} &= 1, \\ \boldsymbol{\alpha} &\geqslant 0. \end{aligned} \tag{4.12}$$

We solve the quadratic programming problem given by (4.11) using the conjugate gradient (CG) method [121]. We use CVXOPT[2] to solve (4.12) to obtain pivothoods α_k. We rank the features w_k in the descending order of their corresponding α_k and select the top-ranked features as pivots for a UDA method.

The computational complexity of the quadratic programming problem is dominated by the eigenvalue decomposition of $\mathbf{U}^\top\mathbf{U}$, which is of the dimensionality $K \times K$. Recall that K is the total number of features in the feature space representing the source and target domain instances. Computing the eigenvalue decomposition of a $K \times K$ square matrix is $\mathcal{O}(K^3)$. However, we can use randomised truncated Eigensolvers to compute the top-ranked eigenvalues (corresponding to the best pivots), without having to perform the full eigenvalue decomposition [71].

4.1.2 Experiments

Because the purpose of selecting pivots is to perform UDA, and because we cannot determine whether a particular feature is suitable as a pivot by manual observation, the most direct way to evaluate a pivot selection method is to use the pivots selected by that method in a state-of-the-art UDA method and evaluate the relative increase/decrease in performance in that DA task. For this purpose, we select SCL and SFA, which are UDA methods and perform a cross-domain sentiment classification task. We use the same hyperparameters from the original papers. The task here is to learn from labelled reviews from the source domain (a collection of reviews about a particular product) and predict sentiment for a different target domain (a collection of reviews about a different product). It is noteworthy that sentiment classification is used here purely as an evaluation task, and our goal is not to improve the performance of sentiment classification itself but to evaluate pivot selection methods. Therefore, we keep all other factors other than the pivots used by the UDA methods fixed during our evaluations.

We use the multi-domain sentiment dataset [16], which contains Amazon user reviews for the four product categories *Books* (**B**), *Electronic appliances* (**E**), *Kitchen appliances* (**K**), and *Dvds* (**D**). For each domain we have 1000 positive and 1000 negative reviews, and ca. 15,000 unlabelled reviews. We use the standard split of $800 \times 2 = 1600$ train and $200 \times 2 = 400$ test reviews. We generate 12 DA tasks by selecting one of the four domains as the source and another as the target. We represent a review using a bag-of-features consisting of unigrams and bigrams, excluding a standard stop words list. We drop features that occur less than 5 times in the entire dataset to remove noise. A binary logistic regression classifier with an ℓ_2 regulariser is trained in each setting to develop a binary sentiment classifier. The regularisation coefficient is tuned using the *music*

[2]`http://cvxopt.org/`

domain as a development domain, which is not part of the train/test data. Although different classifiers could be used in place of logistic regression, by using a simpler classifier we can readily evaluate the effect of the pivots on the UDA performance. The classification accuracy on the target domain's test labelled reviews is used as the evaluation measure.

On average, for a domain pair we have $K = 2648$ features. We set the number of pivots selected to the top-ranked 500 pivots in all domain pairs. Later in Section 4.1.2.4 we study the effect of the number of pivots on the performance of the proposed method. We use the publicly available $D = 300$ dimensional GloVe[3] (trained using 42B tokens from the Common Crawl) and CBOW[4] (trained using 100B tokens from Google News) embeddings as the word representations required by TSP. For bigrams not listed in the pretrained models, we sum the embeddings of constituent unigrams following prior work on compositional approaches for creating phrase (or sentence) embeddings using word embeddings [5, 81]. We denote TSP trained using GloVe and CBOW embeddings respectively by **T-GloVe** and **T-CBOW**. By using two different types of word embeddings we can evaluate the dependance (if any) of TSP on a particular type of word embedding learning algorithm. Later in Section 4.1.2.4, we evaluate the effect of counting-based and prediction-based word embeddings on the performance of TSP when used for computing source and target centroids respectively in (4.1) and (4.2). We use the inverse document frequency (IDF) [150] as the feature salience ϕ and PPMI as h in our experiments.[5]

4.1.2.1 Cross-Domain Sentiment Classification

To evaluate the effect on performance of a UDA method when we select pivots using the proposed TSP and previously proposed pivots selection methods, we compare the performance of SCL and SFA in 12 different adaptation tasks from a source **S** to a target domain **T** as shown in Tables 4.2 and 4.3. We use the binomial exact test at two significance levels to evaluate statistical significance. The *no-adapt* (**NA**) lower-baselines, which simply applies a model trained using the source domain labelled data on the target domain's test data without performing DA, produced a near random-level performance indicating the importance of performing DA for obtaining good performance.

In SCL, T-CBOW reports the best performance in 8 domain-pairs, whereas T-GloVe in 3. Although in B-E and K-D pairs respectively PMI$_U$ and PMI$_L$ report the best results, the difference in performance with T-CBOW is not significant. Overall, T-CBOW is the best method in SCL closely followed by T-GloVe. For each of the previously proposed pivot selection methods except

[3]`http://nlp.stanford.edu/projects/glove/`
[4]`https://code.google.com/archive/p/word2vec/`
[5]Performance of TSP was found to be robust over a wide-range of ϕ and h combinations.

S-T	NA	SCL									
		FREQ$_L$	FREQ$_U$	MI$_L$	MI$_U$	PMI$_L$	PMI$_U$	PPMI$_L$	PPMI$_U$	T-CBOW	T-GloVe
B-E	52.03	69.75	68.25	68.75	65.75	69.50	**75.75***	69.50	67.50	73.25	75.25*
B-D	53.51	70.25	73.25	74.25	59.75	76.50	72.00	76.50	70.50	**77.50**	77.00
B-K	51.63	76.25	74.25	78.25	63.50	80.00*	79.50	80.00*	77.00	**83.25****	82.00**
E-B	51.02	60.50	65.25	66.25	55.75	64.75	63.00	64.25	60.50	**68.75**	67.25
E-D	50.94	68.00	67.75	68.00	66.25	70.50	67.00	71.50	65.50	73.25	**74.00***
E-K	56.00	81.00	80.50	82.50	80.50	86.25*	77.50	85.75*	77.25	**87.50****	**87.50****
D-B	52.50	72.00	69.25	72.00	56.25	74.75	68.50	75.75*	69.50	**76.75***	75.50*
D-E	53.25	71.75	70.50	74.25	66.00	74.25	65.25	74.00	65.25	**76.25**	75.75
D-K	54.39	70.75	75.25	74.00	57.25	80.50	77.25	80.25	79.75	83.00**	**84.00****
K-B	51.29	66.75	67.75	68.50	56.00	74.00*	70.00	74.00*	69.25	**75.25***	74.25*
K-E	54.86	74.00	74.25	75.50	78.00	80.00*	72.25	80.00*	71.75	**82.00****	81.25*
K-D	50.94	67.00	65.75	68.00	60.00	**71.50**	67.50	**71.50**	68.75	70.75	70.75
AVG	52.70	70.67	71.00	72.52	63.75	75.21	71.30	75.25	70.21	**77.30***	77.04*

Table 4.2: Classification accuracy of SCL using pivots selected by TSP (T-CBOW & T-GloVe) and prior methods. For each domain pair, the best results are given in bold font. The last row is the average across all the domain pairs. Statistically significant improvements over the FREQ$_U$ baseline according to the binomial exact test, are shown by "*" and "**" respectively at $p = 0.01$ and $p = 0.001$ levels.

frequency, we observe that the performance is always better when we use labelled data (denoted by subscript L) than unlabelled data (denoted by subscript U) to select pivots. Recalling that labelled data is only available from the source domain and is the number is smaller compared to the unlabelled data available from both domains, frequency as a pivot selection method is unable to identify useful pivots from labelled data as demonstrated by the lower performance of FREQ$_L$ compared to FREQ$_U$.

In SFA, T-GloVe reports the best performance in 5 domain-pairs, whereas T-CBOW in 3. Among the previously proposed pivot selection methods, FREQ$_U$ performs best in 5 domain-pairs. However, we observe that overall, T-GloVe and T-CBOW outperform all the other pivot selection methods. Moreover, the difference between performance reported by T-CBOW and T-GloVe is not significant, indicating that TSP is relatively robust against the actual word embedding method being used.

Overall, we observe that SCL benefits more from accurate pivot selection than SFA. SCL learns separate linear predictors for each pivot using non-pivots (i.e. features other than pivots) as features, whereas SFA builds a bi-partite graph between pivots and non-pivots on which eigenvalue decomposition is applied to obtain a lower-dimensional embedding of pivots. Therefore, the effect of pivots on SCL is more direct and critical than in SFA.

S-T	NA	SFA									
		FREQ$_L$	FREQ$_U$	MI$_L$	MI$_U$	PMI$_L$	PMI$_U$	PPMI$_L$	PPMI$_U$	T-CBOW	T-GloVe
B-E	52.03	70.50	74.00	73.25	66.00	74.00	71.00	74.00	70.50	**74.75**	73.75
B-D	53.51	71.50	**78.00**	69.50	60.00	72.75	74.75	72.75	73.00	77.75	**78.00**
B-K	51.63	72.75	74.25	73.00	66.50	78.50	75.75	78.50	75.00	**81.00***	80.75*
E-B	51.02	64.75	64.50	64.00	57.25	65.75	59.00	64.00	57.75	64.50	**67.00**
E-D	50.94	67.50	**74.50**	63.25	60.75	71.50	65.00	71.50	61.50	73.50	71.50
E-K	56.00	81.00	82.50	78.25	71.75	85.50	79.00	85.25	78.00	**87.00**	85.50
D-B	52.50	74.25	**79.00**	69.50	62.00	73.50	73.00	73.75	67.75	**79.00**	78.00
D-E	53.25	72.50	75.50	71.75	65.75	69.00	68.75	69.00	66.50	74.25	**76.75**
D-K	54.39	73.75	76.75	74.75	56.50	81.00	79.75	81.00	79.00	81.50	**83.25***
K-B	51.29	67.75	70.00	69.00	58.00	66.50	**71.25**	66.50	**71.25**	69.00	70.00
K-E	54.86	80.50	**84.50**	79.25	70.25	77.25	71.75	77.25	72.50	80.50	79.75
K-D	50.94	67.25	**77.75**	67.75	60.50	68.00	71.00	68.00	71.00	72.25	72.50
AVG	52.70	72.00	75.94	71.10	62.94	73.60	71.67	73.46	70.31	76.25	**76.40**

Table 4.3: Classification accuracy of SFA using pivots selected by TSP (T-CBOW & T-GloVe) and prior methods. For each domain pair, the best results are given in bold font. The last row is the average across all the domain pairs. Statistically significant improvements over FREQ$_U$ baseline according to the binomial exact test are shown by "*" and "**" respectively at $p = 0.01$ and $p = 0.001$ levels.

4.1.2.2 Effect of the mixing parameter

λ	T-CBOW	T-GloVe
0	unlike complaint fair i+havent name	wont whether yes complete so+good
10^{-5}	sent+it of+junk im+very fast+and an+excellent	go+wrong of+junk im+very i+called an+excellent
10^{-4}	of+junk value+for i+called fast+and it+comes	is+excellent sent+it im+very an+excellent good+price
10^{-3}	an+excellent of+junk i+called fast+and pleased+with	sent+it stopped+working of+junk is+perfect very+happy
10^{-2}	i+love of+junk dont+buy i+called very+happy	i+love of+junk i+called a+great very+happy
1.0	return stopped of+junk dont+buy i+called	stopped of+junk dont+buy i+called very+happy

Table 4.4: Top 5 pivots selected for adapting from **E** to **K** by T-CBOW and T-GloVe for different mixing parameter values. Bigrams are denoted by "+".

To evaluate the effect of the mixing parameter on the pivots selected by TSP, we use the pivots selected by TSP for different λ values with SCL and SFA, and measure the classification accuracy on a target domain's sentiment labelled test reviews. Hyperparameters such as the number of singular vectors used in SCL and the number of latent dimensions used in SFA are tuned using the *music* development domain as the target for each of the four source domains. We show the results for adapting from the **D** to **K** pair with SCL and SFA in Figure 4.1. We observe that when we do not require pivots to be task-specific (i.e. $\lambda = 0$) the accuracy is low. However, when we gradually increase λ thereby enforcing the criterion that the selected pivots must be also task-specific, we

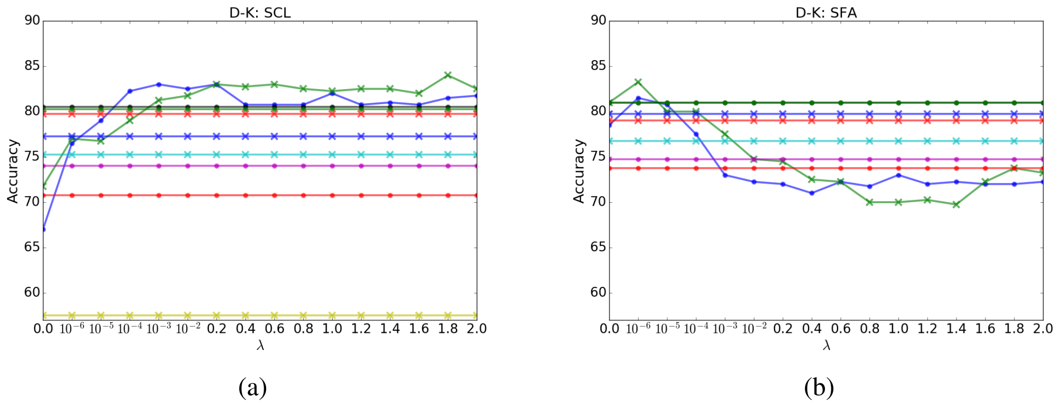


Figure 4.1: Accuracy of SCL and SFA when adapting from **D** source to **K** target (x-axis not to scale).

observe a steady improvement in accuracy, which then saturates.

This result shows the importance of considering both objectives (4.6) and (4.10) appropriately when selecting pivots, instead of focusing on only one of the two. Moreover, performance of TSP is relatively robust for $\lambda > 10^{-3}$, which is attractive because we do not have to fine-tune the mixing-parameter for each UDA setting.

4.1.2.3 Sentiment Polarity of the Selected Pivots

To evaluate whether the pivots selected by TSP are task-specific, we conducted the following experiment. For a particular value of λ, we sort the features in the descending order of their α values, and select the top-ranked 500 features as pivots. Because our task in this case is sentiment classification, we would expect the selected pivots to be sentiment-bearing (i.e. containing words that express sentiment). Next, we compare the selected pivots against the sentiment polarity ratings provided in SentiWordNet [54]. SentiWordNet classifies each synset in the WordNet into positive, negative or neutral sentiment polarities such that any word with a positive sentiment will have a positive score, a negative sentiment a negative score, and zero otherwise. For bigrams not listed in the SentiWordNet, we compute the average polarity of the two constituent unigrams as the sentiment polarity of the bigrams. We consider a pivot to be task-specific if it has a non-zero polarity score.

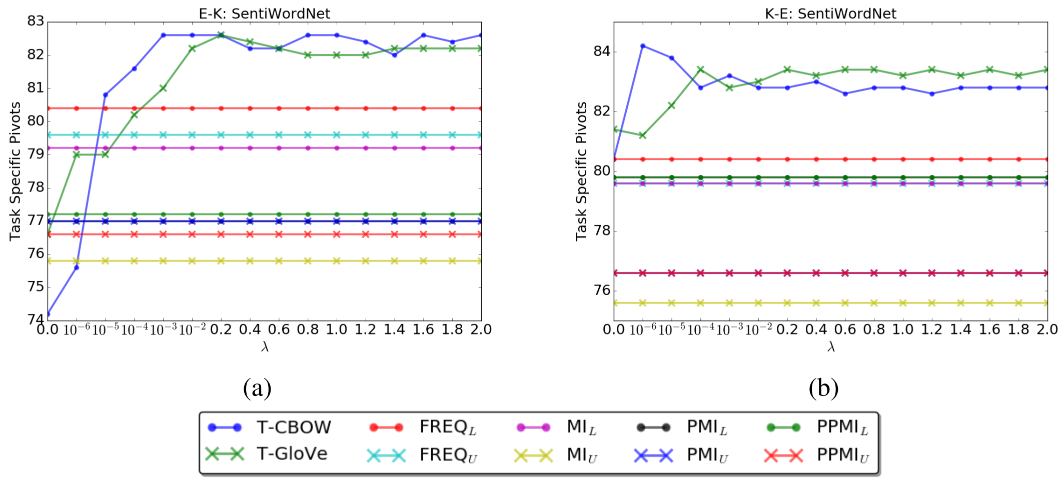


Figure 4.2: Task specific pivots for adapting between **E** and **K** by TSP with $\lambda \in [0, 2]$ (x-axis not to scale).

Figure 4.2 shows the percentage of the task-specific (sentiment-bearing) pivots among the top-500 pivots selected by TSP for different λ values when adapting between **E** and **K**. We observe that initially, when λ is small, the percentage of task-specific pivots is small. However, when we increase λ, thereby encouraging TSP to consider the task-specificity criterion more, we end up with pivots that are more sentiment-bearing. Moreover, when $\lambda > 10^{-3}$ the percentage of the task-specific pivots remains relatively stable, indicating that We have selected all pivots that are sentiment-bearing for the particular domain-pair.

As a qualitative example, we show the top-5 pivots ranked by their pivothood scores by T-CBOW and T-GloVe when adapting between **E** and **K** in Table 4.4. At $\lambda = 0$, we observe that most of the top-ranked pivots are not sentiment-bearing. However, when we increase λ, we observe that more and more sentiment-bearing pivots appear among the top-ranks. This result re-confirms that TSP can accurately capture task-specific pivots by jointly optimising for the two criteria proposed in Section 4.1.1. Interestingly, we observe that many pivots are selected by both T-CBOW and T-GloVe such as *an+excellent*, *i+love*, and *very+happy*. This is encouraging because it shows that TSP depends weakly on the word representation method we use, thereby enabling us to potentially use a wide-range of existing pre-trained word embeddings with TSP.

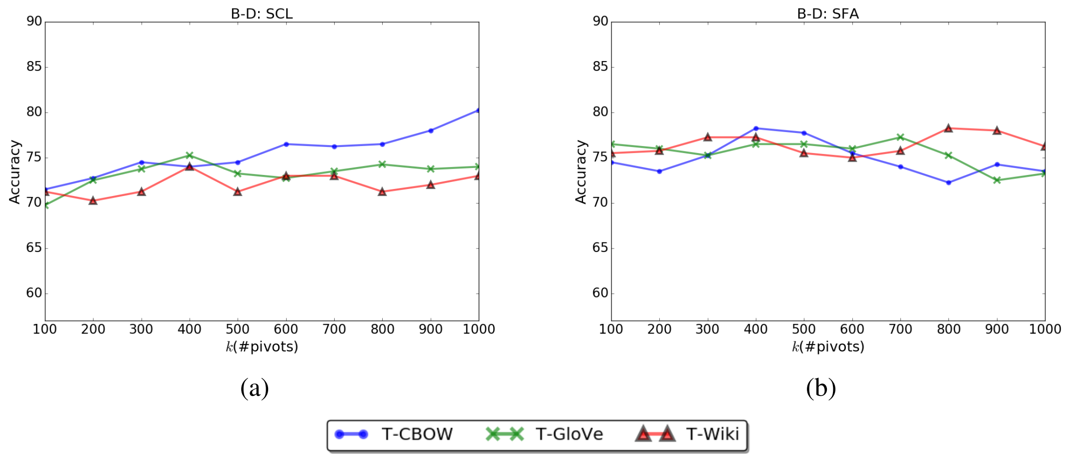


Figure 4.3: Accuracy of SCL and SFA when adapting from **B** source to **D** target, from k (number of pivots) in the range $[100, 1000]$. The mixing parameter is set to $\lambda = 10^{-3}$.

4.1.2.4 Number of Selected Pivots and Effect of Word Embeddings

The number of pivots selected and the word embeddings are external inputs to TSP. In this section, we experimentally study the effect of the number of pivots selected by our proposed TSP and the word embeddings on the performance of the cross-domain sentiment classification. Specifically, we use TSP to select different k numbers of pivots, with three types of word embeddings: pre-trained word embeddings using CBOW (**T-CBOW**), pre-trained word embeddings using GloVe (**T-GloVe**), and counting-based word embeddings computed from Wikipedia (referred to as **T-Wiki**).

Counting-based word embeddings differ from prediction-based word embeddings such as CBOW and GloVe in several important ways [9, 100]. In counting-based word embeddings we represent a target word by a vector where the elements correspond to words that co-occur with the target word in some contextual window. Entire sentences or windows of a fixed number of tokens can be considered as the co-occurrence window. Next, co-occurrences are aggregated over the entire corpus to build the final representation for the target word. Because any word can co-occur with the target word in some contextual window, counting-based word representations are often high dimensional (e.g. greater than 10^5), in comparison to prediction-based word embeddings (e.g. less than 1000 dimensions). Moreover, only a handful of words will co-occur with any given target word even in a large corpus, producing highly sparse vectors. Unlike in prediction-based word embeddings where dimensions correspond to some latent attributes, counting-based word representations are easily interpretable because each dimension in the representation is explicitly assigned to a particular word in the vocabulary. By using both counting-based as well as prediction-based embeddings in the proposed method as the word embeddings used in (4.1) and (4.2), we can evaluate the effect of the word embeddings on the overall performance of the proposed method.

To build the counting-based embeddings (T-Wiki) we selected the January 2017 dump of English Wikipedia[6], and processed it using a Perl script[7] to create a corpus of 4.6 billion tokens. We select unigrams occurring at least 1000 times in this corpus amounting to a vocabulary of size $73,954$. We represent each word by a vector whose elements correspond to the PPMI values computed from the co-occurrence counts between words in this Wikipedia corpus.

To study the effect of the number of pivots k on the performance of TSP, we fix the mixing parameter $\lambda = 10^{-3}$ for all domain pairs and vary $k \in [100, 1000]$. We show the performance of SCL and SFA for the B-D pair respectively in Figures 4.3a and 4.3b. From Figure 4.3a, we observe that, overall for SCL, T-CBOW outperforms the other two embeddings across a wide range of pivot set sizes. Moreover, its performance increases with k, which indicates that with larger pivot sets it can better represent a domain using the centroid. T-Wiki reports lowest accuracies across all k values. Prior work evaluating word embedding learning algorithms has shown that prediction-based word embeddings outperform counting-based word embeddings for a wide-range of language processing tasks [9]. Conversely, for SFA (Figure 4.3b) we do not observe much difference in performance among the different word embeddings. Overall, the best performance is reported using **T-CBOW** with $k = 400$ pivots. This observation is in agreement with the recommendation by Pan et al. [126] to use 500 domain independent (pivot) features for this dataset. Similar trends were observed for all 12 domain pairs.

4.2 Data Imbalance

In many real-world applications involving machine learning methods we frequently encounter two important problems: (a) the training and testing data distributions being different (*data mismatch*) [11, 15, 16], and (b) large discrepancy in terms of the amount of training data available for the different target classes we would like to learn (*data imbalance*) [69, 135, 189]. Data imbalance arises when we have unequal numbers of training instances for the different target classes we would like to learn [25, 28, 93]. For example, in a sentiment classification setting, we might have a disproportionately large amount of positively labelled data to negatively labelled data. If we

[6] `https://dumps.wikimedia.org`
[7] `http://mattmahoney.net/dc/textdata.html`

simply mix all available data and train a classifier, it might be incorrectly biased towards predicting the positive label by default. Under or oversampling methods that respectively select a subset of training instances from the majority class or take multiple samples from the minority class have been proposed to overcome data imbalance issues in machine learning [72].

We study cross-domain POS tagging [153, 154] in which we can encounter both the data mismatch and data imbalance problems discussed above. POS tagging is the task of assigning POS categories such as *noun*, *verb*, *adjective*, *adverb*, etc. to each word in a sentence. POS tagging is one of the fundamental steps in most NLP applications such as dependency parsing, sentiment classification, machine translation and text summarisation. For example, adjectives are known to carry useful information related to the sentiment of a user who has written a review about a product. Consequently, using adjectives as features for training a classifier to predict sentiment has been an effective strategy. In the cross-domain POS setting, we would like to train a POS tagger using data from a source domain and apply the trained POS tagger on a different target domain. For example, we could train a POS tagger using manually annotated Wall Street Journal articles and adapt the learnt POS tagger to tag POS in social media such as tweets. In the UDA of POS taggers we do not assume any POS labelled training data for the target domain.

As we later show in our analysis, the POS distribution of words is highly uneven. Some POS categories such as nouns and adjectives are highly frequent, whereas adverbs are much less frequent. Therefore, when we adapt a POS tagger to a new domain we must take into account the imbalance of training data for the different POS categories. Several heuristic methods have been proposed in prior work on cross-domain POS tagging for selecting pivots. However, to the best of our knowledge, prior work on cross-domain POS tagging has largely ignored this data imbalance issue and have focused purely on the adaptation task. In this section, we study the effect of data imbalance on UDA applied in cross-domain POS tagging. UDA methods first select a subset of features that are common to both source and target domains, which are referred to as *pivots*. Next, a projection is learnt from the source and target domains to the space spanned by the pivots. The source domain's labelled training data can then be used to learn a POS tagger in this shared pivot space. By using common features as pivots we can reduce the dissimilarity between the two domains, thereby improving the accuracy of POS tagging in the target domain.

Our contributions in this section can be summarised as follows:

- We compare the effect of previously proposed pivot selection strategies for selecting pivots for UDA of POS tagging under data imbalance. Specifically, we compare FREQ, MI, PMI and PPMI as heuristics for selecting pivots. These heuristics can be computed either using labelled data or unlabelled data giving rise to two flavours.

- We propose a pivot selection method using the F-score for UDA of POS tagging, aimed at the problem of high imbalance ratio in POS categories. This method prefers categories with lower performance, measured using F-score, when selecting pivots, thereby selecting more pivots to cover low performing categories. We use only labelled data from the source domain training instances when measuring F-scores. In the experiments, we observe that the proposed F-score-based pivot selection method indeed improves the POS tagging accuracy of low-performing categories, thereby improving the overall performance.

4.2.1 Pivot Selection for Unsupervised Cross-domain Part-of-Speech Tagging

The POS tag of a word depends on the POS tags of the preceding words; sequence labellers such as hidden markov models (HMMs) and conditional random fields (CRFs) have been successfully used for learning accurate POS taggers [96]. However, by encoding structural features, it is possible to obtain comparable performance using sequence labellers as well as classifiers on POS tagging [80]. Therefore, in this work we model POS tagging as a multi-class classification problem where for a given word, we must select its correct POS tag from a pre-defined finite set of POS categories. This modelling assumption enables us to straightforwardly extend previously proposed pivot selection methods for cross-domain sentiment classification. However, sentiment classification is often modelled as a binary classification task (positive vs. negative sentiment) whereas POS tagging is a multi-class classification task. For example, the SANCL2012 shared task [132] contains 48 categories.

To extend the pivot selection methods proposed for binary classification tasks (i.e. sentiment classification) to multi-class classification tasks (i.e. POS tagging) we collate all training data for the categories to a single category, except for the POS category of interest. This is similar to building a one vs. rest binary classifier for each POS category. Specifically, the score function $\phi(x, \mathcal{D})$ for a feature x in a set of training instances $\mathcal{D}$ is computed by heuristic pivot selection methods such as: FREQ, MI, PMI and PPMI. The frequency of a feature x in a set of training instances $\mathcal{D}$ is denoted by FREQ($x, \mathcal{D}$). The mutual information between a feature x and a set of instances $\mathcal{D}$ is given by

$$\mathrm{MI}(x, \mathcal{D}) = p(x, \mathcal{D}) \log \left(\frac{p(x, \mathcal{D})}{p(x)p(\mathcal{D})} \right). \tag{4.13}$$

We use "$*$" to denote the sum over the set of features or sets of instances for all the domains, and compute the probabilities in (4.13) using the frequency counts as

$$\begin{aligned} p(x,\mathcal{D}) &= \text{FREQ}(x,\mathcal{D})/\text{FREQ}(*,*),\\ p(x) &= \text{FREQ}(x,*)/\text{FREQ}(*,*),\\ p(\mathcal{D}) &= \text{FREQ}(*,\mathcal{D})/\text{FREQ}(*,*). \end{aligned}$$

Similarly, we compute PMI and PPMI by

$$\text{PMI}(x,\mathcal{D}) = \log\left(\frac{p(x,\mathcal{D})}{p(x)p(\mathcal{D})}\right), \tag{4.14}$$

$$\text{PPMI}(x,\mathcal{D}) = \max\big(\text{PMI}(x,\mathcal{D}),0\big). \tag{4.15}$$

4.2.1.1 Pivot Selection for Unlabelled Data

Unlabelled pivot selection methods use unlabelled data from the source domain and target domain (we use notations $\mathcal{D}_{S_U}$ and $\mathcal{D}_{T_U}$ to denote unlabelled data in the source and the target domains respectively).

For example, FREQ$_U$ can be computed using (4.16) for selecting top-ranked features by occurrence in both domains to be pivots. However, for labelled datasets, pivot selection methods are based on the number of classes, hence the selection process is under multi-class settings as

$$x_U = \min\big(\phi(x,\mathcal{D}_{S_U}),\phi(x,\mathcal{D}_{T_U})\big). \tag{4.16}$$

4.2.1.2 Pivot Selection for Labelled Data

As described above, we follow the idea of one vs. rest binary classification to select pivots based on each known tag for labelled datasets in the source domain. For each POS tag P in m POS tags, we split the labelled datasets into $\mathcal{D}_{P_+}$ ($\boldsymbol{x}$ is labelled as P) and $\mathcal{D}_{P_-}$ ($\boldsymbol{x}$ is NOT labelled as P), then compute the score $\phi(\mathcal{D}_P)$ for this POS tag as

$$\phi(x,\mathcal{D}_P) = \big|\phi(x,\mathcal{D}_{P_+}) - \phi(x,\mathcal{D}_{P_-})\big|. \tag{4.17}$$

The score for each feature x is then computed by the sum of scores from all POS categories as

$$x_L = \sum_{i=1}^{m}\phi(x,\mathcal{D}_{P_i}). \tag{4.18}$$

Under these scoring methods, features with higher score are more likely to be pivots because they occur frequently or they are more associated with labels.

4.2.1.3 Effect of the Label Distribution

In the training datasets (Figure 4.4), there are very popular POS categories (e.g., nouns (NN)) and less popular ones (e.g., symbols (SYM)). However, none of the above-mentioned pivot selection methods take into consideration this imbalance in data when computing the score when selecting a feature as a pivot. A straightforward method to incorporate the distributional information to the pivot selection process is to multiply the score $\phi(x, \mathcal{D}_{P_i})$ of a feature x as a pivot for representing the i-th POS category by the probability q_i of that category, thus

$$q(x) = \sum_{i=1}^{m} q_i \phi(x, \mathcal{D}_{P_i}). \tag{4.19}$$

The pivot selection score $q(x)$ of a feature x given by (4.19) prefers frequent POS categories when selecting pivots.

4.2.1.4 Effect of the F-Score

The pivot selection method described in Section 4.2.1.3 is agnostic to the individual performance on a particular POS category. As we later observe in our experiments (Figure 4.5), the frequency of a POS category is not correlating with the performance obtained for that category by a POS tagger. In other words, some low-frequent as well as high-frequent POS categories appear to be equally difficult for adapting a POS tagger to. Therefore, we need a pivot selection method that is aware of the performance on POS categories.

For this purpose, we propose a novel pivot selection method that uses F-score. In our experiments, we use F-score to reduce the effect of the label imbalance [169]. We first train a POS tagger separately for each POS category P_i using a randomly selected sample from the labelled data from the source domain. Next, we evaluate its performance on a randomly selected (different) sample from the source domain. We compute the F-score for this POS tagger on the i-th POS category. Note that we *do not* use any labelled test data from the target domain for this purpose because in UDA we do not have any labelled data for the target domain. Let us denote the F-score for the i-th POS category to be F_i.

We would like to select pivots from POS categories that have low F_i values to encourage adaptation to those categories. We can consider the reciprocal of the F-scores, $1/F_i$ for this purpose. Unfortunately, $1/F_i$ is not a $[0, 1]$ bounded score such as a probability. Therefore, we compute such a bounded score r_i using the softmax function as

$$r_i = \frac{\exp(1/F_i)}{\sum_{j=1}^{N} \exp(1/F_j)}. \tag{4.20}$$

Here, N is the total number of POS categories. Note that for pivot selection purposes it is sufficient to determine the relative ordering of the features according to their scores $r(x)$. Because (4.20) is monotonically increasing w.r.t. to the reciprocal of the F scores, we can simply use the reciprocal of the F score as

$$r(x) = \sum_{i=1}^{m} r_i \phi(x, \mathcal{D}_{P_i}). \tag{4.21}$$

4.2.1.5 Nouns

Nouns are one of the most popular POS categories. In fact, in our datasets nouns are the majority POS category. As a baseline for selecting pivots from the majority category, we propose a score function for pivot selection that prefers features that occur frequently in the noun category. This baseline demonstrates the performance of a pivot selection method that considers only one POS category such as nouns (NN). This score function x_{NN} is defined as the score from only category NN given by

$$x_{\text{NN}} = \left|\phi(x, \mathcal{D}_{\text{NN}+}) - \phi(x, \mathcal{D}_{\text{NN}-})\right|. \tag{4.22}$$

4.2.2 Experiments

To evaluate the different pivot selection methods described in Section 4.2.1, we use the selected pivots with SCL to perform cross-domain POS tagging.

4.2.2.1 Experimental Data

Following Blitzer et al. [15], we use the Penn Treebank [114] of the Wall Street Journal (WSJ) section 2-21 as the labelled data, and 100,000 WSJ sentences from 1988 as unlabelled data in the source domain. Following Schnabel and Schütze [154], we evaluate on 5 different target domains (newsgroups, weblogs, reviews, answers and emails) from SANCL2012 shared task [132]. The Penn treebank tag annotated Wall Street Journal (**wsj**) is considered as the source domain in all

	Source	Target				
Domains	wsj	newsgroups	weblogs	reviews	answers	emails
#sentences	30,060	1,195	1,016	1,906	1,744	2,450
#tokens	731,678	20,651	24,025	28,086	28,823	29,131
#types	35,933	4,924	4,747	4,797	4,370	5,478
OOV	0.0%	23.1%	19.6%	29.5%	27.7%	30.7%

Table 4.5: Number of sentences, tokens and types in the source and target labelled data. OOV (Out-Of-Vocabulary) is the percentage of types that have not been observed in the source domain (**wsj**) [132].

	Unlabelled				
Domains	newsgroups	weblogs	reviews	answers	emails
#sentences	1,000,000	524,834	1,965,350	27,274	1,194,173
#tokens	18,424,657	10,356,284	29,289,169	424,299	17,047,731
#types	357,090	166,515	287,575	33,425	221,576

Table 4.6: Number of sentences, tokens and types in the target unlabelled data after sentence splitting and tokenisation [132].

experiments. Table 4.5 and Table 4.6 are the statics of the experimental data. All the datasets have been tokenised during pre-processing. Tokens with the occurrence < 5 are removed.

4.2.2.2 Training

To train a POS tagger, we model this task as a multi-class classification problem. We represent each training instance (a POS labelled word in a sentence) by a feature vector. For this purpose, we use two types of features: (a) contextual words and (b) embeddings.

To train a sequential POS tagger, Giménez and Marquez [59] proposed to use a window of features. Following Schanbel and Schütze [153], we imply a window of $2l + 1$ for tagging token x to take the contextual words into account, given by

$$\boldsymbol{x} = (x_{-l}, x_{-l+1}, \ldots, x_0, \ldots, x_{l-1}, x_l). \tag{4.23}$$

In SCL, original features are projected by the binary classifiers θ learnt from pivots and non-pivots (i.e. pivot predictors) after applied singular value decomposition (SVD). These projected

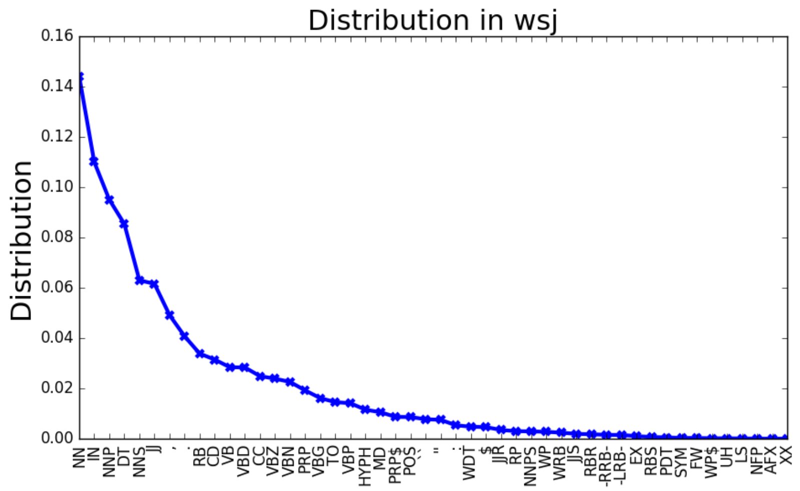


Figure 4.4: Distribution of the 48 PennTreebank POS tags in training data (**wsj**).

features $\theta\boldsymbol{x}$ are influenced by the different sets of pivots selected by the different pivot selection methods. We follow Sapkota et al. [151] to train the final adaptive classifier f only by projected features to reduce the dimensionality, where $\theta\boldsymbol{x} \in \mathbb{R}^h$.

We use $d = 300$ dimensional GloVe [131] embeddings (trained using 42B tokens from the Common Crawl) as word representations. By applying the window, each word $\boldsymbol{w}$ is defined by

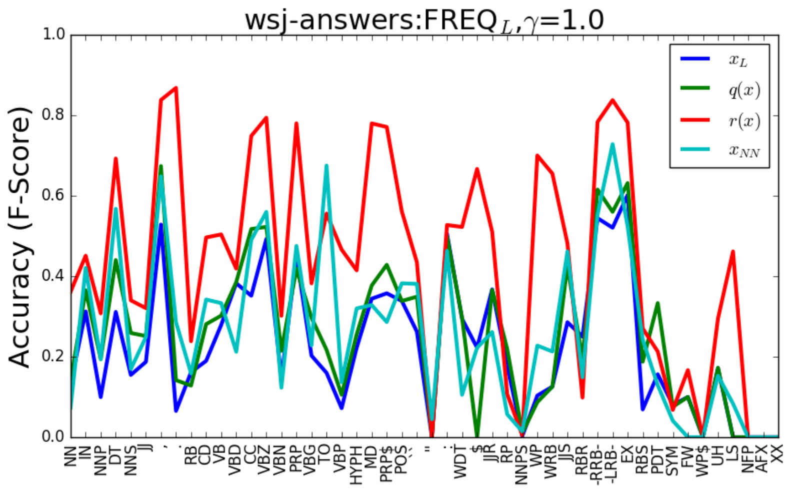
$$\boldsymbol{w} = \boldsymbol{w}_{-l} \oplus \boldsymbol{w}_{-l+1} \oplus \ldots \oplus \boldsymbol{w}_0 \oplus \ldots \oplus \boldsymbol{w}_{l-1} \oplus \boldsymbol{w}_l, \tag{4.24}$$

where $\oplus$ is vector concatenation and $\boldsymbol{w} \in \mathbb{R}^d$.

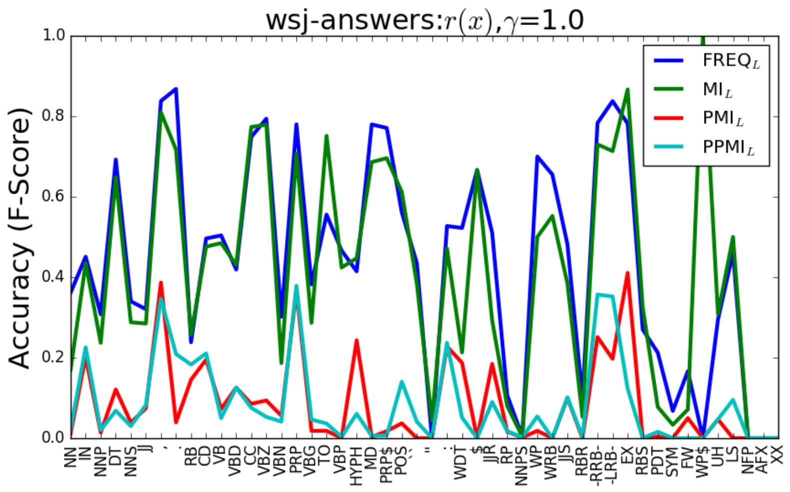
We combine two types of features by introducing a mixing parameter γ. We train an logistic regression classifier f with an ℓ_2 regulariser on $[\gamma\theta\boldsymbol{x}, \boldsymbol{w}]$.

4.2.2.3 Classification Accuracy

Accuracy (the percentage of correct predictions) is not a suitable measurement for datasets with large numbers of labels, as it cannot show the effect on imbalanced data from the various labels. Therefore, we use the macro-average F-score to measure the classification accuracy for each POS tag when a particular pivot selection strategy is applied to SCL. Here, the F-scores are computed



(a) Different labelled data strategies using FREQ$_L$.



(b) Different pivot selection methods using $r(x)$.

Figure 4.5: F-score for the 48 PennTreebank POS tags (left to right: high to low distribution in training data, as shown in Figure 4.4) for adapting from **wsj** to **answers** under mixing parameter $\gamma = 1.0$.

using the target domain's test labelled instances as

$$\text{Precision}(P_i) = \frac{\text{no. of correctly predicted words as category } P_i}{\text{total no. of test words in the target domain}}, \tag{4.25}$$

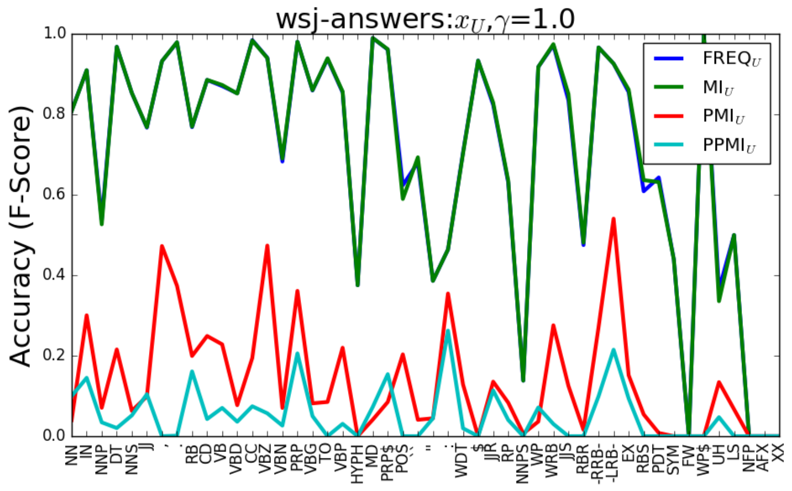


Figure 4.6: F-score for different pivot selection methods using unlabelled datasets.

$$\text{Recall}(P_i) = \frac{\text{no. of correctly predicted words as category } P_i}{\text{total no. of test words belonging to category } P_i}, \tag{4.26}$$

$$\text{F-score}(P_i) = \frac{2 \times \text{Precision}(P_i) \times \text{Recall}(P_i)}{\text{Precision}(P_i) + \text{Recall}(P_i)}. \tag{4.27}$$

4.2.3 Results

In the Figure 4.5a, we show the F-scores for the different POS tags obtained by adapting a POS tagger from **wsj** source domain to the **answers** target domain. Here, we select pivots using the FREQ$_L$ method. x_L denotes the level of performance we obtain if we had simply used the pivots selected by FREQ$_L$ without adjusting for the imbalance of data. $q(x)$, $r(x)$ and x_{NN} correspond to the pivot selection methods described respectively in Sections 4.2.1.3, 4.2.1.4 and 4.2.1.5. The POS tags are arranged in the horizontal axis in the descending order of their frequency in the source domain. The mixing parameter γ is fixed to 1 in this experiment and we later study its effect on the performance.

Figure 4.5a shows that $r(x)$ is the best multi-label strategy for FREQ$_L$. Similar results were obtain when $r(x)$ was combined with other pivot selection methods (MI, PMI and PPMI), and on other target domains. We show the results for the **wsj-answers** adaptation setting. We observe that probability of a POS tag ($q(x)$), or selecting pivots from the majority category (x_{NN}), performs at a similar level to not performing any adjustments due to data imbalance (x_L).

Next, we study the effect of the proposed F-score-based pivot selection method, $r(x)$, with different labelled pivot selection methods. Figure 4.5b shows that F-score by FREQ is consistently better than others for all labelled methods. Figure 4.6 shows that FREQ is also one of the good pivot selection methods for unlabelled datasets, MI$_U$ is closely following FREQ$_U$. These two results agree with the observation made by [16] that FREQ works better for POS tagging as a pivot selection strategy. Overall, PMI or PPMI with any multi-class pivot selection strategy proposed in this section do not work well on datasets with large numbers of categories. A possible reason is that PMI and PPMI do not weight the amount of information obtained about one random event by observing another by the joint probability of the two events [23].

4.2.3.1 Effect on Mixing Parameter

Method	x_L	$q(x)$	$r(x)$	x_{NN}	x_L	$q(x)$	$r(x)$	x_{NN}
γ	FREQ$_L$				MI$_L$			
0.01	**0.6993**	**0.6982**	**0.6985**	**0.6992**	**0.6986**	**0.6993**	**0.6993**	**0.7006**
0.1	0.6927	0.6910	0.6975	0.6877	0.6857	0.6890	0.6930	0.6977
1	0.2246	0.2604	0.4370	0.2649	0.2407	0.2461	0.3533	0.3689
10	0.4366	0.4328	0.4824	0.4290	0.4317	0.4314	0.4589	0.4725
100	0.6890	0.6909	0.6957	0.6959	0.6900	0.6892	0.6860	0.6931
	PMI$_L$				PPMI$_L$			
0.01	**0.7001**	**0.7025**	**0.6966**	**0.7034**	**0.6996**	**0.6977**	**0.6939**	**0.7002**
0.1	0.5270	0.6775	0.5005	0.5113	0.6992	0.4118	0.5133	0.4666
1	0.1254	0.1492	0.0846	0.1151	0.6955	0.0907	0.0836	0.0956
10	0.3296	0.4359	0.3225	0.3423	0.6811	0.3085	0.3198	0.3236
100	0.6732	0.6906	0.6779	0.6636	0.6609	0.6621	0.6611	0.6839

Table 4.7: F-score for pivot selection strategies with mixing parameter $\gamma = \{0.01, 0.1, 1, 10, 100\}$. Highest F-score for each strategy is bolded. x_L, $q(x)$, $r(x)$ and x_{NN} denote data imbalance strategies by (4.18), (4.19), (4.21) and (4.22) respectively.

In Section 4.2.2.2, we defined a mixing parameter γ for the combination of two types of features. Table 4.7 shows that all labelled pivot selection methods share the same trend for $\gamma = \{0.01, 0.1, 1, 10, 100\}$. The highest F-score is obtained with 0.01. These F-scores are closer to each other for different pivot selection methods when γ towards zero because we reduce the weight of pivot predictors from SCL and pretrained word embeddings are not influenced by the pivot selection method. All unlabelled pivot selection methods also follow this trend. The differences between F-scores reported by the different pivot selection methods with the optimal value of γ for that method are not statistically significant, which indicates that pretrained word embeddings can be used to overcome any disfluencies introduced by the pivot selection methods if the mixing parameter is carefully selected. We differ the study of learning the best combinations of pretrained word embedding-based features and pivot predictors to future work.

4.3 Summary

In this chapter, we proposed TSP, a task-specific pivot selection method that simultaneously requires pivots to be both similar in the two domains as well as task-specific. TSP jointly optimises the two criteria by solving a single quadratic programming problem. The pivots selected by TSP improve the classification accuracy in multiple cross-domain sentiment classification tasks, consistently outperforming previously proposed pivot selection methods. Moreover, comparisons against SentiWordNet reveal that indeed the top-ranked pivots selected by TSP are task-specific. We conducted a series of experiments to study the behaviour of the proposed method with various parameters and the design choices involved such as the mixing parameter, number of pivots used, UDA method where the selected pivots are used, and the word embeddings used for representing features when computing the domain centroids. Our experimental results show that TSP can find pivots for various pairs of domains and improve the performance of both SCL and SFA when compared to the performance obtained by using pivots selected by prior heuristics. Moreover, our analysis shows that it is important to use both unlabelled data (available for both source and target domains) as well as labelled data (available only for the source domain) when selecting pivots.

Furthermore, We compare the effect of previously proposed pivot selection strategies for UDA of POS tagging under data imbalance. We propose a combination of pivot selection method and labelled data strategy ($\text{FREQ}_L + r(x)$) that works better than other combinations in the experiments. We also show that the classification accuracy on a single category does not improve using a single category strategy (e.g. x_{NN}).

Chapter 5

Core-Periphery Decomposition based UDA

Short-texts are abundant on the Web and appear in various different formats such as micro-blogs [98], Question and Answer (QA) forums, review sites, Short Message Service (SMS), email, and chat messages [37, 161]. Unlike lengthy responses that take time to both compose and read, short responses have gained popularity particularly in social media contexts. Considering the steady growth of mobile devices that are physically restricted to compact keyboards, which are suboptimal for entering lengthy text inputs, it is safe to predict that the amount of short-texts will continue to grow in the future. Considering the importance and the quantity of the short-texts in various web-related tasks, such as text classification [50, 97], and event prediction [149], it is important to be able to accurately represent and classify short-texts.

Compared to performing text mining on longer texts [68, 159, 182], for which dense and diverse feature representations can be created relatively easily, handling of shorter texts poses several challenges. The number of features that are present in a given short-text will be a small fraction of the set of all features that exist in all of the train instances. Moreover, frequency of a feature in a short-text will be small, which makes it difficult to reliably estimate the salience of a feature using term frequency-based methods. This is known as the *feature sparseness* problem in text classification.

Feature sparseness is not unique to short-text classification but also encountered in cross-domain text classification [15, 16, 21], where the training and test data are selected from different domains with small intersection of feature spaces. In the DA setting, a classifier trained on one domain (*source*) might be agnostic to the features that are unique to a different domain (*target*), which results in a *feature mismatch* problem similar to the feature-sparseness problem discussed above.

Projection-based UDA methods use pivots to project the original feature space to a shared common feature space. We have studied pivot selection to improve them in Chapter 3 and Chapter 4. Considering the effect of data imbalance in UDA, we proposed pivot selection strategies to solve the imbalance problem in Section 4.2.

To address the feature sparseness problem encountered in short-text and cross-domain classification tasks, we propose a novel method that computes related features that can be appended to the feature vectors to reduce the sparsity. Specifically, we decompose a feature-relatedness graph into core-periphery (CP) structures, where a core feature (a vertex) is linked to a set of peripheries (also represented by vertices), indicating the connectivity of the graph. This graph decomposition problem is commonly known as the CP-decomposition [38, 89, 90, 140].

The proposed CP-decomposition algorithm significantly extends existing CP-decomposition methods in three important ways.

- First, existing CP-decomposition methods consider unweighted graphs, whereas edges in feature-relatedness graphs are weighted (possibly nonnegative) real-valued feature-relatedness scores such as PPMI. The proposed CP-decomposition method can operate on edge-weighted graphs.
- Second, considering the fact that in text classification a particular periphery can be related to more than one core, we relax the hard assignment constraints on peripheries and allow a particular periphery attach to multiple cores.
- Third, prior work on feature-based cross-domain sentiment classification methods have used features that are frequent in training (source) and test (target) data as expansion candidates to overcome the feature mismatch problem. Inspired by this, we define *coreness* of a feature as the pointwise mutual information between a feature and the source/target domains. The CP-decomposition algorithm we propose will then compute the set of cores considering both structural properties of the graph as well as the coreness values computed from the train/test data.

To perform feature vector expansion, we first construct a feature-relatedness graph, where vertices correspond to features and the weight of the undirected edge connecting two features represent the relatedness between those two features. Different features and relatedness measures can be flexibly used in the proposed graph construction. In our experiments, we use the simple (yet popular and effective) setting of n-gram features as vertices and compute their relatedness using PPMI. We compute the coreness of features as the sum of the two PPMI values between the feature and the source, and the feature and the target domains.[1] Next, CP-decomposition is performed on this feature-relatedness graph to obtain a set of core-periphery structures. We then rank the set of peripheries of a particular core by their PPMI values, and select the top-ranked peripheries as the expansion features of the core. We expand the core features in training and train a logistic regression-based binary classifier using the expanded feature vectors, and evaluate its performance on the expanded test feature vectors.

We evaluate the effectiveness of the proposed method using benchmark datasets for two different tasks: short-text classification and cross-domain sentiment classification. Experimental results on short-text classification show that the proposed method consistently outperforms previously proposed feature expansion-based methods for short-text classification and even some of the sentence embedding learning-based methods. Moreover, the consideration of coreness during the CP-decomposition improves the text classification accuracy. In cross-domain sentiment classification experiments, the proposed method outperforms previously proposed feature-based methods such as SCL [15].

5.1 Related Work

A complementary approach to overcome feature-sparseness is to learn a (potentially lower dimensional) dense feature representation for the training and test instances that suffer from feature sparseness, and train and evaluate classifiers in this dense feature space instead of the original sparse feature space. Skip-thought vectors [87] encodes a sentence into a lower-dimensional dense vector using bidirectional long short-term memory (bi-LSTM), whereas FastSent [73] learns sentence embeddings by predicting the words in the adjacent sentences in a corpus, ignoring the word ordering. Paragraph2Vec [99] jointly learns sentence and word embeddings that can mutually predict each other in a short-text such as a paragraph in a document. Sequential Denoising Autoencoder (SDAE) [73] transforms an input sentence into an embedding by a look-up table consisting of pre-trained word embeddings and attempts to reconstruct the original sentence embedding from a masked version. Sentence embedding learning methods such as skip-thought vectors, FastSent, SDAE etc. require a large amount of unlabelled texts for training such as 80 million sentence Toronto books corpus, which might not be available for specialised domains. As shown in our

[1]In short-text classification experiments, coreness is computed using unlabelled training and test instances.

experiments, the proposed methods perform competitively with these embedding-based methods, while not requiring any additional training data, other than the small (typically less than 50,000 sentences) benchmark training datasets.

In the CP-decomposition problem, one seeks a partition of vertices into two groups called a core and a periphery. The core vertices are densely interconnected and the peripheral vertices are sparsely interconnected. The core and peripheral vertices may be densely interconnected or sparsely interconnected. Various algorithms have been developed to find a single core-periphery structure [38, 140] or multiple core-periphery structures [89, 90] in a graph. Many existing algorithms assume that each vertex belongs to only one core-periphery structure. This assumption is problematic for text classification because a peripheral vertex can belong to multiple core-periphery structures. To circumvent this problem, here we present a novel algorithm for the CP-decomposition that allows a peripheral vertex to belong to more than one core-periphery structures. Some existing CP-decomposition algorithms allow peripheral vertices to belong to multiple core-periphery structures [152, 176, 177]. These algorithms detect non-overlapping communities (i.e., groups of densely interconnected vertices) in a graph. Then, they regard vertices that do not belong to any community as peripheral vertices. Therefore, the detected peripheries might not be strongly related to the associated cores because they are not densely interconnected with the cores in general. Another CP-decomposition algorithm allows communities to overlap and regard the vertices belonging to many communities as a core [178]. Then, the detected peripheral vertices may be densely interconnected because they belong to the same community. In contrast to these algorithms, the present algorithm seeks peripheries that are densely interconnected with the associated cores while sparsely interconnected with other peripheral vertices.

5.2 CP-decomposition-based Feature Expansion

Our proposed method consists of three steps: (a) building a feature-relatedness graph (§5.2.1), (b) performing CP-decomposition on the feature-relatedness graph (§5.2.2 and §5.2.3) and (c) using the core-peripheries from the decomposition to perform feature expansion (§5.2.4). Next, we describe each of those steps in detail.

5.2.1 Feature-Relatedness Graph

Given a set of texts, we build a feature-relatedness graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V}$ is the set of vertices corresponding to the features, $\mathcal{E}$ is the set of undirected edges between two vertices in $\mathcal{G}$ and the weight of the edge $e_{ij} \in \mathcal{E}$ connecting two features i and j is given by the W_{ij} element of the weight matrix $\mathbf{W}$. Let us denote the number of vertices and edges respectively by N and M (i.e. $|\mathcal{V}| = N$ and $|\mathcal{E}| = M$). Different types of features such as n-grams, part-of-speech sequences, named entities, dependency relations etc. can be used as vertices in the feature-relatedness graph. Moreover, different relatedness measures such as co-occurrence frequency, pointwise mutual information, χ^2, log-likelihood ratio etc. can be used to compute the weights assigned to the edges. For simplicity, in this chapter, we represent each text-document using the set of unigrams extracted from that document, and use PPMI to compute a nonnegative $\mathbf{W}$. we connect two words if PPMI values between them are greater than zero. This formulation is used for both short-text classification and cross-domain sentiment classification experiments conducted in the chapter.

5.2.2 Core-Periphery Decomposition

Given a feature-relatedness graph $\mathcal{G}$ created using the process described in Section 5.2.1, we propose a method that decomposes $\mathcal{G}$ into a set of overlapping core-periphery structures. A core-periphery structure assumed in this chapter consists of one core vertex and an arbitrarily number of peripheral vertices that are adjacent (i.e., directly connected) to the core vertex.[2] Therefore, a core-periphery structure forms a star graph. We further assume that a core belongs only to one core-periphery structure, but a periphery can belong to multiple core-periphery structures.

Let $\mathcal{C} \subseteq \mathcal{V}$ be the set of cores and P_i be the set of peripheries associated with the core $i (\in \mathcal{C})$. We regard that a core-periphery structure is a good pair if the core is adjacent to its peripheries with large edge weights. One goodness measure is the sum of edge weights between the core i and peripheries, which is given by $\sum_{j \in \mathcal{P}_i} W_{ij}$. This quantity should be larger than the value expected from a null model (i.e., randomised graph) for the detected core-periphery structure to be meaningful. We seek $\mathcal{C}$ and $\mathcal{P}_i$ ($\forall i \in C$) by maximising

$$Q = \sum_{i \in \mathcal{C}} \sum_{j \in \mathcal{P}_i} W_{ij} - \sum_{i \in \mathcal{C}} \sum_{j \in \mathcal{P}_i} \mathbb{E}[W_{ij}], \tag{5.1}$$

where $\mathbb{E}[W_{ij}]$ is the expected edge weight between vertices i and j in the null model. The first term on the right-hand side of (5.1) is the total weights of the edges between the cores and peripheries. The second term is the expected value of the first term according to the null model. Therefore, a large positive Q value indicates that cores and peripheries are connected with large edge weights.

[2]In the remainder of the chapter, we refer to core vertices as cores and peripheral vertices as peripheries to simplify the terminology.

To compute $\mathbb{E}[W_{ij}]$, we must specify a null model. We consider a simple null model where any pair of vertices is adjacent by an edge with an equal expected weight [53]. Then, we can rewrite (5.1) as

$$Q = \sum_{i \in \mathcal{C}} \sum_{j \in \mathcal{P}_i} (W_{ij} - p), \tag{5.2}$$

where p is the average edge weight of the original graph given by

$$p = \frac{2}{N(N-1)} \sum_{i,j \in \mathcal{V}, i \neq j} W_{ij}. \tag{5.3}$$

We maximise Q as follows. Given a set of cores $\mathcal{C}$, it is easy to find peripheries that maximise Q. Suppose a core i and a vertex $j \notin \mathcal{P}_i$, which may belong to one or more different core-periphery structures. Adding the vertex j to $\mathcal{P}_i$ increases Q, if $W_{ij} - p$ is positive. Therefore, $\mathcal{P}_i$ associated with core i must be the neighbours of vertex i with an edge weight of $W_{ij} > p$. Therefore, we have

$$\max_{\mathcal{C}} \max_{\mathcal{P}_i, i \in \mathcal{C}} Q = \max_{\mathcal{C}} \sum_{i \in \mathcal{C}} \sum_{j \in \mathcal{V} \setminus \mathcal{C}} \tilde{W}_{ij}, \tag{5.4}$$

where

$$\tilde{W}_{ij} = \max(W_{ij} - p, 0). \tag{5.5}$$

(5.4) indicates that the maximisation of Q is equivalent to partitioning of the set of vertices $\mathcal{V}$ into $\mathcal{C}$ and $\mathcal{V} \setminus \mathcal{C}$ such that the sum of edge weights given by (5.5) between $\mathcal{C}$ and $\mathcal{V} \setminus \mathcal{C}$ is maximised. This is known as the max-cut problem [62]. However, solving the max-cut problem is NP-hard [79]. Therefore, we use the Kernighan-Lin's algorithm [82] to find a good (but generally a suboptimal) solution.

It should be noted that Q is conserved when we regard $\mathcal{V} \setminus \mathcal{C}$ as the cores and $\mathcal{C}$ as peripheries. This is because Q is the sum of edge weights between $\mathcal{C}$ and $\mathcal{V} \setminus \mathcal{C}$. For example, suppose a graph with a single core-periphery structure as shown in Figure 5.1a. By regarding the core as a periphery and vice versa, we have another assignment of the core-periphery structure achieving the same Q value as shown in Figure 5.1b. Although Q is the same in the two assignments, we would like to prioritise the core-periphery structure shown in Figure 5.1a, because we would like to have a smaller set of cores than peripheries. Therefore, we regard $\mathcal{C}$ as the set of cores if $|\mathcal{C}| < |\mathcal{V} \setminus \mathcal{C}|$; otherwise we regard $\mathcal{C}$ as the set of peripheries.

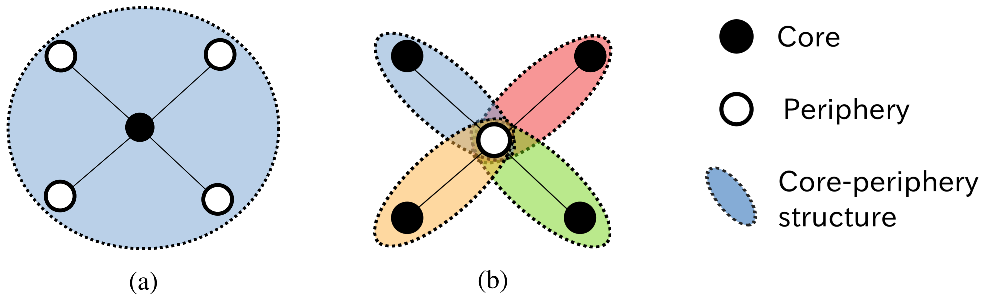


Figure 5.1: Core-periphery structures with an equal quality, Q. Each filled and empty circles indicate core and peripheral vertices, respectively. Each shared region indicates a core-periphery structure.

5.2.3 Semi-supervised Core-Periphery Decomposition

The objective given by (5.4) depends only on $\mathcal{G}$ and does not consider any prior linguistic knowledge that we might have about which features are appropriate as cores. For example, for cross-domain sentiment classification, it has been shown that features that express similar sentiment in both source and target domains are suitable as pivots [16]. To incorporate this information, we integrate the *coreness* of words into the objective as

$$\max_{\mathcal{C}} \max_{\mathcal{P}_i, i \in \mathcal{C}} Q = \max_{\mathcal{C}} \sum_{i \in \mathcal{C}} \sum_{j \in \mathcal{V} \setminus \mathcal{C}} \tilde{W}_{ij} + \lambda \sum_{i \in \mathcal{C}} \text{coreness}(i). \tag{5.6}$$

In (5.6), coreness(i) is a nonnegative value that indicates the appropriateness of i as a core. Hyper-parameter λ adjusts the importance we would like to give to coreness as opposed to determining cores based on the graph structure. We tune λ using a held out portion of the training data in our experiments. Different measures can be used to pre-computed the coreness values from the train/test data such as FREQ, MI, PMI, PPMI etc, which have been proposed in prior work on DA for selecting pivots [15, 16, 22]. In this work, we use PPMI to pre-compute the coreness for a word i as

$$\text{coreness}(i) = \left(\text{PPMI}(i, \mathcal{D}_{\text{train}}) - \text{PPMI}(i, \mathcal{D}_{\text{test}})\right)^2. \tag{5.7}$$

Here, $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ are respectively the set of training and test data (or in the case of DA selected from the source and the target domains).

Dataset	SCL				No Expansion	CP-decomposition		
	FREQ	MI	PMI	PPMI		Non-overlapping	Overlapping w/o coreness	Overlapping w/ coreness
TR	67.60	66.12	67.44	63.21	78.86	80.34	80.56	**80.86**
CR	77.85	74.83	78.52	75.50	80.87	83.89	83.89	**84.40**
SUBJ	87.65	82.15	85.65	82.75	88.05	89.75	90.15	**90.48**
MR	64.68	58.07	64.26	59.10	73.55	75.23	74.95	**75.66**
AVG	74.45	70.29	73.97	70.14	80.33	82.30	82.39	**82.85**

Table 5.1: Results for the short-text classification task. For each dataset, the best results are shown in bold.

5.2.4 Feature Expansion

To overcome feature-sparseness in training and test instances, we expand features that are cores by their corresponding peripheral sets. Specifically, for each core $i \in \mathcal{C}$, we sort its peripheries $\mathcal{P}_i$ by their coreness values and select the top-k ranked peripheries as the expansion features for a core i if it appears in a document. The values of these expansion features are set to their PPMI values with the corresponding core after ℓ_1 normalising over the set of expansion features in each instance. The effect of k on performance is experimentally studied later.

5.3 Experiments

We evaluate the proposed method on two tasks: short-text classification (a non-DA task) and cross-domain sentiment classification (a DA task). For short-text classification we use the Stanford sentiment treebank (TR)[3], customer reviews dataset (CR) [75], subjective dataset (SUBJ) [128] and movie reviews (MR) [129]. For DA we use Amazon multi-domain sentiment dataset [16] containing product reviews from four categories: Books (B), DVDs (D), Electronics (E) and Kitchen Appliances (K). Each category is regarded as a domain and has 1000 positive and 1000 negative reviews, and a large number of unlabelled reviews.[4] We train a classifier on 12 domain pairs adapting from source to target (S-T): B-D, B-E, B-K, D-B, D-E, D-K, E-B, E-D, E-K, K-B, K-D, K-E. For the short-text classification datasets, we use the official train/test split.

We represent each instance (document) using a bag-of-features consisting of unigrams. Stop words are removed using a standard stop words list. We train an ℓ_2 regularised binary logistic regression classifier with each dataset, where the regularisation coefficient is tuned via 5-fold cross

[3]https://nlp.stanford.edu/sentiment/treebank.html

[4]Blitzer et al. [16] considered 4 and 5 star rated reviews as positive and 1 or 2 as negative in sentiment.

S-T	SCL					CP-decomposition		
	FREQ	MI	PMI	PPMI	No Expansion	Non-overlapping	Overlapping w/o coreness	Overlapping w/ coreness
B-D	72.75	65.50	71.50	69.25	75.00	75.75	**76.75**	76.38
B-E	72.75	71.00	**74.50**	66.00	71.00	71.00	69.75	69.75
B-K	77.25	64.00	**80.50**	77.25	78.25	78.25	77.75	78.00
D-B	71.00	53.00	66.25	65.50	74.00	74.25	74.25	**75.25**
D-E	72.00	67.00	72.75	**74.75**	**74.75**	73.75	73.00	**74.75**
D-K	**79.75**	57.50	79.00	76.75	79.25	78.00	79.25	79.25
E-B	62.75	57.25	66.25	60.25	**69.50**	68.50	68.75	68.75
E-D	64.50	62.75	65.50	62.75	73.25	71.75	73.25	**73.50**
E-K	82.00	77.75	81.25	79.50	**84.25**	84.00	82.50	84.00
K-B	65.75	52.50	68.00	68.75	**70.00**	**70.00**	69.75	69.50
K-D	67.25	53.75	66.75	68.50	72.75	72.00	72.75	**73.63**
K-E	77.25	74.50	74.50	74.75	79.00	79.75	79.00	**80.50**
AVG	72.08	63.04	72.23	70.33	75.08	74.75	74.73	**75.27**

Table 5.2: Results for DA tasks. For each S-T pair, the best results are shown in bold. The last row shows the average of performance over the 12 S-T pairs.

Methods	**TR**	**CR**	**SUBJ**	**MR**
No Expansion	76.31	81.54	88.05	73.35
FTS [111]	76.47	62.41	50.15	66.83
SCL [15]	67.60	78.52	87.65	64.68
SFA [126]	60.08	70.13	79.00	59.57
Proposed	**80.86**	**84.40**	**90.48**	**75.66**

Table 5.3: Proposed vs. feature-based methods for short-text classification.

validation.

5.3.1 Classification Accuracy

We use the classification accuracy on the test data (i.e. ratio between the number of correctly classified test instances and the total number of test instances in the dataset) as the performance evaluation measure. As baselines we evaluate the classification accuracy without expanding features (**No Expansion**), expanding the features by a non-overlapping version of the CP-decomposition method where a single periphery will be assigned to only a single core, overlapping CP-decomposition with and without the consideration of coreness (described respectively in Sections 5.2.2 and 5.2.3). We apply SCL with pivots selected from four different criteria (FREQ, MI, PMI and PPMI) for each S-T pair in the DA datasets. Strictly speaking, SCL is a DA method but if we can apply to short-text classification tasks as well if we consider training and test datasets respectively as a source and a target domain and select pivots using some selection criterion. Results on the short-text and DA tasks are summarised respectively in Tables 5.1 and 5.2.

As shown in Table 5.1, all variants of the CP-decomposition outperform the **No Expansion** baseline and the best performance is reported by the overlapping CP-decomposition considering the coreness values. According to binomial test results, there is no statistical significance in Table 5.1. SCL performs poorly on this non-DA task, indicating that it is specifically customised for DA tasks.

Table 5.3 compares the performance of the proposed method (i.e., overlapping version of the CP-decomposition with coreness) against FTS (§2.1.3), a previously proposed feature expansion method and DA methods such as SCL and SFA applied to short-text classification. We observe that the proposed method consistently outperforms FTS, which uses frequently occurring features as expansion candidates. This result implies that frequency of a feature alone does not enable us to find useful features for expanding sparse feature vectors. The suboptimal performance of SFA and SCL for short-text classification indicates that, despite the fact that the feature-mismatch problem in DA has some resemblance to the feature-sparseness problem in short-text classification, applying DA methods to short-text classification is not effective. As shown in Table 5.2, proposed method reports equal or the best performance for 10 out of 12 domain pairs indicating that it is effective not only for short-text classification but also for DA. However, the improvements reported in Table 5.2 are not statistically significant (according to Clopper-Pearson confidence intervals [35] computed at $p < 0.01$), implying that CP-decomposition is less effective on DA datasets, which contain longer (on average 5-10 sentence reviews) texts.

We compare the proposed method against the state-of-the-art embedding-based short-text classification methods in Table 5.4. For skip-thought vectors [87], Paragraph2Vec [99], FastSent [73] and SDAE [73] provided by Hill et al. [73], we show the published results on MR, CR and SUBJ.[5] CNN represents the convolutional neural network-based document-level embedding learning method proposed by Kim [84]. The proposed method reports the best results on CR, whereas skip-thought does so for MR and SUBJ datasets. An interesting future research direction would be to combine feature-expansion method and document-level embedding methods to further improve the accuracy of short-text classification.

An example feature expansion is shown in Table 5.5, where 6 cores are expanded by the overlapping version of the CP-decomposition method without using coreness and one core with the proposed method. Top 5-ranked peripheries are shown for each core, which are used as the expansion features. We observe that many cores are found without constraining the CP-decomposition

[5]These methods have not been evaluated on the TR dataset.

Methods	MR	CR	SUBJ
Skip-thought [87]	**76.5**	80.1	**93.6**
Paragraph2Vec [99]	74.8	78.1	90.5
FastSent [73]	70.8	78.4	88.7
SDAE [73]	74.6	78.0	90.8
CNN [84]	76.1	79.8	89.6
Proposed	75.7	**84.4**	90.5

Table 5.4: Proposed vs. document-level embedding-based methods for short-text classification.

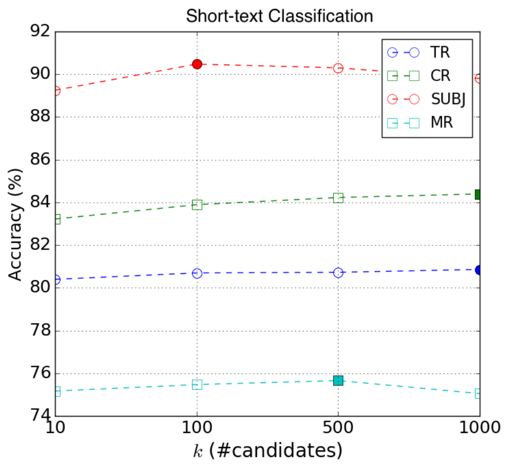
Sentence:	The film makes a strong case for the importance of the musicians in creating the motown sound.						
Methods:	Overlapping w/o coreness						Overlapping w/ coreness
Cores:	film	strong	case	musicians	creating	sound	motown
Peripheries:	tribeca	willed	neko	remixers	irritation	puget	discographer
	remakes	fliers	genitive	trombonists	populating	stereophonic	gordy
	grossing	syllabic	accusative	bandleaders	abolishing	nootka	supremes
	slasher	roderick	dative	saxophonists	duopoly	mcmurdo	stax
	blaxploitation	oxidizing	eeml	clarinetists	soundscapes	blaster	dozier

Table 5.5: An example of cores and top 5 peripheries chosen by overlapping CP-decomposition with/without coreness ($k = 5$). This example sentence in TR is classified incorrectly using the method without coreness (and the **No Expansion** baseline) but correctly after considering coreness.

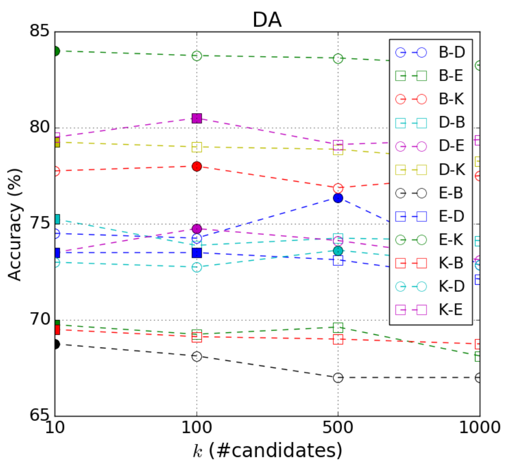
by coreness, introducing noisy expansions resulting in an incorrect prediction. Moreover, although by integrating coreness into the CP-decomposition process we have only a single matching core, *motown*, it is adequate for making the correct prediction. *motown* is a music company, which is expanded by a more general periphery *discographer*, which is a type of music performer, helping the final classification. Consideration of coreness improves the classification accuracy in both short-text classification as well as DA.

In both Tables 5.1 and 5.2, the non-overlapping version performs poorly compared to the overlapping counterpart. With non-overlapping CP-decomposition, peripheries are not allowed to connect to multiple cores. This results in producing a large number of cores each with a small number of peripheries, which does not help to overcome the feature-sparseness because each core will be expanded by a different periphery.

Figure 5.2 shows the effect of the number of expansion candidates k on the performance of the proposed overlapping CP-decomposition with coreness. For short-text classification (Figure 5.2a), the accuracy increases for $k \geq 100$ (TR and CR obtain the best for $k = 1000$). For DA (Figure 5.2b), $k \leq 100$ yields better performance in most of the domain pairs (10 out of 12). For all 12 domain pairs, the accuracy achieved a peak when $k \leq 500$.



(a)



(b)

Figure 5.2: Number of expansion candidates for the proposed method. The marker for the best result for each dataset is filled.

5.4 Summary

We proposed a novel algorithm for decomposing a feature-relatedness graph into core-periphery structures considering coreness of a feature. Our experimental results show that the induced core-periphery structures are useful for reducing the feature sparseness in short-text classification and cross-domain sentiment classification tasks, as indicated by their improved performance.

Chapter 6

Self-training based UDA

Based on the initial input of the training data, the prior work in UDA can be categorised into: feature-based (§2.1) and instance-based approaches (§2.2). In the previous chapters, we studied feature-based UDA methods and proposed novel UDA methods as well as pivot selection strategies. From this chapter on words, we consider instance-based UDA methods. A machine learning model trained using data from one domain (source domain) might not necessarily perform well on a different (target) domain when their distributions are different. DA considers the problem of adapting a machine learning model such as a classifier that is trained using a source domain to a target domain. In particular, in UDA [15, 16, 126] we do not assume the availability of any labelled instances from the target domain but a set of labelled instances from the source domain and unlabelled instances from both source and the target domains.

Two main approaches for UDA can be identified from prior work: **projection-based** and **self-training**.

Projection-based methods for UDA learn an embedding space where the distribution of features in the source and the target domains become closer to each other than they were in the original feature spaces [15]. For this purpose, the union of the source and target feature spaces is split into domain-independent (often referred to as *pivots*) and domain-specific features using heuristics such as mutual information or frequency of a feature in a domain. A projection is then learnt between those two feature spaces and used to adapt a classifier trained from the source domain labelled data. For example, methods based on different approaches such as graph-decomposition *spectral feature alignment* [126] or autoencoders [109] have been proposed for this purpose.

Self-training [2, 181] is a technique to iteratively increase a set of labelled instances by training a classifier using current labelled instances and applying the trained classifier to predict pseudo-labels for unlabelled instances. High confident predictions are then appended to the current labelled dataset, thereby increasing the number of labelled instances. The process is iterated until no additional pseudo-labelled instances can be found. Self-training provides a direct solution to the lack of labelled data in the target domain in UDA [51, 115, 138]. Specifically, the source domain's labelled instances are used to initialise the self-training process and during subsequent iterations labels are inferred for the target domain's unlabelled instances, which can be used to train a classifier for the task of interest.

So far in UDA projection-learning and self-trained approaches have been explored separately. An interesting research question we ask and answer positively in this chapter is whether *can we improve the performance of projection-based methods in UDA using self-training?* In particular, recent work on UDA [120] has shown that minimising the entropy of a classifier on its predictions in the source and target domains is equivalent to learning a projection space that maximises the correlation between source and target instances. Motivated by these developments, we propose **Self-Adapt**, a method that combines the complementary strengths of projection-based methods and self-training methods for UDA.

The proposed method consists of three steps.

- First, using labelled instances from the source domain we learn a projection ($\mathcal{S}_{\text{prj}}$) that maximises the distance between each source domain labelled instance and its nearest neighbours with opposite labels. Intuitively, this process will learn a projected feature space in the source domain where the margin between the opposite labelled nearest neighbours is maximised, thereby minimising the risk of misclassifications. We project the source domain's labelled instances using $\mathcal{S}_{\text{prj}}$ for the purpose of training a classifier for predicting the target task labels such as positive/negative sentiment in cross-domain sentiment classification or part-of-speech tags in cross-domain part-of-speech tagging.
- Second, we use the classifier trained in the previous step to assign pseudo labels for the (unlabelled) target domain instances. Different strategies can be used for this label inference process such as selecting instances with the highest classifier confidence as in self-training or checking the agreement among multiple classifier as in tri-training.
- Third, we use the pseudo-labelled target domain instances to learn a projection for the target domain ($\mathcal{T}_{\text{prj}}$) following the same procedure used to learn $\mathcal{S}_{\text{prj}}$. Specifically, we learn a projected feature space in the target domain where the margin between the opposite pseudo-labelled nearest neighbours is maximised. We project labelled instances in the source domain and pseudo-labelled instances in the target domain respectively using $\mathcal{S}_{\text{prj}}$ and $\mathcal{T}_{\text{prj}}$, and use

those projected instances to learn a classifier for the target task.

As an evaluation task, we perform cross-domain sentiment classification on the Amazon multi-domain sentiment dataset [16]. Although most prior work on UDA have used this dataset as a standard evaluation benchmark, the evaluations have been limited to the four domains *books*, *dvds*, *electronic appliances* and *kitchen appliances*. We too report performances on those four domains for the ease of comparison against prior work. However, to reliably estimate the generalisability of the proposed method, we perform an additional extensive evaluation using 16 other domains included in the original version of the Amazon multi-domain sentiment dataset.

Results from the cross-domain sentiment classification reveal several interesting facts. A baseline that uses $\mathcal{S}_{\text{prj}}$ alone would still outperform a baseline that uses a classifier trained using only the source domain's labelled instances on the target domain test instances, without performing any adaptations. This result shows that it is useful to consider the label distribution available in the source domain to learn a projection even though it might be different to that in the target domain.

In addition, training a classifier using the pseudo-labelled target domain instances alone, without learning $\mathcal{T}_{\text{prj}}$ further improves performance. This result shows that pseudo labels inferred for the target domain unlabelled instances can be used to overcome the issue of lack of labelled instances in the target domain.

Moreover, if we further use the pseudo-labelled instances to learn $\mathcal{T}_{\text{prj}}$, then we observe a significant improvement of performance across all domain pairs, suggesting that UDA can benefit from both projection learning and self-training.

These experimental results support our claim that it is beneficial to combine projection-based and self-training-based UDA approaches. Moreover, the proposed method outperforms all self-training based domain adaptation methods such as tri-training [158, 190] and is competitive against neural domain adaptation methods [56, 109, 144, 148].

6.1 Self-Adaptation (Self-Adapt)

In UDA, we are given a set of positively ($\mathcal{S}_L^+$) and negatively ($\mathcal{S}_L^-$) labelled instances for a source domain $\mathcal{S}$ ($\mathcal{S}_L = \mathcal{S}_L^+ \cup \mathcal{S}_L^-$), and sets of unlabelled instances $\mathcal{S}_U$ and $\mathcal{T}_U$ respectively for the source and target domain $\mathcal{T}$. Given a dataset $\mathcal{D}$, we are required to learn a classifier $f(\boldsymbol{x}, y; \mathcal{D})$ that returns the probability of a test instance $\boldsymbol{x}$ taking a label y. For simplicity, we consider the pairwise adaptation from a single source to single target, and binary ($y \in \{-1, 1\}$) classification as the target task. However, Self-Adapt can be easily extended to multi-domain and multi-class UDA.

We represent an instance (document/review) x by a bag-of-n-gram (BonG) embedding [6], where we add the pre-trained d-dimensional word embeddings $\boldsymbol{w} \in \mathbb{R}^d$ for the words $w \in x$ to create a d-dimensional feature vector $\boldsymbol{x} \in \mathbb{R}^d$ representing x. Self-adapt consists of three steps: (a) learning a source projection using $\mathcal{S}_L$ (§6.1.1), (b) pseudo labelling $\mathcal{T}_U$ using a classifier trained on the projected $\mathcal{S}_L$ (§6.1.2); (c) learning a target projection using the pseudo-labelled target instances, and then learning a classifier f for the target task (§6.1.3).

6.1.1 Source Projection Learning ($\mathcal{S}_{\text{prj}}$)

In UDA, the adaptation task does not vary between the source and target domains. Therefore, we can use $\mathcal{S}_L$ to learn a projection for the source domain $\mathcal{S}_{\text{prj}}$ where the separation between an instance $x \in \mathcal{S}_L$ and its opposite-labelled nearest neighbours is maximised. Specifically, for an instance x we represent the set of k of its nearest neighbours $\text{NN}(\boldsymbol{x}, \mathcal{D}, k)$ selected from a set $\mathcal{D}$ by a vector $\phi(\boldsymbol{x}, \mathcal{D}, k)$ as the sum of the word embeddings of the neighbours given by

$$\phi(\boldsymbol{x}, \mathcal{D}, k) = \sum_{u \in \text{NN}(\boldsymbol{x}, \mathcal{D}, k)} \theta(x, u)\boldsymbol{u}. \tag{6.1}$$

Here, the weight $\theta(x, u)$ is computed using the cosine similarity between $\boldsymbol{u}$ and $\boldsymbol{x}$, and is normalised s.t. $\sum_{u \in \text{NN}(\boldsymbol{x}, \mathcal{D}, k)} \theta(x, u) = 1$. Other similarity measures can also be used instead of cosine, for example, Euclidean distance [171]. Then, $\mathcal{S}_{\text{prj}}$ is defined by the projection matrices $\mathbf{A}_+ \in \mathbb{R}^{d \times d}$ and $\mathbf{A}_- \in \mathbb{R}^{d \times d}$ and is learnt by maximising the objective O_L given by

$$O_L(\mathbf{A}_+, \mathbf{A}_-) = \sum_{\boldsymbol{x} \in \mathcal{S}_L^+} \left|\left|\mathbf{A}_+\boldsymbol{x} - \mathbf{A}_-\phi(\boldsymbol{x}, \mathcal{S}_L^-, k)\right|\right|_2^2 + \sum_{\boldsymbol{x} \in \mathcal{S}_L^-} \left|\left|\mathbf{A}_-\boldsymbol{x} - \mathbf{A}_+\phi(\boldsymbol{x}, \mathcal{S}_L^+, k)\right|\right|_2^2, \tag{6.2}$$

we initialise $\mathbf{A}_+$ and $\mathbf{A}_-$ to the identity matrix $\mathbf{I} \in \mathbb{R}^{d \times d}$ and apply Adam [86] to find their optimal values denoted respectively by $\mathbf{A}_+^*$ and $\mathbf{A}_-^*$. Other gradient descent optimisation algorithms can be applied to (6.2), such as AdaGrad [52] and AdaDelta [185]. Finally, we project $\mathcal{S}_L$ using the learnt $\mathcal{S}_{\text{prj}}$ to obtain a projected set of source domain labelled instances $\mathcal{S}_L^* = \mathbf{A}_+^* \circ \mathcal{S}_L^+ \cup \mathbf{A}_-^* \circ \mathcal{S}_L^-$. Here, we use the notation $\mathbf{A} \circ \mathcal{D} = \{\mathbf{A}\boldsymbol{x} | \boldsymbol{x} \in \mathcal{D}\}$ to indicate the application of a projection matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ on elements $\boldsymbol{x} \in \mathbb{R}^d$ in a dataset $\mathcal{D}$.

6.1.2 Pseudo Label Generation (PL)

In UDA, we do not have labelled data for the target domain. To overcome this issue, inspired by prior work on self-training approaches to UDA, we train a classifier $f(\boldsymbol{x}, y; \mathcal{S}_L^*)$ on $\mathcal{S}_L^*$ first and then

Algorithm 6.1 Pseudo Label Generation

```
Input: source domain positively labelled data $\mathcal{S}_L^+$,
       source domain negatively labelled data $\mathcal{S}_L^-$,
       source domain positive transformation matrix $\mathbf{A}_+$,
       source domain negative transformation matrix $\mathbf{A}_-$,
       target domain unlabelled data $\mathcal{T}_U$,
       a set of target classes $Y = \{+1, -1\}$,
       classification confidence threshold $\tau$.
Output: target domain pseudo-labelled data $\mathcal{T}_L'$
  $\mathcal{S}_L^* \leftarrow \mathbf{A}_+^* \mathcal{S}_L^+ \cup \mathbf{A}_-^* \mathcal{S}_L^-$
  $\mathcal{T}_L' \leftarrow \emptyset$
  for $\boldsymbol{x} \in \mathcal{T}_U$ do
    $y \in Y, p(t = y|\boldsymbol{x}) = f(\boldsymbol{x}, y; \mathcal{S}_L^*)$ {probability of $\boldsymbol{x}$ belongs to class $y$}
    if $p(t = y|\boldsymbol{x})) > \tau$ then
      $\mathcal{T}_L' \leftarrow \mathcal{T}_L' \cup \{(\boldsymbol{x}, y)\}$
    end if
  end for
  return $\mathcal{T}_L'$
```

use this classifier to assign pseudo labels for the target domain's unlabelled data $\mathcal{T}_U$, if the classifier is more confident than a pre-defined threshold τ. Algorithm 6.1 returns a pseudo-labelled dataset $\mathcal{T}_L'$ for the target domain. According to the classical self-training [2, 181], $\mathcal{T}_L'$ will be appended to $\mathcal{S}_L^*$ and the classifier is retrained on this extended labelled dataset. The process is repeated until no further unlabelled instances can be assigned labels with confidence higher than τ. Plank [133] found multiple iterations do not improve the performance in UDA beyond the first iteration. We observe a similar trend in our preliminary experiments. Therefore, we limit the number of iterations to one as shown in Algorithm 6.1. Doing so also speeds up the training process over classical self-training, which retrains the classifier and iterates.

6.1.3 Target Projection Learning ($\mathcal{T}_{\text{prj}}$)

Armed with the pseudo-labelled data generated via Algorithm 6.1, we can now learn a projection for the target domain, $\mathcal{T}_{\text{prj}}$, following the same procedure we proposed for learning $\mathcal{S}_{\text{prj}}$ in Section 6.1.1. Specifically, $\mathcal{T}_{\text{prj}}$ is defined by the two target-domain projection matrices $\mathbf{B}_+ \in \mathbb{R}^{d \times d}$ and $\mathbf{B}_- \in \mathbb{R}^{d \times d}$ that maximises the distance between each pseudo-labelled target instance x and its k opposite labelled nearest neighbours selected from positively ($\mathcal{T}_L'^+$) and negatively ($\mathcal{T}_L'^-$)

pseudo-labelled instances. The objective O'_L for this optimisation problem is given by

$$O'_L(\mathbf{B}_+, \mathbf{B}_-) = \sum_{\boldsymbol{x} \in \mathcal{T}'^{+}_L} \left|\left|\mathbf{B}_+\boldsymbol{x} - \mathbf{B}_-\phi(\boldsymbol{x}, \mathcal{T}'^{-}_L, k)\right|\right|_2^2 + \sum_{\boldsymbol{x} \in \mathcal{T}'^{-}_L} \left|\left|\mathbf{B}_-\boldsymbol{x} - \mathbf{B}_+\phi(\boldsymbol{x}, \mathcal{T}'^{+}_L, k)\right|\right|_2^2. \quad (6.3)$$

Likewise with $\mathcal{S}_{\text{prj}}$, $\mathbf{B}_+$ and $\mathbf{B}_-$ are initialised to the identify matrix $\mathbf{I} \in \mathbb{R}^{d \times d}$, and Adam is used to find their maximisers denoted respectively by $\mathbf{B}^*_+$ and $\mathbf{B}^*_-$. We project the target domain pseudo-labelled data using $\mathcal{T}_{\text{prj}}$ to obtain $\mathcal{T}'^{*}_L = \mathbf{B}^*_+ \circ \mathcal{T}'^{+}_L \cup \mathbf{B}^*_- \circ \mathcal{T}'^{-}_L$. Finally, we train a classifier $f(\boldsymbol{x}, y; \mathcal{S}^*_L \cup \mathcal{T}'^{*}_L)$ for the target task using both source and target projected labelled instances $\mathcal{S}^*_L \cup \mathcal{T}'^{*}_L$. Any binary classifier can be used for this purpose. In the experiments, we use ℓ_2 regularised logistic regression following prior work in UDA [15, 20, 126]. Moreover, by using a simple linear classifier, we can decouple the projection learning step from the target classification task, thereby more directly evaluate the performance of the former.

6.2 Experiments

The proposed method *does not* assume any information about the target task and can be in principle applied for any domain adaptation task. We use cross-domain sentiment classification as an evaluation task in this chapter because it has been used extensively in prior work on UDA, thereby enabling us to directly compare the performance of the proposed method against previously proposed UDA methods. In particular, we use the Amazon multi-domain sentiment dataset, originally created by Blitzer et al. [16], as a benchmark dataset in my experiments. This dataset includes Amazon Product Reviews from four categories: Books (**B**), DVDs (**D**), Electronic Appliances (**E**) and Kitchen Appliances (**K**). Considering each category as a domain[1], we can generate $\binom{4}{2} = 12$ pair-wise adaptation tasks involving a single source and a single target domain.

An Amazon product review is assigned 1-5 star rating and product reviews with 4 or 5 stars are labelled as positive, whereas 1 or 2 star reviews are labelled as negative. 3 star reviews are ignored because of their ambiguity. In addition to the labelled reviews, the Amazon multi-domain dataset contains a large number of unlabelled reviews for each domain. We use the official balanced train and test dataset splits, which has 800 (pos), 800 (neg) training instances and 200 (pos), 200 (neg) test instances for each domain. We name this dataset as the Multi-domain Adaptation Dataset (**MAD**).

One issue that is often raised with **MAD** is that it contains only four domains. In order to ro-

[1]Multiple reviews might exist for the same product within the same domain. Products are not shared across domains.

Dataset	Domain	Train	Test	Unlabel
MAD	books (**B**)	1600	400	6000
	dvd (**D**)	1600	400	34741
	electronics (**E**)	1600	400	13153
	kitchen (**K**)	1600	400	16785
EMAD	apparel	1426	360	7152
	baby	1260	340	2256
	beauty	602	258	1291
	camera_and_photo	1326	359	5309
	computer_and_video_games	486	251	1213
	gourmet_food	168	201	267
	grocery	414	230	1180
	health_and_personal_care	1408	360	5125
	jewelry_and_watches	312	218	589
	magazines	1152	354	2121
	music	1568	399	15041
	outdoor_living	358	225	172
	software	1264	343	375
	sports_and_outdoors	1402	360	3628
	toys_and_games	1426	360	11047
	videos	1172	399	15094

Table 6.1: Number of train, test and unlabelled reviews for **MAD** and **EMAD**.

bustly evaluate the performance of a UDA method we must evaluate on multiple domains. Therefore, in addition to **MAD**, we also evaluate on an extended dataset that contains 16 domains. We name this dataset as the Extended Multi-domain Adaptation Dataset (**EMAD**). The reviews for the 16 domains contained in **EMAD** were also collected by Blitzer et al. [16], but were not used in the evaluations. The same star-based procedure used in **MAD** is used to label the reviews in **EMAD**. We randomly select 20% of the available labelled reviews as test data and construct a balanced training dataset from the rest of the labelled reviews (i.e. for each domain we have equal number of positive and negative labelled instances in the train datasets). Likewise in **MAD**, we generate $\binom{16}{2} = 240$ pair-wise domain adaptation tasks from **EMAD**. The statistics of **MAD** and **EMAD** are shown in Table 6.1.

We train an ℓ_2 regularised logistic regression as the target (sentiment) classifier, in which we tune the regularisation coefficient using validation data. We randomly select 10% from the target domain labelled data, which is separate from the train or test data. We tune regularisation coefficient in $[0.001, 0.01, 0.1, 1]$. We use 300 dimensional pre-trained GloVe word embeddings [131] to create BonG embeddings for uni and bigrams. We found that a maximum of 100 epochs to be sufficient to reach convergence in all projection learning tasks for all domains.

6.2.1 Effect of Projection Learning and Pseudo-Labelling

The proposed method consists of 3 main steps as described in Section 6.1: source projection learning, pseudo labelling and target projection learning. Using **MAD**, in Table 6.2, we compare the relative effectiveness of these three steps towards the overall performance in UDA using $k = 1$ for all the steps. Specifically, we consider the following baselines:

- **NA:** No-adaptation. Learn a classifier from $\mathcal{S}_L$ and simply use it to classify sentiment on target domain test instances, without performing any domain adaptation.

- $\mathcal{S}_{\textbf{prj}}$**:** Learn a source projection $\mathcal{S}_{\text{prj}}$ and apply it to project $\mathcal{S}_L$ to obtain $\mathcal{S}_L^* = \mathbf{A}_+^* \circ \mathcal{S}_L^+ \cup \mathbf{A}_-^* \circ \mathcal{S}_L^-$. Train a sentiment classifier using $\mathcal{S}_L^*$ and use it to classify target domain test instances.

- $\mathcal{S}_{\textbf{prj}}$ **+PL:** Use the classifier trained using $\mathcal{S}_L^*$ on target domain unlabelled data to create a pseudo-labelled dataset $\mathcal{T}'_L$. Train a sentiment classifier on $\mathcal{S}_L^* \cup \mathcal{T}'_L$ and use it to classify target domain test instances.

- $\mathcal{S}_{\textbf{prj}}$ **+**$\mathcal{T}_{\textbf{prj}}$**:** This is the proposed method including all three steps. A target projection $\mathcal{T}_{\text{prj}}$ is learnt using $\mathcal{T}'_L$ and is applied to obtain $\mathcal{T}'^*_L = \mathbf{B}_+^* \circ \mathcal{T}'^+_L \cup \mathbf{B}_-^* \circ \mathcal{T}'^-_L$. Finally, a sentiment classifier is trained using $\mathcal{S}_L^* \cup \mathcal{T}'^*_L$ and used to classify target domain test instances.

With all methods, we keep $k = 1$ in the nearest neighbour feature representation in (6.1) for the ease of comparisons. Confidence threshold τ is tuned in the range $[0.5, 0.9]$ using cross-validation on a development dataset. From Table 6.2 we observe that $\mathcal{S}_{\text{prj}}$ consistently outperforms **NA**, showing that even without using any information from the target domain, it is still useful to learn source domain projections that discriminates instances with opposite labels. When we perform pseudo labelling on top of source projection learning ($\mathcal{S}_{\text{prj}}$ +**PL**) we observe a slight but consistent improvement in all domain-pairs. However, when we use the pseudo labelled instances to learn a target projection ($\mathcal{S}_{\text{prj}}$ +$\mathcal{T}_{\text{prj}}$) we obtain the best performance in all domain-pairs. Moreover, the obtained results are significantly better under the stricter $p < 0.001$ level over the **NA** baseline in 7 out of 12 domain-pairs.

Table 6.4 shows the classification accuracy for the **EMAD**. We show the average classification accuracy for adapting to the same target domain instead of showing all 240 domain-pairs for **EMAD**. Likewise in **MAD**, we observe in **EMAD** we obtain the best results when we use both source and target projections. Interestingly, we observe that the proposed method adapting well even to the domains with smaller numbers of unlabelled data such as **gourmet_food** (168 labelled

S-T	NA	$\mathcal{S}_{\text{prj}}$	$\mathcal{S}_{\text{prj}}$ +PL	$\mathcal{S}_{\text{prj}}$ +$\mathcal{T}_{\text{prj}}$
B-D	73.50	74.25	74.50	**76.25**
B-E	64.00	73.25**	73.50**	**77.50****
B-K	68.50	75.25*	76.00*	**78.75****
D-B	74.00	**75.50**	**75.50**	**75.50**
D-E	64.75	71.25*	71.50*	**74.25****
D-K	71.50	75.00	76.25	**79.75****
E-B	67.25	74.50*	75.25**	**76.25****
E-D	66.75	67.75	68.00	**69.50**
E-K	76.00	**81.00**	**81.00**	**81.00**
K-B	63.00	73.75**	73.75**	**75.00****
K-D	71.25	71.25	71.00	**72.50**
K-E	69.00	77.50**	78.00**	**78.25****

Table 6.2: Target domain test data classification accuracy for the different steps in the proposed method ($k = 1$). S-T denotes adapting from a source S to a target T domain. The best result for each domain-pair is bolded. Statistically significant improvements over **NA** according to the binomial exact test are shown by "*" and "**" respectively at $p = 0.01$ and $p = 0.001$ levels.

and 267 unlabelled train instances). This is encouraging because it shows that the proposed method can overcome the lack of labelled instances via pseudo labelling and projection learning.

6.2.2 Comparisons against Self-Training

Ruder and Plank [144] evaluated classical general-purpose semi-supervised learning methods proposed for inducing pseudo labels for unlabelled instances using a seed set of labelled instances in the context of UDA. They found that tri-training to outperform more complex neural state-of-the-art UDA methods. Considering the fact that **Self-Adapt** is performing pseudo-labelling, similar to other self-training methods, it is interesting to observe how well it compares against classical self-training methods for inducing labels [2, 158, 181, 190] when applied to UDA. Specifically, we consider the classical self-training [2, 181] (**Self**), Tri-training [190] (**Tri**) and Tri-training with diversification [158] (**Tri-D**). For each of those methods, we use the labelled data in the source domain as seeds and induce labels for the unlabelled data in the target domain. Table 6.3 reports the results on **MAD**.

We re-implement the classical self-training methods considered by Ruder and Plank [144] and evaluated them against the proposed self-adapt on the same datasets, feature representations and settings to conduct a fair comparison. All classical self-training methods were trained using the

S-T	**NA**	**Self**	**Tri**	**Tri-D**	**Self-Adapt**
B-D	73.50	74.25	74.75	**77.25**	**77.25**
B-E	64.00	65.00	73.25**	72.00**	**78.50****
B-K	68.50	70.75	73.25	73.75	**79.00****
D-B	74.00	73.00	77.00	76.00	**81.00***
D-E	64.75	65.25	73.75**	70.50*	**75.50****
D-K	71.50	71.75	75.75	74.00	**79.75****
E-B	67.25	68.25	68.50	74.25*	**76.25****
E-D	66.75	66.25	68.25	**73.50***	69.50
E-K	76.00	76.50	**82.50***	81.00	**82.50***
K-B	63.00	63.00	70.00*	73.00**	**75.00****
K-D	71.25	71.00	71.25	72.00	**72.75**
K-E	69.00	68.75	73.00	75.50*	**79.00****

Table 6.3: Target domain test data classification accuracy of classical self-training methods when applied to UDA (k is selected using a validation dataset).

source domain labelled instances $\mathcal{S}_L$ as seed data. As discussed in Section 6.1.2, similar to **Self-Adapt**, we observed that the performance did not significantly increase beyond the first iteration for any of the classical self-training methods in UDA. Consequently, we compare all classical self-training methods for their peak performance, obtained after the first iteration. We tune the confidence threshold τ for each method using validation data and found the optimal value of τ to fall in the range $[0.6, 0.9]$. k is a hyperparameter selected using validation dataset for **Self-Adapt** in comparison.

Experimental results on **MAD** and **EMAD** are shown respectively in Tables 6.3 and 6.5. From those Tables, we observe that **Self-Adapt** for most of the domain pairs performs similarly or slightly worse than **NA**. Although **Tri** and **Tri-D** outperform **NA** on all cases, we found that those two methods are highly sensitive to the seed instances used to initialise the pseudo-labelling process. We find the proposed **Self-Adapt** to outperform all classical self-training based methods in 11 out of 12 domain pairs in **MAD** and in all 16 target domains in **EMAD**, showing a strong and robust improvement over classical self-training methods. This result shows that by combining source and target domain projections with self-training, we can obtain superior performance in UDA in comparison to using classical self-training methods alone.

Target Domain	**NA**	$\mathcal{S}_{\textbf{prj}}$	$\mathcal{S}_{\textbf{prj}}$ **+PL**	$\mathcal{S}_{\textbf{prj}}$ **+**$\mathcal{T}_{\textbf{prj}}$
apparel	71.39	75.31	75.66	**76.68***
baby	68.00	71.06	71.37	**72.63**
beauty	69.66	73.20	73.18	**73.70**
camera_and_photo	68.46	73.46	74.02	**74.54***
computer_and_video_games	61.78	67.38	67.85	**68.11***
gourmet_food	72.34	81.13**	82.89**	**83.15****
grocery	71.01	76.90*	77.57 *	**78.14***
health_and_personal_care	65.85	68.38	68.67	**69.24**
jewelry_and_watches	68.90	77.86**	79.11**	**79.79****
magazines	65.28	71.54*	71.45*	**72.16***
music	66.27	70.69	70.89	**71.41**
outdoor_living	71.97	76.77	78.01*	**78.93***
software	65.72	69.56	69.60	**70.31**
sports_and_outdoors	66.56	70.18	70.67	**71.02**
toys_and_games	69.57	72.82	73.22	**73.65**
video	64.19	67.59	68.37	**68.97**

Table 6.4: Average classification accuracy on each target domain in **EMAD** for the steps in the proposed method ($k = 1$).

6.2.3 Comparisons against Neural UDA

In Table 6.6, we compare **Self-Adapt** against the following neural UDA methods on **MAD**: Variational Fair Autoencoder [109] (**VFAE**), Domain-adversarial Neural Networks [56] (**DANN**), Asymmetric Tri-training [148] (**Asy-Tri**), and Multi-task Tri-training [144] (**MT-Tri**). We select these methods as they are the current state-of-the-art for UDA on **MAD**, and report the results from the original publications in Table 6.6.

Although only in 3 out of 12 domain-pairs **Self-Adapt** is obtaining the best performance, the difference of performance between **DANN** and **Self-Adapt** is not statistically significant. Although **MT-Tri** is outperforming **Self-Adapt** in 8 domain-pairs, it is noteworthy that **MT-Tri** is using a larger feature space than that of **Self-Adapt**. Specifically, **MT-Tri** is using 5000 dimensional tf-idf weighted unigram and bigram vectors for representing reviews, whereas **Self-Adapt** uses a 300 dimensional BonG representation computed using pre-trained GloVe vectors. Moreover, prior work on neural UDA have not used the entire unlabelled datasets and have sampled a smaller subset due to computational feasibility. For example, **MT-Tri** uses only 2000 unlabelled instances for each domain despite the fact that the original unlabelled datasets contain much larger numbers of reviews. Our preliminary experiments revealed that the performance of neural UDA methods to

Target Domain	**NA**	**Self**	**Tri**	**Tri-D**	**Self-Adapt**
apparel	71.39	71.38	73.96	73.89	**76.68**
baby	68.00	67.96	69.71	69.84	**72.63**
beauty	69.66	70.54	71.73	70.49	**73.70**
camera_and_photo	68.46	68.44	71.31	71.28	**74.54****
computer_and_video_games	61.78	62.28	64.65	64.93	**68.11****
gourmet_food	72.34	74.10	75.39	77.88	**83.15****
grocery	71.01	71.83	75.22	74.29	**78.14***
health_and_personal_care	65.85	66.08	67.10	67.07	**69.24**
jewelry_and_watches	68.90	71.04	76.67**	76.21**	**79.79****
magazines	65.28	65.34	67.58	67.47	**72.16***
music	66.27	66.22	68.82	68.27	**71.41**
outdoor_living	71.97	74.01	76.21	74.79	**78.93***
software	65.72	65.47	67.36	67.53	**70.31**
sports_and_outdoors	66.56	66.87	69.22	68.05	**71.02**
toys_and_games	69.57	70.03	71.76	72.32	**73.65**
video	64.19	64.88	66.34	67.49	**68.97**

Table 6.5: Average classification accuracy on each target domain in **EMAD** of classical self-training based methods when applied to UDA.

be sensitive to the unlabelled datasets used.[2] However, **Self-Adapt** does not require sub-sampling of unlabelled data and uses all the available unlabelled data for UDA. During the pseudo-labelling step, **Self-Adapt** automatically selects a subset of unlabelled target instances that are determined to be confident by the classifier more than a pre-defined threshold τ. The ability to operate on a lower-dimensional feature space and obviating the need to subsample unlabelled data are properties of the proposed method that are attractive when applying UDA methods on large datasets and across multiple domains.

6.2.4 Parameter Sensitivity

Figure 6.1 shows the effect on **Self-Adapt** of neighbourhood size ($k = 1, 5, 10$) for the source projection step. We observe that the performance increases initially and peaks around $k = 5$ for most of the domain pairs (8 out of 12). Increasing the neighbourhood size enables the projection learning method to utilise more local information but larger neighbourhoods include unrelated instances that act as noise.

[2]We report the results from the original publication.

S-T	VFAE	DANN	Asy-Tri	MT-Tri	Self-Adapt
B-D	79.90	78.40	80.70	**81.47**	77.25
B-E	79.20	73.30	**79.80**	78.62	78.50
B-K	81.60	77.90	**82.50**	78.09	79.00
D-B	75.50	72.30	73.20	77.49	**81.00****
D-E	78.60	75.40	77.00	**79.66**	75.50
D-K	82.20	78.30	**82.50**	81.23	79.75
E-B	72.70	71.10	73.20	73.43	**76.25**
E-D	**76.50**	73.80	72.90	75.05	69.50
E-K	85.00	85.40	86.90	**87.07**	82.50
K-B	72.00	70.90	72.50	73.60	**75.00**
K-D	73.30	74.00	74.90	**77.41**	72.75
K-E	83.80	84.30	84.60	**86.06**	79.00

Table 6.6: Classification accuracy compared with neural adaptation methods. The best result is bolded. Statistically significant improvements over **DANN** according to the binomial exact test are shown by "*" and "**" respectively at $p = 0.01$ and $p = 0.001$ levels.

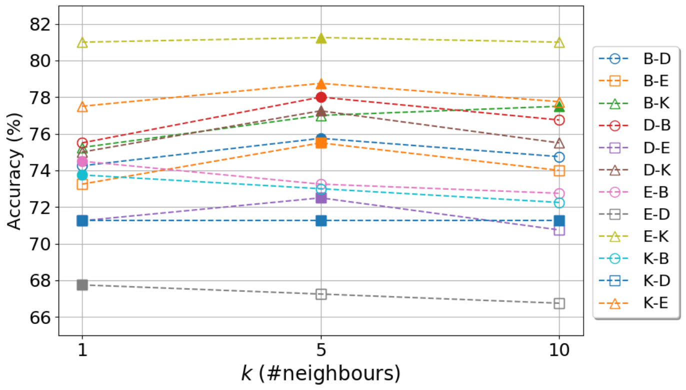


Figure 6.1: Number of neighbours k for $\mathcal{S}_{\text{prj}}$ in **MAD**. The marker for the best result for each domain-pair is filled.

6.2.5 Effect of Pseudo Label Generation

Table 6.7 we show some examples from pseudo label generation in $\mathcal{T}_U$ adapting from **B** to **D**. Labels are not available for $\mathcal{T}_U$, we show the truth label annotated by authors and predicted labels are generated following steps in Algorithm 6.1.

Review	P	T
Entertaining and where my educational fought. Great grandfather visual.	+	+
Civil sides this both sides follows the a great battle. Battle tells you're into activities of movie.	+	+
Garbage! Uninteresting piece! Most boring movie, this movie is piece of ever of garbage.	-	-
The story without giving feels like ending hot hot sequel. Really bit unsatisfactory.	-	-
Fantastic third season. Good first 8 murder, and I finish the complete series.	+	+

Table 6.7: Examples of **B-D** for **PL**. "P" denotes predicted label by **PL**, and "T" denotes truth label by manual annotation.

6.3 Summary

In this chapter, we proposed **Self-Adapt**, a UDA method that combines projection learning and self-training. Using the labelled data from the source domain, **Self-Adapt** first learns a source projection, which is then used to project source domain's labelled data. Next, a classifier is trained on the projected data, which is used to induce pseudo labels for the target domain. Using the pseudo labels, **Self-Adapt** learns a transformation for the target domain. Our experimental results on two datasets for cross-domain sentiment classification show that projection learning and self-training have complementary strengths and jointly contribute to improve UDA performance.

Chapter 7

Negative Transfer in UDA

Many machine learning processes have different training and testing distributions [186], thus leading to the problem of DA. Most DA methods consider adapting to a target domain from a single source domain [15, 16, 56]. The goal of DA is to transfer salient information from the source domain to obtain a model suitable for a particular target domain [31]. In the previous chapters, we studied UDA methods that learn an adaptive model from a single source domain. In practice, however, training data can come from multiple sources. For example, in sentiment classification, if we consider each category as a domain, we will have product reviews from multiple domains.

Feature-based methods, such as SCL [15, 16] and SFA [126], fail in multi-source settings because it is challenging to find pivots across all sources such that a shared projection can be learnt. Similarly, instance-based methods, for example, SDA [61] and mSDA [30] using autoencoders have been proposed, where the model is trained to minimise the loss between the original inputs and their reconstructions. Not all of the source domains are appropriate for learning transferable projections for a particular target domain. Often, it is difficult to find a suitable single source to adapt from, and one must consider multiple sources. Adapting from an unrelated source can result in poor performance on the given target, which is known as *negative transfer* [70, 141].

Prior proposals for multi-source UDA can be broadly classified into methods that: (a) first select a source domain and then select instances from that source domain to adapt to a given target domain test instance [56, 70, 85, 188]; (b) pool all source domain instances together and from this pool select instances to adapt to a given target domain test instance [27, 134]; (c) pick a source domain and use all instances in that source (source domain selection) [155]; and (d) pick all source domains and use all instances (utilising all instances) [7, 18, 174].

In contrast, in this chapter, we propose a UDA method for multi-source UDA and make the following contributions:

- We propose a self-training-based pseudo-labelling method for learning an attention model for multi-source UDA. The proposed method learns *domain-attention* weights for the source domains per test instance. Based on the learnt attention scores, we are able to find appropriate sources to adapt to a given target domain.
- Unlike adversarial neural networks based approaches [56, 70], our proposed method does not require rule-based labelling of instances for training.
- We evaluate the performance of the proposed method against pivot- and instance-based approaches. The proposed method performs competitively against previously proposed multi-source UDA methods and is able to provide evidence for its predictions.

7.1 Multi-Source Domain Attention

Let us assume that are given N source domains, $S_1, S_2, \ldots, S_N$, and required to adapt to a target domain T. Moreover, let us denote the labelled instances in S_i by $\mathcal{S}_i^L$ and unlabelled instances by $\mathcal{S}_i^U$. For T we have only unlabelled instances $\mathcal{T}^U$ in the UDA setting. Our goal is to learn a classifier to predict labels for the target domain instances using $\mathcal{S}^L = \cup_{i=1}^N \mathcal{S}_i^L$, $\mathcal{S}^U = \cup_{i=1}^N \mathcal{S}_i^U$ and $\mathcal{T}^U$. We denote labelled and unlabelled instances in S_i by respectively x_i^L and x_i^U, whereas instances in T are denoted by x_T. To simplify the notation, we drop the superscripts L and U when it is clear from the context whether the instance is respectively labelled or not.

The steps of our proposed method can be summarised as follows: (a) use labelled and unlabelled instances from each of the source domains to learn classifiers that can predict the label for a given instance. Next, develop a majority voter and use it to predict the *pseudo-labels* for the target domain unlabelled instances $\mathcal{T}^U$ (§7.1.1); (b) compute a *relatedness map* between the target domain's pseudo-labelled instances, $\mathcal{T}^{L*}$, and source domains' labelled instances $\mathcal{S}^L$ (§7.1.2); (c) compute *domain-attention* weights for each source domain (§7.1.3); (d) jointly learn a model based on the relatedness map and the domain-attention weights for predicting labels for the target domain's test instances (§7.1.4).

7.1.1 Pseudo-Label Generation

In UDA, we have only unlabelled data for the target domain. Therefore, we first introduce pseudo-labels for the target domain instances $\mathcal{T}^U$ by self-training [2] following Algorithm 7.1. Specifically, we first train a predictor f_i for the i-th source domain using only its labelled instances $\mathcal{S}_i^L$ using a base learner Γ (Line 1-2). Any classification algorithm that can learn a predictor f_i that can compute the probability, $f_i(x, y)$, of a given instance x belonging to the class y can be used as Γ. In our experiments, we use logistic regression for its simplicity and popularity in prior UDA work [18, 19]. Next, for each unlabelled instance in the selected source domain, we compute the probability of it belonging to each class and find the most probable class label. If the probability of the most likely class is greater than the given confidence threshold $\tau \in [0, 1]$, we will append that instance to the current labelled training set. This enables us to increase the labelled instances for the source domains, which is important for learning accurate classifiers when the amount of labelled instances available is small. After processing all unlabelled instances in domain S_i we train the final classifier f_i for that domain using all initial and pseudo-labelled instances. We predict a pseudo-label for a target domain instance as the majority vote, f^*, over the predictions made by the individual classifiers f_i.

Algorithm 7.1 Multi-Source Self-Training

```
Input: source domains' labelled instances $\mathcal{S}_1^L, \ldots, \mathcal{S}_N^L$,
       source domains' unlabelled instances $\mathcal{S}_1^U, \ldots, \mathcal{S}_N^U$,
       target domain's unlabelled instances $\mathcal{T}^U$,
       target classes $\mathcal{Y}$, base learner $\Gamma$,
       and the classification confidence threshold $\tau$.
       Output: multi-source self-training classifier $f^*$.
1:  for $i = 1$ to $N$ do
2:    $\mathcal{L}_i \leftarrow \mathcal{S}_i^L$
3:    $f_i \leftarrow \Gamma(\mathcal{L}_i)$
4:    for $x \in \mathcal{S}_i^U$ do
5:      $\hat{y} = \arg\max_{y \in \mathcal{Y}} f_i(x, y)$
6:      if $f_i(x, \hat{y}) > \tau$ then
7:        $\mathcal{L}_i \leftarrow \mathcal{L}_i \cup \{(x, \hat{y})\}$
8:      end if
9:    end for
10:   $f_i \leftarrow \Gamma(\mathcal{L}_i)$
11: end for
12: return majority voter $f^*$ over $f_1, \ldots, f_N$.
```

Selecting the highest confident pseudo-labelled instances for the purpose of training a classifier for the target domain has been a popular as done in prior work [2, 144, 158, 190] does not guarantee that those instances will be the most suitable ones for adapting to the target domain, which was not considered during the self-training stage. For example, some target instances might not be good prototypical examples of the target domain and we would not want to use the pseudo-labels induced for those instances when training a classifier for the target domain. To identify instances in the target domain that are better prototypes, we first encode each target instance by a vector and select the instances that are closest to the centroid, $\boldsymbol{c}_T$, of the target domain instances given by

$$\boldsymbol{c}_T = \frac{1}{|\mathcal{T}^U|} \sum_{x \in \mathcal{T}^U} \boldsymbol{x}. \tag{7.1}$$

In the case of text documents x, their embeddings, $\boldsymbol{x}$, can be computed using numerous approaches such as using bi-directional LSTMs [116] or transformers [139]. In our experiments, we use the Smoothed Inversed Frequency (SIF) proposed by Arora et al. [5], which computes document embeddings as the weighted-average of the pre-trained word embeddings for the words in a document. Despite being unsupervised, SIF has shown strong performance in numerous semantic textual similarity benchmarks [3]. Using the centroid computed in (7.1), similarity for target instance to the centroid is computed using the cosine similarity given by

$$\mathrm{sim}(\boldsymbol{x}, \boldsymbol{c}_T) = \frac{\boldsymbol{x}^\top \boldsymbol{c}_T}{||\boldsymbol{x}||\,||\boldsymbol{c}_T||}. \tag{7.2}$$

Other distance measures such as the Euclidean distance can also be used. We use cosine similarity here for its simplicity. We predict the labels for the target domain unlabelled instances, $\mathcal{T}^U$, using f^*, and select the instances with the top-k highest similarities to the target domain according to (7.2) as the target domain's pseudo-labelled instances $\mathcal{T}^{L*}$.

7.1.2 Relatedness Map Learning

Not all of the source domain instances are relevant to a given target domain instance and the performance of a classifier under domain shift can be upper bounded by the $\mathcal{H}$-divergence between a source and a target domain [10, 11, 83]. To model the relatedness between a target domain instance and each instance from the N source domains, we use the pseudo-labelled target domain instances $\mathcal{T}^{L*}$ and source domains' labelled instances $\mathcal{S}_i^L$ to learn a *relatedness map*, ψ_i, between a target domain instance $\boldsymbol{x}_T(\in \mathcal{T}^{L*})$ and a source domain labelled instance $\boldsymbol{x}_i^L$ $(\in \mathcal{S}_i^L)$ as given by

$$\psi_i(\boldsymbol{x}_T, \boldsymbol{x}_i^L) = \frac{\exp(\boldsymbol{x}_T{}^\top \boldsymbol{x}_i^L)}{\sum_{\boldsymbol{x}' \in \mathcal{S}_i^L} \exp(\boldsymbol{x}_T{}^\top \boldsymbol{x}')}. \tag{7.3}$$

With the help of the relatedness map, ψ_i, we can determine how well each instance in a source domain contributes to the prediction of the label of a target domain's instance.

7.1.3 Instance-based Domain-Attention

To avoid negative transfer, we dynamically select the source domain(s) to use when predicting the label for a given target domain instance. Specifically, we learn *domain-attention*, $\theta(\boldsymbol{x}_T, \mathcal{S}_i)$, for each source domain, S_i, conditioned on x_T as given by

$$\theta(\boldsymbol{x}_T, \mathcal{S}_i) = \frac{\exp({\boldsymbol{x}_T}^\top \boldsymbol{\phi}_i)}{\sum_{j=1}^{N} \exp({\boldsymbol{x}_T}^\top \boldsymbol{\phi}_j)}. \tag{7.4}$$

$\boldsymbol{\phi}_i$ can be considered as a *domain embedding* for S_i and has the same dimensionality as the instance embeddings. During training, to prevent activation outputs from exploding or vanishing, we initialise $\boldsymbol{\phi}_i$ using Xavier initialisation [60] and normalise such that $\forall \boldsymbol{x}_T, \sum_{i=1}^{N} \theta(\boldsymbol{x}_T, \mathcal{S}_i) = 1$.

7.1.4 Training

We combine the relatedness map (Section 7.1.2) and domain-attention (Section 7.1.3) and predict the label, $\hat{y}(x_T)$, of a target domain instance x_T using

$$\hat{y}(x_T) = \sigma\left(\sum_{i=1}^{N} \sum_{\boldsymbol{x}_i^L \in \mathcal{S}_i^L} y(x_i^L)\, \psi_i(\boldsymbol{x}_T, \boldsymbol{x}_i^L)\, \theta(\boldsymbol{x}_T, \mathcal{S}_i)\right). \tag{7.5}$$

Here, $\sigma(z) = 1/(1+\exp(-z))$ is the logistic sigmoid function and $y(x_i^L)$ is the label of the source domain labelled instance x_i^L.

First, we use the target instances, $x \in \mathcal{T}^{L*}$, with inferred labels $y^*(x)$ (computed using f^* produced by Algorithm 7.1) as the training instances and predict their labels, $\hat{y}(x)$, by (7.5). The cross entropy error, $E\left(\hat{y}(x), y^*(x)\right)$ for this prediction is given by

$$E\left(\hat{y}(x), y^*(x)\right) = -\lambda(x)\big(1 - y^*(x)\big) \log\big(1 - \hat{y}(x)\big) - \lambda(x) y^*(x) \log\big(\hat{y}(x)\big), \tag{7.6}$$

$\lambda(x)$ a rescaling factor computed using the normalised similarity score as

$$\lambda(x) = \frac{\mathrm{sim}(\boldsymbol{x}, \boldsymbol{c}_T)}{\sum_{\boldsymbol{x}' \in \mathcal{T}^{L*}} \mathrm{sim}(\boldsymbol{x}', \boldsymbol{c}_T)}. \tag{7.7}$$

We minimise the cross-entropy error given by (7.6) using Adam [86] for the purpose of learning the domain-embeddings, ϕ_i. The initial learning rate in Adam was set to 10^{-3} using a subset of $\mathcal{T}^{L*}$ held-out as a validation dataset. Other gradient descent optimisation algorithms can also be used, such as AdaGrad [52] and AdaDelta [185].

7.2 Experiments

To evaluate the proposed method, we use the multi-domain Amazon product review dataset compiled by Blitzer et al. [16]. This dataset contains product reviews from four domains: Books (**B**), DVD (**D**), Electronics (**E**) and Kitchen Appliances (**K**). Following Guo et al. [70], we conduct experiments under two different splits of this dataset as originally proposed by Blitzer et al. [16] (**Blitzer2007**) and by Chen et al. [30] (**Chen2012**). Table 7.1 shows the number of instances in each dataset. By using these two versions of the Amazon review dataset, we can directly compare the proposed method against relevant prior work. Next, we describe how the proposed method was trained on each dataset.

For **Blitzer2007**, we use the official train and test splits where each domain contains 1600 labelled training instances (800 positive and 800 negative), and 400 target test instances (200 positive and 200 negative). In addition, each domain also contains 6K-35K unlabelled instances. We use 300 dimensional pre-trained GloVe embeddings [131] following prior work [18, 174] with SIF [5] to create document embeddings for the reviews.

In **Chen2012**, each domain contains 2000 labelled training instances (1000 positive and 1000 negative), and 2000 target test instances (1000 positive and 1000 negative). The remainder of the instances are used as unlabelled instances (ca. 4K-6K for each domain). We use the publicly available[1] 5000 dimensional tf-idf vectors produced by Zhao et al. [188]. We use a multilayer perceptron (MLP) with an input layer of 5000 dimensions and 3 hidden layers with 500 dimensions. We use final output layer with 500 dimensions as the representation of an instance.

For each setting, we follow the standard input representation methods as used in prior work. It also shows the flexibility of the proposed method to use different (embedding vs. BoW) text representation methods. We conduct experiments for cross-domain sentiment classification with multiple sources by selecting one domain as the target and the remaining three as sources. The statistics for the two settings are shown in Table 7.1.

[1] `https://github.com/KeiraZhao/MDAN/`

Target	Source	Train	Test	Unlabel
Blitzer2007 [15]				
B	D,E,K	1600×3	400	6000
D	B,E,K	1600×3	400	34741
E	B,D,K	1600×3	400	13153
K	B,D,E	1600×3	400	16785
Chen2012 [30]				
B	D,E,K	2000×3	2000	4465
D	B,E,K	2000×3	2000	5586
E	B,D,K	2000×3	2000	5681
K	B,D,E	2000×3	2000	5945

Table 7.1: Number of train, test and unlabelled instances for Amazon product reviews.

7.2.1 Comparisons against Prior Work

We evaluate the proposed method in two settings. In Table 7.2, we compare our method against the following methods on **Blitzer2007** dataset:

uni-MS: is the baseline model, trained on the union of all source domains and tested directly on a target domain without any DA. uni-MS has been identified as a strong baseline for multi-source DA [7, 70, 188].

SCL: Structural Correspondence Learning [15, 16] is a single-source DA method, trained on the union of all source domains and tested on the target domain. We report the published results from Wu and Huang [174].

SFA: Spectral Feature Alignment [126] is a single-source DA method, trained on the union of all source domains, and tested on the target domain. We report the published results from Wu and Huang [174].

SST: Sensitive Sentiment Thesaurus [18, 19] is the SoTA multi-source DA method on **Blitzer2007**. We report the published results from Bollegala et al. [18].

SDAMS: Sentiment Domain Adaptation with Multiple Sources proposed by Wu and Huang [174]. We report the results from the original paper.

AMN: End-to-End Adversarial Memory Network [104] is a single-source DA method, trained on the union of all source domains, and tested on the target domain. We report the published results from Ding et al. [49].

In Table 7.3, we compare our proposed method against the following methods on **Chen2012**.

T	uni-MS	SCL	SFA	SST	SDAMS	AMN	Proposed
B	80.00	74.57	75.98	76.32	78.29	79.75	**83.50**
D	76.00	76.30	78.48	78.77	79.13	79.83	**80.50**
E	74.75	78.93	78.08	83.63*	**84.18****	80.92*	80.00*
K	85.25	82.07	82.10	85.18	**86.29**	85.00	86.00

Table 7.2: Classification accuracies (%) for the proposed method and prior work on **Blitzer2007**. Statistically significant improvements over **uni-MS** according to the Binomial exact test are shown by "*" and "**" respectively at $p = 0.01$ and $p = 0.001$ levels.

mSDA: Marginalized Stacked Denoising Autoencoders proposed by Chen et al. [30]. We report the published results from Guo et al. [70].

DANN: Domain-Adversarial Neural Networks proposed by Ganin et al. [56]. We report the published results from Zhao et al. [188].

MDAN: Multiple Source Domain Adaptation with Adversarial Learning proposed by Zhao et al. [188]. We report the published results from the original paper.

MoE: Mixture of Experts proposed by Guo et al. [70]. We report the published results from the original paper.

T	uni-MS	mSDA	DANN	MDAN	MoE	Proposed
B	79.46	76.98	76.50	78.63	79.42	**79.68**
D	82.32	78.61	77.32	80.65	**83.35**	82.96
E	84.93	81.98	83.81	85.34	**86.62**	85.30
K	86.71	84.26	84.33	86.26	**87.96**	87.48

Table 7.3: Classification accuracies (%) for the proposed method and prior work on **Chen2012**.

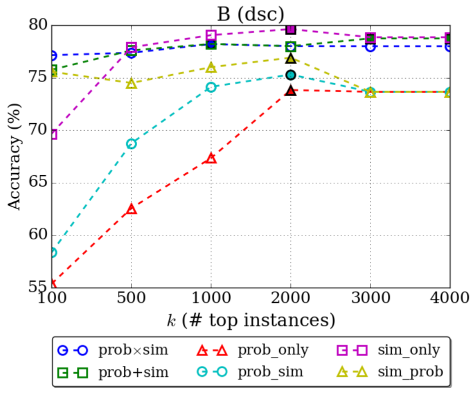
From Table 7.2 and Table 7.3, we observe that the proposed method obtains the best classification accuracy on Books domain (**B**) in both settings, which is the domain with the smallest number of unlabelled instances.

7.2.2 Effect of Self-Training

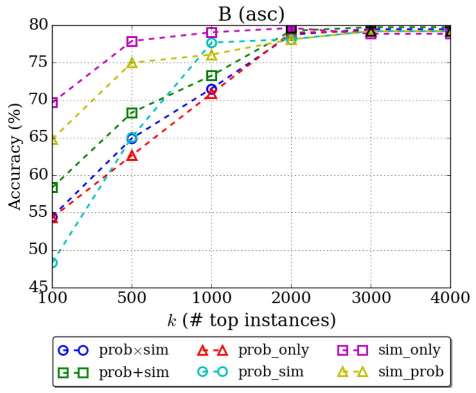
As described in Section 7.1.1, our proposed method uses self-training to generate pseudo-labels for the target domain unlabelled instances. In Table 7.4, we compare self-training against alternative pseudo-labelling methods on **Chen2012**: Self-Training (**Self**) [2, 27], Union Self-Training (**uni-Self**) [7], Tri-Training (**Tri**) [190] and Tri-Training with Disagreement (**Tri-D**) [158]. In Table 7.4, we observe that all semi-supervised learning methods improve only slightly over uni-MS (no adapt baseline). Therefore, pseudo-labelling step alone is insufficient for DA. Moreover, we observe that all semi-supervised methods perform comparably.

T	uni-MS	Self	uni-Self	Tri	Tri-D
B	79.46	79.60	79.46	**79.61**	79.51
D	82.32	**82.49**	82.35	82.35	82.35
E	84.93	84.97	84.93	**84.99**	84.93
K	87.17	87.18	87.17	87.15	**87.23**

Table 7.4: Classification accuracies (%) for semi-supervised methods on **Chen2012**.



(a) prob sorted in descending order



(b) prob sorted in ascending order

Figure 7.1: The number of selected pseudo-labelled instances k on **Blitzer2007** is shown on the x-axis. prob denotes prediction confidence from the pseudo classifier trained on the source domains, sim denotes the similarity to the target domain, asc and dsc respectively denote sorted in ascending and descending order (only applied to prob related selection methods, sim is always sorted in dsc). prob_only denotes using only prediction confidence, sim_only denotes using only target similarity. prob_sim indicates selecting by prob first and then sim (likewise for sim_prob). prob×sim denotes using the product of prob and sim, and prob+sim denotes using their sum. The marker for the best result of each method is filled.

7.2.3 Pseudo-labelled Instances Selection

When selecting the pseudo-labelled instances from the target domain for training a classifier for the target domain, we have two complementary strategies: (a) select the most confident instances according to f^* (denoted by *prob*) or (b) select the most similar instances to the target domain's centroid (denoted by *sim*). To evaluate the effect of these two strategies and their combinations (i.e

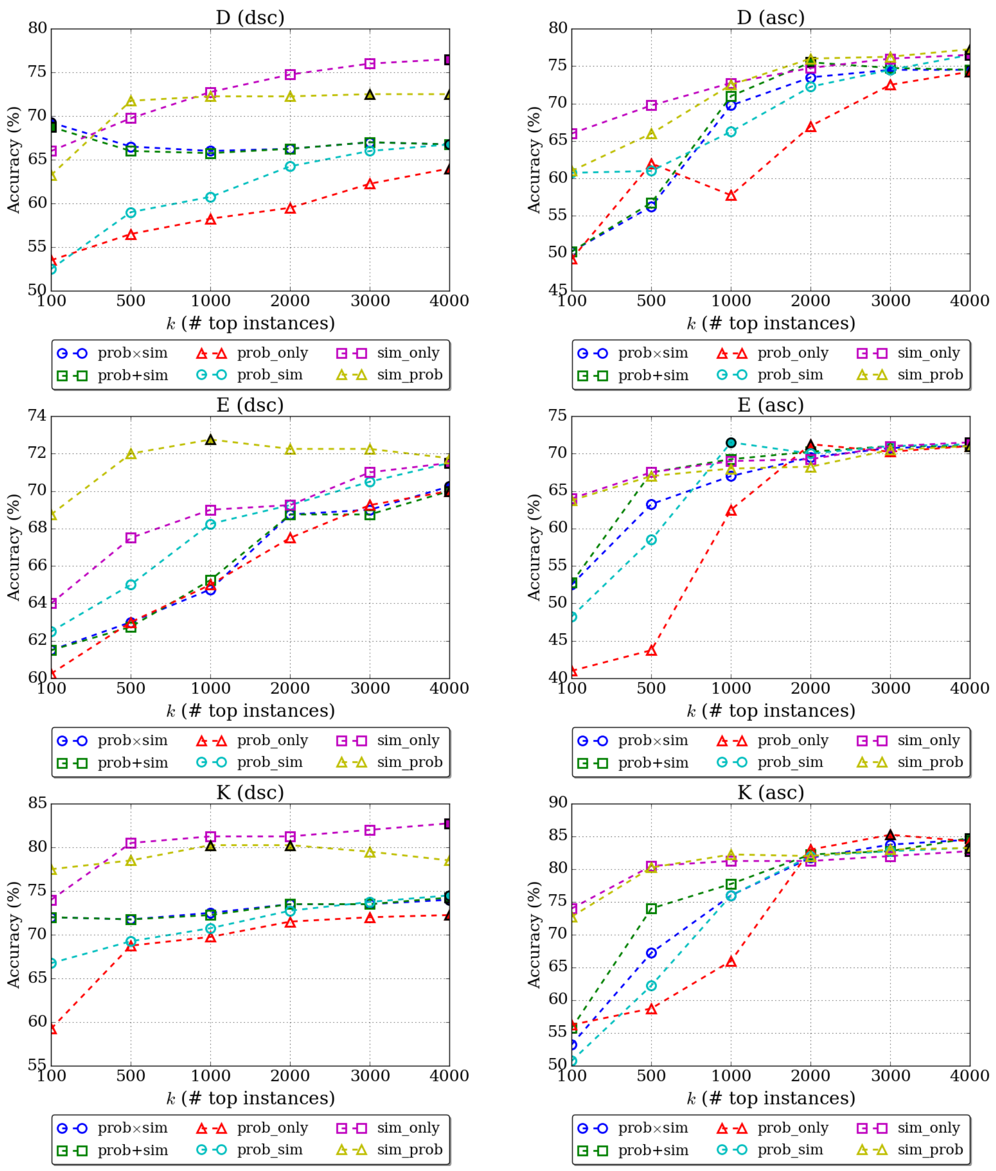


Figure 7.2: Effect of selection methods in **PL** step when different selection criteria are used on the other target domains (**D**, **E** and **K**) in **Blitzer2007**. prob is sorted in descending order for the figures in the first column (left), and in ascending order for the figures in the second column (right).

Example (1) *Why anybody everest feet would want reading this? ... pure pleasure why 29028 feet account this?... It's a pleasure to read.*

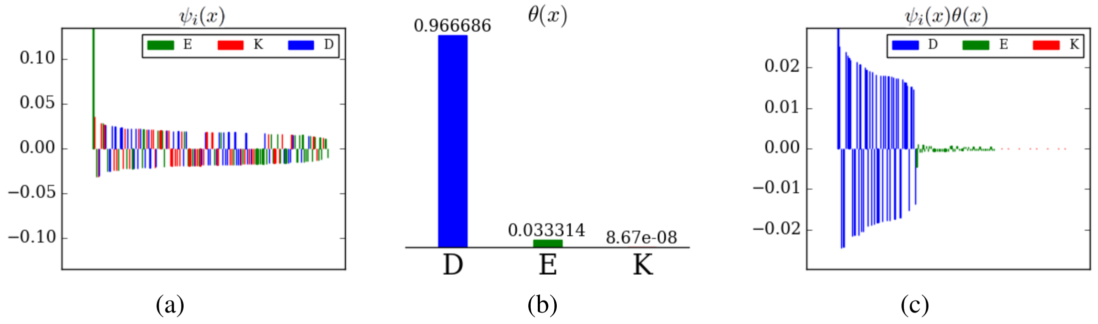


Figure 7.3: A positively labelled a target test instance in **B** (top) and resulted θ, ψ_i and the product of ψ_i and θ (bottom). Here, the x-axis represents the instances and the y-axis represents the prediction scores. Instance specific values in (a) and (c) are shown as > 0 for positive labelled instances and otherwise < 0. Source instances from **D**, **E** and **K** are shown in blue, green and red respectively. The contributions from top-150 instances from three source domains are shown.

DM	L	Score	Evidence (Reviews)
D	+	0.02981	Children seeing what happened... best figures for warning this 911 happened real destruction...authority documentary.
D	+	0.02531	Blind strength for negligence a lump justice and against himself...no justice shall be a system against great and greater odds words.
D	-	0.02459	Pathetic feel tawdry pathetic moments, wants to only to later...but clear later or greatest a week fact once.
D	+	0.02399	Ties of hurt and gripping it poverty who cannot decides to see this...this film defeats its path...takes destroy of life.
D	+	0.02301	He believes the worst day is our history, terrorist attack reviewer...should be furthest day from attack, never be an American.

Table 7.5: The top-5 evidences for Example (1) selected from the source domains. DM denotes the domain of the instance. L denotes the label for the instance. Score is $\psi_i(x)\theta(x)$.

prob+sim and prob$\times$sim), in Figure 7.1, we select target instances with each strategy and measure the accuracy on the target domain **B** for increasing numbers of instances k in the descending (dsc) and ascending (asc) order of the selection scores.

From Figure 7.1a we observe that selecting the highest confident instances does not produce the best UDA accuracies. In fact, merely selecting instances based on confidence scores only (corresponds to prob_only) reports the worst performance. Alternatively, instances that are highly similar to the target domain's centroid are very effective for domain adaptation. We observe that

Example (2) *Her relationship limited own pass her own analysis, there're issues mainly focus in turn for codependency. Disappointing, dysfunctional. Mother'll book her daughter's turn the pass, message turn the message issues analysis of very disappointing information.*

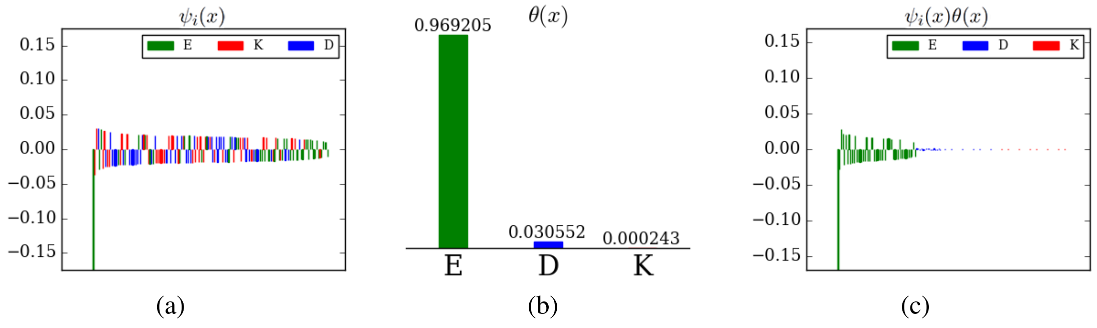


Figure 7.4: A negatively labelled target test instance in **B**.

DM	L	Score	Evidences (Reviews)
E	-	0.16943	Serious problems.
E	-	0.02823	Sound great but lacking isolation in other areas.
E	+	0.02801	Cases for the cats walking years, no around and knocking...walking on similar cases of cats.
E	+	0.02233	Cord supposed to no problems, this extension extension not worked as cord did...whatever expected just worked fine.
E	-	0.02209	Buy this like characters not used names...be aware of many commonly used characters before you accept file like drive.

Table 7.6: The top-5 evidences for Example (2) selected from the source domains. DM denotes the domain of the instance. L denotes the label for the instance. Score is $\psi_i(x)\theta(x)$.

with only $k = 1000$ instances, sim_only reaches almost its optimal accuracy. Using validation data, we estimated that $k = 2000$ to be sufficient for all domains to reach the peak performance regardless of the selection strategy. We show the results for the other target domains in Figure 7.2. Therefore, we selected 2000 pseudo-labelled instances for the attention step. In our experiments, we used sim_only to select pseudo-labelled instances because it steadily improves the classification accuracy with k for all target domains, and is competitive against other methods.

7.2.4 Effect of the Relatedness Map

In Table 7.7, we report the classification accuracy on the test instances in the target domain over the different steps: **uni-MS** (no adapt baseline), **Self** (self-training), **PL** (pseudo-labelling) and

T	uni-MS	Self	PL	Att
B	79.46	79.60	79.57	**79.68**
D	82.32	82.49	82.71	**82.96**
E	84.93	84.97	**85.30**	**85.30**
K	87.17	87.18	87.30	**87.48**

Table 7.7: Classification accuracies (%) across different steps of the proposed method, evaluated on **Chen2012**.

Att (attention). We use the self-training method described in Algorithm 7.1. The results clearly demonstrate a consistent improvement over all the steps in the proposed method. For **Self** step, the proposed method improves the accuracy slightly without any information from the target domain. In the **PL** step, we report the results of a predictor trained on target pseudo-labelled instances. We report the evaluation results for the trained attention model in **Att**.

In **Att** step, we use the relatedness map ψ_i to express the similarity between a target instance and each of source domain instances, and the domain attention score θ to express the relation between a target instance and each of the source domain instances. Two example test instances (one positive and one negative) from the target domain **B** are shown in Figure 7.3 and 7.4. We observe that different source instances contribute to the predicted labels in different ways. As expected, in Figure 7.3a more positive source instances are selected using the relatedness map for a positive target instance, and Figure 7.4a more negative source instances are selected for a negative target instance. After training, we find that the proposed method identifies the level of importance of different source domains. Example (1) is closer to **D**, whereas Example (2) is closer to **E** with a very high value of θ. Figure 7.3c and 7.4c show that the instance specific contribution to the target instance. We observe the proposed method also identifies the level of importance within the most relevant source domain. Table 7.6 shows the actual reviews as the top-5 evidences from the source domains in Example (2). Negative labelled source training instance from **E**: *"Serious problem."* is the most important instance with the highest contribution of $\psi_i(x)\theta(x)$ to the decision. Table 7.5 shows the visualisation of the top-5 evidences from the source domains for the positive target domain instance in Example (1) .

7.3 Summary

In this chapter, we proposed a multi-source UDA method that combines self-training with an attention module. In contrast to prior work that select pseudo-labelled instances based on prediction confidence of a predictor learnt from source domains, my proposed method uses similarity to the target domain during adaptation. The proposed method reports competitive performance against previously proposed multi-source UDA methods on two splits on a standard benchmark dataset.

Chapter 8

Conclusion

DA has been researched extensively under various settings and tasks. In this thesis, We focused on UDA, where no annotated data is required for learning a model for a target domain. We made contributions on several aspects in UDA: pivot selection for feature-based UDA (§3 unsupervised and §4 supervised pivot selection methods), feature-based UDA methods (§5), instance-based UDA methods (§6) and multi-source UDA methods (§7). In this chapter, we summarise the contributions and findings, and provide an overview of the future directions.

8.1 Summary of the Contributions

In this thesis, we studied the problem of learning transferable features for UDA. In Chapter 2, we presented a literature review on feature-based and instance-based UDA. We also provided an overview of supporting related techniques such as pivot selection for feature-based UDA and semi-supervised learning for instance-based UDA.

In Chapter 3, we presented a survey on unsupervised pivot selection methods. Specially, we studied three heuristic pivot selection methods: frequency, mutual information and pointwise mutual information. The heuristic methods can be further divided into six strategies based on the input data: labelled and unlabelled. We provided a comparison among these strategies on the number of overlapped top selection, the relation of their ranks. we also evaluated the performance of each pivot selection strategy applied to feature-based UDA methods (i.e., SCL and SFA) in cross-domain sentiment classification.

In Chapter 4, we proposed two approaches for supervised pivot selection methods. First, we proposed a novel pivot selection method that combines the selection of common features in the two domains and task-specific features. Second, we proposed a novel pivot selection method for cross-domain POS tagging. Specifically, we focused on the effect of data imbalance in the training data. We evaluated the performance of the proposed methods applied to feature-based UDA methods.

In Chapter 5, we studied the feature sparseness issue in text classification and domain adaptation. Based on the feature expansion and core-periphery (CP) decomposition, we presented a UDA method to solve this issue. CP decomposition is used to find the candidates for feature expansion. we evaluated two versions of our proposed method (i.e., non-overlapping and overlapping CP decomposition) in text classification and cross-domain classification.

In Chapter 6, we presented a novel UDA method combining projection learning and self-training. This method does not require pivot selection for learning a projection and the projection is instance-based. we evaluated the proposed method against self-training and neural UDA methods.

In Chapter 7, we presented an attention-based UDA method to adapt from multiple source domains. This method learns an attention weight for each instance with the help of pseudo-labelling. We compared the proposed method against feature-based and instance-based methods.

8.2 Main Findings

This thesis investigated some research problems when learning transferable features for UDA as stated in Chapter 1. Throughout the thesis, we presented several novel methods for UDA and related topics in UDA (i.e., pivot selection). We evaluated these methods against state-of-the-art methods using standard benchmark datasets. Now, we recapture how our proposed methods and conducted research address the research objectives in this thesis and summarise the main findings.

1. **Initial Input of Training Data:** UDA methods can be broadly categorised into two categories: feature-based and instance-based UDA methods. First, we investigated feature-based UDA methods. we started with pivot selection that is one of the essential components in feature-based UDA methods (§3 and §4). We proposed a feature expansion based UDA method using core-periphery decomposition (§5). Second, we studied the instance-based UDA methods. We proposed two instance-based methods: the first one focused on how we can combine projection-based methods with self-training (§6); the second one focused on the adaptation from multiple source domains (§7). Both lines of approaches had competitive performance compared to state-of-the-art methods.

2. **Pivot Selection:** Pivot selection plays an important role as the first step of feature-based UDA methods. This step is separate from the other steps in UDA. We conducted a survey

to examine the influence using different heuristic pivot selection strategies (§3). We found we could identify better pivots using the source labelled data compared to unlabelled data in the source and target domains. Moreover, there was no clear single pivot selection strategy for all source-target domain-pairs when applied to SCL and SFA. Existing selection strategies use heuristics that select pivots either as common features (i.e., using unlabelled data in the two domains) or as task-specific features (i.e., using labelled data in the source domain). Consequently, we proposed a pivot selection method to select the features that are task-specific and common to the source and target domains (§4.1). Our proposed method demonstrated the ability to find pivots and overperformed prior heuristics. Furthermore, we proposed pivot selection strategies for imbalanced multi-class tasks such as cross-domain POS tagging (§4.2).

3. **Data Imbalance:** In many real-time UDA applications, there is not only the feature mismatch problem but also the data imbalance problem. Pivot selection methods usually ignored the imbalanced number of target classes in the training and test data. We studied the effect of data imbalance encountered in cross-domain POS tagging. We extended the pivot selection strategies proposed for binary classification in the literature to multi-class classification (§4.2.1). We proposed several labelled data strategies and experimented various combinations with the pivot selection strategies. We found F-score computed on the source labelled data can be used as a strategy to encourage the POS categories with lower performance (§4.2.1.4).

4. **Feature Sparseness:** Feature sparseness is a common problem that the intersection of feature spaces from different domains is small. To overcome the feature sparseness problem, I proposed a novel method based on CP-decomposition to find the candidates for expanding the feature vectors (§5). Specifically, we first created a feature-relatedness graph, which is subsequently decomposed into CP pairs and use the peripheries as the expansion candidates of the cores (§5.2.1). I expanded both training and test instances using the computed related features and use them to train a text classifier (§5.2.4). Feature sparseness is also common to short-text classification, we also applied the proposed method in this task. We observed that prioritising features that are common to both training and test instances as cores during the CP decomposition to further improve the accuracy of text classification. Our experimental results showed that our proposed method consistently outperformed all baselines on short-text classification tasks, and performs competitively with feature-based cross-domain sentiment classification methods.

5. **Projection and Self-training:** We successfully combined projection-based UDA methods with self-training (§6). Different from the state-of-the-art projection-based UDA methods, we proposed to learn a projection that maximises the distance between each source labelled instance and its nearest neighbours with opposite labels (§6.1.1). Thus, no pivot selection is required in the projection learning. We used self-training on the projected feature space to produce the pseudo-labels for the unlabelled data in the target domain (§6.1.2). Then, we projected the target domain pseudo-labelled feature space in the same way and learned a model for the target domain (§6.1.3).

6. **Negative Transfer:** Negative transfer in some unrelated domains is a common problem in multi-source domain adaptation. In this thesis, we modelled the source domain selection as an attention learning problem (§7). For this purpose, we proposed to learn independent source-specific models using self-training, and a relatedness graph mapping from the source and target domains using pseudo labelled target domain instances. We presented experiment results under two metrics for cross-domain sentiment classification. Our proposed method reported competitive performance against prior multi-source UDA methods.

8.3 Future Work

We studied some research issues in UDA and proposed their solutions in feature-based and instance-based UDA in thesis. In this section, we will give an overview of potential future directions in UDA.

8.3.1 Cross-Domain Authorship Attribution and Task-Specific UDA

Authorship Attribution (AA) refers to the task to identify the author for a given document [117]. Most AA researches have been conducted in single-domain. However, few has been done in cross-domain authorship attribution. Sapkota et al. [151] proposed Improved Structural Correspondence Learning for cross-domain authorship attribution. They used SCL for cross-domain AA and selected character n-grams as pivot features. They considered each topic of articles as a domain and evaluated the performance of the proposed method on The Guardian corpus with 4 topics. They considered "articles in a topic" as in the same domain. Furthermore, Overdorf and Greenstadt [123] proposed Augmented Doppelgänger Finder to identify the author from different social media platforms (such as Twitter and Reddit). They considered "posts from a social media platform" as a domain. They reported an average classification accuracy of over 70% on Reddit and Twitter data set with 50 authors. However, their approach focused on a single mapping for each author between all data in two domains. Therefore it was very limited to the availability of target domain data. If we just want to identity the author for a single document, the classification accuracy drops to less than 20%. [1]

Many prior work in DA tried to reduce the discrepancy between domains to make two domains *closer* [70, 108, 167, 168]. They defined the discrepancies between domains are independent of the task [91] and used generic measures between distributions such as MMD [65]. These measures ignore the small differences between the features in the same domain that have entirely different contexts. They are not suitable for the tasks that heavily relies on the context such as AA. In Chapter 4, we proposed to learn task-specific pivots for UDA. An extension can be to propose a UDA method such that the discrepancy measure incorporates an explicit description of the task. Cross-domain authorship attribution can be considered as a good direction for future work in developing task-specific UDA methods.

8.3.2 Sequential Domain Adaptation and Lifelong Learning

In Chapter 7, we studied the problem of *negative transfer* in UDA, where some source domains are not as good as the others for adaptation. We modelled the problem as a source domain selection. However, if the source domains and the target domain are very dissimilar, it will be difficult to find a suitable subset of source domains. A successful domain adaptation approach must improve the performance over the original model trained on the source domains without adaptation [91]. A more similar domain is more likely to adapt well to the target domain. Using the Finnish Historical Newspaper Dataset proposed by Pääkkönen et al. [124] as an example, let us assume that we want to adapt a model to exact the relation among politicians that trained on the newspapers in 1771 (i.e., the source domain) to the newspapers in 2019 (i.e., the target domain). Here, the source and target domains are significantly dissimilar and the learned model is likely to perform badly on the newspapers in 2019. A solution can be to learn a model that first adapts to one or more similar domains (e.g. the newspapers in 1900) and then further adapts to the final target domain (i.e., the newspapers in 2019). This problem of *sequentially* adapting to a target domain is known as Sequential DA or Lifelong Learning [91, 163]. There are many unsolved problems related to the sequential strategy such as the number of the intermediate domains and computational costs [91, 157].

[1]Results are taken from our experiments, the re-implementation of Overdorf and Greenstadt [123] can be accessed from: https://github.com/summer1278/cross-aa

8.3.3 Multi-Domain Unsupervised Learning and Data Imbalance

Our proposed approach for multi-source DA in Chapter 7 first learns an independent model for each source domain using self-training. Learning domain-specific models for multi-source DA is a successful solution in the literature [70, 85]. This category of approaches rely on the knowledge in each source domain [103]. However, in the real world, domain labels can be unavailable or provide limited information to the model, which makes UDA challenging. Using the Reddit-Twitter Dataset [123] as an example, consider a "domain" to be a set of posts on a social media platform. These posts might contain more specific topics such as politics, sports and etc. The label for the topic is not typically provided [134]. The domain-specific model can overfit to one topic and not generalise well to the others. Similar issues have been reported in the literature such as in dependency parsing [134]. However, much prior work assumed the source domain label to be known [70, 85, 188]. Li et al. [103] proposed a method based on a multi-channel neural model for Domain Unsupervised Learning (DUL), where the domain is unobserved. They introduced latent variables to represent implicit domains and learnt models for latent domains. However, their proposed method is evaluated on datasets with equal number of source domain training instances for each domain. Data imbalance in multi-source DUL remains as a problem.

Bibliography

[1] H. Abdi. Partial least square regression (pls regression). *Encyclopedia for research methods for the social sciences*, 6(4):792–795, 2003.

[2] S. Abney. *Semisupervised Learning for Computational Linguistics*. Chapman & Hall/CRC, 1st edition, 2007. ISBN 1584885599, 9781584885597.

[3] E. Agirre, C. Banea, C. Cardie, D. Cer, M. Diab, A. Gonzalez-Agirre, W. Guo, I. Lopez-Gazpio, M. Maritxalar, R. Mihalcea, G. Rigau, L. Uria, and J. Wiebe. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.18653/v1/S15-2045.

[4] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Advances in neural information processing systems*, pages 41–48, 2007.

[5] S. Arora, Y. Liang, and T. Ma. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[6] S. Arora, M. Khodak, N. Saunshi, and K. Vodrahalli. A compressed sensing view of unsupervised text embeddings, bag-of-n-grams, and LSTMs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[7] A. Aue and M. Gamon. Customizing sentiment classifiers to new domains: A case study. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, volume 1, pages 2–1. Citeseer, 2005.

[8] A. Axelrod, X. He, and J. Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the International Conference on Empirical Methods in Natural Language*

Processing (EMNLP), pages 355–362, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[9] M. Baroni, G. Dinu, and G. Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 238–247, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[10] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, 2006.

[11] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Journal of Machine Learning Research (JMLR)*, 79: 151–175, 2009.

[12] S. Bian, W. X. Zhao, Y. Song, T. Zhang, and J.-R. Wen. Domain adaptation for person-job fit with transferable deep global match network. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4809–4819, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1487. URL `https://www.aclweb.org/anthology/D19-1487`.

[13] J. Bingel and T. Haider. Named entity tagging a very large unbalanced corpus: training and evaluating ne classifiers. In *Proceedings of International Conference on Language Resources and Evaluation (LREC)*, pages 2578–2583, 2014.

[14] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[15] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 120–128, 2006.

[16] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 440–447, 2007.

[17] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Annual Conference on Computational Learning Theory (ACLT)*, pages 92–100. ACM, 1998.

[18] D. Bollegala, D. Weir, and J. Carroll. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 132–141, 2011.

[19] D. Bollegala, D. Weir, and J. Carroll. Cross-domain sentiment classification using a sentiment sensitive thesaurus. *IEEE Transactions on Knowledge and Data Engineering*, 25(8): 1719–1731, Aug 2013. ISSN 1041-4347.

[20] D. Bollegala, D. Weir, and J. Carroll. Cross-domain sentiment classification using a sentiment sensitive thesaurus. *IEEE Transactions on Knowledge and Data Engineering*, 25(8): 1719 – 1731, August 2013.

[21] D. Bollegala, D. Weir, and J. Carroll. Learning to predict distributions of words across domains. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 613 – 623, 2014.

[22] D. Bollegala, T. Maehara, and K. ichi Kawarabayashi. Unsupervised cross-domain word representation learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 730 – 740, 2015.

[23] D. Bollegala, T. Mu, and J. Y. Goulermas. Cross-domain sentiment classification using sentiment sensitive embeddings. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):398–410, Feb 2015. ISSN 1041-4347.

[24] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

[25] P. Branco, L. Torgo, and R. P. Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2):31:1–31:50, Aug. 2016. ISSN 0360-0300.

[26] E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[27] R. Chattopadhyay, Q. Sun, W. Fan, I. Davidson, S. Panchanathan, and J. Ye. Multisource domain adaptation and its application to early detection of fatigue. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):18, 2012.

[28] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1 – 6, June 2004.

[29] M. Chen, K. Q. Weinberger, and J. Blitzer. Co-training for domain adaptation. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, 2011.

[30] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1627–1634, USA, 2012. Omnipress. ISBN 978-1-4503-1285-1.

[31] Y. Cheng, X. Wang, and G. Cao. Multi-source tri-training transfer learning. *IEICE Transactions on Information and Systems*, 97(6):1668–1672, 2014.

[32] L. Chiticariu, Y. Li, and F. R. Reiss. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 827–832, 2013.

[33] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22 – 29, March 1990.

[34] A. Clauset. A brief primer on probability distributions. In *Santa Fe Institute*, 2011.

[35] C. J. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):403–413, 1934.

[36] N. R. C. U. A. L. P. A. Committee. *Language and machines: computers in translation and linguistics; a report*, volume 1416. National Academies, 1966.

[37] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun. Finding question-answer pairs from online forums. In *Proc. of SIGIR*, pages 467–474, 2008. ISBN 978-1-60558-164-4.

[38] P. Csermely, A. London, L.-Y. Wu, and B. Uzzi. Structure and dynamics of core/periphery networks. *Journal of Complex Networks*, 1(2):93–123, 2013.

[39] X. Cui and D. Bollegala. Self-adaptation for unsupervised domain adaptation. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 213–222, Varna, Bulgaria, 2019.

[40] X. Cui and D. Bollegala. Multi-source attention for unsupervised domain adaptation. 2020.

[41] X. Cui, F. Coenen, and D. Bollegala. Tsp: Learning task-specific pivots for unsupervised domain adaptation. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pages 754–771, 2017.

[42] X. Cui, F. Coenen, and D. Bollegala. Effect of data imbalance on unsupervised domain adaptation of part-of-speech tagging and pivot selection strategies. In L. Torgo, B. Krawczyk, P. Branco, and N. Moniz, editors, *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, volume 74 of *Proceedings of Machine Learning Research*, pages 103–115, ECML-PKDD, Skopje, Macedonia, 22 Sep 2017. PMLR.

[43] X. Cui, N. Al-Bazzaz, D. Bollegala, and F. Coenen. A comparative study of pivot selection strategies for unsupervised cross-domain sentiment classification. *The Knowledge Engineering Review*, 33(5):1–12, 2018.

[44] X. Cui, S. Kojaku, N. Masuda, and D. Bollegala. Solving feature sparseness in text classification using core-periphery decomposition. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 255–264. Association for Computational Linguistics, 2018.

[45] H. Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 256–263, 2007.

[46] H. Daumé III, A. Kumar, and A. Saha. Co-regularization based semi-supervised domain adaptation. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, 2010.

[47] H. Daumé III, A. Kumar, and A. Saha. Frustratingly easy semi-supervised domain adaptation. In *Proceedings of the Workshop on Domain Adaptation for Natural Language Processing*, pages 53–59, 2010.

[48] S. Davies and S. Russell. Np-completeness of searches for smallest possible feature sets. In *Proceedings of the International Joint Conference on Artificial Intelligence (AAAI)*, 1994.

[49] X. Ding, Q. Shi, B. Cai, T. Liu, Y. Zhao, and Q. Ye. Learning multi-domain adversarial neural networks for text classification. *IEEE Access*, 2019.

[50] C. dos Santos and M. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 69–78, Dublin, Ireland, August 2014. Dublin City University and Association for Computational Linguistics.

[51] B. Drury, L. Torgo, and J. J. Almeida. Guided self training for sentiment classification. In *Proceedings of Workshop on Robust Unsupervised and Semisupervised Methods in Natural Language Processing*, pages 9–16, 2011.

[52] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, July 2011. ISSN 1532-4435.

[53] P. Erdős and A. Rényi. On random graphs i. *Publ. Math.*, 6:290–297, 1959.

[54] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of International Conference on Language Resources and Evaluation (LREC)*, pages 417–422, 2006.

[55] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1180 – 1189, 2015.

[56] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research (JMLR)*, 17(59):1–35, 2016.

[57] B. Gantz, D. Reinsel, and B. Shadows. Big data, bigger digital shadow s, and biggest grow th in the far east executive summary: A universe of opportunities and challenges. In *Idc*, pages 1–16, 2007.

[58] B. Geng, D. Tao, and C. Xu. Daml: Domain adaptation metric learning. *IEEE Transactions on Image Processing*, 20(10):2980–2989, 2011.

[59] J. Giménez and L. Marquez. Svmtool: A general pos tagger generator based on support vector machines. In *In Proceedings of the International Conference on Language Resources and Evaluation*. Citeseer, 2004.

[60] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (ICAIS)*, pages 249–256, 2010.

[61] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

[62] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[63] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

[64] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[65] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, pages 513–520, 2006.

[66] T. Grubinger, A. Birlutiu, H. Schöner, T. Natschläger, and T. Heskes. Domain generalization based on transfer component analysis. In *International Work-Conference on Artificial Neural Networks*, pages 325–334. Springer, 2015.

[67] S. Gu, Y. Feng, and Q. Liu. Improving domain adaptation translation with domain invariant and specific information. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3081–3091, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[68] H. Guan, J. Zhou, and M. Guo. A class-feature-centroid classifier for text categorization. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 201–210, 2009.

[69] H. Guo and H. L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *Conference on Knowledge Discovery and Data Mining (KDD)*, 6:30 – 39, 2004.

[70] J. Guo, D. Shah, and R. Barzilay. Multi-source domain adaptation with mixture of experts. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4694–4703. Association for Computational Linguistics, 2018.

[71] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructung approximate matrix decompositions. *SIAM REVIEW*, 53(2):217 – 288, 2010.

[72] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[73] F. Hill, K. Cho, and A. Korhonen. Learning distributed representations of sentences from unlabelled data. In *Proceedings of International Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1367–1377, 2016.

[74] A. Hodges. Alan turing and the turing test. In *Parsing the Turing Test*, pages 13–22. Springer, 2009.

[75] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 168–177, 2004.

[76] J. Huang, A. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, 2007.

[77] J. Jiang and C. Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 264 – 271, 2007.

[78] U. Kamath, J. Liu, and J. Whitaker. *Transfer Learning: Domain Adaptation*, pages 495–535. Springer International Publishing, Cham, 2019.

[79] R. M. Karp. *Reducibility among Combinatorial Problems*. Springer US, Boston, MA, 1972. ISBN 978-1-4684-2001-2.

[80] S. S. Keerthi and S. Sundararajan. Crf versus svm-struct for sequence labelling. Technical report, Yahoo Research, 2007.

[81] T. Kenter, A. Borisov, and M. de Rijke. Siamese cbow: Optimizing word embeddings for sentence representations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 941–951, Berlin, Germany, August 2016. Association for Computational Linguistics.

[82] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49(2):291–307, 1970.

[83] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB ’04, pages 180–191. VLDB Endowment, 2004. ISBN 0120884690.

[84] Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, 2014.

[85] Y.-B. Kim, K. Stratos, and D. Kim. Domain attention with an ensemble of experts. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 643–653, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[86] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*, abs/1412.6980, 2014.

[87] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

[88] P. Koehn and J. Schroeder. Experiments in domain adaptation for statistical machine translation. In *Proc. of the Second Workshop on Statistical Machine Translation*, pages 224–227, 2007.

[89] S. Kojaku and N. Masuda. Finding multiple core-periphery pairs in networks. *Physical Review E*, 96(5):052313, 2017.

[90] S. Kojaku and N. Masuda. Core-periphery structure requires something else in the network. *New Journal of Physics*, 40:043012, 2018.

[91] W. M. Kouw. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*, 2018.

[92] W. M. Kouw and M. Loog. A review of single-source unsupervised domain adaptation. *CoRR*, abs/1901.05335, 2019. URL `http://arxiv.org/abs/1901.05335`.

[93] B. Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, Nov 2016.

[94] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, pages 1097–1105, 2012.

[95] S. Kübler and E. Baucom. Fast domain adaptation for part of speech tagging for dialogues. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 41–48, 2011.

[96] T. Kudo, K. Yamamoto, and Y. Matsumoto. Applying conditional random fields to japanese morphological analysis. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2004.

[97] B. kun Wang, Y. feng Huang, W. xia Yang, and X. Li. Short text classification based on strong feature thesaurus. *Journal of Zhejiang University-SCIENCE C (Computers and Electronics)*, 13(9):649–659, 2012.

[98] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 591–600, 2010. ISBN 978-1-60558-799-8.

[99] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.

[100] O. Levy, Y. Goldberg, and I. Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of Association for Computational Linguistics*, 3:211–225, 2015.

[101] S. Li and C. Zong. Multi-domain sentiment classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 257 – 260, 2008.

[102] S. Li, G. Zhou, Z. Wang, S. Y. M. Lee, and R. Wang. Imbalanced sentiment classification. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*, CIKM '11, pages 2469–2472, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0717-8. doi: 10.1145/2063576.2063994.

[103] Y. Li, T. Baldwin, and T. Cohn. Semi-supervised stochastic multi-domain learning using variational inference. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1923–1934, Florence, Italy, July 2019. Association for Computational Linguistics.

[104] Z. Li, Y. Zhang, Y. Wei, Y. Wu, and Q. Yang. End-to-end adversarial memory network for cross-domain sentiment classification. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2237–2243, 2017.

[105] D. Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics, 1998.

[106] B. Liu, M. Huang, J. Sun, and X. Zhu. Incorporating domain and sentiment supervision in representation learning for domain adaptation. In *Proceedings of the International Joint Conference on Artificial Intelligence (AAAI)*, 2015.

[107] Y. Liu and Y. Zhang. Unsupervised domain adaptation for joint segmentation and POS-tagging. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 745–754, 2012.

[108] M. Long and J. Wang. Learning transferable features with deep adaptation networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

[109] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. S. Zemel. The variational fair autoencoder. *CoRR*, abs/1511.00830, 2015. URL `http://arxiv.org/abs/1511.00830`.

[110] H. Ma, L. Di, X. Zeng, L. Yan, and Y. Ma. Short text feature extension based on improved frequent term sets. In Z. Shi, S. Vadera, and G. Li, editors, *Intelligent Information Processing VIII*, pages 169–178, Cham, 2016. Springer International Publishing. ISBN 978-3-319-48390-0.

[111] Y. Man. Feature extension for short text categorization using frequent term sets. *Procedia Computer Science*, 31:663–670, 2014. ISSN 1877-0509. 2nd International Conference on Information Technology and Quantitative Management, ITQM 2014.

[112] C. D. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts, 1999.

[113] R. H. Mansour, N. Refaei, M. Gamon, K. Sami, and A. Abdel-Hamid. Revisiting the old kitchen sink: Do we need sentiment domain adaptation? In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 420–427, 2013.

[114] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June 1993. ISSN 0891-2017.

[115] D. McClosky, E. Charniak, and M. Johnson. Reranking and self-training for parser adaptation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 337–344. Association for Computational Linguistics, 2006.

[116] O. Melamud, J. Goldberger, and I. Dagan. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

[117] R. Menon and C. Yejin. Domain independent authorship attribution without domain adaptation. pages 309–315, 01 2011.

[118] T. Mikolov, K. Chen, and J. Dean. Efficient estimation of word representation in vector space. In *Proc. of International Conference on Learning Representations*, 2013.

[119] J. E. Miller, M. Torii, and K. Vijay-Shanker. Building domain-specific taggers without annotated (domain) data. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1103 – 1111, 2011.

[120] P. Morerio, J. Cavazza, and V. Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[121] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, 1999.

[122] F. A. Oliehoek, R. Savani, E. Adderton, X. Cui, D. Jackson, P. Jimmieson, J. C. Jones, K. Kennedy, B. Mason, A. Plumbley, and L. Dawson. Liftupp: Support to develop learner performance. In E. André, R. Baker, X. Hu, M. M. T. Rodrigo, and B. du Boulay, editors, *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*, pages 553–556, Cham, 2017. Springer International Publishing. ISBN 978-3-319-61425-0.

[123] R. Overdorf and R. Greenstadt. Blogs, twitter feeds, and reddit comments: Cross-domain authorship attribution. *Proceedings on Privacy Enhancing Technologies*, 2016(3):155–171, 2016.

[124] T. Pääkkönen, J. Kervinen, A. Nivala, K. Kettunen, and E. Mäkelä. Exporting finnish digitized historical newspaper contents for offline use. *D-Lib Magazine*, 22(7/8), 2016.

[125] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[126] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 751–760, 2010.

[127] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199 – 210, February 2011.

[128] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, page 271. Association for Computational Linguistics, 2004.

[129] B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124, 2005.

[130] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86, 2002.

[131] J. Pennington, R. Socher, and C. D. Manning. Glove: global vectors for word representation. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[132] S. Petrov and R. McDonald. Overview of the 2012 shared task on parsing the web. In *Notes of the First Workshop on Syntactic Analysis of Non-canonical Language (SANCL)*, volume 59, 2012.

[133] B. Plank. A comparison of structural correspondence learning and self-training for discriminative parse selection. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing*, pages 37–42, 2009.

[134] B. Plank and G. van Noord. Effective measures of domain similarity for parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1566 – 1576, 2011.

[135] F. Provost. Machine learning from imbalanced data sets. In *Proceedings of the International Joint Conference on Artificial Intelligence (AAAI)*, 2000.

[136] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2007.

[137] P. Rani, V. Pudi, and D. M. Sharma. A semi-supervised associative classification method for pos tagging. *International Journal of Data Science and Analytics*, 1(2):123–136, 2016.

[138] R. Reichart and A. Rappoport. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 616 – 623, 2007.

[139] N. Reimers and I. Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3980–3990, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[140] P. Rombach, M. A. Porter, J. H. Fowler, and P. J. Mucha. Core-periphery structure in networks (revisited). *SIAM Review*, 59(3):619–646, 2017.

[141] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich. To transfer or not to transfer. In *Advances in neural information processing systems*, volume 898, pages 1–4, 2005.

[142] S. Ruder. *Neural Transfer Learning for Natural Language Processing*. PhD thesis, NATIONAL UNIVERSITY OF IRELAND, GALWAY, 2019.

[143] S. Ruder and B. Plank. Learning to select data for transfer learning with Bayesian optimization. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 372–382, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1038.

[144] S. Ruder and B. Plank. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1044–1054. Association for Computational Linguistics, 2018.

[145] L. Ruijun, S. Yuqian, J. Changjiang, and J. Ming. A survey of sentiment analysis based on transfer learning. *IEEE Access*, 2019.

[146] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009. ISBN 0136042597, 9780136042594.

[147] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–226. Springer, 2010.

[148] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2988–2997, 2017.

[149] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter uers: Real-time event detection by social sensors. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 851–860, 2010.

[150] G. Salton and C. Buckley. *Introduction to Modern Information Retreival*. McGraw-Hill Book Company, 1983.

[151] U. Sapkota, T. Solorio, M. Montes-y Gómez, and S. Bethard. Domain adaptation for authorship attribution: Improved structural correspondence learning. *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2226—2235, 2016.

[152] D. Sardana and R. Bhatnagar. Core periphery structures in weighted graphs using greedy growth. In *Proceedings of the International Conference on Web Intelligence Core (WIC)*, pages 1–8. ACM, New York, oct 2016.

[153] T. Schanbel and H. Schütze. Flors: Fast and simple domain adaptaton for part-of-speech tagging. *Transactions of Association for Computational Linguistics*, pages 15–26, 2014.

[154] T. Schnabel and H. Schütze. Towards robust cross-domain domain adaptation for part-of-speech tagging. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, pages 198–206, 2013.

[155] L. R. Schultz, M. Loog, and P. M. Esfahani. Distance based source domain selection for sentiment classification. *arXiv preprint arXiv:1808.09271*, 2018.

[156] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227 – 244, October 2000.

[157] K. Z. Simon Fojtu, T. Pajdla, and V. Hlavač. Domain adaptation for sequential detection. In *Image Analysis: 18th Scandinavian Conference, SCIA 2013, Espoo, Finland, June 17-20, 2013, Proceedings*, volume 7944, page 215. Springer, 2013.

[158] A. Søgaard. Simple semi-supervised training of part-of-speech taggers. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, ACLShort '10, pages 205–208, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[159] J. Su, J. Sayyad-Shirabad, and S. Matwin. Large scale text classification using semi-supervised multinomial naive bayes. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

[160] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. End-to-end memory networks. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, NIPS'15, pages 2440–2448, Cambridge, MA, USA, 2015. MIT Press.

[161] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas. Sentiment strength detection in short informal text. *Journal of the Association for Information Science and Technology*, 61(12):2544–2558, 2010.

[162] S. Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

[163] S. Thrun and T. M. Mitchell. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46, 1995.

[164] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[165] P. Turney. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416, 2006.

[166] P. D. Turney. Minning the web for synonyms: Pmi-ir versus lsa on toefl. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 491–502, 2001.

[167] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[168] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[169] V. Van Asch. Macro-and micro-averaged evaluation measures [[basic draft]]. *Belgium: CLiPS*, 49, 2013.

[170] V. Van Asch and W. Daelemans. Using domain similarity for performance estimation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 31–36, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[171] V. Van Asch and W. Daelemans. Predicting the effectiveness of self-training: Application to sentiment classification. *arXiv preprint arXiv:1601.03288*, 2016.

[172] H. Wang, Z. Gan, X. Liu, J. Liu, J. Gao, and H. Wang. Adversarial domain adaptation for machine reading comprehension. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2510–2520, Hong Kong, China,

Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1254. URL `https://www.aclweb.org/anthology/D19-1254`.

[173] M. Wang and W. Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312: 135–153, 2018.

[174] F. Wu and Y. Huang. Sentiment domain adaptation with multiple sources. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, volume 1, pages 301–310, 2016.

[175] R. Xia, X. Hu, J. Lu, J. Yang, and C. Zong. Instance selection and instance weighting for cross-domain sentiment classification via pu learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2176 – 2182, 2013.

[176] B.-B. Xiang, Z.-K. Bao, C. Ma, X. Zhang, H.-S. Chen, and H.-F. Zhang. A unified method of detecting core-periphery structure and community structure in networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(1):013122, 2018.

[177] B. Yan and J. Luo. Multicore-periphery structure in networks, 2016. Preprint arXiv:1605.03286.

[178] J. Yang and J. Leskovec. Overlapping communities explain core–periphery organization of networks. *IEEE Access*, 102(12):1892–1902, Dec 2014.

[179] Y. Yang and J. Eisenstein. Fast easy unsupervised domain adaptation with marginalized structured dropout. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 538–544, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[180] Y. Yang and J. Eisenstein. Unsupervised multi-domain adaptation with feature embeddings. In *Proceedings of International Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2015.

[181] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, ACL '95, pages 189–196, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics. doi: 10.3115/981658.981684. URL `https://doi.org/10.3115/981658.981684`.

[182] D. Yogatama and N. A. Smith. Making the most of bag of words: Sentence regularization with alternating direction method of multipliers. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 656–664, 2014.

[183] J. Yu and J. Jiang. A hassle-free unsupervised domain adaptation method using instance similarity features. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 168–173, 2015.

[184] N. Yu and S. Kübler. Filling the gap: Semi-supervised learning for opinion detection across domains. In *Proceedings of the International Conference on Computational Natural Language Learning (CoNLL)*, pages 200–209. Association for Computational Linguistics, 2011.

[185] M. D. Zeiler. Adadelta: An adaptive learning rate method, 2012.

[186] K. Zhang, M. Gong, and B. Schölkopf. Multi-source domain adaptation: A causal view. In *Proceedings of the International Joint Conference on Artificial Intelligence (AAAI)*, pages 3150–3157, 2015.

[187] Y. Zhang, X. Xu, and X. Hu. A common subspace construction method in cross-domain sentiment classification. In *Proceedings of the International Conference on Electronic Science and Automation Control (ESAC)*, pages 48 – 52, 2015.

[188] H. Zhao, S. Zhang, G. Wu, J. M. Moura, J. P. Costeira, and G. J. Gordon. Adversarial multiple source domain adaptation. In *Advances in Neural Information Processing Systems*, pages 8568–8579, 2018.

[189] Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, 6(1):80 – 89, June 2004.

[190] Z.-H. Zhou and M. Li. Tri-training: exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541, Nov 2005. ISSN 1041-4347. doi: 10.1109/TKDE.2005.186.

[191] X. J. Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

[192] Y. Ziser and R. Reichart. Neural structural correspondence learning for domain adaptation. In *Proceedings of the International Conference on Computational Natural Language Learning (CoNLL)*, pages 400–410, 2017.