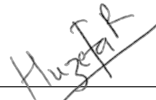


UNSUPERVISED CROSS-DOMAIN AND CROSS-LINGUAL METHODS FOR
TEXT CLASSIFICATION, SLOT-FILLING, AND QUESTION-ANSWERING

by

Jitin Krishnan
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computer Science


Committee:



Dr. Huzefa Rangwala, Dissertation Director



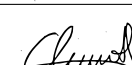
Dr. Hemant Purohit, Dissertation Co-Director



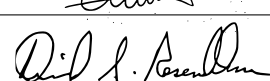
Dr. Jessica Lin, Committee Member

Carlotta Domeniconi

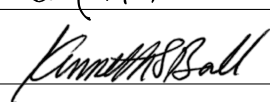
Dr. Carlotta Domeniconi, Committee Member



Dr. Antonios Anastasopoulos, Committee Member



Dr. David Rosenblum, Department Chair



Dr. Kenneth Ball, Dean, Volgenau School
of Engineering

Date: 04/23/2021

Spring Semester 2021
George Mason University
Fairfax, VA

Unsupervised Cross-Domain and Cross-Lingual Methods for Text Classification,
Slot-Filling, and Question-Answering

A dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy at George Mason University

By

Jitin Krishnan
Master of Science
George Mason University, 2016
Bachelor of Science
University of Virginia, 2012

Director: Dr. Huzefa Rangwala, Professor
Department of Computer Science

Spring Semester 2021
George Mason University
Fairfax, VA

Copyright © 2021 by Jitin Krishnan
All Rights Reserved

Dedication

This dissertation is dedicated to my parents, Mr. Balakrishnan Chellaton and Dr. Vanaja Taliyl, to my wife, Chandini Narayan, and to my younger brother, Hrishikesh. Without their love, support, and encouragement this work would not have been possible.

Acknowledgments

I thank my advisors Dr. Huzefa Rangwala and Dr. Hemant Purohit for the amazing guidance they have given me throughout my PhD life. I thank them for their patience, hours of discussions, and motivating me to push research boundaries. I would also like to express my gratitude towards Dr. Antonios Anastasopoulos for his exceptional guidance with NLP technical concepts. I would also like to thank my additional committee members, Dr. Jessica Lin and Dr. Carlotta Domeniconi for their valuable guidance and support.

I thank the U.S. National Science Foundation grants IIS-1815459 and IIS-1657379 for partially supporting my research. I thank my labmates Yasas Senarath and Rahul Pandey for COVID-19 data annotation along with the Montgomery County CERT volunteers led by Steve Peterson. I thank Chandini Narayan, Sujay Das, and Arun Krishna for data annotation in Malayalam and Hindi. I also thank GMU’s ARGO team and their support in running my experiments.

I thank Sneha Mehta, Raj Patel, Angeela Acharya, Huangxin Wang, Qian Hu, and all of my labmates and peers for valuable research discussions and advice they have given along the way.

I thank Patrick Coronado, Ming Sun, Ruoyu Li, and Nathan Hurst for showing me that AI/Machine Learning can be fun and also is very impactful for our society at large.

Finally, I thank my entire family and friends for their constant support and encouragement. I will forever be grateful.

Table of Contents

| | Page |
|------------------------------------------------------------------------------------|------|
| List of Tables | viii |
| List of Figures | x |
| List of Abbreviations | xii |
| Abstract | xiv |
| 1 Introduction | 1 |
| 1.1 Motivation | 3 |
| 1.2 Problem Statement | 4 |
| 1.3 Contributions | 5 |
| 1.4 Dissertation Outline | 8 |
| 2 Background & Related Work | 9 |
| 2.1 Notations | 9 |
| 2.2 Cross-Domain | 10 |
| 2.2.1 Text Classification | 10 |
| 2.2.2 Application in Crisis Management | 13 |
| 2.2.3 Generative Question Answering | 16 |
| 2.3 Cross-Lingual | 17 |
| 2.3.1 Text Classification & Application in Crisis Management | 17 |
| 2.3.2 Intent Prediction and Slot-Filling | 19 |
| 2.3.3 Transliterated Text Classification | 23 |
| 3 Diversity-Based Generalization for Unsupervised Cross-Domain Text Classification | 27 |
| 3.1 Summary | 27 |
| 3.2 Methodology | 28 |
| 3.2.1 Problem Definition | 28 |
| 3.2.2 Models & Concepts | 29 |
| 3.3 Experimental Evaluation | 35 |
| 3.3.1 Datasets | 35 |
| 3.3.2 Experiments | 36 |
| 3.3.3 Baselines | 37 |
| 3.4 Results & Discussion | 40 |

| | | |
|-------|---------------------------------------------------------------------------------------------------------------------|----|
| 3.5 | Key Takeaways | 44 |
| 4 | Unsupervised and Interpretable Domain Adaptation: Application in Crisis Management | 45 |
| 4.1 | Summary | 45 |
| 4.2 | Methodology | 47 |
| 4.2.1 | Problem Definition | 47 |
| 4.2.2 | Models & Concepts | 48 |
| 4.3 | Experimental Evaluation | 52 |
| 4.3.1 | Datasets | 52 |
| 4.3.2 | Baselines | 54 |
| 4.3.3 | Experiments | 54 |
| 4.4 | Results & Discussion | 56 |
| 4.5 | Key Takeaways | 64 |
| 5 | Domain Adversarial Masking and Regeneration for Cross-Domain Generative Question Answering | 66 |
| 5.1 | Summary | 66 |
| 5.2 | Methodology | 66 |
| 5.2.1 | Problem Definition | 66 |
| 5.2.2 | Models & Concepts | 67 |
| 5.3 | Experimental Evaluation | 70 |
| 5.3.1 | Datasets | 70 |
| 5.3.2 | Experiments | 71 |
| 5.4 | Results & Discussion | 72 |
| 5.5 | Key Takeaways | 73 |
| 6 | Attention Realignment and Pseudo-Labeling for Interpretable Cross-Lingual Classification of Crisis Tweets | 74 |
| 6.1 | Summary | 74 |
| 6.2 | Methodology | 75 |
| 6.2.1 | Problem Definition | 75 |
| 6.2.2 | Models & Concepts | 76 |
| 6.3 | Experimental Evaluation | 80 |
| 6.3.1 | Datasets | 80 |
| 6.3.2 | Experiments | 80 |
| 6.4 | Results & Discussion | 81 |
| 6.5 | Key Takeaways | 83 |

| | | |
|-------|------------------------------------------------------------------------------------------------------|-----|
| 7 | Multilingual Code-Switching for Zero-Shot Cross-Lingual Intent Prediction and Slot-Filling | 84 |
| 7.1 | Summary | 84 |
| 7.2 | Methodology | 85 |
| 7.2.1 | Problem Definition | 85 |
| 7.2.2 | Models & Concepts | 86 |
| 7.3 | Experimental Evaluation | 89 |
| 7.3.1 | Datasets | 89 |
| 7.3.2 | Experiments | 90 |
| 7.3.3 | Baselines & Upper Bound | 90 |
| 7.4 | Results & Discussion | 91 |
| 7.5 | Key Takeaways | 99 |
| 8 | Cross-Lingual Text Classification of Transliterated Hindi and Malayalam | 100 |
| 8.1 | Summary | 100 |
| 8.2 | Methodology | 100 |
| 8.2.1 | Problem Definition | 100 |
| 8.2.2 | Models & Concepts | 102 |
| 8.3 | Experimental Evaluation | 105 |
| 8.3.1 | Datasets | 105 |
| 8.3.2 | Experiments | 107 |
| 8.4 | Results & Discussion | 109 |
| 8.5 | Key Takeaways | 113 |
| 9 | Conclusion & Future Work | 114 |
| 9.1 | Conclusion | 114 |
| 9.2 | Future Work | 115 |
| 9.2.1 | Crisis Tweet Representation | 115 |
| 9.2.2 | Interpretability | 115 |
| 9.2.3 | Language Families | 116 |
| 9.2.4 | Aligning Multilingual Knowledge Graphs for Crisis | 116 |
| 9.2.5 | Transliteration | 117 |
| | Bibliography | 118 |

List of Tables

| Table | Page |
|-----------------------------------------------------------------------|------|
| 2.1 Notations | 10 |
| 3.1 Amazon and Crisis Tweet Dataset Statistics | 36 |
| 3.2 Implementation Details for Diversity Models | 37 |
| 3.3 Diversity Performance Evaluation on Amazon Reviews | 41 |
| 3.4 Diversity Performance Evaluation on Crisis Tweets | 42 |
| 3.5 Runtime of Diversity Models | 42 |
| 3.6 Additional Experiments with Diversity Models | 42 |
| 3.7 Evaluation of Attention Aggregation Methods | 44 |
| 4.1 MT-DAAN Notations | 48 |
| 4.2 TREC Dataset Statistics | 52 |
| 4.3 Implementation Details for MT-DAAN | 55 |
| 4.4 Performance of Deep Neural Models on Amazon Reviews | 57 |
| 4.5 Performance of Deep Neural Models on TREC Dataset | 58 |
| 4.6 MT-DAAN Results on TREC Dataset | 58 |
| 4.7 MT-DAAN Results on COVID-19 Dataset | 59 |
| 4.8 Comparison of Word Vector Models | 59 |
| 4.9 Improving MT-DAAN with Additional Web Data | 61 |
| 5.1 Evaluation of Amazon QA and COVID QA Datasets using BLEU Scores . | 71 |
| 6.1 Notations for Attention Realignment | 77 |
| 6.2 Appen Dataset Statistics | 79 |
| 6.3 Implementation Details for Attention Realignment | 80 |
| 6.4 Performance Evaluation of Attention Realignment | 82 |
| 7.1 Selected Language Families | 87 |
| 7.2 Datasets for Code-Switching | 88 |
| 7.3 Performance Evaluation of Code-Switching on MultiATIS++ | 91 |
| 7.4 Performance Evaluation of Code-Switching on Crisis Data | 93 |
| 7.5 Runtime for Joint Training with Code-Switching | 93 |
| 8.1 Statistics of Transliteration Test Data | 105 |

| | | |
|-----|------------------------------------------------------------------|-----|
| 8.2 | Transliteration Performance Evaluation | 108 |
| 8.3 | Evaluation of Teacher-Student Model on Source Language | 108 |
| 8.4 | Evaluation of Teacher-Student Model on Original Script | 109 |
| 8.5 | Teacher-Student Model Runtime | 113 |

List of Figures

| Figure | Page |
|------------------------------------------------------------|------|
| 1.1 Cross-Domain Text Classification Example | 2 |
| 1.2 Cross-Lingual Text Classification Example | 2 |
| 1.3 Implication of Our Work in the Crisis Domain | 3 |
| 1.4 Overall Problem Definition | 4 |
| 2.1 t-SNE of mBERT Embeddings on Parallel Data | 20 |
| 2.2 Transliteration Examples | 24 |
| 2.3 t-SNE of Transliterated Embeddings | 24 |
| 3.1 Diversity Model Architecture | 28 |
| 3.2 Tri-training Setup with Diversity | 34 |
| 3.3 Attention Aggregation Methods | 35 |
| 3.4 Diversity Model Attention Visualization | 39 |
| 3.5 Optimal Number of Attention Heads | 40 |
| 4.1 Crisis Tweets Domain Adaption | 46 |
| 4.2 Multi-Task Domain Adaptation Setup | 47 |
| 4.3 Visualizing Attention on Crisis Tweets | 62 |
| 4.4 Interpretability on COVID-19 Tweets | 63 |
| 5.1 QA Domain Adaptation Examples | 67 |
| 5.2 Masking and Regeneration Architecture | 68 |
| 5.3 Masking and Regeneration Examples | 72 |
| 6.1 Cross-Lingual Tweet Classification Example | 75 |
| 6.2 Attention Realignment Architecture | 76 |
| 6.3 Attention Visualization with Realignment | 81 |
| 7.1 Slot-Filling Example | 85 |
| 7.2 Language Family Analysis | 94 |
| 7.3 Code-Switching Training Runtime | 94 |
| 7.4 mBERT Performance with Code-Switching | 95 |
| 7.5 XLM-R Performance with Code-Switching | 95 |

| | | |
|-----|---------------------------------------------------------------|-----|
| 7.6 | Impact of Code-Switching | 96 |
| 7.7 | mBERT vs XLM-R for Code-Switching | 97 |
| 7.8 | Freezing mBert Layers for Code-Switching | 98 |
| 8.1 | Teacher-Student Model Overview | 101 |
| 8.2 | Transliteration Examples | 101 |
| 8.3 | Teacher-Student Model with Joint Training. | 103 |
| 8.4 | New Transliteration Dataset | 104 |
| 8.5 | Unsupervised Alignment | 110 |
| 8.6 | Freezing mBERT layers for the Teacher-Student Model | 111 |

List of Abbreviations

| | |
|--------------------|-------------------------------------------------------------|
| NLP | Natural Language Processing |
| NLU | Natural Language Understanding |
| QA | Question Answering |
| MTL | Multi-Task Learning |
| MLM | Masked Language Model |
| CNN | Convolutional Neural Network |
| RNN | Recurrent Neural Network |
| SVM | Support Vector Machines |
| KG | Knowledge Graph |
| GCN | Graph Convolutional Network |
| TREC-IS | Text Retrieval Conference - Incident Streams |
| MHA | Multi-Head Attention |
| MHAD | Multi-Head Attention with Diversity |
| ST | Single-Task Attention Network |
| ST-DAAN | Single-Task Domain Adversarial Attention Network |
| MT-DAAN | Multi-Task Domain Adversarial Attention Network |
| Tukey’s HSD | Tukey’s Honest Significant Difference [1] |
| LSTM | Long Short-Term Memory [2] |
| BiLSTM | Bidirectional Long Short-Term Memory [3] |
| GRU | Gated Recurrent Unit [4] |
| T5 | Text-to-Text Transfer Transformer [5] |
| XLM | Cross-Lingual Masked Language Modeling [6] |
| XLM-R | Cross-Lingual Masked Language Modeling in a Roberta way [7] |
| BERT | Bidirectional Encoder Representations from Transformers [8] |

mBERT Multilingual BERT [8]
MUSE Multilingual Unsupervised and Supervised Embeddings [9]
VecMap Cross-Lingual Word Embedding Mappings [10]
SDA Stacked Denoising Autoencoders [11]
mSDA Marginalized Stacked Denoising Autoencoders [12]
DAmSDA Domain Adversarial Marginalized Stacked Denoising Autoencoders [11]
XNLI Cross-lingual Natural Language Inference [13]
GloVe Global Vectors for Word Representation [14]
MT-Tri Multi-Task Tri-training [15]
DANN Domain-Adversarial training of Neural Networks [16]
AMN Adversarial Memory Network [17]
HATN Hierarchical Attention Network [18]
IATN Interactive Attention Transfer Network [19]
MemN2N End-To-End Memory Networks [20]
QACNN Query-based Attention CNN [21]
CliniQG4QA Generating Diverse Questions for Domain Adaptation of Clinical QA [22]
YAGO Yet Another Great Ontology [23]
MTransE Multilingual KG Embeddings for Cross-lingual Knowledge Alignment [24]
JAPE Cross-Lingual Entity Alignment via Joint Attribute-Preserving Embedding [25]
t-SNE t-Distributed Stochastic Neighbor Embedding [26]

Abstract

UNSUPERVISED CROSS-DOMAIN AND CROSS-LINGUAL METHODS FOR TEXT CLASSIFICATION, SLOT-FILLING, AND QUESTION-ANSWERING

Jitin Krishnan, PhD

George Mason University, 2021

Dissertation Director: Dr. Huzefa Rangwala

Transfer learning has significantly revolutionized modern machine learning systems by instilling the ability to use the knowledge gained from solving one problem for another. It has also helped to adapt and build models that can be generalized beyond the distributions that they are trained on. This dissertation explores and presents novel techniques for two transfer learning problems (cross-domain and cross-lingual) in the field of Natural Language Processing, for the tasks of text classification, slot-filling, and question-answering. This is particularly motivated by scenarios such as crisis management in which labeled data from a new domain or language cannot be easily obtained during an ongoing crisis to train models. A cross-domain setup (also known as domain adaptation) adapts a model trained on one domain (e.g., Tweets posted during Hurricane Harvey) to another (e.g., Tweets posted during Hurricane Florence). A cross-lingual setup adapts a model trained on one language (e.g., English) to another (e.g., Hindi). Essentially, our goal is to bring seemingly dissimilar distributions into a comparable representation based on a task at hand, so that a model trained on data from one domain/language can be generalized to another.

We make three contributions for the task of domain adaptation, focusing on text data in English: (a) We show that machine learning architectures that ensure sufficient diversity can generalize better. In the context of text classification, this is achieved by enforcing orthogonality constraints within and across attention-based neural models, in a fully unsupervised manner unlike traditional methods that require unlabeled data from the target. (b) For text classification in low-resource scenarios (e.g., crisis tweets), where there exist multiple domains and multiple tasks, a setup with domain discrimination while sharing a few internal layers for multiple tasks can generalize well to an unseen domain. (c) For the task of generative question-answering, we propose an adversarial method of masking domain specific words and regenerating them using a sequence-to-sequence language model trained using unlabeled target data. The purpose of this approach is to construct pseudo-labeled target data from the labeled source data.

We also make three contributions for the task of cross-lingual learning: (a) In the context of text classification, we show that an attention realignment method that enforces the model to distinguish task-specific versus language-specific words can improve cross-lingual performance. (b) For the task of joint learning of intent prediction and slot-filling (intent being sentence-level label and slot being word-level label), randomized switching of phrases in a sentence to various other languages is shown to generalize well on unseen languages. (c) Finally, we enhance the modern multilingual language models with the ability to classify transliterated text.

Practical implications of our work are demonstrated on Twitter posts collected during various natural disasters that span different languages. Due to the generalizability of our models across domains and languages, they can be immediately deployed to aid emergency services during crisis events to extract relevant information. Towards this goal and for designing models that can explain the predictions for crisis management, interpretability of models is also explored. Furthermore, for some of our tasks, we release newly labeled crisis datasets for the research community.

Chapter 1: Introduction

Text documents or words transcribed from speech are ubiquitous data sources created from the fundamental nature of human communication. We see text in books, TV, movies, social media, email, and other plethora of sources on a daily basis. Information extracted and insights gained from such textual data can be used to build exciting applications such as virtual assistants, summarizing news content, classifying spam, detecting urgent tweets posted during a crisis, analyzing social media interactions, etc. Under the hood of such applications lie complex machine learning models that are ready to make predictions on an unseen data point. Quality of such models largely depend on the quality of data on which they are trained on. And, with more labeled data comes robustness and the ability to tackle very specific problems. However, in many real-world problems such as an ongoing crisis event, obtaining labeled data for training supervised or semi-supervised models is tedious and mostly unrealistic. Consider situations where we need to classify Twitter posts based on *urgency* or identify words that represent resource needs such as *food*, *water*, *shelter*, etc. In such scenarios, it is imperative to make use of past data and adapt it to the new situation. For example, machine learning models trained on labeled data from past crisis events can be used for an ongoing crisis event with minimal or no data from the new event. Similarly, if the past data is in a different language, we would like the models to transfer that knowledge to a language that we are interested in. These are typically known as cross-domain (also domain adaptation) and cross-lingual problems respectively. Creating machine learning models with this setup in mind for various tasks forms the fundamental motivation behind this dissertation.

To elaborate on an example, consider domain adaptation applied to text classification. A benchmarking task in text classification is sentiment analysis [27], where textual data can be classified as either *positive* or *negative*. For example, consider *reviews* posted by

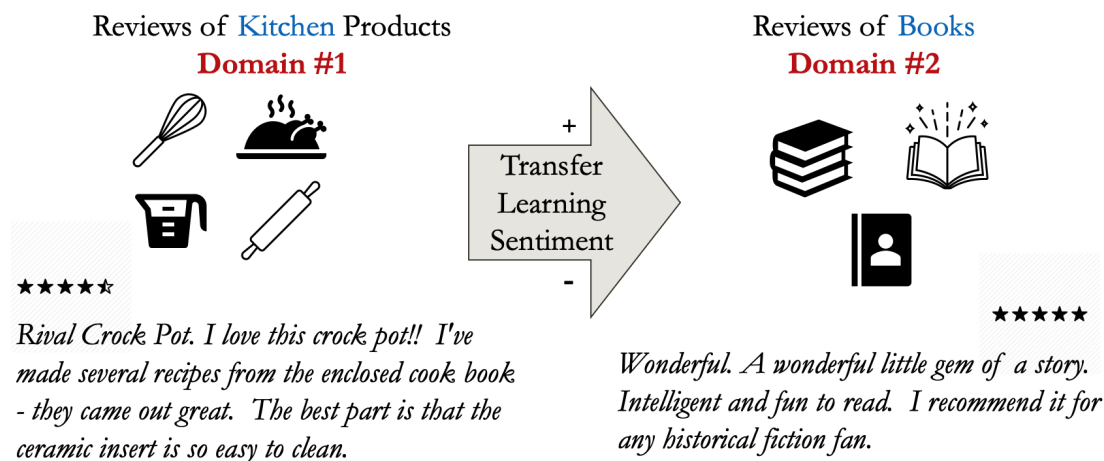


Figure 1.1: An example of cross-domain text classification. A cross-domain model will be trained on one domain and evaluated on a different domain.

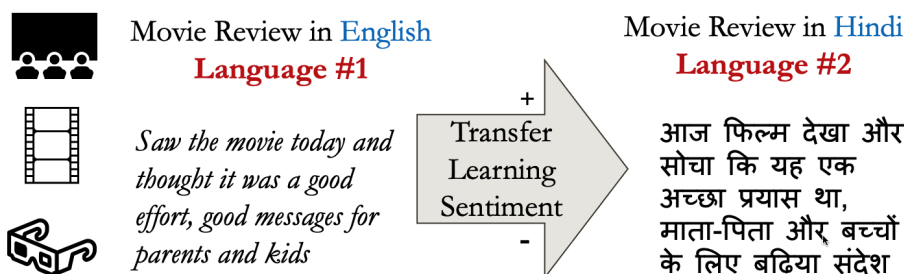


Figure 1.2: An example of cross-lingual text classification. A cross-lingual model will be trained on one language and evaluated on a different language.

customers for movies, restaurants, or items purchased on Amazon. By training a simple binary classification model using such reviews, for which treating all reviews less than 3 stars as *negative* and more than 3 stars as *positive*, we could predict binary ratings for reviews that have no rating associated with it or distinguish unrated comments posted by the public into *positive* or *negative* to study the sentiment behind them. In the cross-domain setting that we interested in, the setup is slightly tweaked such that the training and test datasets come from two different domains. E.g., *Kitchen Product Reviews* versus *Book Reviews* as shown in Fig. 1.1. Similarly, a cross-lingual setting assumes that the test data comes from a different language. For example, a model can be trained using reviews in *English* and

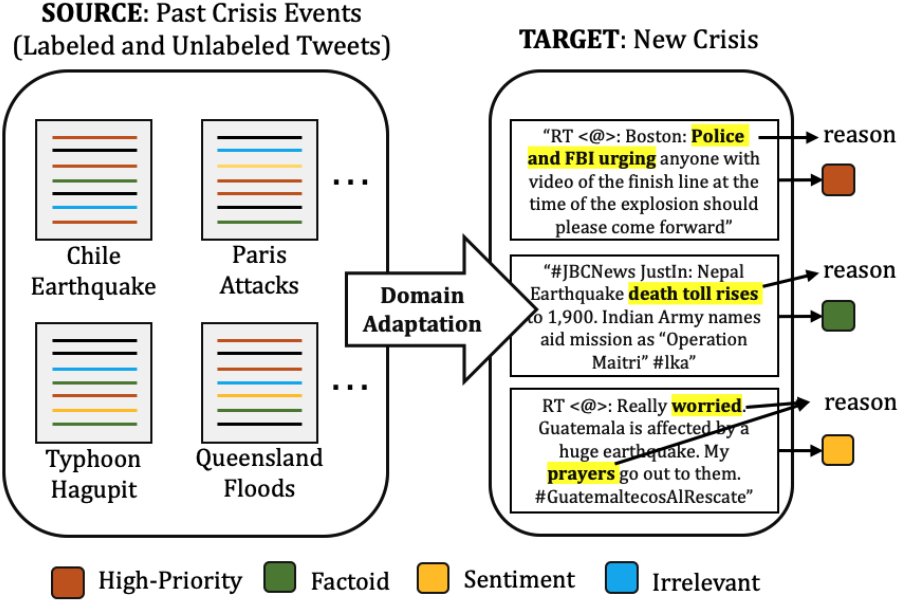


Figure 1.3: An application scenario in the crisis domain to interpretably predict labels for tweets collected during an ongoing crisis using only the past crisis data, given a) unavailability of labeled data in the ongoing event, and b) need for interpretability of machine reasoning behind data filtering for emergency managers.

evaluated on reviews in *Hindi*, as shown in Fig. 1.2. As evident from the generic nature of the examples shown, this setup can be expanded to varieties of tasks, other than just classification, such as Slot-Filling, Question-Answering, Named Entity Recognition, Natural Language Inference, Knowledge Graph Construction, etc. In this dissertation, we focus on three such tasks: Classification, Slot-Filling, and Question-Answering.

1.1 Motivation

The primary motivation of this dissertation is to solve key machine learning challenges in order to build models that can extract relevant information from social media posts during crisis situations such as Hurricanes or Earthquakes to aid emergency managers for an efficient and timely allocation of resources. Cross-domain and cross-lingual settings gain prominence particularly in crisis situations because it is impractical to label new datasets

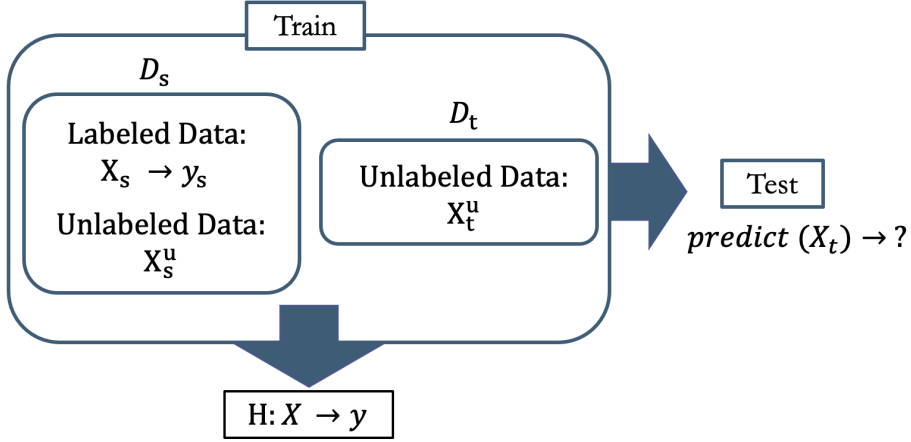


Figure 1.4: Unsupervised cross-domain and cross-lingual setup of our work (s = source, t = target, H = hypothesis). The goal is to train a model with the ability to generalize beyond the original distribution (D_s).

during an ongoing crisis. Such situations require models that can be immediately deployed regardless of a change in domain or language. Thus, the practical implication of our work is demonstrated in this context throughout. An example is shown in Fig. 1.3. This example also shows the importance of *interpretability* in model predictions. As language technologies continue to improve and include various domains and languages, transferring knowledge gained from one domain/language to another still remains a challenging problem. The quality of such machine learning models lie in their ability to generalize beyond their original distribution, which is the theoretical motivation behind our work.

1.2 Problem Statement

We define our problem statement commonly for both cross-domain and cross-lingual tasks. The training dataset is called as *source* and the evaluation dataset is called as *target*. In a cross-domain setting, source and target are two domains in the same language, while in a cross-lingual setting, source and target are two languages in the same domain. Given a source (D_s) and a target (D_t) distribution, the goal is to train a model using data from D_s and predict examples from the completely (or partially) unseen D_t . X_s and X_t represent the

set of labeled data from source and target distributions respectively with their corresponding ground truth labels y_s and y_t . X_t and y_t are used only for evaluation purposes. In scenarios where unlabeled data is available, we also define X_s^u and X_t^u from the source and target distributions respectively. Note that some of our models completely avoid X_t^u . Superscript *pred* represents predicted result. This is briefly outlined below and shown in Fig. 1.4:

Input: $X_s, y_s, X_s^u, (X_t^u \text{ if needed})$

Goal: $y_t^{pred} \leftarrow predict(X_t)$.

1.3 Contributions

We make six overall contributions with three each for cross-domain and cross-lingual problems. Each contribution has its dedicated chapter (3 to 8) and described briefly below.

Cross-Domain Text Classification (Chapter 3)

For our first contribution, we address the challenge of improving generalizability of neural sequence models in the context of domain adaption in unsupervised text classification task. We introduce novel diversity-based models focusing on attention layers and evaluate them on the benchmark datasets of Amazon reviews. To further improve generalizability and utilize additionally available unlabeled source data, a tri-training procedure is defined with an additional diversity constraint between the attention layers of the tri-trained classifiers. Addressing the existing evaluation gap in component-level performance analysis, a systematic and incremental creation of our models is shown by creating strong unsupervised baselines and improving upon existing work. A key advantage of our model is that it does not require any target data for training; making it universally adaptable to any domain. Our models are computationally cheaper (training converges quickly) when compared to the state-of-the-art. Diversified attention can provide better quality of attended words which can be used for various downstream tasks such as knowledge graph construction.

Cross-Domain Text Classification for Multi-Domain Dataset with Limited Labels (Chapter 4)

For our second contribution, we address the problems of data sparsity and limited labels in the context of cross-domain text classification. We construct a multi-task learning architecture with different tasks sharing a common layer along with a multi-domain discriminator. We apply this technique to filter tweets for crisis management by training four different classification tasks across ten different crisis events under domain shift. Our design consists of dedicated attention layers for each task for interpretability and a domain classifier branch to promote the model to be domain-agnostic. The interpretability provided by the attention layers is demonstrated for the predictions made by the classifiers, with the goal to aid emergency services in a more meaningful way. Furthermore, through experiments, we show that deep networks struggle with small datasets, and that this can be improved by sharing the base layer for multi-task learning and domain adversarial training.

Cross-Domain Generative Question Answering (Chapter 5)

For our third contribution, we address the task of generative question answering under domain shift. Generative QA is a setup in which the model produces free form answers to a question. We propose an improvement upon the state-of-the-art T5 [5] transformer by introducing an adversarial masking and regeneration method. This is evaluated on the Amazon QA dataset. And, a practical implication is demonstrated by applying it to answer questions regarding COVID-19 from tweets.

Cross-Lingual Text Classification (Chapter 6)

For our fourth contribution, we address the challenge of improving cross-lingual text classification while exploring interpretability. We propose a novel attention realignment method to promote the task classifier to be more language agnostic, which in turn tests the effectiveness of multilingual knowledge capture of the state-of-the-art XLM-R model [7]. This

is enhanced with a pseudo-labelling procedure which further improves the model’s generalization capability. Furthermore, incorporating the attention-based mechanism allows to perform an interpretability analysis on the model, by comparing how words are attended in the original versus translated versions. Experiments are conducted on a multilingual dataset consisting of tweets posted during various crisis events.

Cross-Lingual Intent Prediction and Slot Filling (Chapter 7)

For our fifth contribution, we address a Natural Language Understanding (NLU) task of intent prediction and slot-filling. This is a classification task that jointly predict the intent (sentence-level label) and slots (word-level labels) of a sentence/phrase. To enhance the language neutrality of Multilingual BERT (mBERT) [8] for fine-tuning, we propose a data augmentation method via multilingual code-switching. By code-switching into different language families, it is shown that potential linguistic relationships between a family and a target language can be identified and studied. This could help foster zero-shot cross-lingual research in low-resource languages. A key advantage of our model is that, with enhanced generalizability, it can be deployed with an out-of-the-box functionality as training using code-switched data is conducted independent of the target language, as compared to [28,29]. Methods of first machine translation of the source data into the known target language, followed by fine-tuning (referred ‘translate-train’) [30–33] require a separate model to be trained for each language. Furthermore, a new human-annotated tweet dataset, collected during Haiti earthquake disaster, for intent prediction and slot filling in English and Haitian Creole is publicly released.

Cross-Lingual Classification of Transliterated Text (Chapter 8)

For our sixth contribution, we address the task of classifying transliterated text (text which is transferred from the alphabet one language to another). We propose a novel Teacher-Student method to address the alignment problem for contextual representations of transliterated text (and show its efficacy on Hindi and Malayalam). Two newly labeled datasets

are released; a binary *sentiment* dataset of Malayalam movie reviews and a binary *relevancy* dataset of tweets posted during North India and Kerala flood crises.

The work (Chapters 3 to 8) presented in this dissertation are also published/submitted/to-be-submitted at peer reviewed venues [34–37].

1.4 Dissertation Outline

To summarize, this dissertation is primarily split into two types of research problems in NLP: cross-domain and cross-lingual transfer learning. A background and related work of all the tasks are provided in Chapter 2. Chapters 3, 4, and 5 tackle cross-domain problems of text classification, its application in the crisis domain, and question-answering respectively. Chapters 6, 7, and 8 tackle cross-lingual problems of text classification, slot-filling, and transliterated text classification respectively. Finally, the conclusion and future directions are presented in Chapter 9.

Chapter 2: Background & Related Work

This chapter introduces the background work related to cross-domain and cross-lingual problems addressed in this dissertation. We assume that the readers are familiar with the basic concepts in machine learning and text classification. Among the vast amount of research done in this area, we focus specifically on a subset of topics that is relevant to us. This includes an exploration of the state-of-the-art techniques and a literature review of domain adaptation in text classification and generative question-answering tasks, cross-lingual text classification and slot filling tasks, and their applications in crisis management. These form the basis and benchmarks for the new models presented in this dissertation.

2.1 Notations

This section discusses the commonly used notations throughout the dissertation. A summary is provided in Table 2.1. Individual chapters may define new notations or terminologies for specific concepts or tasks. s and t subscripts are used to represent source and target versions of data respectively. Some of our methods use unlabeled data as well as their soft/pseudo-labeled versions. These are represented with u and pl superscripts. Given a dataset X of text samples, x represents a single data point. In problems such as text classification, the feature vector for each word present in x comes from word vector models such as fastText [38] or language models such as BERT [8]. For example, $x^{<k>}$ will have a dimension of 300 if using fastText. We will describe them in more detail in the upcoming sections and chapters.

Table 2.1: Notations

| Notation | Definition |
|-------------------------------|----------------------------------------------------------------|
| s (<i>subscript</i>) | Source Domain/Language Data |
| t (<i>subscript</i>) | Target Domain/Language Data |
| u (<i>superscript</i>) | Unlabeled Data |
| pl (<i>superscript</i>) | Pseudo-labeled Data |
| $pred$ (<i>superscript</i>) | Predicted Result (as compared to ground truth) |
| \mathcal{D} | Distribution |
| X | Set of labeled data |
| y | Set of ground truth labels for X |
| H | Hypothesis |
| x | input text (i.e., a sample from X) |
| $x^{<k>}$ | feature vector representing k -th word of the input text x |
| T_x | Number of words in the input text |
| \vec{a} | Attention vector produced from the input text |
| $\alpha^{<k>}$ | Attention weight of k -th word |
| T_α | Number of attention heads |
| $a^{<k>}$ | Activation from k -th word |

2.2 Cross-Domain

2.2.1 Text Classification

In NLP, domain adaptation of sequence classification problems has several applications ranging from sentiment analysis [27] to classifying social media posts during crisis events [39]. Knowledge learned from one domain, book reviews for instance, can be adapted to predict examples from a different domain such as reviews of electronics. Similarly, information about resource-need events learned from one natural disaster can be adapted to predict events from an ongoing crisis. With the publication of Amazon reviews dataset [27] consisting of around 25 different domains, cross-domain sentiment analysis became a common way to evaluate machine learning models for domain adaptation in text.

The best methods in this line of research largely remain semi-supervised with the best performing models utilizing unlabeled target data during training. We consider the essential

criterion for no supervision in domain adaptation as having zero knowledge about the target domain beforehand; even if it is unlabeled. Thus, our proposed method can be viewed either as a strong baseline for future semi-supervised cross-domain research or as a new direction in fully unsupervised domain adaptation. Our work is scoped to the following setting: **a)** *single-task transfer*, **b)** *single source and target*, and **c)** *without unlabeled target data available during training*. The unsupervised methods proposed in our work are compared and contrasted with the existing semi-supervised counterparts. Supervised or minimally supervised approaches are not considered in this work.

Early works on domain adaptation such as Structural Correspondence Learning [40] made use of unlabeled target data to find a joint representation by automatically inducing correspondences among features from different domains. The importance of a good feature representation was later formally analyzed with a generalization bound by Ben-David et al. [41]. These studies realized the importance of finding commonality in features or pivots and minimizing the difference between the domains. Pan et al. [42] proposed a spectral feature alignment method to align domain-specific and domain-independent words into unified clusters via simultaneous co-clustering in a common latent space. Later, introduction of deep learning and neural networks helped remedy the problems of manual pivot selection and discrete feature representations. In order to learn better higher level representations, Stacked Denoising Autoencoders (SDA) [11] were introduced. Along with SDA, a more efficient version called marginalized SDA [12] with low computational cost and scalability has been utilized successfully in cross-domain tasks [43, 44]. Domain-Adversarial training of Neural Networks (DANN) [16] was proposed to effectively utilize unlabeled target data to create a classifier that is indiscriminate toward different domains. In DANN, a negative gradient (gradient reversal) from a domain classifier branch is back-propagated to promote the features at the lower layers of the network incapable of discriminating domains. DANN became an essential component in many works that followed. Recent works such as Adversarial Memory Network (AMN) [17] bring interpretability by using attention to capture the pivots. Along with attention, they effectively use gradient reversal to learn domain

indiscriminate features. Hierarchical Attention Network (HATN) [18] expands upon AMN by first extracting pivots and then jointly training a pivot and non-pivot networks. Interactive Attention Transfer Network (IATN) [19], another closely related work to AMN and HATN, showed the importance of attending ‘aspect’ information. Another line of research, that approached domain adaption through innovation in training procedure, is tri-training [45, 46]. Tri-training utilizes three independently trained classifiers; of which one is trained only on unlabeled target data, pseudo-labeled by the other two. The final prediction is done by majority voting. Multi-task tri-training (MT-Tri) [15], on the other hand, introduced an orthogonality constraint between the two classifiers such that it can be trained jointly reducing the compute time. This constraint is one of the inspirations for our work. All of these recent works used unlabeled target data for training classifiers. Our goal is to show that similar performance is achievable without using unlabeled target data.

Based on how the dataset is used, approaches to domain adaptation can vary as minimally-supervised, semi-supervised, or unsupervised. Minimally-supervised approaches such as Aligned Recurrent Transfer [47] utilize some labeled data from the target domain. Semi-supervised approaches such as DANN, AMN, HATN, or IATN utilize unlabeled target data making it a more realistic scenario in terms of usability where collecting labeled target data is expensive. However, many state-of-the-art semi-supervised methods, strikingly, never compare with strong fully unsupervised baselines where no target data is used. Newer methods have started using word vectors [48] for their input word representations. However, the baselines they compare with, utilize 5000-dimension feature vector of the most frequent unigrams and bigrams as the input representation. In addition, many recent works present a complex system without conducting a component-wise analysis which makes it unclear as to how much each component (word vectors, gradient reversal, or attention) contributed to the performance boost as compared to a simple DANN architecture. To address these evaluation gaps, we perform a systematic and incremental construction of architectures such that individual performance gain is realized.

2.2.2 Application in Crisis Management

A crucial implication of this work is the crisis domain as shown in Fig. 4.2. During the sudden onset of a crisis situation, social media platforms such as Twitter provide valuable information to aid crisis response organizations in gaining real-time situational awareness [49]. Effective analysis of important information such as affected individuals, infrastructure damage, medical emergencies, or food and shelter needs can help emergency responders to make time-critical decisions and allocate resources in the most efficient and effective manner [50–57].

Several machine learning systems have been deployed to help towards this humanitarian goal of converting real-time social media streams into actionable knowledge. Classification being the most common task, researchers have designed models [39, 54, 58–61] that classify tweets into various categories such as priority, affected individuals, type of damage, type of assistance needed, usefulness of the tweet, etc. Social media streams are short, informal, and abbreviated; with potential linguistic errors and sometimes contextually ambiguous. These inherently challenging properties of tweets make their classification task and formulation less trivial when compared to traditional text classification tasks.

We address two practically important and underdeveloped aspects of current research in the crisis-domain to classify relevant social web posts (e.g., “*please send medical supplies, paramedics, temporary shelters Asap #NepalQuakeRelief*”): **a)** a fully unsupervised domain adaptation and **b)** interpretability of predictions. A fully unsupervised domain adaptation uses **no data** from the ongoing crisis to train the model. [58] showed that their convolutional neural network (CNN) model does not require feature engineering and performed better than the state-of-the-art methods; one of their models being completely unsupervised [58]. Similarly, [39] designed a CNN architecture with adversarial training on graph embeddings, but utilizing unlabeled target data. Our goal is to construct an unsupervised model that does not require any unlabeled target data with the capability of being interpretable. The problem of data sparsity and limited labels is addressed, in particular, by designing a multi-task classification model with domain adversarial training; which to the author’s knowledge

is not explored in the crisis domain. In prior works, when a top performing model produces an accuracy of 78%, for instance, it is unclear what that score really represent and how trustworthy it is. An interpretable model, such as the one presented in our work, can present with a convincing evidence of which words the classifier deems important when making a certain prediction. This also brings additional benefits of using them in downstream tasks such as knowledge graph construction.

For filtering social web data for crisis management, traditional domain adaptation models do not suffice and need customized expansions due to the following reasons: **a)** Collecting and using large unlabeled target data from the new/ongoing crisis event may not be practically viable, thus, we aim for a fully unsupervised modeling. **b)** Having access to unlabeled data from multiple crisis events can alleviate the above problem to an extent by using it to train the domain classifier branch to push the model to be domain independent. **c)** Due to the low-resource nature of the dataset, binary classifiers may miss important lower level features that can be potentially improved by a multi-task model that shares the lower layers of the network for all the tasks. This is also evident from our results in Table 4.4 and 4.5, which show that deep models that perform much better than simple models on Amazon reviews do not significantly outperform them on TREC tweet dataset for crises.

Multi-Task Learning. In the context of low-resource datasets, a method that gained prominence is Multi-Task Learning (MTL). MTL solves multiple tasks at the same time with a goal to improve the overall generalization capability of the model [62]. Within the context of Deep Learning, MTL is performed by sharing (or constraining) lower level layers and using dedicated upper level layers for various tasks. A rich overview of MTL in Deep Neural Networks is presented by Ruder (2017) [63]. MTL has been a successful strategy over the past few years for many research explorations such as relationship networks [64] in computer vision and Sluice networks [65] in natural language processing. Similar problems in domain adaptation of semantic classification and information retrieval were addressed by jointly learning to leverage large amounts of cross-task data [66]. In low resource datasets

such as for crises, the chance of overfitting is very high. Thus, it seems intuitively better for the model to find a shared representation capturing different tasks and not just one, such that feature commonalities across tasks can be exploited; which is the motivation for our second contribution.

Interpretability. Our cross-domain models also make use of a popular methodology called *attention*. Attention mechanism [67], originally designed for machine translation problems, has become one of the most successful and widely used methods in deep learning that can look at a part of a sentence at a time like humans. This is particularly useful because of its ability to construct a context vector by weighing on the entire input sequence unlike previous sequence-to-sequence models [68] that used only the last hidden state of the encoder network (typically BiLSTM [3], LSTM [2], or GRU [4]). For example, in a sentence, the context vector is a dot product of the word activations and weights associated with each word; thus leading to an improved contextual memorization, especially for long sentences. Our method incorporates such attention mechanisms to enhance interpretability of the classifier. With more and more machine learning systems being adopted by diverse application domains, transparency in decision-making inevitably becomes an essential criteria, especially in high-risk scenarios [69] where trust is of utmost importance. With deep neural networks, including natural language systems, shown to be easily fooled [70,71], there has been many promising ideas that empower machine learning systems with the ability to explain their predictions [72–74]. [75] presents a survey of interpretability in machine learning, which provides a taxonomy of research that addresses various aspects of this problem. Similar to the work by [76], we employ an attention-based approach to evaluate model interpretability applied to the crisis-domain.

2.2.3 Generative Question Answering

So far, we addressed the task of classification in cross-domain context. Now, we explore a different and one of the most popular tasks in Natural Language Processing: Question-Answering (QA). Large neural language models that are pre-trained on unlabeled text have seen tremendous success in recent times when fine-tuned on downstream tasks [5, 8, 77–79]; QA being one of them. There are several variants of QA tasks, the typical ones based purely on language models are defined by either i) explicitly providing a contextual text, alongside the question, from which the answer is extracted (extractive QA) [80–83], or ii) providing no context and letting the model to answer from an external knowledge base or from the underlying pre-trained language model (open-domain and generative QA) [84–87]. This work restricts the problem to the specific setting of generative QA given a contextual text alongside the question under domain shift as shown in Fig. 5.1.

Domain adaptation research in QA tasks have focused on approaches such as domain adversarial training [88–90], extending vocabulary of the language model [91], and adding a mixture of experts layer [92]. [93] comprehensively explores QA on multiple choice questions based on MemN2N [20] and QACNN [21]. However, the most common models such as BERT [8] are *encoder-only*, but generative models typically require a *decoder* to produce free-form answers, as advocated in models such as T5 [5]. Domains where a dedicated NER tagger is available, like Clinical QA, synthetic data generation approaches like CliniQG4QA [22, 94] which first extracts answer phrases and then predicts diverse question phrases from the unlabeled target have found promising results. Recently, an approach to generate question-answer pairs from the target domain using text-to-text language models, was proven to be successful for extractive QA [95, 96]. However, synthetic QA generation is non-trivial for generative QA as the answer is not always an exact span from the context. Therefore, as a deviation from these works, we design an alternative QA generation method from the target with the help of the source domain using masking and regeneration. We assume that the two domains share some semantic similarity (eg., how users answer questions about

different genres of products on Amazon such as *Music Instruments* versus *Office Products*).

2.3 Cross-Lingual

A cross-lingual setting is typically described as a scenario in which a model trained for a particular task in one language (e.g. English) should be able to generalize well to a different language (e.g. Japanese). While a semi-supervised solution [97, 98] assumes some target language data is available, a zero-shot solution [30, 99, 100] assumes none is available at training time. This is particularly significant in real world problems such as extracting relevant information during a new disaster [35, 101] and hate speech detection [102, 103], where the target language might be of low-resource or unknown. In such scenarios, it is crucial that models can generalize well to unseen languages.

2.3.1 Text Classification & Application in Crisis Management

Social media platforms such as Twitter provide valuable information to aid emergency response organizations in gaining real-time situational awareness during the sudden onset of crisis situations [49]. Extracting critical information about affected individuals, infrastructure damage, medical emergencies, or food and shelter needs can help emergency managers make time-critical decisions and allocate resources efficiently [50, 53–55, 104, 105]. Researchers have designed numerous classification models to help towards this humanitarian goal of converting real-time social media streams into actionable knowledge [39, 54, 58, 59, 61]. Recently, with the advent of multilingual models such as multilingual BERT [8] and XLM [6], researchers have started adopting them to multilingual disaster tweets [106, 107]. Since XLM-R [79] has been shown to be the most superior model in cross-lingual language understanding, our work leverages XLM-R to explore the aspects of cross-lingual transfer of knowledge and interpretability.

We address two questions in particular. First is to examine whether XLM-R is effective in capturing multilingual knowledge by constructing a custom model over it to analyze if a

model trained using English-only tweets will generalize to multilingual data and vice-versa. An example is shown in Fig. 6.1. Social media streams are generally different from other text, given the user-generated content. For example, tweets are usually short with possibly errors and ambiguity in the behavioral expressions. These properties in turn make the classification task or extracting representations a bit more challenging. Second question is to examine whether word translations will be equally attended by the attention layers. For instance, the words with higher attention weights in a sentence in Haitian Creole such as “*Tanpri nou bezwen tant avek dlo nou zon silo mesi*” should align with the words in its corresponding translated tweet in English “*Please, we need tents and water. We are in Silo, Thank you!*”. The core idea is that if ‘*dlo*’ in the Haitian tweet has a higher weight, so should its English translation ‘*water*’. This word-level language agnostic property can promote machine learning models to be more interpretable. This also brings several benefits to downstream tasks such as knowledge graph construction using keywords extracted from tweets. In situations where data is available only in one language, this similarity in attention would still allow us to extract relevant phrases in cross-lingual settings. To the best of the author’s knowledge in crisis analytics domain, aligning attention in cross-lingual setting is not attempted before. The classification experiments are focused only to tweets containing ‘*request*’ intent, which will be expanded to other behaviors, tasks, and datasets in the future.

There are numerous prior works (*c.f.* surveys [49,108]) that focus specifically on disaster related data to perform classification and other rapid assessments during an onset of a new disaster event. Crisis period is an important but challenging situation, where collecting labeled data during an ongoing event is very expensive. This problem led to several works on domain adaptation techniques in which machine learning models can learn and generalize to unseen crisis event [16,18,34,40]. In the context of crisis data, [58] designed a convolutional neural network model which does not require any feature engineering and [39] [39] designed a CNN architecture with adversarial training on graph embeddings. [35] showed that sharing a common layer for multiple tasks can improve performance of tasks with limited labels.

In multilingual or cross-lingual direction, many works [9, 109] tried to align word embeddings (such as fastText [38]) from different languages into the same space so that a word and its translations have the same vector. These models are superseded by models such as multilingual BERT [8] and XLM-R [79] that produce contextual embeddings which can be pretrained using several languages together to achieve impressive performance gains on multilingual use-cases.

Attention mechanism [67, 110] is one of the most widely used methods in deep learning that can construct a context vector by weighing on the entire input sequence which improves over previous sequence-to-sequence models [2, 3, 68]. As the model produces weights associated with each word in a sentence, this allows for evaluating interpretability by comparing the words that are given priority in original versus translated tweets.

With more and more machine learning systems being adopted by diverse application domains, transparency in decision-making inevitably becomes an essential criteria, especially in high-risk scenarios [69] where trust is of utmost importance. With deep neural networks, including natural language systems, shown to be easily fooled [71], there has been many promising ideas that empower machine learning systems with the ability to explain their predictions [72, 74]. [75] presents a survey of interpretability in machine learning, which provides a taxonomy of research that addresses various aspects of this problem. Similar to the work by [76], we employ an attention-based approach to evaluate model interpretability applied to the crisis-domain.

2.3.2 Intent Prediction and Slot-Filling

Intent prediction and slot filling are important NLU tasks and significant for real world problems. They are studied extensively for goal-oriented dialogue systems currently, such as Amazon’s Alexa, Apple’s Siri, Google Assistant, and Microsoft’s Cortana. Finding the ‘intent’ behind the user’s query and identifying relevant ‘slots’ in the sentence to engage in a dialogue are essential for an effective conversational assistance. For example, users might want to ‘*play music*’ given the slot labels ‘*year*’ and ‘*artist*’ [111], or they may want to ‘*book*

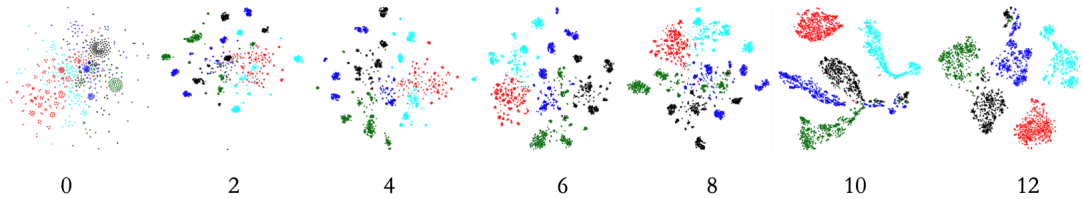


Figure 2.1: t-SNE plot of embeddings across the 12 multi-head attention layers of multilingual BERT. Parallely translated sentences of MutiATIS++ dataset are still clustered according to the languages: English (black), Chinese (cyan), French (blue), German (green), and Japanese (red).

a flight’ given the slot labels ‘*airport*’ and ‘*locations*’ [112]. A strong correlation between the two tasks has made jointly trained models successful [113–116]. In a cross-lingual setting, the model should be able to learn this joint task in one language and transfer knowledge to another [30, 117, 118]. This is the premise of our work.

Highly effective multilingual models such as mBERT [8] and XLM-R [7] have shown success across several multilingual tasks in recent years. In the zero-shot cross-lingual transfer setting with an unknown target language, a typical solution is to use pre-trained transformer models and fine-tune to the downstream task using the monolingual source data [30]. However, previous work [119] has shown that existing transformer-based representations may exhibit systematic deficiencies for certain language pairs. Previous work [119] has shown that existing transformer-based representations may exhibit systematic deficiencies for certain language pairs. Fig. 2.1 shows that the representations across the 12 multi-head attention layers of mBERT are still clustered according to the languages. This leads to a fundamental challenge that we address in this work: enhancing the language neutrality so that the fine-tuned model is generalizable across languages for the downstream task. To this goal, we introduce a data augmentation method via multilingual code-switching, where the original sentence in English is code-switched into randomly selected languages. For example, chunk-level code-switching creates sentences with phrases in multiple languages as

shown in Fig. 7.1. We show that this can lead to a better performance in the zero-shot setting such that mBERT can be fine-tuned for all languages (not just one) with a monolingual source data.

Further, we show how code-switching with different language families impact the model’s performance on individual target languages. Cross-lingual study of language families largely remains unexplored for NLU tasks. For instance, while it might be intuitive that Sino-Tibetan language family can aid a task in Hindi, results indicating that Turkic language family may help Japanese can reveal intriguing inter-family relationships and how they are aligned in the underlying language model’s vector space.

Cross-Lingual Transfer Learning. Researchers have studied cross-lingual tasks in various settings such as sentiment/sequence classification [99, 120, 121], named entity recognition [122–124], parts-of-speech tagging [31, 125, 126], and natural language understanding [30, 117, 127]. The methodology for most of the current approaches for cross-lingual tasks fall into the following three categories: **a)** multilingual representations from pre-trained or fine-tuned models such as mBERT [8] or XLM-R [7], **b)** machine translation followed by alignment [31–33], or **c)** a combination of both [30]. Before transformer models, effective approaches included domain adversarial training to extract language-agnostic features [44, 128] and word alignment methods such as MUSE [9] to align fastText word vectors [129]. Recently, [130] has shown that having shared parameters in the top layers of the multi-lingual encoders can be used to align different languages quite effectively on tasks such as XNLI [13].

Monolingual models for joint slot filling and intent prediction have used methods such as attention-based RNN [131] and attention-based BiLSTM with a slot gate [113] on benchmark datasets such as ATIS [112] and SNIPS [111]. These methods have shown that a joint method can enhance both tasks and slot filling can be conditioned on the learned intent. An interrelated mechanism was introduced [114] to iteratively learn the relationship between the two tasks. Recently, BERT-based approaches [115, 116] have shown improved results.

On the other hand, cross-lingual versions of this joint task include a low-supervision based approach for Hindi and Turkish [117], new dataset for Spanish and Thai [118], and the most recent work of MultiATIS++ [30] creating a comprehensive dataset in 9 languages; which is used to benchmark our results.

The joint task mentioned above in a pure zero-shot learning is the motivation of our work. Zero-shot is described as the setting where the model sees a new distribution of examples during test time [100, 132, 133]. It is common for machine translation based methods to translate source data to the target language before training. We assume that target language is unknown during training, so that our model is generalizable across languages.

Code-Switching. Linguistic code-switching is a phenomenon where multilingual speakers alternate between languages. Recently, monolingual models have been adapted to code-switched text in several tasks such as entity recognition [134], part-of-speech tagging [135, 136], sentiment analysis [137], and language identification [138–140]. Recently, [141] have proposed a pipeline to sample code-mixed documents using minimal supervision. [28] allows randomized code-switching to include the target language, as shown in their Figure 3. In our context for example, if the target language is German, we ensure that there is no code-switching to German during training. We consider this distinction essential to evaluate a true zero-shot learning scenario and prevent any bias when comparing with translate-and-train. Another recent work by [29] presents a non-zero-shot approach that performs code-switching to target languages. [142] presents a code-switching based method to improve the ability of multilingual language models for factual knowledge retrieval. Code-switching is usually done at the word-level. However, our results favor chunk-level switching over word-level as the latter may bring more noise to the code-switched version when compared to the original meaning of the sentence. A contemporary work by [143] makes use of both word and phrase-level code-mixing to switch to a set of languages to perform adversarial training for XNLI [13]. Code-switching and other data augmentation techniques have been applied to the pre-training stage in recent works [144–146], however we do not address

pre-training in this work. Highly popular pre-trained multilingual models such as XLM-R is also likely to be exposed to code-switched data, as it is trained using common-crawl; for which an analysis is given in Section 7.4. We focus primarily on mBERT which largely remain monolingual at the sentence level to identify the impact of code-switching during fine-tuning. In addition to studying multilingual slot filling and language families, another key distinction of our method is that we fully ignore the target language during training to represent a fully zero-shot scenario.

2.3.3 Transliterated Text Classification

Transliteration is a common phenomenon where a word from the alphabet of one language is transferred to another. A significant number of native Hindi or Malayalam speakers use Latin script instead of Devanagari or Brahmic scripts for a wide range of social media interactions such as posting tweets, updating Facebook status, commenting on YouTube videos, and writing reviews for restaurants/movies. This behavior occurs because keyboard optimizations focus primarily on English [147] and languages such as Hindi or Malayalam can be very time consuming to type on small devices. If users prefer the original script, the current solution is to back-transliterate the romanized text to the original language [148]. Examples of this transliteration process for Hindi and Malayalam are shown in Fig. 2.2. In the first row, English and Hindi are translations of each other, while Latin-transliterated (or *romanized*) Hindi is phonetically identical to Hindi but written using Latin characters.

In this context, we explore state-of-the-art language models such as multilingual BERT [8, mBERT] and XLM-R [79] to improve their multilingual generalizability through inclusion of romanized Hindi and Malayalam. Previous work [119] has shown that existing transformer-based representations may exhibit systematic deficiencies for certain language pairs. This deficiency also appears in transliterated sentences. The t-SNE plot, across various transformer layers, of 3-way parallel datasets consisting of sentences in source (English), target (Malayalam/Hindi), and their romanizations are clearly clustered in their own

| translated | | transliterated |
|-------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| English | Hindi | Romanized Hindi |
| Saw the movie today and thought it was a good effort, good messages for kids. | आज फिल्म देखने के लिए और सोचा कि यह एक अच्छा प्रयास था, बच्चों के लिए बढ़िया संदेश. | aj film dekhane ke liye aur socha ki yaha ek achcha prayas tha, bachchon ke lie badiya sandesh. |
| English | Malayalam | Romanized Malayalam |
| I was very disappointed in the movie. | ഞാൻ സിനിമയിൽ വളരെ നിരാശനായിരുന്നു. | njaan sinimayil valare niraashanaayirunnu. |

Figure 2.2: Examples of English sentences and corresponding translations and transliterations.

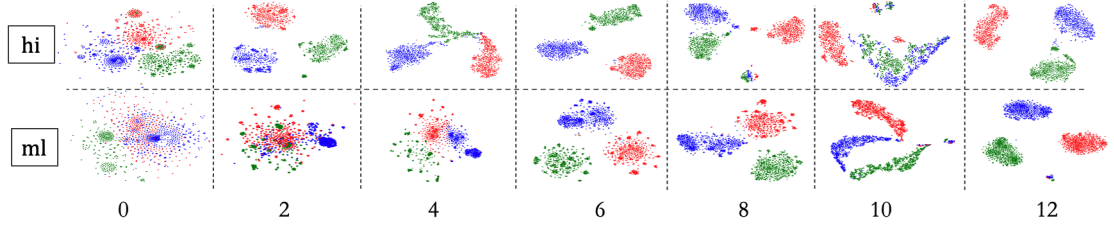


Figure 2.3: t-SNE plot of embeddings across the multi-head attention layers shows that the alignment deficiency identified in previous works [119] also extends to transliteration. XLM-R on Hindi (top). mBERT on Malayalam (bottom). English (blue), Original Script (green), and Latin-Transliteration (red).

groups, even though they match semantically, as shown in Fig. 2.3. Our work aims to provide a general solution towards alleviating this issue, by designing a generic and extensible architecture that can be used for aligning cross-lingual sentence representations in general.

Our problem setting is related to language alignment works where static [9, 149] or contextual [150–152] word representations from different languages are aligned to a shared vector space. Such methods primarily use a parallel word corpus and either design an alignment loss or explicitly perform rotations (transformations) on the representations. This work deviates from these methods in three aspects: (a) we focus on sentence-level representation (in this case as produced by the [CLS] classification BERT token; (b) we create a *synthetic 3-way* parallel corpus out of the source data using machine translations/transliterations,

and (c) by using a Teacher-Student training scheme, the final representations of the 3 language variants (source, target, and romanized target) are aligned to the same space as the source language embeddings as produced by the original pre-trained model.

Our Teacher-Student model is inspired from knowledge distillation methods [153] that are intended to transfer knowledge from a complex model (Teacher) to a simpler model (Student). This approach has been utilized for various tasks such as distillation to reduce the dimension of word embeddings [154], distilling BERT for text generation [155], and self-knowledge distillation [156]. A similar approach is taken where the Teacher model acts as an anchor by freezing all the layers, while the Student model is fine-tuned based on our optimization procedure to align sentences in English, their target translations, and transliterations.

The practical implications of the work are demonstrated by applying the idea on a naturally-occurring transliterated tweet dataset posted during the North India and Kerala flood crises. A model that can immediately handle transliterated tweets by producing embeddings in the same space as that of English tweets can immensely benefit information systems for emergency services, by utilizing the vast amount of crisis response models trained in English [35, 101, 157, 158].

Sentiment analysis in Hindi spans a variety of tasks such as analysis of movie reviews [159], building subjective lexica for product reviews and blogs [160], analysis on tweets [161], aspect-based sentiment [162], predicting elections [163], and analysis on code-mixed text [137]. Code-mixing is another related phenomenon where multilingual speakers alternate between languages, often in the same sentence. Both code-mixing and transliteration are studied for Hindi and Marathi texts using supervised learning methods by [164]. We restrict our analysis to transliteration, although our dataset may contain code-mixed text. Recently, [141] have proposed a pipeline to sample code-mixed documents using minimal supervision. In cross-lingual context, researchers have used linked WordNets [165] and cross-lingual word embeddings [166] using MUSE [9] and VecMap [10] to bridge the language gap,

later addressing code-mixing and transliteration. With the advent of large pre-trained language models, we take a step further in this direction to enhance mBERT/XLM-R to cover transliterations for fine-tuning it to the downstream task of sentiment analysis.

Meanwhile, Malayalam also has seen several works on sentiment analysis [167–170]. Recently, a new Malayalam-English code-mixed corpus [171] has been constructed by scraping YouTube comments. This corpus primarily consists of romanized sentences with some code-mixing. After converting this dataset into its original script to obtain the parallel corpus, this can also be used as an additional dataset for our model evaluation.

Translate and train has been a popular methodology [30–33] that utilizes the power of existing Machine Translation tools [172, 173] to perform cross-lingual tasks by augmenting the original source dataset with the target-translated data before training. This kind of training could enhance the performance of multilingual representations by fine-tuning the pre-trained models such as mBERT or XLM-R, creating a pseudo-supervised environment where the model now has access to data in the target language. We follow the same approach to create strong baselines as well as for the Teacher-Student model.

To summarize, in this chapter, we presented the relevant background and related work necessary to follow the materials presented in the dissertation. This included an exploration of state-of-the-art techniques and a literature review of our tasks in the context of cross-domain and cross-lingual learning. In the upcoming chapters, we dive into the specific tasks/challenges and present solutions with extensive empirical evaluation.

Chapter 3: Diversity-Based Generalization for Unsupervised Cross-Domain Text Classification

3.1 Summary

This chapter¹ [34] focuses on the problem of unsupervised domain adaptation applied to text classification. Domain adaptation approaches seek to learn from a source domain and generalize it to an unseen target domain. At present, the state-of-the-art domain adaptation approaches for subjective text classification problems are semi-supervised; and use unlabeled target data along with labeled source data. We propose a novel method for domain adaptation of *single-task* text classification problems based on a simple but effective idea of diversity-based generalization that does not require unlabeled target data. Diversity plays the role of promoting the model to better generalize and be indiscriminate towards domain shift by forcing the model not to rely on same features for prediction. We apply this concept on the most explainable component of neural networks, the attention layer. To generate sufficient diversity, we create a multi-head attention model and infuse a diversity constraint between the attention heads such that each head will learn differently. We further expand upon our model by tri-training and designing a procedure with an additional diversity constraint between the attention heads of the tri-trained classifiers. Extensive evaluation using the standard benchmark dataset of Amazon reviews and a newly constructed dataset of Crisis events shows that our fully unsupervised method matches with the competing semi-supervised baselines. Our results demonstrate that machine learning architectures that ensure sufficient diversity can generalize better; encouraging future research to design ubiquitously usable learning models without using unlabeled target data.

¹Published in the proceedings of the 19th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD). URL: <https://arxiv.org/pdf/2002.10937.pdf>

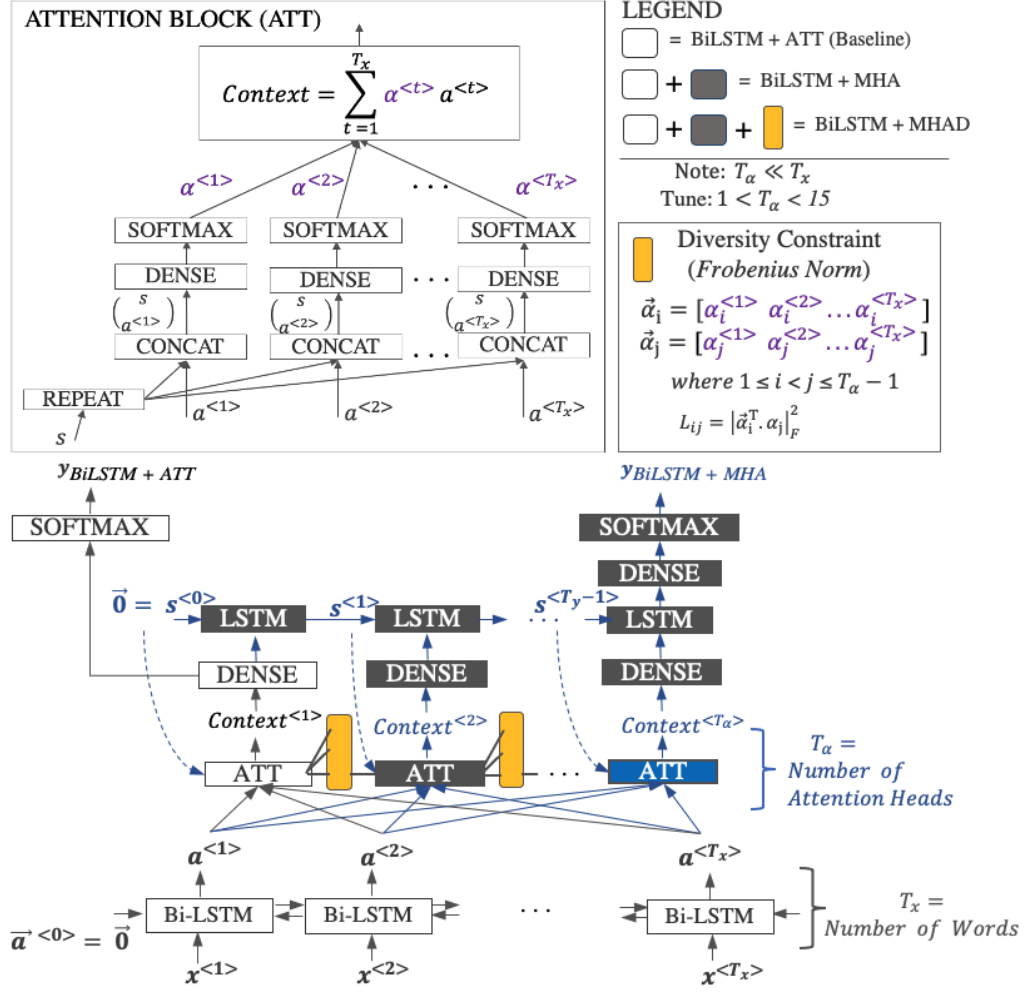


Figure 3.1: Complete architecture of the multi-head attention model with diversity.

3.2 Methodology

3.2.1 Problem Definition

Given a source (D_s) and a target (D_t) domain, the goal is to train a classifier using data **only** from D_s and predict examples from the completely unseen D_t . X_s and X_t represent the set of labeled data from source and target domains respectively with their corresponding ground truth labels y_s and y_t . X_t and y_t are used for testing purposes only. X_s^u and X_t^u represent unlabeled data available from the source and target domain respectively. X_t^u

(used in all of our competing models either for adversarial training or tri-training) is **never** used in our models. Finally, $[\cdot]^{pl}$ represents data that is pseudo-labeled by the classifier. To summarize:

Input: X_s, y_s (and X_s^u for tri-training)

Output: $y_t^{pred} \leftarrow predict(X_t)$

3.2.2 Models & Concepts

We introduce 4 models with one integral concept: *diversity*. Fig. 3.1 provides an overview of the first two models and Fig. 3.2 provides an overview of the last two. First is a multi-head attention baseline created to understand the naturally occurring diversity when multiple attention heads are connected. The second model enforces this diversity as a constraint such that all heads learn different features. The third model puts together three diversity-based classifiers and tri-trains them. Tri-training procedure in itself consists of an additional diversity constraint which forces two of the classifiers to learn differently. This is a one-step tri-training procedure intended for scenarios where no unlabeled source data is additionally available. When it is available, a full tri-training can be done until convergence, which is the fourth model.

What is *diversity*? Simply put, we define *diversity* as the dissimilarity between the objects in a group. In the context of machine learning, these objects can be classifiers or particular components of a model such as attention heads. One way to enforce this is by introducing an orthogonality constraint. In works such as [15], a constraint is introduced between two classifiers such that they learn slightly different features leading to better generalization during the test phase. Note that the opposite of *diversity* is simply using the same component multiple times. For example, a simple tri-training procedure may just use three of the same classifiers and take a majority vote for prediction [45,46]. This method still produces a better result as compared to a single model because, with random initialization of weights, three copies of the same model may still learn to focus on different aspects

and help avoid getting stuck in local minima. In Multi-task tri-training (MT-Tri) [15], the explicitly defined orthogonality constraint forces the three classifiers to learn differently. In our work, we apply this idea of diversity specifically to attention heads within and across the models that we design.

BiLSTM+MHA: Multi-Head Attention for Sequence Classification. BiLSTM+ATT is a standard baseline attention architecture constructed using BiLSTM [2, 3] and attention mechanism [67, 110]. Bidirectional Long Short-Term Memory (BiLSTM) units have been successfully used in sequence modeling tasks because of their effectiveness in representing forward and backward dependencies in a sequence. For example, meanings of words like ‘good’ and ‘bad’ can be changed when they are prefixed with ‘not’ or suffixed with ‘but’. Attention, on the other hand, provides task-specific benefits by attending the most relevant words such as ‘excellent’ or ‘poor’ in sentiment analysis. Attention and BiLSTM have been successfully combined previously for tasks such as relation extraction [174] to capture important semantic information in a sentence.

BiLSTM+MHA is an extension of the BiLSTM+ATT baseline by adding multiple attention heads as shown in Fig. 3.1. This is similar to machine-translation-like architecture [110] where each attention head leads to an LSTM cell with memory carried from previous cells to predict the next word. To customize it to classification purpose, we simply use the output from the final LSTM cell. Setting the classification task this way gives more leniency for the model to learn, remember, and generalize. Multiple attention heads can learn differently and what is learned from the previous heads is transferred to the next. However, this does not guarantee diversity as we do not know if the attention heads will in fact learn differently. In order to enforce diversity, we introduce the following models.

BiLSTM+MHAD: Multi-Head Attention with Diversity. In order to guarantee that these attention heads learn differently and forcing the model not to rely on the same

Algorithm 1 One-Step Diversity Tri-training

Input: X_s **Output:** m_1, m_2, m_3

- 1: $m_1, m_2 \leftarrow \text{joint_diversity_train_models}(X_s)$
 - 2: $m_3 \leftarrow \text{diversity_train_model}(X_s)$
 - 3: apply majority vote over m_i
-

features, we create a *diversity constraint*, an additional loss term shown below.

$$L_d = \frac{1}{k} \sum_{i=1}^{T_\alpha-2} \sum_{j=i+1}^{T_\alpha-1} \|\vec{\alpha}_i^T \cdot \vec{\alpha}_j\|_F^2 ; \text{ where } i \neq j \quad (3.1)$$

where $k = \frac{(T_\alpha-2)(T_\alpha-1)}{2}$, the total number of combinations. T_α is the total number of attention heads. $\vec{\alpha}_i$ and $\vec{\alpha}_j$ are i^{th} and j^{th} attention heads and $\|\cdot\|_F^2$ is the squared Frobenius norm, similar to the orthogonality constraint used in [15]. We leave the last attention head from this loss term so that we have one layer that learns freely without any constraints. The complete architecture of this diversity-based model is shown in Fig. 3.1. Resulting overall loss function, consisting of a binary cross entropy loss term and the diversity loss term, for N training examples is shown below.

$$L(\theta) = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)] + \gamma L_d \quad (3.2)$$

where γ is the hyperparameter to control how much diversity to be enforced within the model.

BiLSTM+MHAD-Tri-I: One-Step Diversity Tri-training. To further expand the concept of diversity, we tri-train the BiLSTM+MHAD models by adapting the multi-task tri-training procedure by [15]. In addition to applying the diversity constraint within each classifiers, an additional orthogonality loss is enforced between first two models m_1 and m_2 .

The third model m_3 is left out from the joint training. The loss term is shown below.

$$L_o = \frac{1}{k} \sum_{i=1}^{T_\alpha} \sum_{j=1}^{T_\alpha} \|(\vec{\alpha}_i^{[m_1]})^T \cdot \vec{\alpha}_j^{[m_2]}\|_F^2 \quad (3.3)$$

where $k = \frac{(T_\alpha-1)(T_\alpha)}{2}$. $\vec{\alpha}^{[m_1]}$ and $\vec{\alpha}^{[m_2]}$ are the attention heads for models m_1 and m_2 respectively. T_α is the total number of attention heads of each model. The total tri-training diversity loss is given below.

$$L_{dtri} = \beta_1 L_o + \beta_2 L_d \quad (3.4)$$

where β_1 and β_2 are the hyperparameters to control how much diversity to be enforced within and between the models.

For one-step diversity tri-training show in Algorithm 1, we jointly train m_1 and m_2 with tri-training diversity loss L_{dtri} . m_3 is separately trained as a BiLSTM+MHAD model. For predictions, a majority voting rule is applied over the three classifiers. The overall loss function for N training examples is given below.

$$L(\theta) = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)] + L_{dtri} \quad (3.5)$$

BiLSTM + MHAD-Tri-II: Tri-training until Convergence. Full tri-training, shown in Algorithm 2 and Fig. 3.2, utilizes additionally available unlabeled source data. While the first two classifiers m_1 and m_2 are jointly trained on labeled source data, the third classifier m_3 is solely dedicated to the train unlabeled data pseudo-labeled by m_1 and m_2 . Similar to [15], we define a threshold value τ such that at least one out of the two models should predict with probability greater than τ to be considered successfully pseudo-labeled. We set τ to be 0.7. Starting with second iteration, m_1 is trained jointly with m_2 using a combination of labeled source data and unlabeled source data pseudo-labeled by m_2 and m_3 . During joint-training, we give priority to the primary model by setting the loss weights accordingly.

Algorithm 2 Tri-training [15] - [Modified](#)

Input: X_s, X_s^u **Output:** m_1, m_2, m_3

```
1: while convergence condition is not met do
2:   for  $i \in 1..3$  do
3:      $X^{pl} \leftarrow \emptyset$ 
4:     for  $x \in X_s^u$  do
5:       if  $p_i(x) = p_k(x)(j, k \neq i)$  then
6:          $X^{pl} \leftarrow X^{pl} \cup \{(x, p_j(x))\}$  {where  $p(x)$  = predicted label}
7:       end if
8:     end for
9:     if  $i = 3$  then
10:       $m_3 \leftarrow \text{diversity\_train}(X^{pl})$  {Eq. 4.2}
11:     else if  $i = 1$  then
12:       $m_1 \leftarrow \text{joint\_diversity\_train}(X_s \cup X^{pl}, m_2)$  {Eq. 8.3}
13:     else
14:       $m_2 \leftarrow \text{joint\_diversity\_train}(X_s \cup X^{pl}, m_1)$  {Eq. 8.3}
15:     end if
16:   end for
17: end while
18: apply majority vote over  $m_i$ 
```

For example, while joint-training m_1 with m_2 , losses for the models can be minimized in a 2 : 1 ratio, giving priority to m_1 . We continue this process until a convergence condition is met: $m_1 \approx m_2 \approx m_3$.

Other Aggregation Methods. Multi-Head Aggregated Attention Network shown in Fig. 3.3 is a simple variation of the multi-head attention model used in machine translation tasks [175]. In the figure, T_x represents a constant that corresponds to the number of words in a sentence passed to the BiLSTM layer. Sentences with words greater than T_x are stripped and with words lower than T_x are padded. Activations for each word from the BiLSTM layer are passed through the attention block to generation the attention vector α consisting of weights that represents importance of each word in the sentence. In a traditional attention network, the context vector is given by product of the attention vector and the word activations:

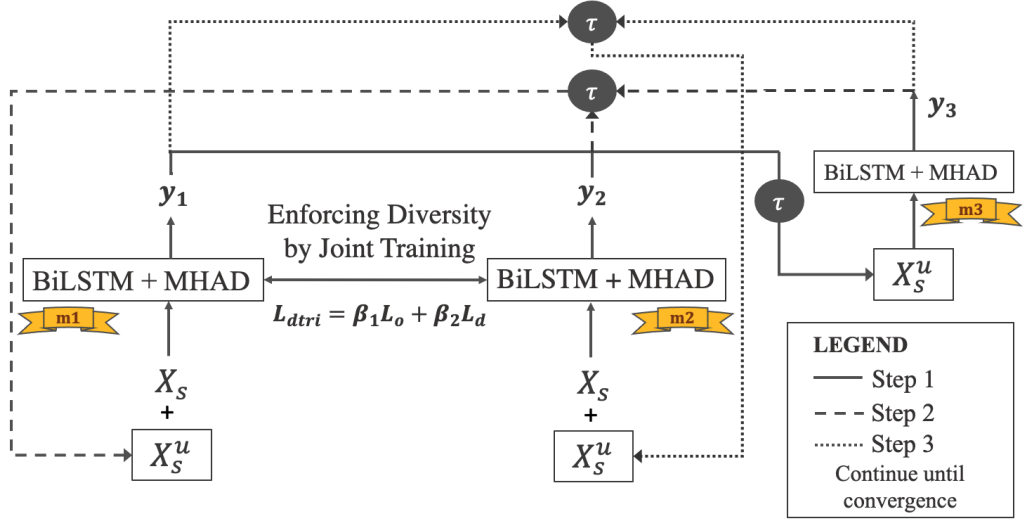


Figure 3.2: Tri-training BiLSTM+MHAD models

$$Context = \sum_{k=1}^{T_x} \alpha^{<k>} a^{<k>} \quad (3.6)$$

Aggregating T_α multiple attention layers can add stability to the resulting layer with more feature inclusions. Each word attention of dimension T_x is aggregated across T_α attention layers to form a resulting vector of dimension T_x as shown below:

$$\overrightarrow{\alpha_{agg}} = \left[\alpha_{agg}^{<k>} \quad \forall 1 \leq k \leq T_x \ni \alpha_{agg}^{<k>} = \sum_{h=1}^{T_\alpha} \alpha_h^{<k>} \right] \quad (3.7)$$

The aggregated context is given by:

$$Context_{agg} = \sum_{k=1}^{T_x} \alpha_{agg}^{<k>} a^{<k>} \quad (3.8)$$

which is then passed through dense and softmax layers to predict the label.

Having multiple attention layers brings two new hyperparameter: aggregation method

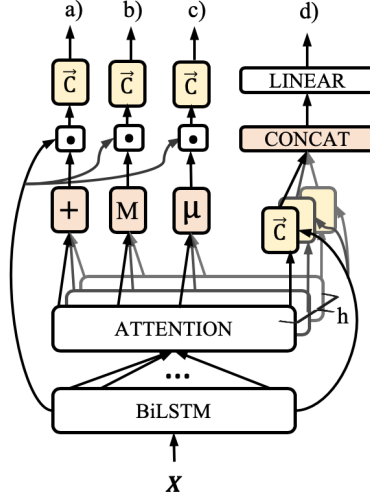


Figure 3.3: Multi-Head Attention Aggregation Methods: **a)** Sum (+), **b)** Max (M), **c)** Mean (μ), **d)** Concatenation. \vec{C} = Context and $h = [1, T_\alpha]$.

and number of layers. Performance variation with different types of aggregation methods such as ‘*sum*’, ‘*average*’, or ‘*max*’ and finding the optimal number of attention layers (T_α) are analyzed in the *Results and Discussion* section.

3.3 Experimental Evaluation

3.3.1 Datasets

Benchmark Dataset of Amazon Reviews. We use the standard benchmark Amazon reviews dataset² [27] which is widely used for cross-domain sentiment analysis. We consider four domains: Books (B), Kitchen (K), DVD (D), and Electronics (E). For a fair evaluation of the architectures, we use the exact same raw dataset³ used by our top competitor model HATN [18], which is a part of Blitzer’s original raw dataset. We also use the same 300-dimensional word vectors⁴ [48]. Table 3.1 summarizes this dataset.

²<http://www.cs.jhu.edu/~mdredze/datasets/sentiment/>

³https://github.com/hsqmlzno1/HATN/tree/master/raw_data

⁴<https://code.google.com/archive/p/word2vec/>

Table 3.1: Dataset Statistics

| | Positive | Negative | Unlabeled | Average Number of Tokens | Vocab |
|-----------------|----------|----------|-----------|--------------------------|--------|
| <i>Books</i> | 3000 | 3000 | 9750 | 182.0 | 105920 |
| <i>DVD</i> | 3000 | 3000 | 11843 | 197.5 | 117619 |
| <i>Kitchen</i> | 3000 | 3000 | 13856 | 102.0 | 52972 |
| <i>Elec.</i> | 3000 | 3000 | 17009 | 119.3 | 72458 |
| <i>Harvey</i> | 1122 | 960 | 10001 | 17.2 | 23562 |
| <i>Florence</i> | 201 | 1475 | 10001 | 17.1 | 26380 |
| <i>Irma</i> | 313 | 596 | 10001 | 15.3 | 20764 |

Crisis Dataset (Tweets). Additionally, we construct a new dataset consisting of Twitter posts (tweets) collected during three hurricane crises by *CitizenHelper* [176] system: *Harvey* and *Irma* in 2017, and *Florence* in 2018. Similar to sentiment classification, our goal here is to classify whether a tweet text indicates an event or not. Using the crowd-sourcing platform Figure-Eight⁵, three workers at minimum were assigned to give binary label to each tweet. We define events to be actions that involve at least one noun/entity. Events could be past, present, or future actions. It could also be questions, news, or instructions about actions. Some examples are: ‘*A rescues B*’, ‘*A is sending food to B*’, ‘*A will move to location B*’, and so on. Table 3.1 summarizes this dataset. Unfortunately, the labeled dataset for Florence and Irma consists of very low number of positive events. Consequently, we set up the experiments such that we train only on *Harvey* and test on *Florence* and *Irma*.

3.3.2 Experiments

Experimental Setup. We follow the traditional cross domain sentiment classification set up where each experiment consists of a source domain (S) and a target domain (T). A model will be trained on source data and tested on target data, represented as $S \rightarrow T$. We

⁵<https://www.figure-eight.com>

Table 3.2: Implementation Details

| | |
|-------------------------|----------------------------------------------------------------------------------|
| Deep Learning Library | Keras |
| Optimizer | Adam [$lr = 0.005$, $\beta_{1} = 0.9$, $\beta_{2} = 0.999$, $decay = 0.01$] |
| Maximum Epoch | 40 |
| Early Stopping Patience | 3 |
| Batch Size | 32 |
| Validation Split | 0.15 |
| Dropout | 0.4 |
| T_{α} | 5 |
| T_x | 200 |
| τ | 0.7 |

use all available labeled target data for testing. Crisis dataset is balanced before training and testing.

Implementation Details. We use Keras deep learning library with Adam optimizer. Implementation details are shown in Table 3.2. To keep the model simple, we do not change T_x and T_{α} further. Tri-training is stopped at 85% agreement. We set $\gamma = 0.01$, $\beta_1 = 0.05$ and $\beta_2 = 0.01$. These values are obtained by performing a basic hyperparameter tuning using grid search. Keras initializes the *LSTM* and *Dense* layers with *glorot_uniform* (or Xavier) [177] initializer. Addressing convergence to avoid local minima issues, we use a validation split of 0.15, with a patience of 3, and conduct 5 independent runs for each of the 12 cross-domain combinations.

3.3.3 Baselines

Adversarial Learning Based Methods. DANN [16] introduced adversarial training by making use of unlabeled target domain data. Earlier layers of the deep neural network architecture are made domain invariant through back-propagating a negative gradient using a jointly trained domain classifier. It uses 5000-dimension feature vector of the most frequent unigrams and bigrams as the input representation. DAmSDA [44], on the other hand, uses

mSDA [12] representation instead. We report the scores for DAmSDA and DANN from HATN [18]. For DANN, additionally, we create a customized implementation (**DANN⁺**) using BiLSTM and word vectors. This modified architecture simply consists of a shared BiLSTM layer followed by a dense layer for sentiment classification and the same BiLSTM layer followed by a gradient reversal layer and a dense layer for domain classification. Note that the accuracy for our improved DANN is **+3.2%** higher than what is reported in HATN.

Tri-training Based Methods. Multi-task tri-training (MT-Tri) [15] conducts tri-training on a multilayer perceptron model with an orthogonality loss between the final layers to enforce diversity between the jointly trained models. Unlabeled target data pseudo-labeled by the first two classifiers are fed to the third classifier. Three classifiers are optimized until none of the models’ predictions change. We improve upon this model (**MT-Tri⁺**) by using word vectors and BiLSTM.

Attention Based Methods. Recent works such as AMN [17], HATN [18], and IATN [19] use attention to identify sentiment pivots. Utilizing unlabeled target data, gradient reversal is an essential component in their models for domain classification. AMN expands DANN to an attention-based model. HATN improves AMN further by building a pivot and non-pivot networks. The pivot network (P-Net) performs the same task as AMN by extracting pivots. The non-pivot network (NP-Net) takes a transformed input that hides previously extracted pivots which is then jointly trained with P-Net. IATN⁶ incorporates ‘aspect’ information in addition to the sentence attentions. It reports a 0.8% increase in performance as compared to HATN (85.9% versus 85.1%). IATN uses the same input settings and the dataset as HATN with one difference: 200-dimensional word vectors instead of 300. Meanwhile, we use the exact same dataset and GoogleNews word vectors used by HATN for all our experiments for both reproducibility as well as blind comparison.

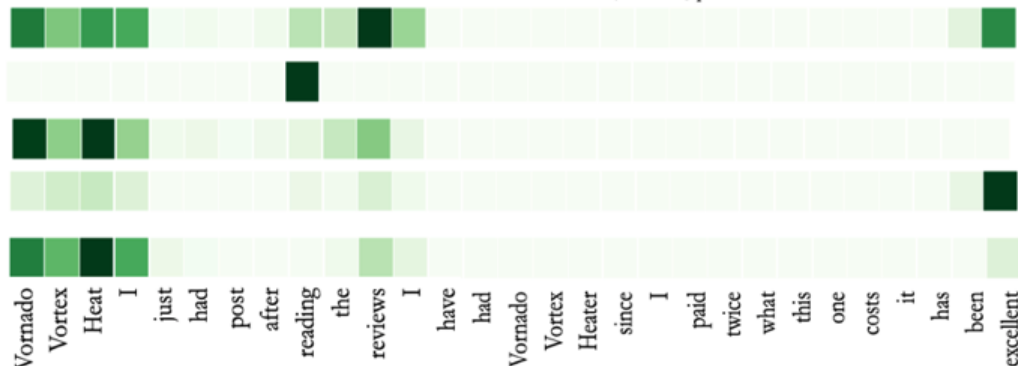
⁶<https://github.com/1146976048qq/IATN>

Review Text: Vornado Vortex Heat. I just had to post after reading the reviews. I have had a Vornado Vortex Heater since 1994, I paid twice what this one costs and it has been excellent. $y_{true} = +ve$

BiLSTM+ATT: $y_{pred} = -ve$

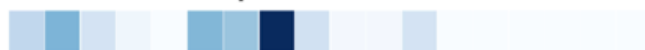


BiLSTM + MHAD with 5 diverse attention layers: $y_{pred} = +ve$



Review Text: Butter dish not tall enough. It looks nice but the lid touches the top of the butter and sticks to it. $y_{true} = -ve$

BiLSTM+ATT: $y_{pred} = +ve$



BiLSTM + MHAD with 5 diverse attention layers: $y_{pred} = -ve$

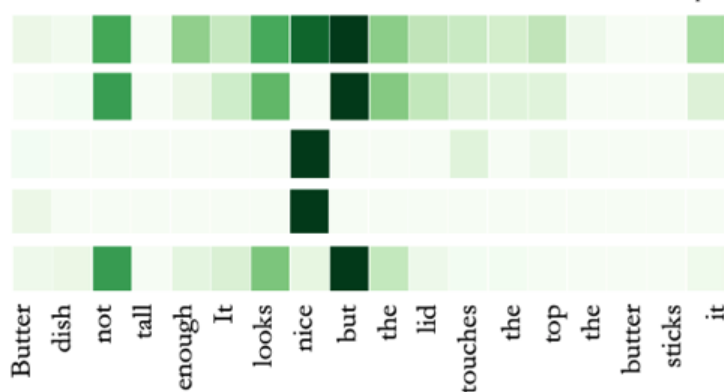


Figure 3.4: Two examples of *kitchen* review predictions by BiLSTM + ATT and BiLSTM + MHAD models trained on *book* reviews. When a single attention head fails to attend key words like ‘excellent’ or ‘but’, at least one of the diverse heads tends to make up for it.

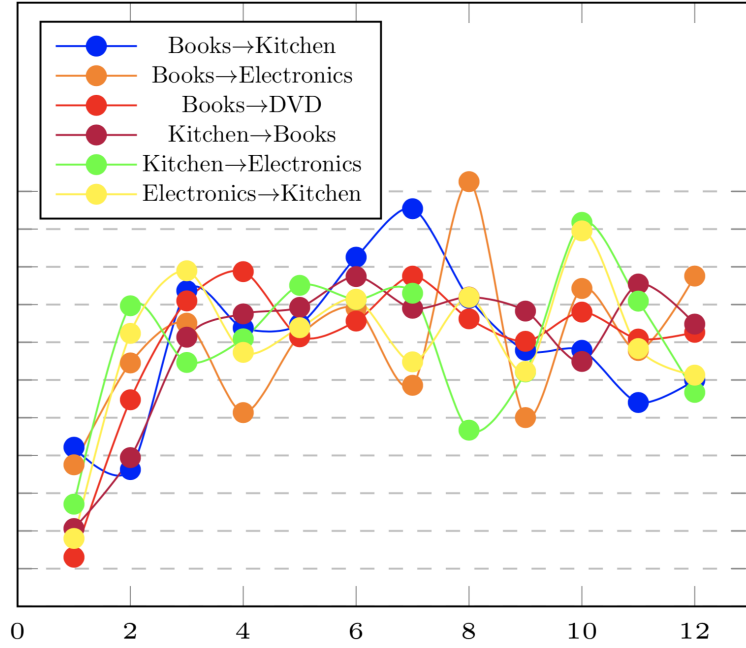


Figure 3.5: Performance change with increase in number of attention heads for six $S \rightarrow T$ combinations.

Strong Unsupervised Baselines. To study component-wise performance, we construct two strong unsupervised baselines from standard neural network architectures: BiLSTM and BiLSTM+ATT. BiLSTM consists of traditional BiLSTM units with the final unit making the prediction. BiLSTM+ATT, as shown in Fig. 3.1, adds a single attention layer on top of BiLSTM and the prediction is based on the output from the attention layer. Note that these two baselines still produce strong results and provide a reference for how much improvement following models make.

3.4 Results & Discussion

Tables 3.3, 3.4, and 3.5 show the competitive nature of our fully unsupervised methods when compared with the existing semi-supervised counterparts. Our experiments showed incrementally improving results when each component is added to the baselines. Attention with diversity improved the single-attention baseline and tri-training with diversity improved it

Table 3.3: Classification accuracy scores showing that unlabeled target data is not necessary to achieve strong performance. +: improved implementations, \diamond : reproduced implementations, \spadesuit : strong unsupervised baselines constructed from standard neural network architectures, *: reported scores from [18, 19] (see description). Our scores are averaged over 5 independent runs. **Note that only the models in the bottom table are fully unsupervised as they do not use any unlabeled target data.**

| S \rightarrow T | DANN* | DAmSDA* | IATN* | DANN ⁺ | MT-Tri ⁺ | AMN \diamond /P-Net | HATN \diamond |
|-------------------|-------|---------|-------|-------------------|---------------------|--------------------------|-----------------|
| B \rightarrow D | 83.42 | 86.12 | 86.80 | 82.85 | 84.67 | 87.07 | 87.70 |
| B \rightarrow E | 76.27 | 79.02 | 86.50 | 81.03 | 84.62 | 82.98 | 86.20 |
| B \rightarrow K | 77.90 | 81.05 | 85.90 | 82.01 | 84.78 | 84.85 | 87.08 |
| K \rightarrow B | 74.17 | 80.55 | 84.70 | 79.38 | 80.98 | 83.50 | 84.83 |
| K \rightarrow D | 75.32 | 82.18 | 84.40 | 79.04 | 78.89 | 82.83 | 84.73 |
| K \rightarrow E | 85.53 | 88.00 | 87.60 | 86.00 | 85.87 | 86.72 | 89.08 |
| E \rightarrow B | 73.53 | 79.92 | 81.80 | 78.92 | 80.64 | 83.28 | 83.62 |
| E \rightarrow K | 84.53 | 85.80 | 88.70 | 86.43 | 89.62 | 89.80 | 90.12 |
| E \rightarrow D | 76.27 | 82.63 | 84.10 | 77.83 | 79.97 | 83.37 | 83.87 |
| D \rightarrow B | 80.77 | 85.17 | 87.00 | 84.32 | 85.67 | 87.85 | 88.02 |
| D \rightarrow E | 76.35 | 76.17 | 86.90 | 81.74 | 84.48 | 84.65 | 86.78 |
| D \rightarrow K | 78.15 | 82.60 | 85.80 | 83.29 | 85.05 | 84.28 | 87.00 |
| AVG | 78.52 | 82.43 | 85.90 | 81.78 | 83.77 | 85.10 | 86.59 |

| S \rightarrow T | BiLSTM \spadesuit | BiLSTM +ATT \spadesuit | BiLSTM +MHA | BiLSTM +MHAD | BiLSTM +MHAD-Tri-I | BiLSTM +MHAD-Tri-II |
|-------------------|---------------------|-----------------------------|----------------|-----------------|-----------------------|------------------------|
| B \rightarrow D | 84.19 | 87.44 | 87.29 | 87.54 | 87.76 | 87.46 |
| B \rightarrow E | 83.61 | 83.90 | 85.36 | 85.63 | 85.75 | 86.08 |
| B \rightarrow K | 83.87 | 85.21 | 86.04 | 87.06 | 87.34 | 87.68 |
| K \rightarrow B | 80.52 | 82.15 | 83.11 | 83.70 | 84.19 | 84.23 |
| K \rightarrow D | 78.28 | 80.17 | 81.50 | 82.27 | 82.11 | 83.34 |
| K \rightarrow E | 86.33 | 87.30 | 88.60 | 88.81 | 88.98 | 89.22 |
| E \rightarrow B | 80.58 | 82.10 | 83.55 | 83.67 | 83.96 | 84.33 |
| E \rightarrow K | 88.07 | 88.19 | 89.61 | 89.96 | 90.07 | 91.05 |
| E \rightarrow D | 78.08 | 81.93 | 82.77 | 82.93 | 82.87 | 82.81 |
| D \rightarrow B | 83.93 | 87.72 | 87.77 | 88.22 | 88.51 | 88.74 |
| D \rightarrow E | 82.98 | 84.57 | 84.75 | 85.93 | 85.79 | 86.21 |
| D \rightarrow K | 84.38 | 85.45 | 86.50 | 86.73 | 86.74 | 87.37 |
| AVG | 82.90 | 84.68 | 85.57 | 85.98 | 86.17 | 86.54 |

even further. Using additionally available unlabeled source data proved to be fruitful for most of the domains. Note that for crisis dataset we only use *Harvey* for training because labeled data for *Florence* and *Irma* was just too low.

An implication of the diversity-based attention heads is shown in Fig. 3.4. Diversity

Table 3.4: Classification accuracy scores for crisis dataset.

| | HATN | BiLSTM +ATT | BiLSTM +MHA | BiLSTM +MHAD | BiLSTM +MHAD-Tri-I | BiLSTM +MHAD-Tri-II |
|-------------------|-------|----------------|----------------|-----------------|-----------------------|------------------------|
| H \rightarrow F | 80.01 | 74.88 | 74.32 | 75.69 | 76.00 | 78.11 |
| H \rightarrow I | 58.53 | 63.84 | 64.32 | 65.10 | 65.02 | 64.38 |

Table 3.5: Training time in d-hh:mm:ss for H \rightarrow F on a Dual Intel(R) Xeon(R) Gold 5120 CPU@2.2GHz with 28 cores and 1.5TB RAM.

| HATN | BiLSTM+MHA | BiLSTM+MHAD | BiLSTM+MHAD-Tri-I | BiLSTM+MHAD-Tri-II |
|------------|------------|-------------|-------------------|--------------------|
| 1-08:31:09 | 00:50:31 | 01:29:28 | 2:07:23 | 6:21:18 |

Table 3.6: Classification accuracy scores for three distinct combinations shown in the additional analysis.

| S \rightarrow T | P-Net | BiLSTM +MHAD-Tri-II |
|----------------------------------------------|-------|---------------------|
| <i>Electronics</i> \rightarrow <i>Yelp</i> | 88.45 | 89.15 |
| <i>Kitchen</i> \rightarrow <i>IMDb</i> | 76.38 | 78.33 |
| <i>Yelp</i> \rightarrow <i>IMDb</i> | 78.75 | 77.28 |

pushes the model not to rely on the same features. First example shows misclassification by a single attention model that attends incorrect sentiment words like ‘Vortex’ and ‘Heat’. However, with diversity, the model is lenient and look for alternate features. At least one of the T_y diverse heads tends to find important words like ‘excellent’. These examples also show that placing diversity on attention layers, rather than on any other hidden layers, provides an explainable understanding of which words the model deems to be important and can be used for subsequent pivot extraction like in AMN or HATN.

Computational Performance. To show that our work is practically useful for all communities alike, experiments are run on a CPU. A sample training time comparison is shown in Table 3.5. HATN needs gradient reversal to utilize unlabeled target data for the domain

classifier branch and pivot extraction for joint training; subsequently making it slower.

Gradient Reversal. To study the impact of gradient reversal procedure with BiLSTM, we conducted experiments with unlabeled target data. The performance of BiLSTM versus DANN⁺ (improved DANN) models in Table 3.3 showed that, with a good dropout value for the BiLSTM units, gradient reversal did not help much. On a similar note, domain adversarial loss was found not to be helpful in tri-training experiments [15]. In our context, we speculate that this might be because the dropout in the BiLSTM layer drops individual words that can lead to a better generalization which is essentially the purpose of gradient reversal. We plan to explore this problem in future for semi-supervised domain adaptation.

Additional Analysis: Generalizability of our models is further tested with three randomly selected experiments using very divergent domains such as Yelp⁷ restaurant reviews and IMDb [178] movie reviews in addition to Amazon reviews; *Electronics (Amazon) → Yelp*, *Kitchen (Amazon) → IMDb*, and *Yelp → IMDb*. We randomly selected 2000 positive and negative reviews from Yelp and IMDb. Their accuracy scores on our final model when compared to PNet⁸ of HATN is shown in Table 3.6. Once again, this shows that unlabeled target data is not always necessary; thus providing us with a fully unsupervised and computationally efficient alternative for domain adaptation in text classification tasks.

Analysis of Aggregation Methods. Attention layers can be aggregated in multiple ways such as ‘average’, ‘sum’, or ‘max’. In our experiments ‘sum’ produced slightly better results. A full comparison of the these three options on the Amazon dataset is shown in Table 3.7.

Adding more attention layers may become computationally expensive when compared to the single-layer counterpart. In order to find the optimal number of layers needed, we conducted experiments on two randomly picked combinations of Kitchen→Books and Books→DVD with layers varying from 1 to 12. The plots are shown in Fig. 3.5. We obtained

⁷<https://www.yelp.com/dataset/challenge>

⁸PNet is the first component of HATN which is computationally faster and within $\sim 1.5\%$ accuracy of HATN

Table 3.7: Evaluation of four attention aggregation methods

| S \rightarrow T | No Attention | | Single-Head | | MH-Sum | | MH-Mean | | MH-Max | | MH-Concat | |
|-------------------|--------------|-------|-------------|-------|--------------|--------------|---------|-------|--------|-------|-----------|-------|
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| B \rightarrow K | 84.45 | 84.52 | 87.22 | 87.23 | 87.55 | 87.59 | 86.97 | 86.96 | 86.84 | 87.17 | 86.88 | 86.86 |
| B \rightarrow E | 84.61 | 84.57 | 85.51 | 85.51 | 86.03 | 86.18 | 85.54 | 85.52 | 85.36 | 85.64 | 86.08 | 86.02 |
| B \rightarrow D | 83.52 | 83.13 | 86.32 | 86.24 | 87.30 | 87.37 | 87.26 | 87.27 | 87.15 | 87.19 | 87.18 | 87.38 |
| K \rightarrow B | 80.67 | 80.87 | 81.85 | 82.46 | 84.22 | 84.19 | 83.22 | 83.28 | 83.61 | 83.40 | 83.93 | 83.17 |
| K \rightarrow E | 87.37 | 87.39 | 87.09 | 87.05 | 88.48 | 88.58 | 87.04 | 86.95 | 87.66 | 86.66 | 88.41 | 88.36 |
| K \rightarrow D | 78.49 | 78.90 | 81.13 | 81.23 | 83.62 | 82.73 | 79.72 | 79.64 | 80.59 | 80.39 | 81.70 | 82.88 |
| E \rightarrow B | 81.18 | 81.24 | 81.50 | 81.71 | 83.53 | 83.87 | 82.55 | 81.95 | 82.69 | 82.23 | 83.39 | 83.35 |
| E \rightarrow K | 89.00 | 89.10 | 89.21 | 89.08 | 90.30 | 90.29 | 89.67 | 89.67 | 89.42 | 89.41 | 90.37 | 90.37 |
| E \rightarrow D | 78.46 | 77.63 | 81.37 | 80.86 | 82.60 | 82.55 | 81.57 | 81.42 | 79.32 | 78.04 | 81.64 | 81.74 |
| D \rightarrow B | 84.83 | 84.50 | 87.02 | 87.02 | 87.54 | 87.56 | 87.28 | 87.43 | 87.65 | 87.57 | 87.57 | 87.45 |
| D \rightarrow K | 85.21 | 85.23 | 86.37 | 86.27 | 87.27 | 87.32 | 86.63 | 86.71 | 86.67 | 86.65 | 87.34 | 87.20 |
| D \rightarrow E | 83.66 | 83.68 | 85.63 | 85.23 | 85.93 | 86.04 | 84.26 | 84.27 | 84.68 | 84.34 | 85.71 | 85.68 |
| AVG | 83.45 | 83.40 | 85.02 | 84.99 | 86.20 | 86.19 | 85.14 | 85.09 | 85.14 | 84.89 | 85.85 | 85.87 |

a similar result as [175] where the optimal number of T_α was around 5-6, past which the improvement seemed insignificant.

3.5 Key Takeaways

In this chapter, we showed that machine learning architectures designed for sufficient diversity can generalize better. Further, unlabeled target data, used often by state-of-the-art models, is not always necessary to produce strong performance for subjective text classification problems. We introduced a novel diversity-based generalization approach for the domain shift problem using a multi-head attention model where attention heads are constrained to learn differently such that the classifier can leverage on alternative features. Experiments on the standard benchmark dataset of Amazon reviews and a newly constructed dataset of Crisis events showed that our fully unsupervised methods can indeed match the competing semi-supervised baselines.

Chapter 4: Unsupervised and Interpretable Domain Adaptation: Application in Crisis Management

4.1 Summary

This chapter¹ [35] focuses on the problem of unsupervised and interpretable domain adaptation in the context of crisis tweet classification as shown in Fig. 4.1. During the onset of a natural or man-made crisis event, public often share relevant information for emergency services on social web platforms such as Twitter. However, filtering such relevant data in real-time at scale using social media mining is challenging due to the short noisy text, sparse availability of relevant data, and also, practical limitations in collecting large labeled data during an ongoing event. We hypothesize that unsupervised domain adaptation through multi-task learning can be a useful framework to leverage data from past crisis events for training efficient information filtering models during the sudden onset of a new crisis. We present a novel method to classify relevant social posts during an ongoing crisis without seeing any new data from this event (fully unsupervised domain adaptation). Specifically, we construct a customized multi-task architecture with a multi-domain discriminator for crisis analytics: *multi-task domain adversarial attention network* (MT-DAAN). This model consists of dedicated attention layers for each task to provide *model interpretability*; critical for real-world applications. As deep networks struggle with sparse datasets, we show that this can be improved by sharing a base layer for multi-task learning and domain adversarial training. The framework is validated with the public datasets of TREC incident streams that provide labeled Twitter posts (tweets) with relevant classes (*Priority*, *Factoid*, *Sentiment*) across 10 different crisis events such as floods and earthquakes. Evaluation of

¹Published in the proceedings of the 2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). URL: <https://arxiv.org/pdf/2003.04991.pdf>

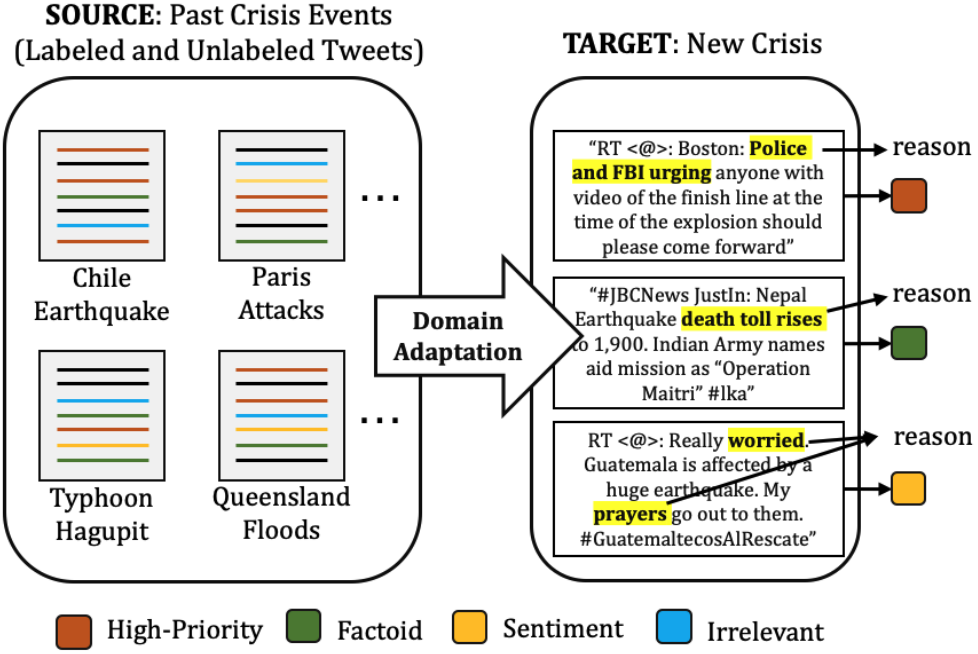


Figure 4.1: **Problem Statement:** Interpretably predict labels for tweets collected during an ongoing crisis using only the past crisis data, given a) unavailability of labeled data in the ongoing event, and b) need for interpretability of machine reasoning behind data filtering for emergency managers.

domain adaptation for crisis events is performed by choosing one target event as the test set and training on the rest. Our results show that the multi-task model outperformed its single-task counterpart. For the qualitative evaluation of interpretability, we show that the attention layer can be used as a guide to explain the model predictions and empower emergency services for exploring accountability of the model, by showcasing the words in a tweet that are deemed important in the classification process. Finally, we show a practical implication of our work by providing a use-case for the COVID-19 pandemic.

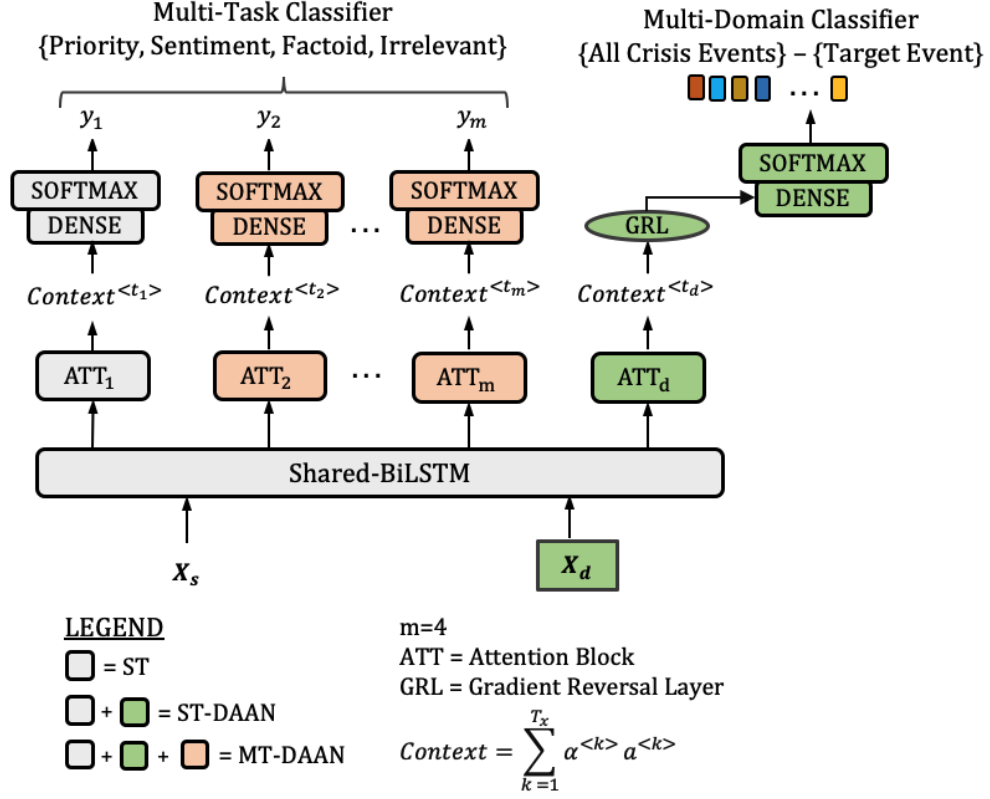


Figure 4.2: Fully Unsupervised Domain Adaptation Set-up for Multi-Task Crisis Tweet Classification.

4.2 Methodology

4.2.1 Problem Definition

Using notations in Table 4.1, consider a set C of all crisis events such as *Guatemala Earthquake* or *Typhoon Yolanda*. The task of unsupervised domain adaptation for crisis analytics is to train a classifier for a specific target crisis (c_t) using labeled (L_{C-c_t}) and unlabeled (U_{C-c_t}) data from all other crises; where $C - c_t$ denotes the set of all crisis events minus the target crisis. We assume that **no** data record from the target crisis is available for training. Following the traditional domain adaptation terminology, $X_s = L_{C-c_t}$ represents the labeled data from the source domain S and $Y_s = y_{C-c_t}$ represents the ground truth labels on which the classifier is trained. And, $X_t = L_{c_t}$ represents the labeled data from

Table 4.1: Notations (Top 5 rows contain new notations introduced).

| Notation | Definition |
|----------------|------------------------------------------------|
| C | Set of all crisis events $\{c_1, c_2, \dots\}$ |
| L_{c_k} | Set of labeled data from the event c_k |
| y_{c_k} | Set of ground truth labels for L_{c_k} . |
| m | Number of tasks (Number of bits in each label) |
| U_{c_k} | Set of unlabeled data from the event c_k |
| T_x | Number of words in a sentence |
| $x^{<k>}$ | k -th word of a sentence |
| $\alpha^{<k>}$ | attention from k -th word |
| $a^{<k>}$ | BiLSTM activation from k -th word |

the target domain T and $Y_t = y_{c_t}$ represents the ground truth labels; both of which are only used for testing the classifier. $X_d = U_{C-c_t}$ represents the unlabeled data from different domains minus the target. To summarize:

Input: X_s, Y_s, X_d

Output: $Y_t^{pred} \leftarrow predict(X_t)$.

4.2.2 Models & Concepts

In the following sections, we describe three models: Single-Task Attention Network (ST), Single-Task Domain Adversarial Attention Network (ST-DAAN), and Multi-Task Domain Adversarial Attention Network (MT-DAAN). ST is the model we adopt from [34] to build the single-task attention based baseline. ST-DAAN is constructed on top of ST to make the model domain agnostic by performing adversarial training using gradient reversal. Finally, MT-DAAN is constructed on top of ST-DAAN with dedicated attention layers for each task on a shared BiLSTM layer. This is shown in Fig. 4.2.

Single-Task Attention Network (ST). We first describe the single-task attention network [34] on top of which we build our models. This model aligns with our goals of interpretability and unsupervised domain adaptation. This BiLSTM based model with Attention gives us three main advantages:

1. Unlike several existing domain adaptation methods that use unlabeled target data to train the domain adversarial component via gradient reversal, this method is a fully unsupervised baseline which also can be customized for multi-task learning.
2. The method uses attention mechanism which in turn weighs each word in a sentence based on its importance. This can be directly utilized for interpretability.
3. The method also runs much faster (only a few minutes), i.e. highly useful in crisis times, as compared to the top performing semi-supervised models such as HATN [18] (hours).

This model [34] consists of a BiLSTM layer which produces T_x activations, each corresponding to a word in the sentence. These activations are passed through *dense* and *softmax* layers and are combined by dot product to produce the context vector $\sum_{k=1}^{T_x} \alpha^{<k>} a^{<k>}$, where $a^{<k>}$ is the BiLSTM activation from k -th word and $\alpha^{<k>}$ is the attention weight of k -th word. Sentences with words greater than T_x are stripped and those with words lower than T_x are padded. This single-task ($m = 1$) attention network is the building block with which rest of the following models are constructed. The single-task loss function is shown below using the standard binary cross entropy loss.

$$L_T = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)] \quad (4.1)$$

where T represents the task, y is the true label, and \hat{y} is the predicted label.

Single-Task Domain Adversarial Attention Network (ST-DAAN). To study the specific contribution of domain adversarial training, we construct a secondary baseline over the ST architecture by constructing an additional branch with gradient reversal layer which is represented by the green blocks in Fig. 4.2. This is a single-task binary classifier with $m = 1$. Domain Adversarial Training of Neural Networks (DANN) [44] was introduced with a goal to confuse the classifier by back-propagating a negative gradient from a separate domain classifier branch (right-most branch, as shown in Fig. 4.2). This makes the classifier agnostic to difference in domains. This back-propagation is implemented using a *gradient reversal layer* [44] which does nothing during the forward pass but pushes a negative gradient ($-\lambda \frac{\partial L_d}{\partial \theta_f}$) during the backward (gradient update) pass. L_d is the domain classification loss, λ is the strength of the reversal, and f represents the lower level layers or features over which the negative gradient update is performed. In our architecture, the goal is to make the BiLSTM layer indiscriminate towards various crisis domains such that the multi-task classification does not depend on the domain from which the tweet/sentence is coming from. The ST-DAAN loss function is shown below.

$$L'_T = L_T + w_d L_d \quad (4.2)$$

where w_d is the domain adversarial loss weight. L_d represents the categorical cross entropy loss for multi-domain discriminator shown below.

$$L_d = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^{|C-c_t|} [y_{ij} \log \hat{y}_{ij}] \quad (4.3)$$

where $C - c_t$ is the set of all crisis events without the target event.

Multi-Task Domain Adversarial Attention Network (MT-DAAN). Building on top of ST-DAAN, we construct MT-DAAN, which is intended to classify problems with multiple tasks or labels. For each task, a dedicated attention layer is allocated from which

it predicts binary labels. The BiLSTM layer remains exactly the same as in the single-task model but multiple attention blocks are added for each task along with a domain classifier. In the architecture decision process, we first investigated a multi-label classifier where all layers are shared with the final softmax layer making multi-label predictions. In low resource settings, constructing a multi-label classifier using a shared architecture is challenging for two reasons: **a)** jointly balancing positive and negative samples across all classes is not trivial and potentially challenging to make it extensible when new classes need to be added, and **b)** attention layer may not always produce class-specific insights as the weights are assigned to train for the combination of labels. On the other hand, in the multi-task architecture with separate attention layers, it is easy to add more classes. If some classes require more training, it is trivial to further tune a model specific to that class. More importantly, $context^{<t_j>}$ vector for j -th task identifies the influential words from each sentence for that specific task. The complete architecture is shown in Fig. 4.2. MT-DAAN loss function is shown below:

$$L_{MT-DAAN} = \sum_{k=1}^m (w_k L_{T_k}) + w_d L_d \quad (4.4)$$

where m is the number of tasks, w_k is the loss weight and L_{T_k} is the loss term for each task, w_d is the domain adversarial loss weight, and L_d is the domain adversarial loss term.

Model Interpretability. The output (α) of the attention layer (ATT) of each task, is a T_x -dimensional vector; T_x being the number of words in the sentence. The context vector ($\sum_{k=1}^{T_x} \alpha^{<k>} a^{<k>}$) is the product of these attention weights and the T_x -dimensional activation (a) from the *BiLSTM* layer. α essentially weighs how much each word in the sentence contributes to the classification result. Thus, α is the component that is evaluated for model interpretability.

Table 4.2: TREC Dataset Statistics; Showing the number of positive samples for each of the 4 classes. P =Priority, F =Factoid, S =Sentiment, and I =Irrelevant.

| CRISIS EVENTS | Total Tweets | Vocab | Avg #words | P | F | S | I |
|------------------------------|--------------|-------|------------|------|-----|-----|-----|
| 2012 Guatemala Earthquake | 154 | 422 | 18.74 | 104 | 108 | 12 | 15 |
| 2013 Typhoon Yolanda | 564 | 1746 | 19.47 | 249 | 46 | 119 | 51 |
| 2013 Australia Bushfire | 677 | 2102 | 20.21 | 152 | 213 | 167 | 36 |
| 2013 Boston Bombings | 535 | 1755 | 19.30 | 147 | 28 | 234 | 198 |
| 2013 Queensland Floods | 713 | 2301 | 19.08 | 293 | 54 | 173 | 215 |
| 2014 Chile Earthquake | 311 | 919 | 16.54 | 48 | 26 | 50 | 10 |
| 2014 Typhoon Hagupit | 1470 | 2893 | 15.36 | 469 | 375 | 276 | 101 |
| 2015 Nepal Earthquake | 2048 | 4026 | 13.77 | 1067 | 377 | 741 | 133 |
| 2015 Paris Attacks | 2066 | 4152 | 18.62 | 306 | 183 | 782 | 429 |
| 2018 Florida School Shooting | 1118 | 2940 | 21.40 | 329 | 64 | 206 | 70 |

4.3 Experimental Evaluation

4.3.1 Datasets

TREC Dataset. TREC-IS² (Text Retrieval Conference - Incident Streams) is a program that encourages research in information retrieval from social media posts with the goal to improve the state-of-the-art social media based crisis analytics solutions. We use the dataset from 2018 track proposal. Statistics of this curated dataset of Twitter downloaded from TREC is shown in Table 4.2. The original dataset consisted of 15 crisis events. However, due to very low data, we trimmed the events and tasks such that there are at least 10 positive samples for each task.

The four tasks used in our experiments are shown below:

1. *Priority*: Different priority levels are assigned for each tweet: *low*, *medium*, *high*, *critical*. We convert this into a binary classification problem where $low = 0$ and $\{medium, high, critical\} = 1$.
2. *Factoid*: ‘Factoid’ is a categorical label that represents if a tweet is stating a fact. Eg: ‘*death toll rises ...*’

²http://dcs.gla.ac.uk/~richardm/TREC_IS/

3. *Sentiment*: ‘Sentiment’ is a categorical label that represents if a tweet represents a sentiment. Eg: ‘*Worried.. Thoughts and prayers.*’
4. *Irrelevant*: ‘Irrelevant’ is a categorical label for tweets that do not provide any relevant information.

Amazon Reviews Dataset. The standard benchmark dataset³ of Amazon reviews [27] is widely used for cross-domain sentiment analysis. We chose four domains: Books (B), Kitchen (K), DVD (D), and Electronics (E). The raw data⁴, a part of Blitzer’s original raw dataset, used in this work is from HATN [18]. This dataset consists of 3000 positive and 3000 negative samples for each of the 4 domains. This dataset is used for two purposes: 1) to validate the performance of the state-of-the-art methods including the single-task baseline and 2) to compare and contrast the performance of deep models when trained with rich versus sparse datasets.

COVID-19 Tweet Dataset. For the COVID-19 use-case, we use Twitter posts collected using CitizenHelper [179] system in March 2020, for the geo-bounding box of the Washington D.C. Metro region. These tweets were annotated by volunteers of regional Community Emergency Response Teams (CERTs), with ‘*Relevant*’ label denoting how relevant a tweet is for crisis response operations. The label values range on a scale of 1-4. We convert them into binary classes by considering values 1 and 2 as $-ve$ (0) class and values 3 and 4 as $+ve$ (1) class. This dataset consists of 4911 tweets with $-ve$ ($Relevant=0$) and 637 tweets with $+ve$ ($Relevant=1$) classes. Following unsupervised domain adaptation criteria, the filtering models are trained using only the TREC dataset and evaluated on the COVID-19 tweets. For each independent run of the experiment, a balanced subset of size 637 for both classes is selected for testing.

³<http://www.cs.jhu.edu/~mdredze/datasets/sentiment/>

⁴https://github.com/hsqmlzno1/HATN/tree/master/raw_data

4.3.2 Baselines

1. **Simple Baselines:** We construct simple baseline classifiers [180]: Logistic Regression (LR) and Support Vector Machines (SVM). The input to these models are constructed by aggregating the 300-dimensional word embeddings of words in each review.

2. **CNN:** A standard Convolutional Neural Network inspired by Kim, 2014 [181] is constructed with the following architecture:

$Word\ Embeddings(T_x, 300) \rightarrow Conv1D(128, 5)$
 $\rightarrow MaxPooling1D(5) \rightarrow Conv1D(128, 5)$
 $\rightarrow MaxPooling1D(5) \rightarrow Conv1D(128, 5)$
 $\rightarrow GlobalMaxPooling1D() \rightarrow Dense(128)$
 $\rightarrow Dense(2) \rightarrow y.$

This is combined with dropouts, *relu* activations, and ending with *softmax* activation producing labels for binary classification. State-of-the-art deep learning methods for existing social media mining approaches of crisis analytics [39, 58] use a similar architecture.

3. **BiLSTM:** This is the bottom-most layer in Fig. 4.2 with the activation $a^{<T_x>}$ passed through the following: $Dense(10) \rightarrow Dense(2) \rightarrow y$ also including dropouts, *relu* activation, and ending with *softmax*.
4. **AMN and HATN:** AMN [17] and HATN [18] are attention-based methods which use gradient reversal to perform domain adversarial training on the unlabeled data from source and target domains. HATN is an extension to AMN by adding the hierarchical component and jointly training pivot and non-pivot networks.

4.3.3 Experiments

Pre-processing. A tweet, as it gets broken down into tokens, undergoes the following pre-processing steps:

Table 4.3: Implementation Details for MT-DAAN

| | |
|-------------------------|------------------------------------------------------------------------------|
| T_x | 200 (Amazon Reviews), 30 (Tweets) |
| Deep Learning Library | Keras |
| Optimizer | Adam [$lr = 0.005$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $decay = 0.01$] |
| Maximum Epoch | 50 |
| Dropout | 0.4 |
| Early Stopping Patience | 3 |
| Batch Size | 32 |
| Validation Split | 0.15 |

1. Contractions such as *didn't* is expanded to *did not* using a basic English contractions dictionary.
2. Simple misspellings are corrected. For example, words with 3 of the same characters appearing consecutively like *goood* is changed to *good*.
3. Numbers and words with no alpha-numeric characters are converted to special tags.
4. Hashtags (or just the # symbol) can optionally be removed.
5. Stop words are selectively removed keeping words such as ‘*but*’ and ‘*not*’ which can impact sentiment phrases such as ‘*not good*’.

Choice of Word Embeddings. We use fastText [38] as our word embeddings for tweets because of its sub-word usage and the ability to create vectors for arbitrary and out-of-vocabulary words. Although there exists many alternatives, picking the one that works well for a specific dataset is not trivial. In the next section (5.4), we compare and contrast the performance of three more options for word vectors: **a)** GoogleNews [48], **b)** GloVe Twitter Embeddings [14], and **c)** CrisisNLP Embeddings [182]. Unlike fastText, we fine-tune these pre-trained vectors using Gensim [183] to create vectors for out-of-vocabulary

words. Vectors for words that are already in the vocabulary are *locked* while tuning for consistency in evaluation.

We first validate the performance of the adopted unsupervised ST model [34] by comparing it with the following standard neural network architectures and state-of-the-art models used for domain adaption in text. We use the standard benchmark dataset of Amazon reviews. Following the traditional domain adaptation experimental setup, each experiment represented as $S \rightarrow T$ consists of a source domain (S) on which the model is trained and a target domain (T) on which the model is tested. Implementation details is shown in Table 4.3.

Input to all the models are word vectors⁵ [48]. The evaluation on amazon reviews shows how well the single-task (ST) model perform when compared to the existing top-performing domain adaptation models on benchmark dataset. Table 4.4 shows accuracy scores on the Amazon cross-domain sentiment analysis dataset. HATN uses unlabeled target data, gradient reversal, explicit pivot extraction, and joint training making it a computationally expensive method. As shown in the experimental evaluation, we use the same Amazon dataset and GoogleNews word vectors for our experiments. ST, being *unsupervised* with no need of unlabeled target data, performed competitively with an overall accuracy of 85.02%; thus establishing a strong fully unsupervised building block for us to build upon.

4.4 Results & Discussion

Crisis Tweets vs Amazon Reviews. Table 4.4 and 4.5 show that deep models struggle with small datasets such as TREC-IS tweets. When ST model outperformed Logistic Regression by $\sim 8\%$ on the Amazon reviews dataset, the difference was only less than 1% with no statistical significance on the TREC-Priority dataset. Note that we conduct experiments with various parameter combinations on the deep models when using tweets. For example, $T_x = 200$ for amazon reviews and $T_x = 30$ for tweets due to the difference in their average word-length. *Books* domain of Amazon reviews has 182 average number of

⁵<https://code.google.com/archive/p/word2vec/>

Table 4.4: Performance comparison (accuracy) various models on the standard benchmark dataset of amazon reviews. Blue colored methods do not use any unlabeled target data; hence relevant in our context. Each reported score is an average of 10 independent runs of each experiment.

| S \rightarrow T | LR | SVM | CNN | BiLSTM | AMN | HATN | BiLSTM+ ATT |
|-------------------|-------|-------|-------|--------|-------|-------|-------------|
| B \rightarrow K | 76.40 | 75.95 | 81.20 | 84.45 | 81.88 | 87.03 | 87.22 |
| B \rightarrow E | 75.53 | 74.05 | 80.44 | 84.61 | 80.55 | 85.75 | 85.51 |
| B \rightarrow D | 81.08 | 81.43 | 82.94 | 83.52 | 85.62 | 87.07 | 86.32 |
| K \rightarrow B | 76.12 | 75.78 | 78.78 | 80.67 | 79.05 | 84.88 | 81.85 |
| K \rightarrow E | 80.37 | 81.20 | 85.17 | 87.37 | 86.68 | 89.00 | 87.09 |
| K \rightarrow D | 73.32 | 74.98 | 76.41 | 78.49 | 79.50 | 84.72 | 81.13 |
| E \rightarrow B | 74.85 | 74.18 | 78.08 | 81.18 | 77.52 | 84.03 | 81.50 |
| E \rightarrow K | 81.85 | 81.85 | 86.59 | 89.00 | 87.83 | 90.08 | 89.21 |
| E \rightarrow D | 75.82 | 75.83 | 78.35 | 78.46 | 85.03 | 84.32 | 81.37 |
| D \rightarrow B | 81.17 | 82.20 | 82.26 | 84.83 | 84.53 | 87.78 | 87.02 |
| D \rightarrow K | 76.42 | 77.58 | 81.09 | 85.21 | 81.67 | 87.47 | 86.37 |
| D \rightarrow E | 72.47 | 73.68 | 79.56 | 83.66 | 80.42 | 86.32 | 85.63 |
| AVG | 77.12 | 77.39 | 80.91 | 83.45 | 82.52 | 86.54 | 85.02 |

tokens per review with a vocab size of 105920. On the other hand, the event with highest number of tweets in the TREC dataset (Paris Attacks) has only 18.62 average number of tokens per tweet with a vocab size of 4152. This difference makes it intuitively challenging to train deep models with several parameters that may lead the model to memorize the entire dataset resulting in poor generalization. Multi-task learning and domain adversarial training try to alleviate this problem by training the shared BiLSTM layer with much more data from different tasks and unlabeled data.

MT-DAAN Performance Evaluation. The primary purpose of the MT-DAAN model is to show that sharing the bottom layer of the model (i.e., shared representation) for different tasks along with domain adversarial training can help improve the generalizability of some of the tasks that are otherwise trained alone in the single-task model. The experiments for MT-DAAN are setup in the same unsupervised way as for single-task. No data from the test crisis is used for training. For example, if we are testing our model for the event ‘*Typhoon Yolanda*’, no data from this crisis is used for training. Note that the domain classifier component uses unlabeled data only from rest of the crisis; making it a

Table 4.5: Performance comparison (accuracy) of unsupervised models on TREC-Priority dataset showing that deep models are **not** strictly superior than simpler models due to data sparsity. Each reported score is an average of 10 independent runs of each experiment. *Source = Everything - Target.*

| Target | LR | SVM | CNN | BiLSTM | BiLSTM+ ATT |
|-----------------------------|-------|-------|-------|--------|-------------|
| Guatemala Earthquake (G) | 60.14 | 56.76 | 60.47 | 65.54 | 59.97 |
| Typhoon Yolanda (Ty) | 65.39 | 65.97 | 63.05 | 65.49 | 65.53 |
| Australia Bushfire (A) | 65.61 | 63.23 | 62.10 | 60.10 | 62.44 |
| Boston Bombings (B) | 71.47 | 75.45 | 69.72 | 71.43 | 72.08 |
| Queensland Floods (Q) | 65.56 | 64.81 | 64.13 | 66.01 | 66.21 |
| Chile Earthquake (C) | 43.09 | 37.94 | 43.37 | 35.45 | 39.23 |
| Typhoon Hagupit (Th) | 49.86 | 46.22 | 49.21 | 54.13 | 52.61 |
| Nepal Earthquake (N) | 57.11 | 55.39 | 58.61 | 60.49 | 61.35 |
| Paris Attacks (P) | 71.43 | 71.72 | 72.50 | 72.14 | 71.31 |
| Florida School Shooting (F) | 58.79 | 63.02 | 58.82 | 59.71 | 60.55 |
| AVG | 60.85 | 60.05 | 60.20 | 61.05 | 61.13 |

Table 4.6: Unsupervised domain adaptation results on TREC dataset showing performance boost for *Priority*, *Factoid*, and *Irrelevant* tasks. However, *Sentiment* task did not show a significant improvement. See performance evaluation section for details. Each reported score is an average of 10 independent runs of each experiment.

| TARGET | Priority | | | | | | Factoid | | | | | |
|--------|----------|-------|---------|-------|--------------|--------------|---------|-------|---------|-------|--------------|--------------|
| | ST | | ST-DAAN | | MT-DAAN | | ST | | ST-DAAN | | MT-DAAN | |
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| G | 59.97 | 62.39 | 69.07 | 69.66 | 69.05 | 69.34 | 68.92 | 68.47 | 79.90 | 80.76 | 84.05 | 97.01 |
| Ty | 65.53 | 65.47 | 66.07 | 63.73 | 67.42 | 67.30 | 80.50 | 84.42 | 82.71 | 85.61 | 84.36 | 86.93 |
| A | 62.44 | 66.69 | 61.07 | 63.42 | 61.93 | 64.28 | 64.58 | 60.69 | 65.64 | 60.53 | 65.04 | 60.13 |
| B | 72.08 | 74.29 | 72.34 | 73.37 | 73.80 | 74.74 | 83.10 | 88.51 | 81.42 | 85.90 | 85.82 | 88.82 |
| Q | 66.21 | 65.94 | 67.19 | 66.97 | 66.74 | 66.46 | 37.56 | 48.90 | 50.46 | 59.82 | 49.52 | 59.21 |
| C | 39.23 | 40.92 | 38.91 | 42.37 | 41.80 | 46.33 | 30.38 | 33.97 | 39.87 | 48.68 | 45.28 | 54.58 |
| Th | 52.61 | 50.59 | 58.97 | 58.94 | 57.50 | 57.52 | 68.98 | 70.79 | 71.42 | 72.44 | 69.49 | 70.08 |
| N | 61.35 | 59.44 | 60.18 | 57.80 | 61.65 | 59.49 | 74.04 | 76.08 | 80.72 | 81.00 | 81.04 | 81.02 |
| P | 71.31 | 76.26 | 70.42 | 74.08 | 74.44 | 77.21 | 75.78 | 80.35 | 82.35 | 84.89 | 82.52 | 85.63 |
| F | 60.55 | 61.75 | 65.47 | 64.07 | 62.51 | 63.24 | 76.73 | 82.67 | 84.55 | 87.51 | 85.80 | 88.15 |
| AVG | 61.13 | 62.37 | 62.97 | 63.44 | 63.68 | 64.59 | 66.06 | 69.49 | 71.90 | 74.71 | 73.29 | 77.16 |

| TARGET | Sentiment | | | | | | Irrelevant | | | | | |
|--------|-----------|-------|---------|--------------|--------------|-------|------------|--------|---------|-------|--------------|--------------|
| | ST | | ST-DAAN | | MT-DAAN | | ST | | ST-DAAN | | MT-DAAN | |
| | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| G | 96.96 | 97.03 | 96.45 | 96.68 | 96.76 | 92.73 | 89.36 | 89.03 | 91.22 | 91.06 | 93.11 | 92.73 |
| Ty | 75.81 | 77.62 | 77.54 | 79.01 | 76.82 | 78.35 | 76.05 | 79.77 | 78.49 | 80.59 | 80.46 | 82.31 |
| A | 75.95 | 77.58 | 78.80 | 79.12 | 78.54 | 78.92 | 35.42 | 47.164 | 53.78 | 65.11 | 51.76 | 63.36 |
| B | 81.39 | 81.11 | 80.73 | 80.70 | 82.13 | 82.10 | 58.15 | 55.73 | 58.15 | 57.43 | 61.49 | 61.45 |
| Q | 81.69 | 80.39 | 81.05 | 81.39 | 81.53 | 81.32 | 65.68 | 65.36 | 67.26 | 65.72 | 67.88 | 67.27 |
| C | 92.69 | 92.91 | 93.10 | 93.21 | 93.62 | 93.68 | 75.16 | 84.98 | 80.46 | 86.38 | 80.64 | 86.56 |
| Th | 84.98 | 85.86 | 85.15 | 86.14 | 85.43 | 86.38 | 63.21 | 75.04 | 71.50 | 78.25 | 70.22 | 77.27 |
| N | 67.75 | 68.42 | 70.20 | 70.51 | 69.96 | 70.31 | 31.79 | 42.10 | 36.97 | 47.41 | 41.49 | 52.87 |
| P | 76.01 | 76.63 | 73.65 | 73.98 | 74.47 | 74.60 | 33.91 | 35.25 | 44.52 | 48.32 | 47.17 | 51.32 |
| F | 68.77 | 71.77 | 67.06 | 70.03 | 68.14 | 71.05 | 32.66 | 40.90 | 44.22 | 55.27 | 47.64 | 58.65 |
| AVG | 80.20 | 80.93 | 80.37 | 81.08 | 80.74 | 80.94 | 56.14 | 61.53 | 62.66 | 67.55 | 64.19 | 69.38 |

fully unsupervised domain adaptation approach. Performance scores of the four tasks (*Priority*, *Factoid*, *Sentiment*, and *Irrelevant*) are shown in Table 4.6. The results show clear

Table 4.7: Unsupervised domain adaptation results for COVID-19 tweets using only the TREC dataset for training. Each reported score is an average of 10 independent runs of each experiment.

| TARGET | Relevant | | | | | |
|----------|----------|-------|---------|-------|---------|-------|
| | ST | | ST-DAAN | | MT-DAAN | |
| | Acc | F1 | Acc | F1 | Acc | F1 |
| COVID-19 | 73.25 | 77.36 | 74.55 | 77.51 | 77.00 | 78.09 |

Table 4.8: Performance comparison (accuracy) of four relevant word embedding models on TREC-Sentiment task showing that the tweet-based embeddings such as Glove or CrisisNLP did not significantly outperform other models.

| TARGET | fastText [38] | GoogleNews [48] | Glove [14] | CrisisNLP [182] |
|-------------------------|---------------|-----------------|------------|-----------------|
| Guatemala Earthquake | 96.96 | 95.72 | 95.27 | 97.97 |
| Typhoon Yolanda | 75.81 | 83.30 | 86.49 | 79.41 |
| Australia Bushfire | 75.95 | 79.35 | 80.24 | 75.86 |
| Boston Bombings | 81.39 | 82.43 | 80.55 | 81.16 |
| Queensland Floods | 81.69 | 84.06 | 84.01 | 81.50 |
| Chile Earthquake | 92.69 | 92.82 | 92.93 | 92.60 |
| Typhoon Hagupit | 84.98 | 87.55 | 84.25 | 88.76 |
| Nepal Earthquake | 67.75 | 68.46 | 73.63 | 67.50 |
| Paris Attacks | 76.01 | 77.67 | 74.49 | 77.43 |
| Florida School Shooting | 68.77 | 66.84 | 66.97 | 65.06 |
| AVG | 80.20 | 81.82 | 81.88 | 80.73 |

performance improvement for *Priority*, *Factoid*, and *Irrelevant* tasks. However, *Sentiment* task did not show significant improvement. We speculate that this is because other tasks do not generalize the bottom layer enough to boost the sentiment classification performance. These results show the usefulness of multi-task learning as well as domain adversarial training where different tasks in multiple domains help each other when the data is sparse and labels are limited.

MT-DAAN Hyperparameters. Apart from the generic hyperparameters mentioned in Table 4.3, multi-task learning has specific hyperparameters that can be tuned to improve performance. The domain adversarial training hyperparameters, w_d and λ , are set to 0.1

and 0.4 respectively (similar to *ST-DAAN*). w_d is the domain adversarial loss weight and λ is the strength of the reversal (refer section 3.3 for more details). Furthermore, the loss weights (w_p , w_f , w_s , and w_i) corresponding to the 4 tasks (*Priority*, *Factoid*, *Sentiment*, and *Irrelevant*), can be set differently to give prominence to the task under consideration. A baseline setting is to set all weights to 1.0. However, this forces the model to focus on all tasks equally. To provide more flexibility for the model to learn specific tasks, the weights can be changed accordingly. For example, $[w_p=1.0, w_f=0.3, w_s=0.1, \text{ and } w_i=0.6]$ is a *Priority* task which gives more prominence to *Irrelevant* class label and less prominence to *Sentiment* class label. We perform a simple grid search for values in range $[0.1-1.0]$ with 0.1 interval to find corresponding weight combination for each task.

MT-DAAN with Additional Web Datasets. In order to understand if additional web resources can further improve the performance of MT-DAAN, we expand it by adding new tasks from Amazon review dataset and Sentiment140 dataset⁶. Architecturally, this adds two more attention branches in Fig. 4.2 and two more domains for the domain classifier. We select *Priority* as our primary classification task and design 5 more experiments:

1. Single Task: Train solely on Priority.
2. MTL: Multi-Task Learning where other classes such as *Factoid*, *Sentiment*, and *Irrelevant* are jointly trained.
3. MTL+Amazon: Multi-Task Learning with an additional task to jointly train and classify Amazon reviews.
4. MTL+Sentiment140: Multi-Task Learning with an additional task to jointly train and classify Sentiment140 positive/negative reviews.

Results are shown in Table 4.9. Both Amazon reviews and Sentiment140 datasets proved to be useful additions to the MTL setup showing that adding additional web resources may in fact help.

⁶<http://help.sentiment140.com/for-students>

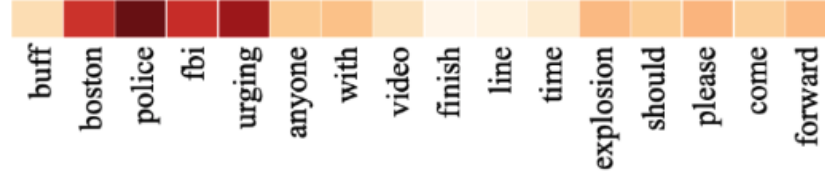
Table 4.9: Performance increases when additional web data is added for classifying TREC *Priority* task. Similar to Table 4.6, scores shown are average and standard deviation of 10 independent runs of each experiment.

| TARGET | MT-DAAN | | +Amazon | | +Sentiment140 | |
|-------------------------|---------|-------|--------------|--------------|---------------|--------------|
| | Acc | F1 | Acc | F1 | Acc | F1 |
| Guatemala Earthquake | 69.05 | 69.34 | 69.59 | 69.84 | 75.00 | 74.45 |
| Typhoon Yolanda | 67.42 | 67.30 | 65.39 | 64.90 | 67.21 | 66.95 |
| Australia Bushfire | 61.93 | 64.28 | 63.39 | 65.85 | 62.05 | 64.47 |
| Boston Bombings | 73.80 | 74.74 | 73.14 | 74.30 | 72.03 | 73.08 |
| Queensland Floods | 66.74 | 66.46 | 67.82 | 67.83 | 65.26 | 65.02 |
| Chile Earthquake | 41.80 | 46.33 | 52.89 | 58.93 | 41.32 | 46.00 |
| Typhoon Hagupit | 57.50 | 57.52 | 57.01 | 57.33 | 59.99 | 60.41 |
| Nepal Earthquake | 61.65 | 59.49 | 60.42 | 57.60 | 61.25 | 59.04 |
| Paris Attacks | 74.44 | 77.21 | 67.46 | 72.01 | 75.67 | 78.02 |
| Florida School Shooting | 62.51 | 63.24 | 65.58 | 65.04 | 66.59 | 65.79 |
| AVG | 63.68 | 64.59 | 64.27 | 65.36 | 64.64 | 65.32 |

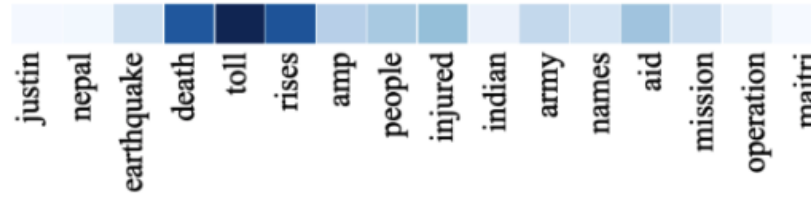
Word Vectors. We use fastText [38] as our word embeddings for tweets because of its sub-word usage and the ability to create vectors for arbitrary and out-of-vocabulary words. Although there exists many alternatives, picking the one that works well for a specific dataset is not trivial. We conducted experiments to classify TREC Sentiment tweets using four choices of word embeddings: *fastText* [38], *GoogleNews* [48], *GloVe* [14], and *CrisisNLP* [182]. Unlike fastText, we fine-tune the other pre-trained vectors using Gensim [183] to create vectors for out-of-vocabulary words. Vectors for words that are already in the vocabulary are *locked* while tuning for consistency in evaluation. Their performance is shown in Table 4.8. All of them performed similarly with Glove performing slightly better than the rest. The tweet-based embeddings such as GloVe or CrisisNLP did not significantly outperform other models. Glove vectors are 200-dimensional while the rest are 300-dimensional which makes the experiment favoring Glove word vectors. This experiment shows that the problem of finding a strictly superior word vector model for tweets still remains a challenging task.

Interpretability (Attention Visualization). The attention weights used to create the context vector by the dot product operation with word activations represent the interpretable layer in our architecture. These weights represent the importance of each word in

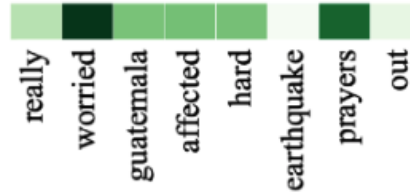
Priority Tweet Example: “RT <@>: BOSTON: POLICE AND FBI URGING ANYONE WITH VIDEO OF THE FINISH LINE AT THE TIME OF THE EXPLOSION SHOULD PLEASE COME FORWARD”



Factoid Tweet Example: “#JBCNews JustIn: Nepal Earthquake death toll rises to 1,900. Indian Army names aid mission as “Operation Maitri” #lka”



Sentiment Tweet Example: “RT <@>: Really worried. Guatemala is affected by a hard earthquake. My prayers go out to them. #Guatemala-tecosAlRescate”



Irrelevant Tweet Example: “RT <@>: Our thoughts are with the people who lost their lives in today’s #earthquake in #Guatemala”

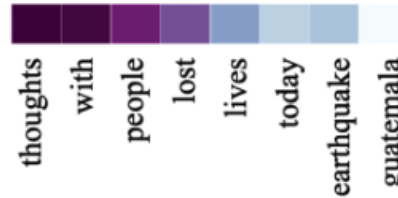


Figure 4.3: Examples of interpretable results using attention; darker the shade, higher the attention. Recall that no data from the crisis-event for testing is used for training the model. Even then, relevant keywords such as ‘police urging’, ‘death toll rises’, ‘worried’, and ‘thoughts with people’ are correctly picked up by the attention layers of their respective tasks.

| Relevant (+ve) | Irrelevant (-ve) |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| #Coronavirus exposes gaps in infection controls for senior-care homes <@> <@> #HarmMatters #ElderCare @covid19 | Prediction: Baby Boom in 9 months aka Corona Kids... #corona #coronacapital19 #quarantine @ Hilliard, Ohio |
| Common symptoms of infection include fever, cough and respiratory symptoms such as shortness of breath and breathing difficulties. If symptoms develop self-isolate immediately and contact your health care provider. | Jennifer, look at the numbers, and turn off the television news. Recovery rate, and who this virus actually impacts. |
| <@> instructs students to call their nurse advice line if they have a fever, respiratory symptoms, have recently traveled internationally or have had contact with someone with the coronavirus. | Well, this is about the ONLY good thing that the #coronavirus brought about!!!! #coronapocalypse #Frozen2 #DisneyPlus #Disneyteacher |
| [Industry News] Revised Guidance for Infection Control and Prevention of COVID-19 in Nursing Homes #nursinghome #covid19 #briggshealthcare | Here is a speedy recovery to <@>, who so responsibly is managing this illness... Incredible times. |

Figure 4.4: Examples of interpretable results using attention for relevancy prediction of COVID-19 tweets. With 77% accuracy, although the highly attended words in the ‘Relevant’ tweets provide some intuitive sense of interpretability, the highlighted words in the ‘Irrelevant’ tweets are somewhat ambiguous because it is unclear if those words are chosen due to their specific or generic nature. This shows both the benefits and challenges of unsupervised and interpretable domain adaptation.

the classification process. Some examples are shown in Figures 4.3 and 4.4. Stronger the color intensity stronger the word attention. In the first example, ‘*boston police urging*’ is the reason why the tweet is classified as +ve priority. Similarly, ‘*death toll rises*’ in the *Factoid* example, ‘*worried, prayers*’ in the *Sentiment* example, and ‘*thoughts with people*’ in the *Irrelevant* example are clear intuitive indicators of +ve predictions. These examples show the importance of having interpretability as a key criterion in crisis domain adaptation tasks for social media.

To the best of our knowledge, in social media mining for crisis analytics, there does not exist a ground truth dataset that highlights the words that explain the labels for tweets. Using our model as a guide, we hope to build a robust evaluation dataset as our immediate

next step so that the models can be quantitatively evaluated using robust trust-evaluation methods such as LIME [72]. It is also crucial to note that binary classification tasks such as sentiment analysis of Amazon reviews has a clear class divide that produces intuitive keywords such as ‘good’, ‘excellent’, or ‘great’ for +ve reviews and ‘bad’, ‘poor’, or ‘horrible’ for –ve reviews. However, for short texts such as tweets shown in Fig. 4.4, ‘relevancy’ can depend on the context and it is unclear which keywords truly represent the examples in the ‘irrelevant’ class.

COVID-19 Use-Case. We show a practical implication of our work by applying it to COVID-19 tweets described in Section 4.3. Our goal is to interpretably predict if a COVID-19 tweet is relevant or not; a binary classification task. The models are trained using only the TREC dataset and evaluated on the COVID-19 tweets (a balanced subset of size 637 for +ve and –ve labels). We found that a combination of ‘*Priority*’ and ‘*Irrelevant*’ labels from TREC performs better to predict COVID-19’s ‘*Relevant*’ label (this can be trivially verified by constructing two binary classifiers). We augment all three methods (*ST*, *ST-DAAN*, and *MT-DAAN*) with an additional condition before label prediction: $R_c = P_t \cap \overline{I_t}$, which means that a COVID-19 tweet is ‘*Relevant*’ only if it is predicted both ‘*Priority*’ = 1 and ‘*Irrelevant*’ = 0. The scores are reported in Table 4.7 and the attention results are shown in Fig. 4.4, demonstrating the effectiveness of our proposed method.

4.5 Key Takeaways

In this chapter, we presented a novel approach of unsupervised domain adaptation with multi-task learning to classify relevant information from Twitter streams for crisis management, while addressing the problems of data sparsity and limited labels. We showed that a multi-task learning model that shares the lower layers of the neural network with dedicated attention layers for each task along with a domain classifier branch can help improve generalizability and performance of deep models in the settings of limited data. Furthermore, we showed that using an attention-based architecture can help in interpreting

the classifier’s predictions by highlighting the important words that justify the predictions. We also presented an in-depth empirical analysis of the state-of-the-art models on both benchmark dataset of Amazon reviews and TREC dataset of crisis events. The application of our generic approach for interpretable and unsupervised domain adaptation within a multi-task learning framework can benefit social media mining systems in diverse domains beyond crisis management.

Chapter 5: Domain Adversarial Masking and Regeneration for Cross-Domain Generative Question Answering

5.1 Summary

This chapter¹ focuses on the problem of domain adaptation in the context of generative question answering as shown in Fig. 5.1. Contextually dependent generative question answering (QA) is yet to be well-studied in cross-domain settings, as compared to their extractive and open-domain variants. We introduce a method of domain-adversarial masking and regeneration to address this cross-domain transfer learning task, with a goal to improve over the state-of-the-art Text-to-Text Transfer Transformer (T5) baseline. We evaluate our method across four domains selected from the Amazon QA dataset. We also demonstrate a practical implication of our work by applying it to COVID-19 tweets. Additionally, we also show a qualitative analysis of the masking and regeneration method.

5.2 Methodology

5.2.1 Problem Definition

Given a source (s) and a target (t) domain, we define C as the context, Q as the question, and A as the answer. Unlike extractive QA, where the answer is a substring from the context, in generative QA the answer may or may not be explicitly present in the context. Refer to Fig. 5.1 for a few examples. The goal is to train a QA system using question-answer pairs from the source, and contexts from both domains to evaluate a partially-unseen target domain. Note, we assume that only unlabeled contexts (C_t^u , i.e., raw sentences without any associated QA pairs) from the target domain are available during training time. Briefly

¹To be submitted as a conference short paper.

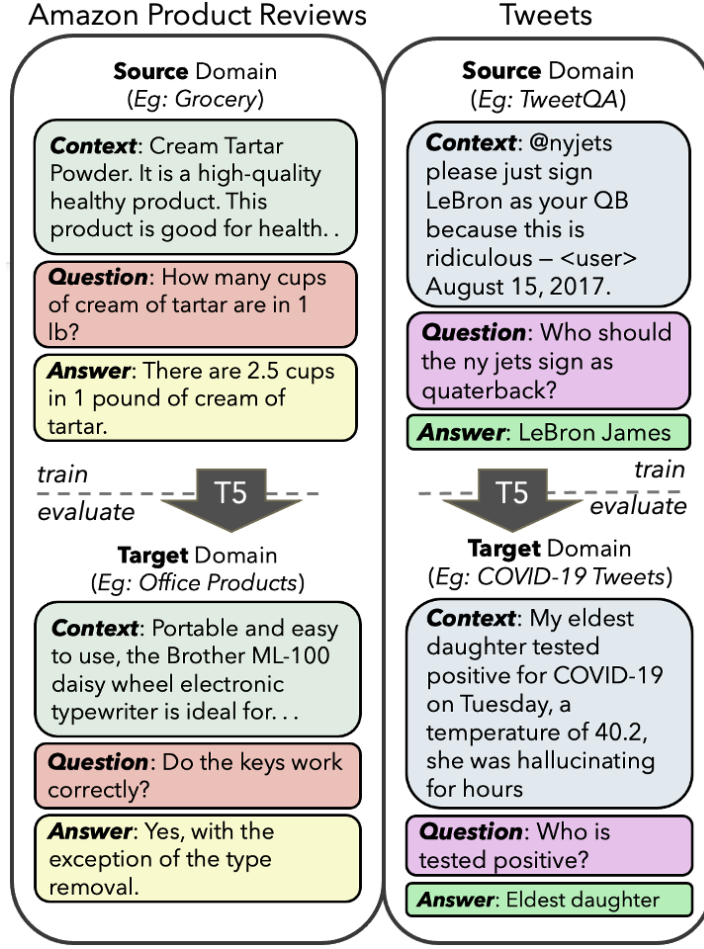


Figure 5.1: Overview of our problem statement showing examples of the cross-domain task of contextual and generative QA using the pre-trained T5 model on i) Amazon product reviews and ii) Tweets.

outlined below:

Input: $X_s = \{C_s, Q_s\}$, $X_t^u = C_t^u$, and $y_s = A_s$,

Goal: $A_t \leftarrow \text{predict}(C_t, Q_t)$.

5.2.2 Models & Concepts

Domain Adversarial Masking. Our first goal is to identify and mask domain-specific words in the source dataset. This can be achieved by constructing an attention-based binary classifier that classifies C_s versus C_t^u . The attention weights play a crucial role in

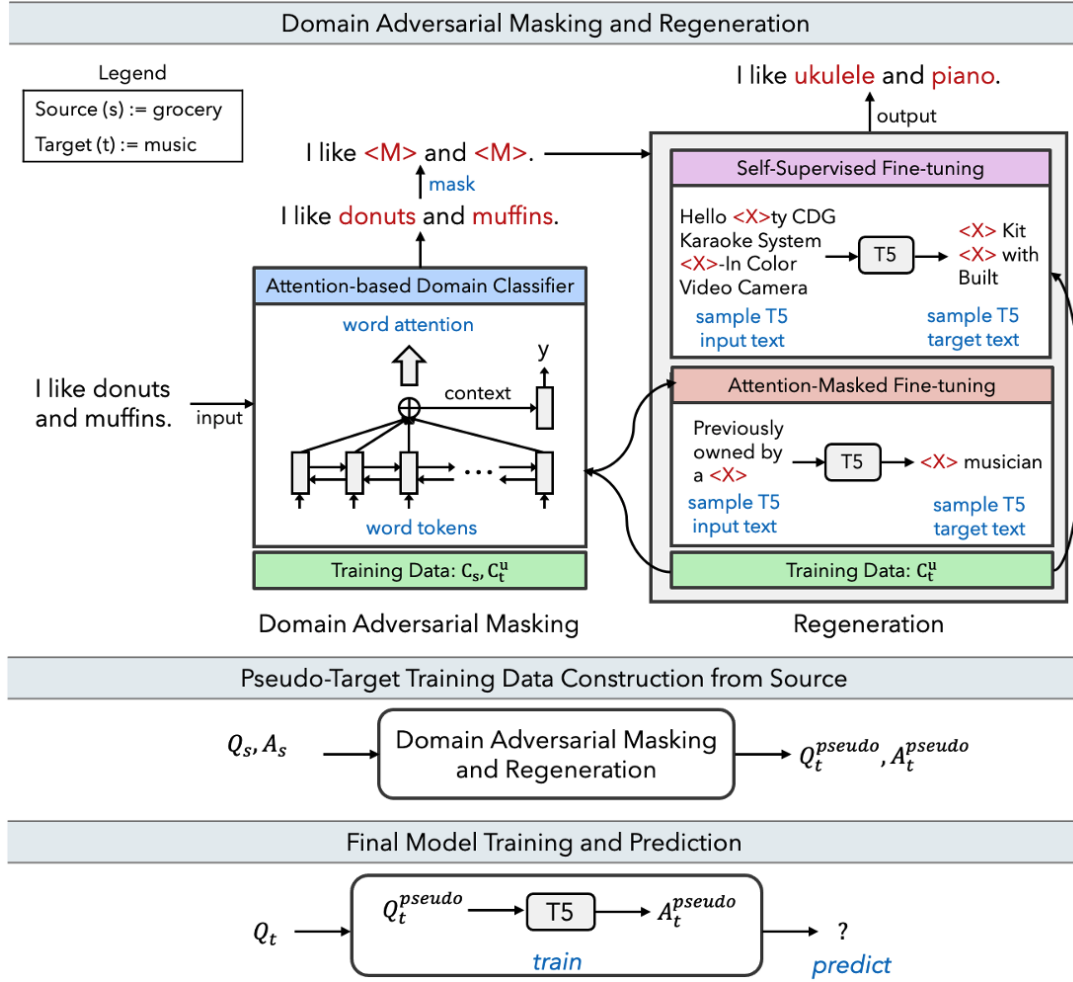


Figure 5.2: A simple example of our pipeline showing the masking process of domain-specific words in the source domain and then regenerating them using target-trained regeneration model (Top). The source QA training data is then converted to pseudo-target data (Middle), which is used to train the final QA model to evaluate ground truth target data (Bottom).

highlighting the important words that assist the classifier’s prediction. We pass word tokens as fastText word vectors [38] through a BiLSTM [2, 3] based architecture, to calculate the context vector as $\sum_{t=1}^m \alpha^{<t>} a^{<t>}$ (m = number of word tokens) to learn the attention [67, 110, 175] weights $\alpha^{<t>}$ for the words activations $a^{<t>}$ from the BiLSTM cells. As shown in Fig. 5.2, the input sentence “I like donuts and muffins” from the *grocery* domain is passed through the classifier and words like *donuts* and *muffins* are highlighted. We mask such

words to construct domain-agnostic sentences, which is then fed to the regeneration model described in the next section. In our model, we chose a simpler form of attention because the weights have naturally interpretable meaning as compared to more complex models such as BERT [8] where interpretability at the word-level for its classification ([CLS]) token is too complex for easy intuitive interpolation [184]. We leave this for future work.

Mask Regeneration. In order to train a QA model in the target domain, ideally we need in-domain data. However, we lack that. Our solution is to construct a dataset that is similar to it. We use the masked domain-agnostic data from the source domain to construct this pseudo target-domain data. This is performed by using two keys strategies over a text-to-text model: i) Self-Supervised Fine-tuning and ii) Attention-Masked Fine-tuning.

i) Self-Supervised Fine-tuning: This method follows the 10-15% masking strategy employed by the standard Masked Language Models (MLMs) that train on the self-supervised task of masked word prediction. In our context, training a text-to-text model on this task using the unlabeled target data empowers the model with target domain knowledge, helping us for the downstream QA task.

ii) Attention-Masked Fine-tuning: Unlike the previous fine-tuning method, instead of randomly choosing characters to mask, attention-based masking masks domain-specific text (e.g., ‘*donuts*’ in *grocery* domain) using the binary domain classifier constructed previously. The idea is to teach the model to fill in the blanks of domain-agnostic sentences created in the adversarial masking step with domain-specific words. Algorithm 3 describes this process in detail. k represents the amount of text to be masked and $\langle M \rangle$ is the masking token.

To summarize, as shown in Fig. 5.2, domain-specific words from the source data is masked and then regenerated using a language model fine-tuned on the target. This way, we can convert any QA datasets from one domain to another before using it for any downstream tasks.

Algorithm 3 Attention-Masking for Fine-tuning of T5 for the Target Domain

Input: C_s, C_t^u
Output: $T5_{finetuned}$
 $DC \leftarrow train_domain_classifier(C_s, C_t^u)$
 $T5_{in} \leftarrow \emptyset, T5_{out} \leftarrow \emptyset$
for $c \in C_t^u$ **do**
 $W \leftarrow get_attention_weights(c, DC)$
 $W_k \leftarrow select_topK(W, k)$ // k in %
 $in \leftarrow \emptyset, out \leftarrow \emptyset$
 for $word \in c$ **do**
 if $word \in W_k$ **then**
 $in \leftarrow in \cup \langle M \rangle$
 $out \leftarrow out \cup word$
 else
 $in \leftarrow in \cup word$
 end
 end
 $T5_{in} \leftarrow T5_{in} \cup in$
 $T5_{out} \leftarrow T5_{out} \cup out$
end
 $T5_{finetuned} \leftarrow train(T5_{in}, T5_{out}, T5_{pretrained})$

5.3 Experimental Evaluation

5.3.1 Datasets

Amazon Reviews. We combine the Amazon question-answer dataset [185, 186] with the product reviews dataset [187] to create our (*Context*, *Question*, *Answer*) triplets. We randomly select three sub-domains with manageable data size: *Musical Instruments* (M), *Office Products* (O), and *Grocery and Gourmet Food* (G). Their dataset sizes are 1117, 717, and 1879 respectively. 20% from each domain is kept aside as the test set.

Tweets. Tweet data is used to show the practical utility of our work. Here, the source domain is the TweetQA (TQa) dataset [188] consisting of tweets, questions about the tweet content, and free-form answers. The target domain is a COVID-19 (CQa) dataset [189]. Our goal is to show cross-domain transfer learning by training on tweets unrelated to COVID to answer COVID-related questions from tweets. The original COVID data is a slot-filling

Table 5.1: BLEU scores on Amazon and Tweet Datasets. *: TT = Target-Train, the model trained and evaluated on target data, serving as an upper bound.

| Source \rightarrow Target | M \rightarrow O | M \rightarrow G | O \rightarrow M | O \rightarrow G | G \rightarrow M | G \rightarrow O | TQa \rightarrow CQa |
|-----------------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-----------------------|
| T5 (Original) | 35.89 | 34.42 | 39.37 | 36.52 | 37.88 | 39.89 | 03.17 |
| + <i>Ours</i> | 36.74 | 35.41 | 38.30 | 36.54 | 38.24 | 39.20 | 03.34 |
| + <i>TT</i> * | 34.99 | 33.60 | 35.12 | 33.60 | 35.12 | 34.99 | 03.17 |

dataset that extract events each tweets. We convert them to a QA dataset as each slot is essentially answering a specific COVID-related question. Refer to [189] for more details about the slots.

5.3.2 Experiments

Experimental Setup. We follow the traditional unsupervised cross-domain experimental setup of train the model on a source domain and testing on a target domain, represented as $S \rightarrow T$. This also assumes that only unlabeled data (C_t^u) from the target domain is available during training time. For evaluation, we use BLEU scores which measure the quality of the predicted answer with respect to a given question.

Implementation Details. Our implementation is in PyTorch [190] with *simpletransformers*² library which is based on the Hugging Face transformers library [191]. For T5 training, we use Google Colab with K80 GPU and set the maximum epoch to 15, sequence length to 128, and training batch size to 8. We use wandb [192] for experiment tracking.

Baselines. State-of-the-art pre-trained T5 model [5] is our first baseline. We fine-tune this model with the source data and then evaluate on the target. This model is not exposed to any unlabeled target data (C_t^u) during the fine-tuning phase.

²<https://simpletransformers.ai>

| Kitchen | Kitchen-Masked (Using Attention-based Domain Classifier) | Kitchen → Books (using T5) |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Examples of Positive Reviews | | |
| Fantastic Flannel sheets. These sheets are warm, soft, and ever so inviting. Garnet red just made for an even better Valentine's Day evening, even though it was 2 weeks later. Truly lovely sheets, would buy more. | <M> Flannel <M> These <M> are and ever so inviting. Garnet red just made for an even better Day even though it was 2 weeks later. Truly lovely would buy more. | Amazing book. Flannel beautiful books. These photos are and ever so inviting. Garnet red just made for an even better Day even though it was 2 weeks later. Truly lovely would buy more. |
| Love them!. These spoons are so nice. The strength is a nice change from all those wooded spoons I've used over the years. | <M> These <M> are so nice. The <M> is a nice change from all those wooded spoons I've used over the years. | Really These illustrations are so nice. The paper is a nice change from all those wooded spoons I've used over the years. |
| Love It. This pan is the best! Nice heavy weight, great size. Can grill 2large or 3 medium sandwiches at the same time ! | <M> This <M> is the Nice heavy great size. Can grill 2large or 3 medium sandwiches at the same time ! | Great This book is the Nice heavy great size. Can grill or 3 medium sandwiches at the same time ! |
| Examples of Negative Reviews | | |
| The color sticks!! I got these sheets and was very pleased with them at first. After a couple of days I observed color sticking to everything around. | The <M> I got these <M> and was very pleased with them at first. After a couple of days I observed color sticking to everything around. | The Beautiful art. I got these books and was very pleased with them at first. After a couple of days I observed color sticking to everything around. |
| save your money. My cats have showed no interest whatsoever in this toy. But I think if I remove the corrugated and ball I'll have a nice palette for mixing paints | save your money. My cats have showed no <M> whatsoever in this toy. But I think if I remove the corrugated and <M> I'll have a nice <M> for mixing <M> | save your money. My cats have showed no book whatsoever in this toy. But I think if I remove the corrugated and paperback titles I'll have a nice book for mixing book libations |
| If you order this item, prepare to wait more than a month!. I ordered this item and received several notices postponing its delivery. | If you <M> this <M> to wait more than a I ordered this item and received several notices postponing its delivery. | If you get this book to wait more than a I ordered this item and received several notices postponing its delivery. |

Figure 5.3: Examples of applying masking to the Kitchen Product domain and regenerating it using the Book domain.

5.4 Results & Discussion

Qualitative examples of masking and regeneration procedure are shown in Fig. 5.3. For this analysis, we use 3 positive and 3 negative reviews from *Kitchen Products* to mask and use the *Books* domain for regeneration. *Kitchen* column shows the original reviews written by the users for Kitchen products, *Kitchen-Masked* column shows the masked version of the same

using an attention-based domain classifier that can highlight domain-specific words in the Kitchen domain, and *Kitchen*→*Books* shows the version in which the masks are regenerated using a T5 transformer pre-trained using unlabeled data from the *Books* domain. We see that the results (column 3) are not perfect, but the transformed data appear to represent the *Books* domain slightly better than what it was in their original form.

Table 5.1 shows our performance evaluation across various combinations of source-to-target transfer learning on QA datasets. When compared the original T5 model, our strategy appears to work on combinations except $O \rightarrow M$ and $G \rightarrow O$. In the context of tweets, the performance of all models were significantly lower compared to the Amazon data. Although there was no one strictly significantly outperforming model, our method indicated an overall positive direction with a +0.3% gain. It is also intriguing to note that training in the target domain itself did not produce any significant gains; which is counter-intuitive and requires further experimental evaluation. We leave a deeper analysis and experimentation of this as future work.

5.5 Key Takeaways

In this chapter, we presented a method of adversarial masking and regeneration to improve upon the state-of-the-art T5 for the task of cross-domain generative QA. A qualitative analysis shows promising transformation of source to target-like data. Our method outperformed the baselines on 5 out of 7 combinations when evaluated across different domains in the Amazon and Tweet QA datasets. The extend of linguistic nuances between the source and target domains may impact masking and regeneration, resulting in the performance differences. We also presented a qualitative evaluation of our method showing a promising direction to generate text from the target domain. A further investigation with a deeper analysis and probing in to the T5 model are left as future work.

Chapter 6: Attention Realignment and Pseudo-Labeling for Interpretable Cross-Lingual Classification of Crisis Tweets

6.1 Summary

This chapter¹ [36] focuses on interpretable cross-lingual text classification. State-of-the-art models for cross-lingual language understanding such as XLM-R [79] have shown great performance on benchmark data sets. However, they typically require some fine-tuning or customization to adapt to downstream NLP tasks for a domain. In this work, we study unsupervised cross-lingual text classification task in the context of crisis domain, where rapidly filtering relevant data regardless of language is critical to improve situational awareness of emergency services. Specifically, we address two research questions: a) Can a custom neural network model over XLM-R trained only in English for such classification task transfer knowledge to multilingual data and vice-versa? b) By employing an attention mechanism, does the model attend to words relevant to the task regardless of the language? To this goal, we present an attention realignment mechanism that utilizes a parallel language classifier to minimize any linguistic differences between the source and target languages. Additionally, we pseudo-label the tweets from the target language which is then augmented with the tweets in the source language for retraining the model. We conduct experiments using Twitter posts (tweets) labelled as a ‘request’ in the open source data set by Appen², consisting of multilingual tweets for crisis response as shown in Fig. 6.1. Experimental results show that attention realignment and pseudo-labelling improve the performance of unsupervised cross-lingual classification. We also present an interpretability analysis by evaluating the performance of attention layers on original versus translated messages.

¹Published in the proceedings of KDD Workshop on Knowledge-infused Mining and Learning (2020). URL: <http://ceur-ws.org/Vol-2657/paper3.pdf>

²<https://appen.com/datasets/combined-disaster-response-data/>

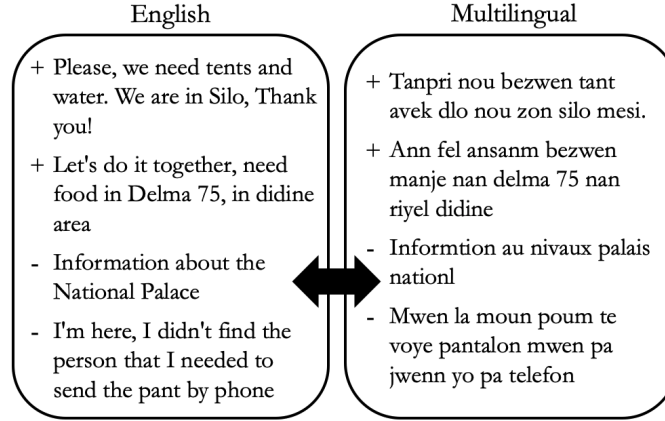


Figure 6.1: Problem Statement: Unsupervised cross-lingual tweet classification, e.g., train a model using English tweets, predict labels for Multilingual tweets, and vice-versa.

6.2 Methodology

6.2.1 Problem Definition

Consider tweets in language A and their corresponding translated tweets in language B. The task of unsupervised cross-lingual classification is to train a classifier using the data only from the source language and predict the labels for the data in the target language. This experimental set up is usually represented as $A \rightarrow B$ for training a model using A and testing on B or $B \rightarrow A$ for training a model using B and testing on A. X refers to the data and y refers to the ground truth labels. The multilingual dataset used in our experiments consists of original multilingual (ml) tweets and their translated (en) tweets in English. To summarize:

Experiment A ($en \rightarrow ml$):

Input: X_{en}, y_{en}, X_{ml}

Output: $y_{ml}^{pred} \leftarrow predict(X_{ml})$

Experiment B ($ml \rightarrow en$):

Input: X_{ml}, y_{ml}, X_{en}

Output: $y_{en}^{pred} \leftarrow predict(X_{en})$

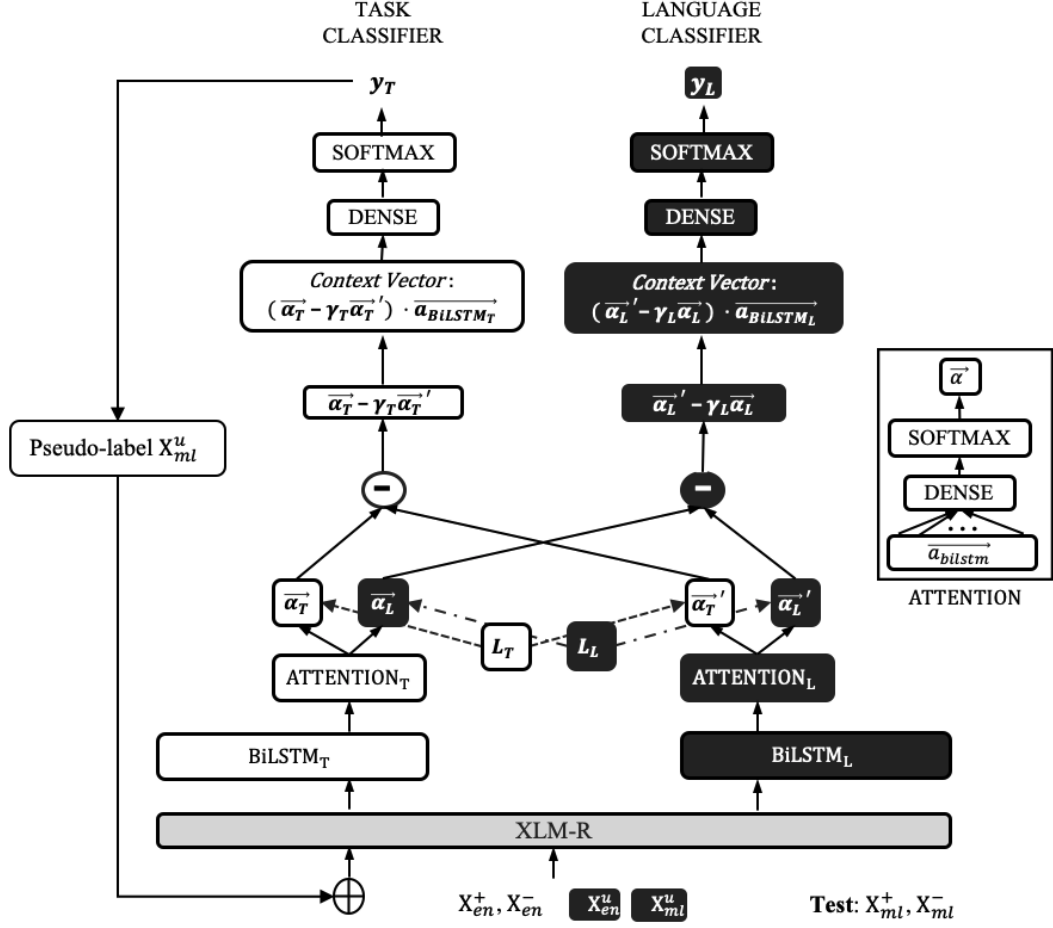


Figure 6.2: Attention Realignment with Pseudo-Labeling over XLM-R model

6.2.2 Models & Concepts

In the following sections, we propose two methodologies to enhance cross-lingual classification: 1) Attention Realignment and 2) Pseudo-Labeling. Attention realignment utilizes a language classifier which is trained in parallel to realign the attention layer of the task classifier such that the weights are more geared towards task-specific words regardless of the language. Pseudo-Labeling further enhances the classifier by adding high quality seeds from the target language that are pseudo-labelled by the task classifier.

Table 6.1: New Notations

| Notation | Definition |
|------------------------|--------------------------------------------------------------------------|
| en | Tweets translated to English (‘message’ column in the dataset) |
| ml | Multilingual Tweets (‘original’ column in the dataset) |
| $\vec{\alpha}$ | Attention Weights |
| T | A component that uses Task-specific data. i.e., + and – ‘Request’ tweets |
| L | A component that uses Language-specific data. i.e., en and ml tweets |
| a_{BiLSTM} | Activation from the BiLSTM layer |
| β, γ, ζ | Hyperparameters |

Attention Realignment by a Parallel Language Classifier. As depicted in Fig. 6.2, model on the left side is the task classifier and the model on the right side is a language classifier that is trained in parallel. The purpose of this language classifier is to pick up aspects that is missed by the XLM-R model. This could be tweet-specific, crisis-specific, or other linguistic nuances that can separate original tweets and translated tweets. Note that semantically, translated words are expected to have similar XLM-R representations.

Attention realignment is a mechanism we introduce to promote the task classifier to be more language independent. The main idea is that the words that are given higher attention in a language classifier should be less important in a task classifier. For example, ‘*dlo*’ in Haitian and ‘*water*’ in English should have the same vector representation in language agnostic models; while the sentence structure, grammar, and other nuances can vary. We enforce this rule by constructing two operations:

1. **Attention Difference:** When a sentence goes through model M1, it also goes through model M2. For the same sentence, this returns two attention layer weights: one from the task classifier ($\vec{\alpha}_T$) and the other from the language classifier ($\vec{\alpha}_T'$). Directly subtracting $\vec{\alpha}_T'$ from $\vec{\alpha}_T$ poses two issues: 1) we do not know whether they are comparable and 2) $\vec{\alpha}_T'$ may have negative values. A simple solution to this is

to normalize both vectors and clip $\vec{\alpha}_T'$ such that it is between 0 and 1. Thus, an attention subtraction step is as follows:

$$\frac{\vec{\alpha}_T}{\|\vec{\alpha}_T\|} - \gamma_T \text{clip}\left(\frac{\vec{\alpha}_T'}{\|\vec{\alpha}_T'\|}, 0, 1\right) \quad (6.1)$$

where γ_T is a hyperparameter to tune the amount of subtraction needed for the task classifier. Similarly, for the language classifier,

$$\frac{\vec{\alpha}_L'}{\|\vec{\alpha}_L'\|} - \gamma_L \text{clip}\left(\frac{\vec{\alpha}_L}{\|\vec{\alpha}_L\|}, 0, 1\right) \quad (6.2)$$

2. **Attention Loss:** Along with attention difference, the model can also be trained by inserting an additional loss function term that penalizes the similarity between the attention weights from the two classifiers. We use the Frobenius norm.

$$L_{At} = \|\vec{\alpha}_T^T \vec{\alpha}_T'\|_F^2 \quad (6.3)$$

$$L_{Al} = \|\vec{\alpha}_L^T \vec{\alpha}_L'\|_F^2 \quad (6.4)$$

for task and language respectively. Resulting final loss function of joint training will be:

$$L(\theta) = \zeta_T(CE_T + \beta_T L_{At}) + \zeta_L(CE_L + \beta_L L_{Al}) \quad (6.5)$$

where β is the hyperparameter to tune the attention loss weight, ζ is the hyperparameter to tune the joint training loss, and CE denotes the binary cross entropy loss,

$$CE = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)] \quad (6.6)$$

Table 6.2: Dataset Statistics for both *en* and *ml*

| | Train | Validation | Test |
|-----------------|--------------|-------------------|-------------|
| <i>Positive</i> | 3554 | 418 | 496 |
| <i>Negative</i> | 17473 | 2152 | 2128 |

It is important to note that the Frobenius norm is **not** simply between the attention weights of the two models but rather between the attention weights produced by the two models on the same input tweet. For example, for a given tweet, the task classifier attends more to task-specific words and the language classifier attends to language-specific words. So the mechanism makes sure that they are distinct.

Pseudo-Labeling. To enhance the model further, we pseudo-label the data in the target language. Pseudo-labelling [193] is first introduced by [193] who showed how a small set of labelled data along with a large amount of unlabelled data can improve a model’s performance. For example, if we are training a model using the English tweets, we use the original tweets before translation for pseudo-labelling. The idea is simply to gather high-quality seeds from the target to retrain the model. Note that, we still do not use any target labels here; still following the unsupervised goal. Thus, for retraining model M1 for $en \rightarrow ml$, the new dataset would consist of: X_{en}^+ and $X_{ml}^{pseudo+}$ as positive examples and X_{en}^- and $X_{ml}^{pseudo-}$ as negative examples.

XLM-R Usage. The recommended feature usage of XLM-R³ is either by fine-tuning to the task or by aggregating features from all the 25 layers. We employ the later to extract the multilingual embeddings for the tweets.

³<https://github.com/facebookresearch/XLM>

Table 6.3: Implementation Details

| | |
|----------------------------------------|--------------------------------------------------------------------------------------|
| T_x | 30 |
| Deep Learning Library | Keras |
| Optimizer | Adam [$lr = 0.005$, $\beta_{a_1} = 0.9$, $\beta_{a_2} = 0.999$, $decay = 0.01$] |
| Maximum Epoch | 100 |
| Dropout | 0.2 |
| Early Stopping Patience | 10 |
| Batch Size | 32 |
| ζ_T | 1 |
| ζ_L | 0.1 |
| $\beta_T, \beta_L, \gamma_T, \gamma_L$ | 0.01 |

6.3 Experimental Evaluation

6.3.1 Datasets

We use the open source dataset from Appen⁴ consisting of multilingual crisis response tweets. The dataset statistics for tweets with ‘request’ behavior labels is shown in Table 8.1. For all the experiments, the dataset is balanced for each split.

6.3.2 Experiments

Each experiment is denoted as $A \rightarrow B$, where A is the data that is used to train the model and B is the data that is used for testing the model. For example, $en \rightarrow ml$ means we train the model using English tweets and test on multilingual tweets.

Models are implemented in Keras and the details are shown in Table 6.3. Hyperparameters β_T , β_L , γ_T , and γ_L are not exhaustively tuned; we leave this exploration for future work.

⁴<https://appen.com/datasets/combined-disaster-response-data/>

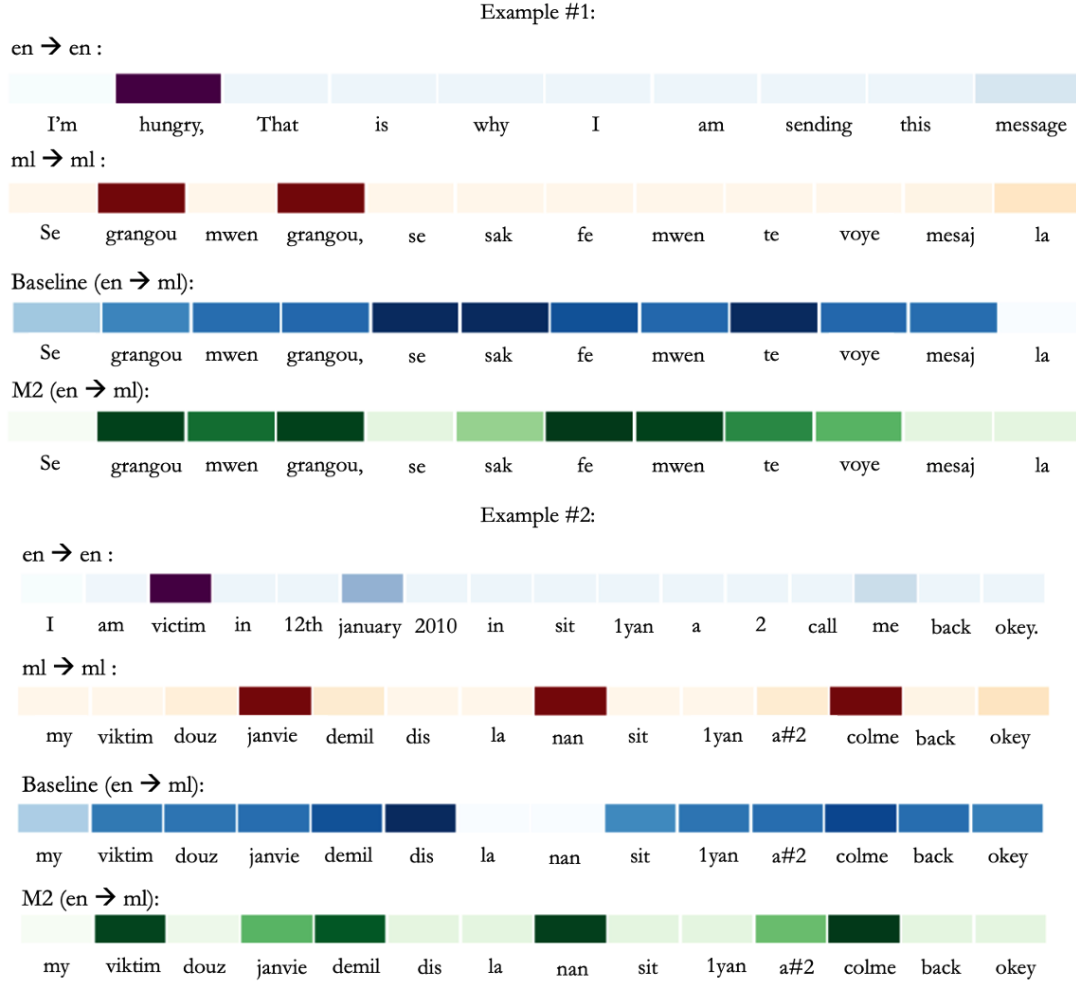


Figure 6.3: Attention visualization example for ‘request’ tweets: words and their attention weights for two tweets in Haitian Creole and its translation in English (darker the shade, higher the attention).

6.4 Results & Discussion

Table 6.4 shows the cross-lingual performance comparison of all the models. The three models are described below:

1. **Baseline:** The baseline model consists of embeddings retrieved from XLM-R trained over BiLSTMs and Attention layers. This is a traditional sequence (text) classifier enhanced with attention mechanism. Activations from the BiLSTM layers are weighed

Table 6.4: Performance Comparison (Accuracy in %) for $Source \rightarrow Target$ ($Source \rightarrow Source$). *Baseline* = XLMR + BiLSTM + Attention. *Model M1* = Baseline + Attention Realignment. *Model M2* = Model M1 + Pseudo-Labeling.

| | Baseline | Model M1 | Model M2 |
|---------------------|-----------------|-----------------|----------------------|
| $en \rightarrow ml$ | 59.98 (80.57) | 62.53 (77.02) | 66.79 (82.39) |
| $ml \rightarrow en$ | 60.93 (70.07) | 65.69 (63.50) | 70.95 (73.84) |

by the attention layer to construct the context vector which is then passed through a dense layer and softmax function to produce the classification output.

2. **Model M1:** Adding attention realignment to the baseline model produces model M1. Attention realignment is achieved through a language classifier which is trained in parallel with the goal to make the task classifier more language agnostic. The attention weights for both task and language classifiers are manipulated by each other during training by a process of subtraction (attention difference) as well a loss component (attention loss). See Section 3.3.
3. **Model M2:** Adding the pseudo-labelling procedure to model M1 produces model M2. Using Model M1 which is trained to be language agnostic, tweets from the target languages are pseudo-labelled. High quality seeds are selected (using Model M1 $p > 0.7$) and augmented to the original training dataset to retrain the task classifier.

Results show that, for cross-lingual evaluation on $en \rightarrow ml$, model M1 outperforms the baseline by +4.3% and model M2 outperforms by +11.4%. On $ml \rightarrow en$, model M1 outperforms the baseline by +7.8% and model M2 outperforms by +16.5%. This shows that both models are effective in cross-lingual crisis tweet classification. An interesting observation to note is that using attention realignment alone decreased the classification performance in the same language, which is brought back up by pseudo-labelling. These scores are shown in brackets in Table 6.4. A deeper investigation in this direction on various other tasks can shed more light on the impact of realignment mechanism.

Interpretability (Attention Visualization). We follow a similar attention architecture shown in [34]. The context vector is constructed as a result of dot product between the attention weights and word activations. This represents the interpretable layer in our architecture. The attention weights represent the importance of each word in the classification process. Two examples are shown in Fig. 6.3. In the first example, both $en \rightarrow en$ and $ml \rightarrow ml$ give attention to the word ‘*hungry*’ (i.e., ‘*grangou*’ in Haitian Creole). Note that these two are results from the models that are trained in the same language in which they are tested; thus, expecting an ideal performance. For the baseline model in the cross-lingual set-up $en \rightarrow ml$, although it correctly predicts the label, the attention weights are more spread apart. In model M2 with attention realignment and pseudo-labelling, although with some spread, the attention weights are shifted more toward ‘*grangou*’. Similarly in example 2, the attention weights in the baseline model are more spread apart. Cross-lingual performance of model M2 aligns more with $en \rightarrow en$ and $ml \rightarrow ml$. These examples show the importance of having interpretability as a key criterion in cross-lingual crisis tweet classification problems; which can also be used for downstream tasks such as extracting relevant keywords for knowledge graph construction.

6.5 Key Takeaways

In this chapter, we presented a novel approach for unsupervised cross-lingual crisis tweet classification problem using a combination of attention realignment mechanism and a pseudo-labelling procedure (over the state-of-the-art multilingual model XLM-R) to promote the model to be more language agnostic. Performance evaluation showed that both models M1 and M2 outperformed the baseline by +4.3% and +11.4% respectively for cross-lingual text classification from English to Multilingual. We also presented an interpretability analysis by comparing the attention layers of the models. It shows the importance of incorporating a word-level language agnostic characteristic in the learning process, when training data is available only in one language.

Chapter 7: Multilingual Code-Switching for Zero-Shot Cross-Lingual Intent Prediction and Slot-Filling

7.1 Summary

This chapter¹ [37] focuses on the task of cross-lingual intent prediction and slot-filling. Predicting user intent and detecting the corresponding slots from text are two key problems in Natural Language Understanding (NLU). In the context of zero-shot learning, this task is typically approached by either using representations from pre-trained multilingual transformers such as mBERT, or by machine translating the source data into the known target language and then fine-tuning. Our work focuses on a particular scenario where the target language is *unknown* during training. To this goal, we propose a novel method to augment the monolingual source data using multilingual code-switching via random translations to enhance a transformer’s language neutrality when fine-tuning it for a downstream task. This method also helps discover novel insights on how code-switching with different language families around the world impact the performance on the target language. Experiments on the benchmark dataset of MultiATIS++ yielded an average improvement of +4.2% in accuracy for intent task and +1.8% in F1 for slot task using our method over the state-of-the-art across 8 different languages². Furthermore, we present an application of our method for crisis informatics using a new human-annotated tweet dataset of slot filling in English and Haitian Creole, collected during Haiti earthquake disaster.

¹To be submitted as a conference paper. URL: <https://arxiv.org/pdf/2103.07792.pdf>

²Languages that have different morphological structures compared to English, such as Hindi, Turkish, Chinese, and Japanese, yielded higher benefits.

| | | | | | | | | | | |
|---------------------------|--------------|----|--------------|--------------|--------------|-------|---------------------|---------------------|---------------------|---------------------|
| Intent | atis_airfare | | | | | | | | | |
| Original (English) | | | | | | | | | | |
| Words | Show | me | round | trip | fares | from | Denver | to | Philadelphia | |
| Slots | O | O | B-round_trip | I-round_trip | O | O | B-fromloc.city_name | O | B-fromloc.city_name | |
| Chunk-level Code-Switched | | | | | | | | | | |
| Words | 给 | 我 | 看看 | ਸੈਰ | tarifas | desde | Denver | إلى | Филадельфия | |
| Slots | O | O | O | O | B-round_trip | O | O | B-fromloc.city_name | O | B-fromloc.city_name |

Figure 7.1: An original example in English from MultiATIS++ dataset and its multilingually code-switched version. In the above code-switching example, the chunks are in Chinese, Punjabi, Spanish, English, Arabic, and Russian. ‘atis_airfare’ represents an intent class where the user seeks price of a ticket.

7.2 Methodology

This section first describes our problem for zero-shot cross-lingual transfer setting, followed by a novel data augmentation method using multilingual code-switching of monolingual source to enhance language neutrality. We then describe language families, followed by the joint training setup.

7.2.1 Problem Definition

Given a source and a set of target languages, the goal is to train a classifier using data only in the source language and predict examples from the completely unseen target languages. We assume the target language is *unknown* during training time, which makes direct translation to target infeasible. In this context, we use code-switching (*cs*) to augment the monolingual source data. Thus, the input and output of our problem can be defined as:

Input: $X_{ut}^s, y_i^s, y_{sl}^s$

Code-Switched Input: $X_{ut}^{cs}, y_i^{cs}, y_{sl}^{cs}$

Output: $y_i^t, y_{sl}^t \leftarrow \text{predict}(X_{ut}^t)$

where X_{ut} represents sentences, y_i their ground truth intent classes, and y_{sl} the slot labels for the words in those sentences. An example sentence, its intent class, and slot labels are shown in Fig. 7.1.

Algorithm 4 Data Augmentation via Multilingual Code-Switching (Chunk-Level)

Input: $X_{ut}^{en}, y_i^{en}, y_{sl}^{en}$
Output: $X_{ut}^{cs}, y_i^{cs}, y_{sl}^{cs}$
 $X_{ut}^{cs} \leftarrow \emptyset, y_i^{cs} \leftarrow \emptyset, y_{sl}^{cs} \leftarrow \emptyset$
 $lset = googletrans.languages - l_T$
for $i \in 1..k$ **do**
 for $i \in 1..len(X_{ut}^{en})$ **do**
 $G^{cs} \leftarrow \emptyset, L^{cs} \leftarrow \emptyset$
 $chunks = slot_chunks(X_{ut}^{en}[i], y_{sl}^{en}[i])$
 for $c \in chunks$ **do**
 $l \leftarrow random.choice(lset)$
 $t \leftarrow translate(c, l)$
 $G^{cs} \leftarrow G^{cs} \cup t$
 $L^{cs} \leftarrow L^{cs} \cup align_label(c, t)$
 end
 $X_{ut}^{cs} \leftarrow X_{ut}^{cs} \cup G^{cs}$
 $y_i^{cs} \leftarrow y_i^{cs} \cup y_i^{cs}[i]$
 $y_{sl}^{cs} \leftarrow y_{sl}^{cs} \cup L^{cs}$
 end
end

7.2.2 Models & Concepts

Multilingual Code-Switching. Multilingual masked language models, such as mBERT [8], are trained using large datasets of publicly available unlabeled corpora such as Wikipedia. Such corpora largely remain monolingual at the sentence level because the presence of intra-sentence code-switched data in written texts is likely scarce. The masked words that needed to be predicted usually are in the same language as their surrounding words. We study how code-switching can enhance the language neutrality of such language models by augmenting it with artificially code-switched data for fine-tuning it to a downstream task. Algorithm 4 explains this code-switching process at the chunk-level. When using slot filling datasets, slot labels that are grouped by BIO [194] tags constitute natural chunks. To summarize the algorithm, we take a sentence, take each chunk from that sentence, perform a translation into a random language using Google’s NMT system [172], and align the slot labels to fit the translation, i.e., label propagation through alignment as the translated sentence do not

Table 7.1: Selected language families to evaluate their impact on a target language.

| Group Name | Languages |
|------------------------|----------------------------------------------------------------------------------|
| Afro-Asiatic | Arabic (ar), Amharic (am), Hebrew (he), Somali (so) |
| Germanic | German (de), Dutch (nl), Danish (da), Swedish (sv), Norwegian (no) |
| Indo-Aryan | Hindi (hi), Bengali (bn), Marathi (mr), Nepali (ne), Gujarati (gu), Punjabi (pa) |
| Romance | Spanish (es), Portuguese (pt), French (fr), Italian (it), Romanian (ro) |
| Sino-Tibetan & Japonic | Chinese (zh-cn), Japanese (ja), Korean (ko) |
| Turkic | Turkish (tr), Azerbaijani (az), Uyghur (ug), Kazakh (kk) |

preserve the number and order of words in the original sentence. At the chunk-level, we use a direct alignment. The BIO-tagged labels are recreated for the translated phrase based on the word tokens. More complex methods can be applied here to improve the alignment of the slot labels such as fast-align [195] or soft-align [30]. Code-Switching at the word-level essentially translates every word randomly, while at the sentence-level translates the entire sentence. During the experimental evaluation process, to build a language neutral model using monolingual source of English data, **all 8 target languages are excluded** from the code-switching procedure to avoid unfair model comparisons, i.e. remove target languages from *lset* in Algorithm 4.

Complexity: The augmentation process is repeated k times per sentence producing a new augmented dataset of size $k \times n$, where n is the size of the original dataset, i.e. space complexity of $\mathcal{O}(k \times n)$. Algorithm 4 has a runtime complexity of $\mathcal{O}(k \times n \times \text{translations/sentence})$ steps assuming constant time for alignment. Word-level requires as many translations as the number of words but sentence-level requires only one. An increase in the dataset size also increases the training time, but an advantage is one model fits all languages.

Language Families. A language family is defined as a group of related languages that are likely coming from the same parent. For example, Portuguese, Spanish, French, Italian, and Romanian are daughter languages derived from Latin [196]. We use language families to study their impact on the target languages. We augment the source language with code-switching from a particular language family. For instance, code-switching the English

Table 7.2: Datasets and statistics.

| Language | Utterances | | | Tokens | | | Intents | Slots |
|-------------------------------|------------|-----|------|--------|-------|-------|---------|-------|
| | train | dev | test | train | dev | test | | |
| MultiATIS++ [30] | | | | | | | | |
| English | 4488 | 490 | 893 | 50755 | 5445 | 9164 | 18 | 84 |
| Spanish | 4488 | 490 | 893 | 55197 | 5927 | 10338 | 18 | 84 |
| Portuguese | 4488 | 490 | 893 | 55052 | 5909 | 10228 | 18 | 84 |
| German | 4488 | 490 | 893 | 51111 | 5517 | 9383 | 18 | 84 |
| French | 4488 | 490 | 893 | 55909 | 5769 | 10511 | 18 | 84 |
| Chinese | 4488 | 490 | 893 | 88194 | 9652 | 16710 | 18 | 84 |
| Japanese | 4488 | 490 | 893 | 133890 | 14416 | 25939 | 18 | 84 |
| Hindi | 1440 | 160 | 893 | 16422 | 1753 | 9755 | 17 | 75 |
| Turkish | 578 | 60 | 715 | 6132 | 686 | 7683 | 17 | 71 |
| Disaster Tweets (New Dataset) | | | | | | | | |
| English | 3518 | 490 | - | 16369 | 4242 | - | 2 | 5 |
| Haitian Creole | - | - | 520 | - | - | 2834 | 2 | 5 |

dataset with Turkic language family and testing on Japanese can reveal how closely the two are aligned in the vector space of a pre-trained multilingual model. From a set of 5 distinct language families, we select a total of 6 groups of languages: Afro-Asiatic [197], Germanic [198], Indo-Aryan [199], Romance [200], Sino-Tibetan and Japonic [201, 202], and Turkic [203]. Germanic, Romance, and Indo-Aryan are branches of the Indo-European language family. Language groups and their selected daughter languages are shown in Table 7.1. Each group is selected based on a target language in the dataset and Afro-Asiatic family is added as an extra group. In experiments, $lset$ in Algorithm 4 will be assigned languages from a specific family.

Joint Training. Joint training is traditionally used for intent prediction and slot filling to exploit the correlation between the two tasks. This is done by feeding the feature vectors of one model to another or by sharing layers of a neural network followed by training the tasks together. So, a standard joint model loss can be defined as a combination of intent (L_i) and slot (L_{sl}) losses. i.e., $L = \alpha L_i + \beta L_{sl}$, where α and β are corresponding task

weights. Prior works [113, 114, 118, 131] that use BiLSTM or RNN are now modified to BERT-based implementations explored in more recent works [30, 115, 116]. A standard *Joint* model consists of BERT outputs from the final hidden state (classification (CLS) token for intent and m word tokens for slots) fed to linear layers to get intent and slot predictions. Assuming h_{cls} represents the CLS token and h_m represents a token from the remaining word-level tokens, the BERT model outputs are defined as [30, 116]:

$$\begin{aligned} p^i &= \text{softmax}(W^i h_{cls} + b^i) \\ p_m^{sl} &= \text{softmax}(W^{sl} h_m + b^{sl}) \quad \forall m \end{aligned} \tag{7.1}$$

with a multi-class cross-entropy loss³ for both intent (L_i) and slots (L_{sl}). We will use this model as our baseline for joint training. Our goal will be to show that code-switching on top of joint training improves the performance. The output of Algorithm 4 will be the input used for joint training on BERT for code-switched experiments.

7.3 Experimental Evaluation

7.3.1 Datasets

Benchmark Dataset. We use the latest multilingual benchmark dataset of MultiATIS++ [30], which was created by manually translating the original ATIS [112] dataset from English (en) to 8 other languages: Spanish (es), Portuguese (pt), German (de), French (fr), Chinese (zh), Japanese (ja), Hindi (hi), and Turkish (tr). The dataset consists of utterances for each language with an ‘*intent*’ label for ‘*flight intent*’ and ‘*slot*’ labels for the word tokens in BIO [194] format. A sample datapoint in English is shown in Fig. 7.1.

New Dataset for Disaster NLU. We construct a new intent and slot filling dataset of tweets collected during natural disasters, in two languages: English and Haitian Creole.

³ $L = -\frac{1}{n} \sum_{i=1}^n [y \log \hat{y}]$

The tweets originally were released by Appen⁴. For English, a language expert coded the tweets, and for Haitian Creole, we used Amazon Mechanical Turk with five annotators. Intent classes include: ‘*request*’ and ‘*others*’. Slot filling consists of 5 labels: ‘*medical_help*’, ‘*food*’, ‘*water*’, ‘*shelter*’, and ‘*other_aid*’. Table 8.1 provides the dataset statistics.

7.3.2 Experiments

We use the traditional cross-lingual task setting where each experiment consists of a source language and a target language. A model is trained on the source data (English) and evaluated on the target data (8 other languages). For code-switching experiments, an English text is augmented with multilingual code-switching before training. Our implementation is in PyTorch [190] and we use the pre-trained *bert-base-multilingual-uncased* [8] with *Bert-ForSequenceClassification* [191] as the mBERT model. Maximum epoch is set to 25 with an early stopping patience of 5, batch size of 32, and Adam optimizer [204] with a learning rate of $5e-5$. We select the best model on the validation set. Consistent with the metrics reported for intent prediction and slot filling evaluation in the past, we also use accuracy for intent and micro F1⁵ to measure slot performance.

7.3.3 Baselines & Upper Bound

Since we assume that target language is not known before hand, **Translate-Train (TT)** [30] method is not a suitable baseline. Rather, we set this to be an upper bound, i.e. translating to the target language and fine-tuning the model should intuitively outperform a generic model. Additionally, we add code-switching to this TT model to assess if augmentation negatively impacts its performance. The zero-shot baselines for the code-switching experiments use an **English-Only** [30] model, which is fine-tuned over the pre-trained mBERT separately for each task and an English-only **Joint** model [116].

⁴<https://appen.com/datasets/combined-disaster-response-data/>

⁵To address class imbalance for slots, we use Micro F1 instead of Macro F1, which is why our F1 scores are inflated when compared to scores in [30].

Table 7.3: Performance evaluation of code-switching with setting $k = 5$. *CS*: Code-Switching. Reported scores are average of 5 independent runs (including a separate code-switched data for each run). m = number of distinct models to be trained. *: modified BERT-based implementations [30, 116]. †: Similar to [28] but modified for slot-filling task and also excluding target language from randomized switching. ♣: The difference is significant with $p < 0.05$ using Tukey HSD (conducted between $Joint_{en-only} + CCS$ versus $Joint_{en-only}$ Baseline for each language).

| Intent Acc. | m | es | de | zh | ja | pt | fr | hi | tr | AVG |
|-----------------------------|-----|--------------|--------------|---------------|---------------|--------------|---------------|---------------|---------------|--------------|
| English-Only Baseline* | 1 | 94.42 | 94.29 | 79.53 | 73.75 | 92.90 | 93.86 | 67.06 | 69.71 | 83.19 |
| $Joint_{en-only}$ Baseline* | 1 | 95.03 | 94.51 | 80.54 | 73.57 | 93.48 | 93.33 | 73.53 | 71.05 | 84.38 |
| Word-level CS† | 1 | 94.18 | 93.92 | 81.67 | 75.48 | 92.54 | 94.18 | 81.19 | 74.22 | 85.92 |
| Sentence-level CS | 1 | 94.60 | 93.53 | 81.21 | 75.01 | 93.10 | 93.24 | 82.37 | 75.11 | 86.02 |
| Chunk-level CS (CCS) | 1 | 95.12 | 95.27 | 83.88 | 74.27 | 94.20 | 93.48 | 82.73 | 77.51 | 87.06 |
| $Joint_{en-only}^* + CCS$ | 1 | 95.48 | 94.51 | 84.43♣ | 76.48♣ | 94.15♣ | 94.89♣ | 85.37♣ | 78.04♣ | 87.92 |
| Upper Bound | | | | | | | | | | |
| Translate-Train (TT)* | 8 | 94.02 | 93.84 | 90.21 | 84.19 | 95.66 | 94.54 | 85.08 | 85.79 | 90.42 |
| $Joint_{TT}^*$ | 8 | 94.16 | 94.24 | 91.56 | 85.98 | 95.75 | 95.01 | 86.45 | 84.95 | 91.01 |
| $Joint_{TT}^* + CCS$ | 8 | 95.48 | 95.41 | 91.60 | 87.17 | 95.34 | 94.60 | 87.94 | 85.93 | 91.68 |
| Slot F1 | m | es | de | zh | ja | pt | fr | hi | tr | AVG |
| English-Only Baseline* | 1 | 96.16 | 96.73 | 83.12 | 78.81 | 95.63 | 95.40 | 77.05 | 88.09 | 88.87 |
| $Joint_{en-only}$ Baseline* | 1 | 96.12 | 96.76 | 84.95 | 79.60 | 95.76 | 95.76 | 77.63 | 88.92 | 89.44 |
| Word-level CS† | 1 | 95.81 | 96.33 | 85.46 | 79.33 | 96.27 | 95.08 | 79.10 | 86.86 | 89.28 |
| Sentence-level CS | 1 | 96.57 | 96.92 | 86.32 | 79.52 | 96.65 | 95.84 | 81.94 | 89.84 | 90.45 |
| Chunk-level CS (CCS) | 1 | 96.68 | 96.82 | 87.10 | 80.00 | 96.46 | 96.31 | 80.95 | 91.60 | 90.51 |
| $Joint_{en-only}^* + CCS$ | 1 | 96.09 | 96.56 | 88.61♣ | 82.28♣ | 96.01 | 95.94 | 82.28♣ | 90.45♣ | 91.03 |
| Upper Bound | | | | | | | | | | |
| Translate-Train (TT)* | 8 | 96.89 | 96.04 | 93.48 | 85.29 | 96.35 | 96.02 | 82.03 | 91.21 | 92.16 |
| $Joint_{TT}^*$ | 8 | 96.92 | 95.66 | 93.64 | 87.84 | 96.11 | 95.95 | 82.98 | 91.15 | 92.53 |
| $Joint_{TT}^* + CCS$ | 8 | 96.98 | 96.27 | 93.37 | 85.87 | 95.88 | 95.44 | 82.00 | 91.31 | 92.14 |

7.4 Results & Discussion

Effect of Multilingual Code-Switching. Table 7.3 describes performance evaluation on the MultiATIS++ dataset. When compared to the state-of-the-art jointly trained English-only baseline, we see a +4.2% boost in intent accuracy and +1.8% boost in slot F1 scores on average by augmenting the dataset via multilingual code-switching **without requiring the target language**. From the significance tests, except for Spanish and German, all other languages were helped by code-switching for intent detection. For slot filling, improvement on Portuguese and French went insignificant. This suggests that code-switching primarily helped languages that are morphologically more different as compared to the source language (English). For example, Hindi and Turkish have the highest intent

performance improvement of +16.1% and +9.8% respectively. And for slots, Hindi and Chinese with +6.0% and +4.3% respectively. Japanese showed +4% improvement for intent and +3.4% for slots.

The running time of the models in Table 7.5 show that code-switching is expensive which can take up to 5 hours for $k = 5$. Its training is also expensive because there is k times more data as compared to the monolingual source data. Increasing the number of code-switchings (k) for a sentence from 5 to 50 improved the performance by +1%, while increasing the run-time by a large margin. So, parameter k should be picked appropriately. Albeit this time cost is for training, with benefits at the prediction stage for real world problems.

In the translate-train (upper bound) scenario, it is not immediately clear if augmentation can help, because data in the same language as the target is always preferred over other languages, or code-switched. However, we show in Table 7.3 that augmentation did not hinder the performance.

For both intent and slot performance, chunk-level model remained robust across the languages. For intent, difference between word-level and sentence-level was insignificant. For slot, sentence-level was in par with chunk-level on average. Thus, we think that code-switching at chunk-level is safer for avoiding semantic discrepancies (as in the word-level) while also capturing better intra-sentence language neutrality.

Evaluation on Disaster Dataset. We found that disaster data is more challenging when compared to the ATIS dataset for transfer learning in NLU. The predictive performance is shown in Table 7.4. Code-Switching improved intent accuracy by +12.5% and slot F1 by +2.3%, which is promising considering that they are tweets. Joint training added +0.9% improvement to intent accuracy, however did not seem to help slot F1. This might imply a lack of strong correlation between the two tasks, i.e. a mention of ‘*food*’ or ‘*shelter*’ in a tweet may not always mean that it is a ‘*request*’ or vice-versa. The upper bound of translate-train method did not perform any better than the randomly code-switched model

Table 7.4: Performance on disaster data in Haitian Creole (ht). *CS* = Code-Switching. Reported scores are average of 5 independent runs (*: modified BERT-based).

| Intent Acc. | ht |
|---------------------------|--------------|
| English-Only Baseline* | 56.12 |
| Chunk-level CS (CCS) | 63.15 |
| $Joint_{en-only}^* + CCS$ | 63.73 |
| Slot F1 | ht |
| English-Only Baseline* | 68.72 |
| Chunk-level CS (CCS) | 70.27 |
| $Joint_{en-only}^* + CCS$ | 70.02 |

Table 7.5: Runtime on Google Colab (K80 GPU for training joint models). *MTT*: Machine Translation to Target. Note that *MTT* and J_{TT} are for one target language (averaged).

| CS (k=5) | MTT | Joint_{en} | Joint_{cs} | Joint_{TT} |
|-----------------|------------|---------------------------|---------------------------|---------------------------|
| 05:04:49 | 1:31:32 | 00:11:50 | 01:06:50 | 00:11:04 |

which seemed counter-intuitive. This might be due to the lack of strong representation for Haitian Creole in the pre-trained model, although it is similar to French.

Impact of Language Families. Results of language family analysis are shown in Fig. 7.2. The input in English is independently code-switched using 6 different language families. Note that the target language is always excluded from the group when evaluating on the same, i.e. Hindi is excluded from Indo-Aryan family when that family is being evaluated on it. Translate-train model is provided as a frame of reference and upper bound. We dropped French and Portuguese from the chart as they fall in to Romance family similar to Spanish. Results show the language families helped their corresponding languages, i.e. Romance helped Spanish, Germanic helped German, and so on; with the exception of Chinese and Japanese. In both cases, Turkic language family helped better than others.

Control Experiments on k . Hyperparameter k controls the amount of code-switched data. $k = 0$ represents original size with no code-switching, $k = 1$ represents original size

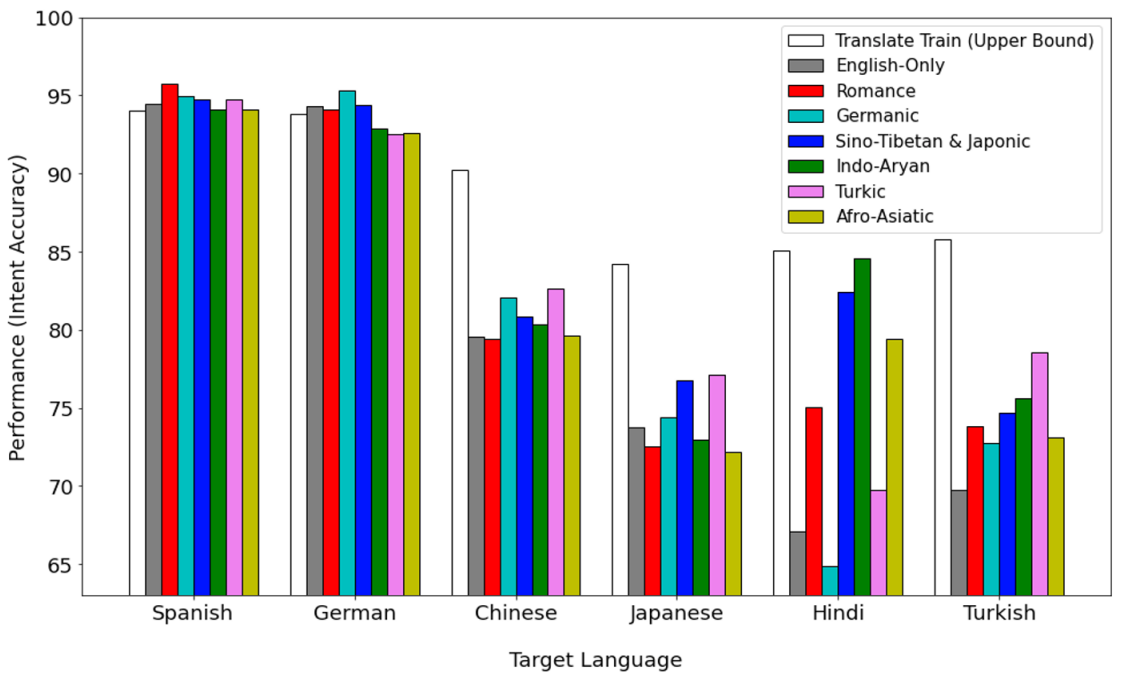


Figure 7.2: Impact of different language groups on the target languages.

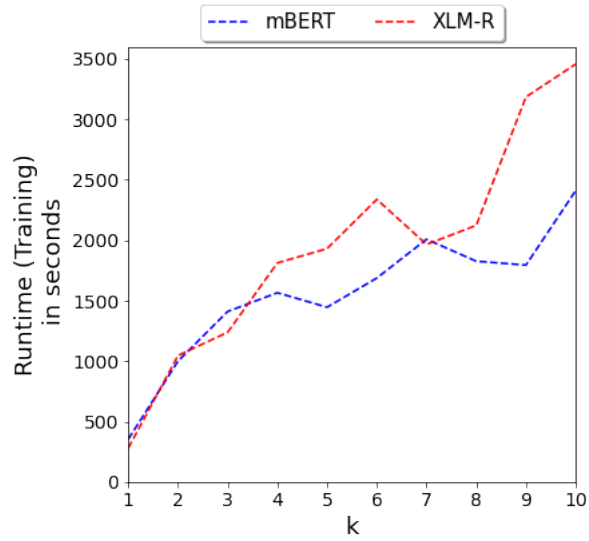


Figure 7.3: Training runtime (Google Colab K80 GPU) as k increases.

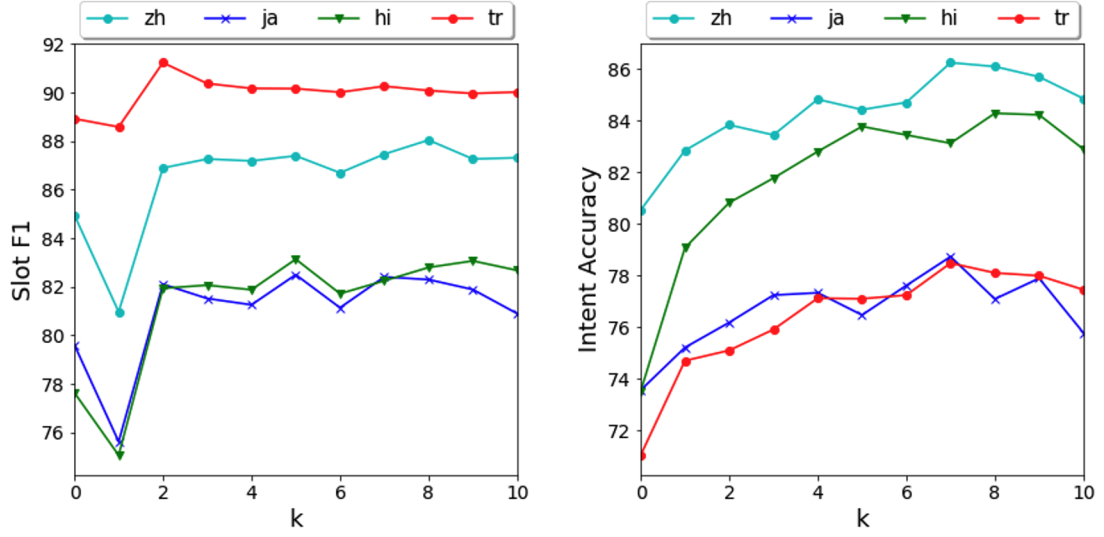


Figure 7.4: Slot F1 and Intent Accuracy as k increases (on mBERT).

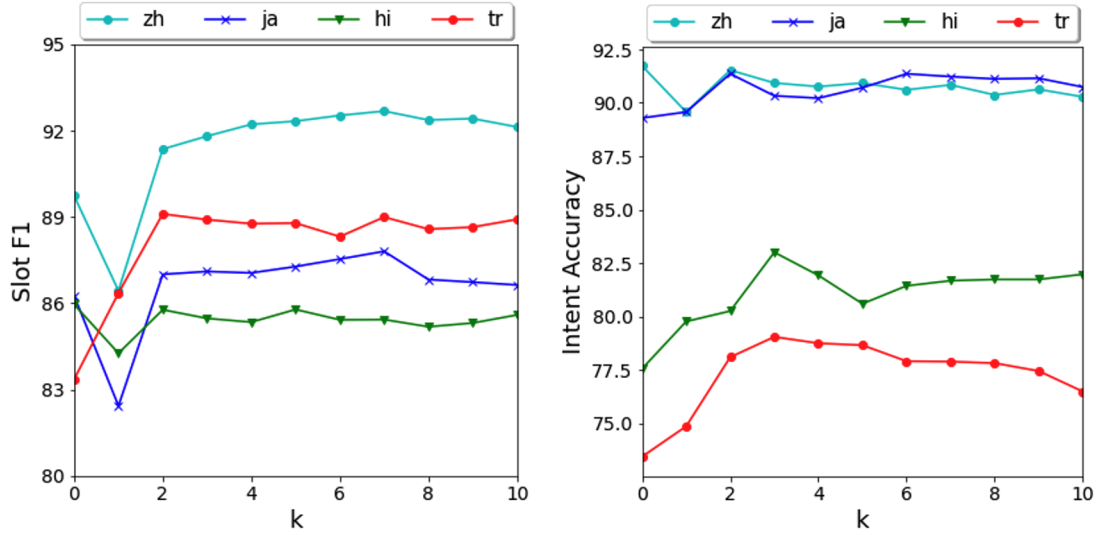


Figure 7.5: Slot F1 and Intent Accuracy as k increases (on XLM-R).

with code-switching, and $k = 10$ means 10-times more code-switched data than the original. The main experiments in Table 7.3 use $k = 5$. Fig. 7.4 shows how varying this parameter k affects the performance. For this analysis, we consider 4 target languages on which code-switching produced significant results in Table 7.3 on both Intent Accuracy and Slot F1:

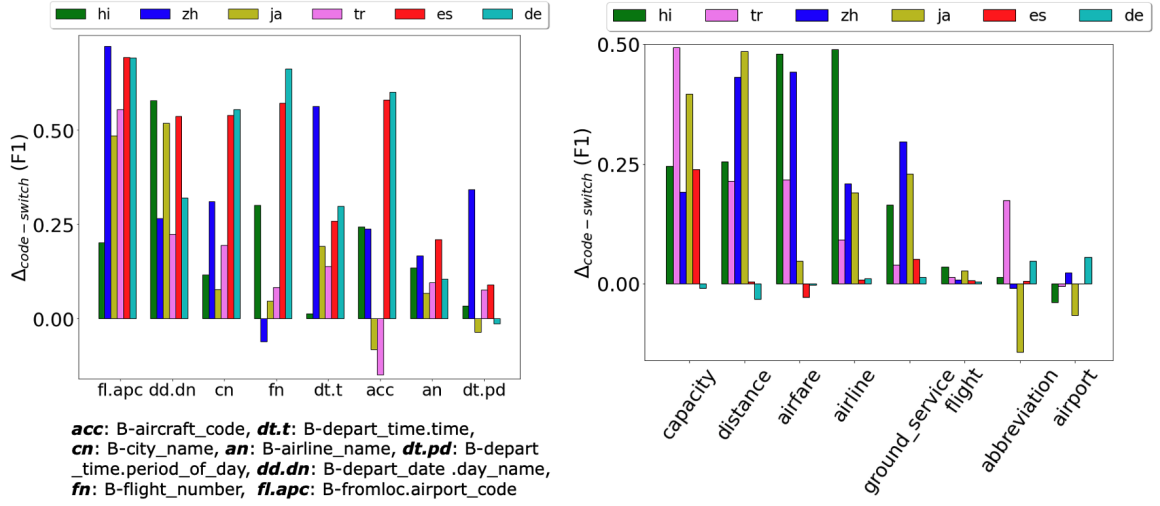


Figure 7.6: Impact of code-switching on selected slot labels (left) and intent classes (right).

Chinese, Japanese, Hindi, and Turkish. Quite intuitively, we observe that as k increases, too much code-switching becomes too expensive in terms of runtime, while performance improvement slowly plateaus. For Slot F1 performance in all four cases, unlike Intent, we observe an interesting dip when $k = 1$, which represents the augmentation having just one copy of code-switching (without the original non-code-switched data), as compared to $k = 0$. Note that this is done to maintain the size; which we speculate is the reason for this dip. This is also evident from the scores shooting up at $k = 2$ when the original data is added. Overall, we see improvement for both Intent and Slot tasks, with Slot F1 plateauing early on. Fig. 7.3 and Table 8.5 show how code-switching impact training runtime. Runtime increases as k increases; thus finding an optimal value of k and specific language groups are essential for downstream applications.

Error Analysis. Selecting intent classes with support > 10 , Fig. 7.6 (right) shows how each class is positively or negatively impacted by code-switching. Improvement was primarily on ‘airfare’, ‘distance’, ‘capacity’, ‘airline’, and ‘ground_service’ which had longer sentences such as ‘Please tell me which airline has the most departures from Atlanta’ when compared to ‘abbreviations’ and ‘airport’ classes that included very short phrases like ‘What

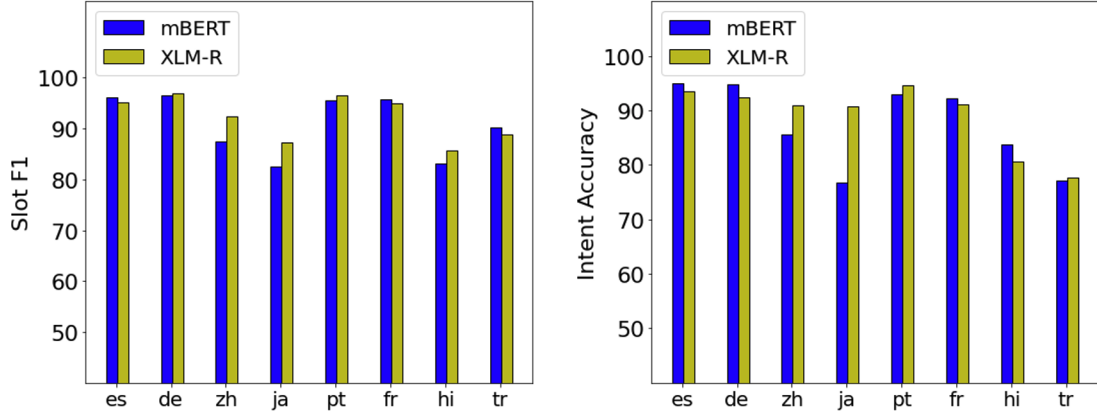


Figure 7.7: mBERT vs XLM-R

does EA mean? However, note that, Spanish and German did not improve much; aligning with our results in Table 7.3. For slot labels in Fig. 7.6 (left), we selected the ones with support > 50 and that have different characteristics, e.g. ‘*name*’, ‘*code*’, etc. The overall trend in slot performance shows improvements for labels such as ‘*day_name*’, ‘*airport_code*’, and ‘*city_name*’ and slight variations in labels such as ‘*flight_number*’ and ‘*period_of_day*’; implying textual slots benefiting over numeric ones.

mBERT versus XLM-R. We conduct an additional analysis on XLM-R [7] and compare it with mBERT [8]. The implementation is very similar in PyTorch [190] but using the pre-trained *xlm-roberta-base* with *RobertaForSequenceClassification* [191] as the XLM-R model. We observe that, setting $k = 5$, XLM-R outperforms mBERT on average (by 2% Intent Accuracy and 1.5% Slot F1). Individually, XLM-R improved Chinese, Japanese, Portuguese, and Turkish for Intent Prediction and German, Chinese, Japanese, Portuguese, and Hindi for Slot Filling as shown in Fig. 7.7. We observe a trend similar to mBERT with k on XLM-R shown in Fig. 7.5. However, for XLM-R, we observe that randomized

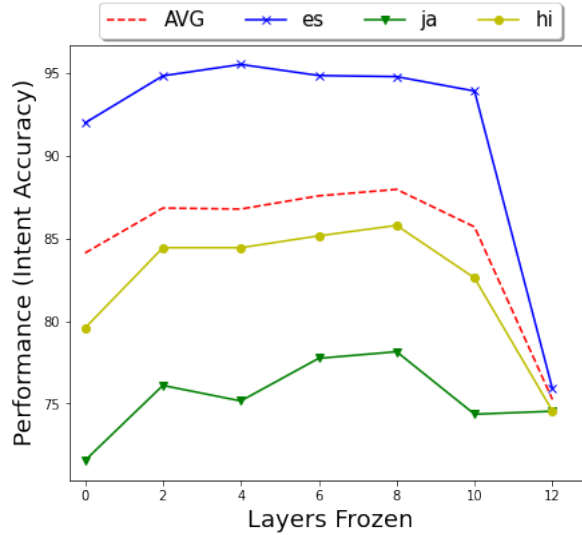


Figure 7.8: Freezing earlier layers and unfreezing a few at the top of the transformer appear to be most optimal.

code-switching did not help Chinese for Intent Prediction and Hindi for Slot F1. If code-switched to a specific language family, instead of switching to random languages, it might improve their performance. A deeper dive into XLM-R and language families are left for future work.

Hyperparameter Tuning. For joint training with same task weights, we tuned α and β using grid search to see the strength of correlation between the tasks. For intent, the (α, β) combination of $(1.0, 0.6)$ performed well, while $(1.0, 1.0)$ for slots. This suggests that intent benefiting slot might be slightly more than slot benefiting intent. Additionally, during fine-tuning, freezing the layers of the transformer affected the model performance as shown in Fig. 8.6. Keeping the first 8 layers frozen gave the best performance. By freezing the earlier layers, the transformer can retain its most fundamental feature information gained from the massive pre-training step, and by unfreezing some top layers, it can undergo fine-tuning.

7.5 Key Takeaways

In this chapter, we showed that augmenting the monolingual input data with multilingual code-switching via random translations at the chunk-level helps a zero-shot model to be more language neutral when evaluated on unseen languages. This approach enhanced the generalizability of pre-trained mBERT when fine-tuning for downstream tasks of intent detection and slot filling. We presented an application of this method using a new annotated dataset of disaster tweets. Further, we studied code-switching with language families and their impact on specific target languages, which can be used to enhance the zero-shot generalizability of models created for low-resource languages.

Chapter 8: Cross-Lingual Text Classification of Transliterated Hindi and Malayalam

8.1 Summary

This chapter¹ focuses on classification of transliterated text. Transliteration (and more typically romanization) is very common on social media, but transliterated text is not adequately handled by modern neural models for various NLP tasks. In this work, we combine data augmentation approaches with a Teacher-Student training scheme to address this issue in a cross-lingual transfer setting. We evaluate our method on transliterated Hindi and Malayalam, also introducing new datasets for benchmarking on real-world scenarios: one on sentiment classification in transliterated Malayalam, and another on crisis tweet classification in transliterated Hindi and Malayalam (related to the 2013 North India and 2018 Kerala floods). Our method yielded an average improvement of +5.6% on mBERT [8] and +4.7% on XLM-R [7] in F1 scores over their strong baselines.

8.2 Methodology

This section first describes our problem for cross-lingual transfer setting for transliterations, followed by the Teacher-Student model.

8.2.1 Problem Definition

Given a source (S) dataset in language σ e.g., in English (en), the goal is to train a classifier such that it can be used to predict examples from a target (T) dataset that consists of

¹Under review as a conference paper.

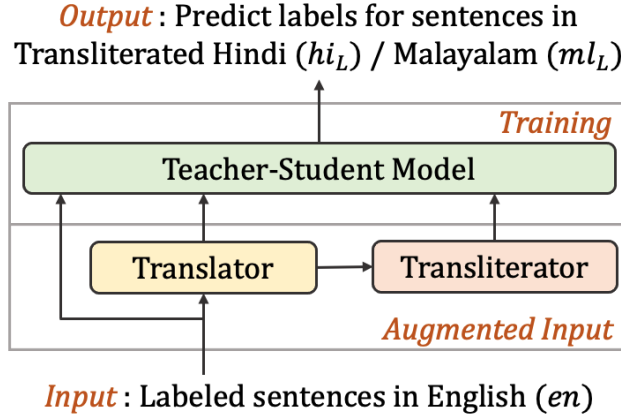


Figure 8.1: Overview of our method to enhance the multilinguality of transformer models such as mBERT or XLM-R to include Latin-transliterations (romanizations) for two Indic languages: Hindi and Malayalam.

| translated | | transliterated | |
|-------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|--|
| English | Hindi | Romanized Hindi | |
| Saw the movie today and thought it was a good effort, good messages for kids. | आज फिल्म देखने के लिए और सोचा कि यह एक अच्छा प्रयास था, बच्चों के लिए बढ़िया संदेश. | aj film dekhane ke liye aur socha ki yaha ek achcha prayas tha, bachchon ke lie badiya sandesh. | |
| English | Malayalam | Romanized Malayalam | |
| I was very disappointed in the movie. | ഞാൻ സിനിമയിൽ വളരെ നിരാശനായിരുന്നു. | njaan sinimayil valare niraashanaayirunnu. | |

Figure 8.2: Examples of English sentences and corresponding translations and transliterations.

transliterations of the target language (τ). To tackle the lack of training data in the transliterated target, as well as the lack of representation alignment between the source sentences X^σ and the transliterated target space X^τ , we propose data augmentation. Specifically, we first create translations $X^{\text{tr}(\sigma,\tau)}$ of the source language sentences in the target language. Then, we also create transliterations of those translated sentences into the source language’s script: $X^{\text{tl}(\text{tr}(\sigma,\tau),\tau)}$.² All of these are matched with the correct labels y^σ from the original

² $\text{tr}(a, b)$ represents translation from language a to b , and $\text{tl}(a, b)$ represents transliteration from language a to b .

dataset. Briefly outlined:

Input: $\mathcal{S} = X^\sigma, y^\sigma$

Augmented Input: $\mathcal{S}' = X^\sigma, X^{\text{tr}(\sigma,\tau)}, X^{\text{tl}(\text{tr}(\sigma,\tau),\sigma)}, y^\sigma$

Goal: $\mathcal{T} = y^\tau \leftarrow \text{predict}(X^\tau)$.

For example, $X^{\text{tl}(\text{tr}(\text{en},\text{hi}),\text{en})}$ represents the data that are the result of first translating from English to Hindi, then romanizing the result. The augmentation process can be performed using any existing machine translation and transliteration tool. An example of this process is shown in Fig. 8.2. In the first row, second column is the result of $\text{tr}(\text{en},\text{hi})$ and the third column is the result of $\text{tl}(\text{tr}(\text{en},\text{hi}),\text{en})$. In the following sections, for ease of representing notations, we simply use ‘tr’ for translated and ‘tl’ for transliterated data from augmented input.

8.2.2 Models & Concepts

Teacher-Student Model & Joint Training. The base component of our proposed model is straightforward: it obtains sentence representations from a pre-trained language model (mBERT, XLM-R, or similar) and uses the [CLS] token to classify the utterance. Our Teacher-Student method uses two such models, jointly trained, as outlined in Fig. 8.3. In this setup, the Teacher model acts as an anchor that does not change, i.e., all its multi-head attention layers are completely frozen. The goal is that the representations produced from training the Student model for the translated and transliterated data will eventually align with the original (Teacher) model’s source language pre-trained representations.

The training consists of two tasks, an unsupervised alignment task and a classification task. The goal of the alignment task is to ensure that the three variants (source, target, and transliterated target) end up with similar representations. As this only requires a 3-way parallel corpus with no labels, it is trained in an unsupervised fashion. The goal of the classification task is to train the model for the given class labels. Joint training on both tasks is necessary for tackling our problem.

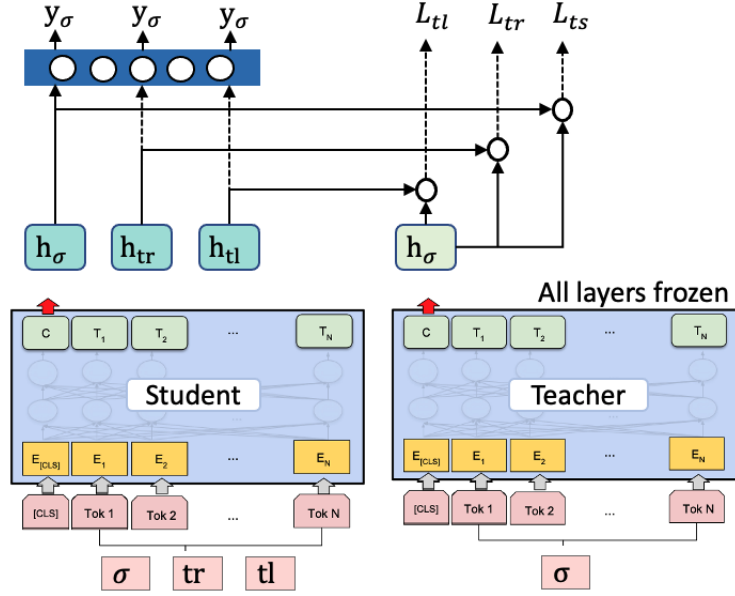


Figure 8.3: Teacher-Student Model with Joint Training.

Since the two tasks we are interested in only require the classification token for making a prediction, we build our loss functions on top of the [CLS] token. We treat its representation as the encoder's output (h), which is directly used to compute the unsupervised loss and passed through linear layers to compute the classification output ($p = \text{softmax}(Wh + b)$). As shown in Fig. 8.3, h_{σ} , h_{tr} , and h_{tl} denote the representations of X^{σ} , $X^{\text{tr}(\sigma, \tau)}$, and $X^{\text{tl}(\text{tr}(\sigma, \tau), \sigma)}$ respectively.

The unsupervised alignment loss consists of three components: \mathcal{L}_{ts} , \mathcal{L}_{tr} , and \mathcal{L}_{tl} . The *Teacher-Student Loss* (\mathcal{L}_{ts}) is defined as the difference between output embeddings produced using source language (X^{σ}) by the Student (s) versus the Teacher (t). The *Translation Loss* (\mathcal{L}_{tr}) is defined as the difference between output embeddings produced using $X^{\text{tr}(\sigma, \tau)}$ on the Student versus X^{σ} on the Teacher. Similarly, the *Transliteration Loss* (\mathcal{L}_{tl}) is defined as the difference between output embeddings produced using $X^{\text{tl}(\text{tr}(\sigma, \tau), \sigma)}$ on the Student versus X^{σ} on the Teacher.

| Malayalam Movie Reviews (\mathbf{ml}) | Transliterated Movie Reviews (\mathbf{ml}_{ro}) | label |
|----------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| സംഘട്ടനരംഗങ്ങൾ മനോഹരമാക്കിയ അശ്റഫ് ഗുരുക്കൾ ഈയൊരു തലത്തിൽ കൈയടി നൽകാം. | Samghattanaramgangal manoharamaakkiya ashraphu gurukkal eeyoru thalatthil kyyati nalkaam. | + |
| വൈകാരികമായ അടുപ്പമോ തമാശകളിലെ കണിശതയോ ജോജോ റാബിറ്റിന് അവകാശപ്പെടാനില്ല. | Vykaarikamaaya atuppamo thamaashakalile kanishathayo jojo raabittinu avakaashappetaanilla. | - |
| North India Floods (\mathbf{hi}_{nf}) | Kerala Floods (\mathbf{ml}_{kf}) | label |
| Reporter:- Aapka Kya Nuksaan Hua H? Flood survivor:- Mera Mard Beh Gaya. Ghar Toot Gaya. Khane Ko Kuch Nahi. Reporter:- Par Nuksaan Kya Hua? | Chavakkad Guruvayoor pradheshangalil bakshanam avishyamullavar thaazhe koduthitulla number il udane bendhapeduka.. Rajah School chavakkad...!!! | <i>Relevant</i> |
| johnson's baby lotion karay apke shishu ki komal towcha ki suraksha. Haha twadi pehn da shishu. Haha. | Kadha Thudarunnu - Aaro Paadunnu Doorey | <i>Irrelevant</i> |

Figure 8.4: Data samples from new datasets. Malayalam and Transliterated-Malayalam movie sentiment data samples (Top). North India and Kerala floods data samples (Bottom).

All distances are measured using the cosine similarity between the two vectors.³ Essentially, all three losses penalize moving away from the Teacher’s representation of the source language:

$$\mathcal{L}_{ts} = \frac{1}{N} \sum d(h_{\sigma}^t, h_{\sigma}^s), \quad \mathcal{L}_{tr} = \frac{1}{N} \sum d(h_{\sigma}^t, h_{tr}^s) \quad (8.1)$$

$$\mathcal{L}_{tl} = \frac{1}{N} \sum d(h_{\sigma}^t, h_{tl}^s),$$

where N represents the number of samples. Thus, the final unsupervised loss (\mathcal{L}_u) defined over the Teacher-Student model can be defined as the weighted sum of the three losses:

$$\mathcal{L}_u = \beta_1 \mathcal{L}_{ts} + \beta_2 \mathcal{L}_{tr} + \beta_3 \mathcal{L}_{tl} \quad (8.2)$$

Meanwhile, the loss function (\mathcal{J}_{joint}) for the sentiment task is a sum of binary cross-entropy (BCE⁴) losses, defined similarly for the three variants of parallel data (Source (σ), Translated (tr), and Transliterated (tl)):

$$\mathcal{J}_{joint} = \sum_{k \in [\sigma, tr, tl]} BCE_k \quad (8.3)$$

³We use $d(a, b) = 1 - \frac{a \cdot b}{|a||b|}$.

⁴ $BCE = -\frac{1}{N} \sum_{i=1}^N y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$

Table 8.1: Statistics for the test datasets.

| Dataset | + | − | Avg. # of words per sentence | Avg. # of chars. per word | Avg. # of chars. per sentence |
|------------------------------------------|-----|-----|---------------------------------|------------------------------|----------------------------------|
| Movie Review Datasets - Sentiment | | | | | |
| Hindi Movie Reviews | 335 | 293 | 154.5 | 4.0 | 613.8 |
| Romanized Hindi (hi _{ro}) | 335 | 293 | 154.5 | 5.0 | 775.8 |
| Malayalam Movie Reviews | 501 | 451 | 10.4 | 9.3 | 108.2 |
| Romanized Malayalam (ml _{ro}) | 501 | 451 | 10.4 | 10.8 | 122.6 |
| Crisis Tweet Datasets - Relevancy | | | | | |
| North India Floods (hi _{nf}) | 206 | 250 | 20.2 | 3.9 | 78.1 |
| Kerala Floods (ml _{kf}) | 109 | 132 | 19.1 | 5.4 | 103.6 |

The overall loss is simply the combination of the unsupervised and the classification loss.

$$\mathcal{L}_{joint.ts} = \mathcal{J}_{joint} + \alpha \mathcal{L}_u \quad (8.4)$$

where α is the hyperparameter that controls the unsupervised loss. To summarize, the Teacher-Student model takes the augmented data (\mathcal{S}') as the input and optimizes over a joint loss function that comprises of an unsupervised component that aligns the translated and transliterated representation into the same vector space as the source language and a supervised component that learns to classify for the task at hand.

8.3 Experimental Evaluation

8.3.1 Datasets

In this section, we describe various datasets used in our experiments.

English Movie Reviews. English movie reviews are sampled from the large IMDb movie review dataset [178] with randomly sampled balanced counts of 5000 positive and 5000 negative reviews for training and 500 each for validation. During training, this dataset is translated to Hindi and Malayalam, and subsequently transliterated. Ground truth Hindi and Malayalam datasets (described in the following sections), are used only for evaluation.

The ‘ro’ notation used in the following section for representing datasets signify $tl(\bullet, en)$; i.e., romanization or Latin-transliteration using transliteration tools.

Hindi Movie Reviews. Notations: (hi, hi_{ro}) . The original Hindi movie review dataset⁵ is constructed from various News Websites. This dataset consists of positive, negative, and neutral movie reviews. We select only positive and negative reviews from both training and validation datasets to construct the test dataset for our cross-lingual setup. From this, we construct the romanized Hindi dataset using the Indic-nlp transliteration tool [205].⁶

Malayalam Movie Reviews - New Dataset I. Notations: ml and ml_{ro} . For Malayalam movie reviews, we construct a new human-labeled dataset from the Samayam News website.⁷ Reviews are lengthy, in general, with plenty of neutral text. So, a native Malayalam speaker was tasked with extracting a few sentences from each movie review such that each positive or negative example in our dataset is highly polar. From this, we construct the romanized Malayalam dataset using the `ml2en` tool.⁸ A few samples from the dataset are shown in Fig. 8.4 and dataset statistics are available in Table 8.1.

Crisis Tweets from Appen. Notation: en . Appen⁹ provides a labeled collection of tweets posted during various natural disasters such as earthquakes, floods, and hurricanes. The three most common languages in this dataset are English, Spanish, and Haitian Creole. For our experiments, we focus on the `related` label column signifying the relevancy of tweets. The dataset also contains English translations of non-English tweets. Training and validation datasets are constructed using only the English tweets (`message` column in the Appen dataset). Statistics are shown in Table 8.1.

⁵<https://www.kaggle.com/disisbig/hindi-movie-reviews-dataset>

⁶<https://pypi.org/project/indic-transliteration/>

⁷<https://malayalam.samayam.com/malayalam-cinema/movie-review/articlelist/48225004.cms>

⁸<https://pypi.org/project/ml2en/>

⁹<https://appen.com/datasets/combined-disaster-response-data/>

Crisis Tweets from North India and Kerala Floods - New Dataset II. Notations: hi_{nf} and ml_{kf} . To construct our test datasets for the crisis tweet classification experiments, we collected tweets from the 2013 North India and 2018 Kerala Floods using Twitter API. We filtered these tweets to restrict only to naturally-occurring transliterated Hindi and Malayalam sentences, using a set of transliterated crisis-related keywords such as *madad*, *toofan*, *baarish*, *sahayta*, *floods*, etc., for Hindi and *pralayam*, *vellapokkam*, *vellam*, *sahayam*, *durantham*, *veedukal*, etc., for Malayalam. With the help of a native language expert for each language who are also proficient in English, the tweets are labeled based on contextual-relevancy or relatedness; similar to the **related** label for the English tweets in the Appen dataset. Dataset statistics are shown in Table 8.1 and a few tweet samples are shown in Fig. 8.4.

8.3.2 Experiments

Since we are interested in cross-lingual transfer, we only use the English datasets (IMDb reviews for Sentiment Analysis and Appen Crisis Dataset for Tweet Classification) for training, augmented with their automatic translations and transliterations. We evaluate on all other datasets (c.f. Table 8.1).

Monolingual LM Baselines. Our first baseline uses pre-trained language models from Hugging Face [191] that are monolingually trained on Hindi¹⁰ and Malayalam.¹¹

mBERT/XLM-R Baselines. We also consider baselines using multilingual masked LMs (MLM), specifically mBERT [8] and XLM-R [79]. We compare our models with the following MLM baselines: **1)** a model trained using only in English, **2)** a model trained using English translated to the target language using MarianMT [173], **3)** a transliterated model which is trained using the target-transliterated version of target-translated English data, and **4)** a combination of the three. These baselines use our augmented datasets but do not use the

¹⁰<https://huggingface.co/monsoon-nlp/hindi-tpu-electra>

¹¹<https://huggingface.co/eliasedwin7/MalayalamBERT>

Table 8.2: Performance evaluation (weighted F1) on various romanized datasets shows that our Teacher-Student model outperforms the baselines. \diamond : Dedicated Hindi and Malayalam language models (LM) from Hugging Face; results reflect the best performing model by varying the training data. \clubsuit : The difference is significant with $p < 0.05$ using Tukey HSD (compared against the best baseline model).

| Test Data \rightarrow Models \downarrow | hi_{ro} | | ml_{ro} | | hi_{nf} | | ml_{kf} | | AVG | |
|------------------------------------------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|--------------|--------------|
| | Acc. | F1 | Acc. | F1 | Acc. | F1 | Acc. | F1 | Acc. | F1 |
| Monolingual LM \diamond | 50.38 | 37.99 | 48.76 | 47.55 | 54.39 | 49.06 | 63.35 | 63.33 | 54.22 | 49.48 |
| mBERT Baselines | | | | | | | | | | |
| mBERT _{en} | 47.55 | 41.23 | 48.23 | 41.63 | 56.67 | 55.27 | 57.84 | 57.72 | 52.57 | 48.96 |
| mBERT _{tr} | 51.81 | 48.21 | 51.32 | 45.62 | 52.89 | 47.89 | 58.26 | 57.54 | 53.57 | 49.82 |
| mBERT _{tl} | 55.29 | 54.18 | 61.72 | 56.47 | 56.14 | 55.78 | 58.09 | 57.79 | 57.81 | 56.06 |
| mBERT _{en+tr+tl} | 55.16 | 54.75 | 61.72 | 61.25 | 56.71 | 56.03 | 63.74 | 63.42 | 59.33 | 58.86 |
| Our mBERT models | | | | | | | | | | |
| mBERT-Joint | 55.57 | 55.56 | 64.29 | 63.58 | 57.75 | 56.73 | 51.85 | 53.98 | 57.37 | 57.46 |
| mBERT-Joint-TS | 57.37\clubsuit | 57.36\clubsuit | 65.15\clubsuit | 65.78\clubsuit | 63.22\clubsuit | 63.14\clubsuit | 62.55 | 62.40 | 62.07 | 62.17 |
| XLM-R Baselines | | | | | | | | | | |
| XLM-R _{en} | 50.57 | 45.76 | 50.86 | 47.13 | 58.11 | 56.77 | 61.74 | 60.98 | 55.32 | 52.66 |
| XLM-R _{tr} | 49.52 | 47.67 | 51.72 | 50.16 | 57.11 | 56.95 | 61.74 | 61.72 | 55.02 | 54.13 |
| XLM-R _{tl} | 54.81 | 53.72 | 51.67 | 54.71 | 51.45 | 51.24 | 59.84 | 59.23 | 54.44 | 54.73 |
| XLM-R _{en+tr+tl} | 55.57 | 54.19 | 62.46 | 61.52 | 56.40 | 55.67 | 63.24 | 63.10 | 59.42 | 58.62 |
| Our XLM-R Models | | | | | | | | | | |
| XLM-R-Joint | 56.09 | 55.40 | 62.90 | 63.14 | 53.68 | 52.81 | 62.79 | 62.77 | 58.87 | 58.53 |
| XLM-R-Joint-TS | 57.70\clubsuit | 57.03\clubsuit | 65.93\clubsuit | 65.71\clubsuit | 58.39 | 57.86 | 64.87\clubsuit | 64.87\clubsuit | 61.72 | 61.37 |

Table 8.3: Evaluation (weighted F1) on IMDb **English** test data showing that our model preserves the performance on the source language for Hindi but not for Malayalam. The results in hi and ml are with our modifications. \dagger : See discussion 5.1.

| Train Language \rightarrow Model | en | hi | ml† |
|---------------------------------------|------------|------------------|--------------------------------|
| | (Baseline) | (Joint-TS Model) | (Joint-TS Model) |
| mBERT | 81.06 | 82.35 | 71.25 |
| XLM-R | 83.47 | 84.00 | 74.16 |

Teacher-Student training scheme. For example, mBERT_{tl} represents the mBERT baseline which is trained using the target-transliterated version of target-translated English data.

Our Proposed Models. **Joint-TS** represents our Teacher-Student model shown in Figure 8.3 and Eq. 8.4. We also perform an ablation (the **Joint** model) without the Teacher model (setting $\alpha = 0$ in Eq. 8.4), i.e. this model does not have an anchored Teacher, which means that there is no penalty for representations of parallel sentences being different.

Table 8.4: Evaluation (weighted F1) on Hindi and Malayalam original script showing that our model preserves or outperforms their performance on the baselines except for Malayalam on XLM-R. †: See discussion 5.1.

| Test Language → | hi | ml |
|---------------------|-------|--------------------|
| mBERT _{en} | 54.34 | 54.07 |
| mBERT _{tr} | 60.11 | 66.20 |
| mBERT-Joint-TS | 61.35 | 66.24 |
| XLM-R _{en} | 60.82 | 78.11 |
| XLM-R _{tr} | 59.72 | 71.40 |
| XLM-R-Joint-TS | 62.23 | 76.46 [†] |

Implementation Details. Our implementation is in PyTorch [190] with the transformers library [191]. We use the pre-trained cased multilingual BERT and the pre-trained *xlm-roberta-base* with a standard sequence classification architecture. Maximum epoch is set to 40 with an early stopping patience of 10, batch size to 32, and we use the Adam optimizer [204] with a learning rate of $5e-5$. We select the best model based on the validation set, using both accuracy and weighted F1 as performance measures.

8.4 Results & Discussion

In this section, we discuss our results on the two key aspects of our work: i) performance improvement on transliterated target, and ii) preserving performance on source and target.

Performance on Transliterated Target. Table 8.2 shows the classification performance on transliterated datasets. Averaging the scores, we see a total boost of +5.6% on mBERT and +4.7% on XLM-R in weighted F1 ¹² performance across the 4 romanized datasets. Our top baseline models in each of the masked language models are mBERT_{en+tr+tl} and XLM-R_{en+tr+tl}, which combine the augment data with the original data to fine-tune the model for the classification task. The improvement produced by our model is due to the

¹²With similar trends on accuracy as well.

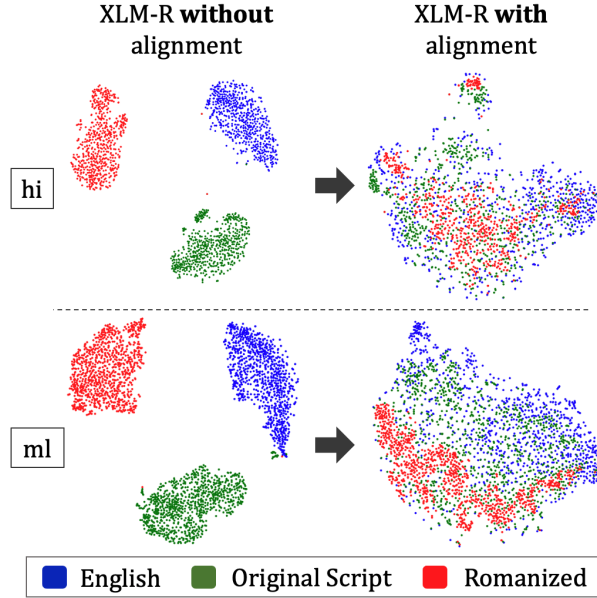


Figure 8.5: Plotting representations from the final transformer layer on the movie review test data shows that unsupervised alignment training brings the representations of the 3 variants in a shared space.

fact that the pre-trained models have likely not seen that many transliterations in these languages. mBERT is trained using data from Wikipedia, while XLM-R uses Common Crawl. As such, XLM-R likely has been trained on at least some code-switched or transliterated data. This is also supported by the scores in Tables 8.2, 8.3, and 8.4, where XLM-R_{en} outperforms mBERT_{en} on all datasets, i.e., XLM-R’s English representations are much more generalizable than mBERT’s for these two languages.

Performance on Source and Target. Table 8.3 shows the performance evaluation on English data (1000 randomly selected positive and negative reviews from the IMDb dataset [178]). Table 8.4 reports the performance evaluation on the original script. These evaluations let us assess whether our Teacher-Student model preserves the performance on the source and non-romanized target languages. We observe that fine-tuning on Hindi preserves and slightly improves performance in English, but training on Malayalam does not (\dagger in Table 8.3; c.f. two bottom columns.) We speculate that this is because, for Malayalam,

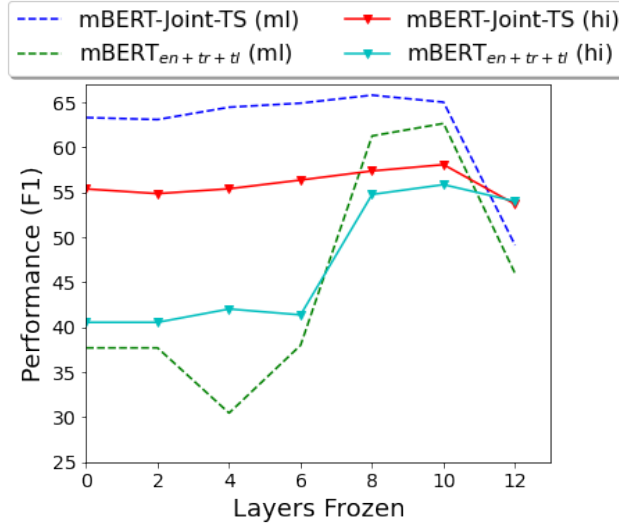


Figure 8.6: Plot showing performance variation by freezing the transformer layers on mBERT. Freezing up to layers 8 to 10 appears to be optimal.

the transliterated embeddings (red) are not blended in well with the others compared to those for Hindi as shown in Fig. 8.5 (right panels).

On the other hand, evaluation on the original script (Table 8.4) shows that our model either preserves or outperforms the baselines trained in English and non-romanized target, except for Malayalam on XLM-R (\dagger in Table 8.4). This exception can be attributed to the power of XLM-R pre-training in producing a more generalizable English representation than the original script (XLM-R_{en} vs. XLM-R_{tr}).

Representation Alignment. Fig. 8.5 shows a visual reason behind our model’s aforementioned performance improvements. While the t-SNE projection of the (3-way parallel) sentence representations for the original model are quite distinct based on language/script (left), our model brings all representations in the same vector space (right). This method of unsupervised training (L_u in Eq. 8.2) is very generic in nature that it can be adapted for any alignment scenarios where different variants or dialects of the same language need to be aligned.

Unsupervised Domain Adaptation. In addition to being cross-lingual, our approach also falls under the paradigm of unsupervised domain adaptation. Our sources and targets do not strictly fall in the same domain, even though the classification tasks are similar. For example, the training data for classifying tweets consists of not only floods, but also other events such as hurricanes, fires, earthquakes, etc. This leads to another application of our method, which can be deployed at the onset of crisis events in any region, with minimal requirement to collect labeled data from the new crisis, or converting the data to native script or English, which might save precious time for crisis response.

Hyperparameter Tuning. Primarily, we tune two hyperparameters: a) α - the weight that is given to unsupervised loss in Eq. 8.4 and b) number of layers to freeze to identify the appropriate amount of pre-trained information to be preserved without alteration. α values are tuned using a simple grid search from a range of [0.01-1.0]. All β values (Eq. 8.2) are set to 1 to prioritize the three variants (source, target, transliterated target) equally. For Joint-TS (Eq. 8.4), best hyperparameters for mBERT are $\alpha_{\text{hi}_{\text{ro}}} = 0.3$ and best $\alpha_{\text{ml}_{\text{ro}}} = 0.05$ and for XLM-R are $\alpha_{\text{hi}_{\text{ro}}} = 0.5$ and best $\alpha_{\text{ml}_{\text{ro}}} = 0.01$. Intuitively, we find that giving the classification loss primary focus and the unsupervised loss secondary focus produced better results. On the other hand, we found that freezing bottom layers and unfreezing a few top layers lead to the best results as shown in Fig. 8.6. The amount of layers to freeze was empirically between 8 to 10.

Model Runtime. The runtime of our Joint-TS model and key baselines is shown in Table 8.5. The mBERT_{en} and XLM-R_{en} model are trained only on English data without any augmentation while mBERT_{en+tr+tl}, XLM-R_{en+tr+tl}, and the Joint-TS models are trained using translated and transliterated target in addition to English data. The additional latency is caused due to the augmented data (two times more data). Our Joint-TS model also consists of unsupervised optimization for alignment, in addition to the augmented data. An interesting observation is that the Teacher-Student model based on mBERT converges faster than its en+tr+tl counterpart and XLM-R, while also having a comparable performance

Table 8.5: Runtime on a single K80 GPU in HH:MM:SS for training a Malayalam model on Crisis data.

| Model → | mBERT _{en} | XLM-R _{en} | mBERT _{en+tr+tl} | XLM-R _{en+tr+tl} | mBERT-Joint-TS | XLM-R-Joint-TS |
|-----------|---------------------|---------------------|---------------------------|---------------------------|----------------|----------------|
| Runtime → | 00:35:55 | 00:55:53 | 02:38:39 | 02:40:48 | 01:30:48 | 04:39:16 |

as shown in Table 8.2.

Ethical Considerations. The tweets extraction procedure followed the Twitter Terms of Service and did not violate privacy policies of individual users. Also, the datasets we share include only Tweet IDs in the public domain. Data statement that includes annotator guidelines for the labeling jobs and other dataset information will be provided with the dataset packet. From a broader impact perspective, our code is open-source and allows NLP technology to be accessible to information systems for emergency services and social scientists in studying a large population in India who use transliterated text for communication in everyday life.

8.5 Key Takeaways

In this chapter, we proposed a Teacher-Student model to enhance the multilinguality of language models such as mBERT/XLM-R so that it can be adapted to perform cross-lingual text classification tasks for *transliterated* Hindi and Malayalam. Experiments showed that our model outperforms traditional fine-tuning and other baselines built on the state-of-the-art. Furthermore, we release two human-annotated datasets: a highly polar Malayalam movie review dataset for sentiment analysis and a dataset of Hindi and Malayalam romanized tweets posted during North India and Kerala floods. Additionally, our method presents a generic and extensible architecture that could be adapted to any language alignment scenarios where the pre-trained models may fall short.

Chapter 9: Conclusion & Future Work

9.1 Conclusion

The main agenda behind this thesis was to study and promote NLP models to generalize well to unseen distributions: to a new domain or to a new language. Leveraging the state-of-the-art architectures as the building blocks, we tackled cross-domain and cross-lingual problems on the tasks of classification, slot-filling, and question-answering. Key takeaways of this dissertation are: (a) infusing diversity in machine learning models can improve generalization, (b) sharing layers between tasks along with a domain discriminator can aid low-resource domain adaptation, (c) a combination of any attention method and a text-to-text transformer can be used to mask domain-specific words in the source domain and construct a pseudo-target data by regeneration, (d) realigning the attention weights using a language discriminator can make help sequence models become more language agnostic, (e) augmenting datasets with code-switching into random languages at the chunk-level can enhance language neutrality of large language models, and (f) a teacher-student alignment method can be used to jointly align transliterated embeddings into a comparable representation as their English/original counterparts. Furthermore, to demonstrate the practical implications of all our methods, they are applied to extract relevant information from tweets posted during various natural disasters, with a vision to aid crisis management.

Reproducibility: Source code and documentation are publicly available at-

<https://github.com/jitinkrishnan>.

9.2 Future Work

9.2.1 Crisis Tweet Representation

As evident from our results, tweets remain a challenging genre of text regardless of the task. In the context of word vector models, we saw from the empirical evaluation of word vector models for practical purposes on challenging datasets that tweet-trained models such as Glove or CrisisNLP did not significantly outperform other models such as fastText or GoogleNews vectors. In the context of transformer models as well, performance of transfer learning on tweets generally remained lower when compared to less ambiguous forms of text such as Amazon product reviews. We also saw the importance of starting with simple baselines and analyzing component-level performance as models get more complex. This is particularly relevant when dealing with low-resource and sparse datasets like tweets. This calls for an in-depth quality analysis of existing tweet datasets and standardizing the annotation procedure.

Another direction is to study the non-transferable components in the context of low-resource tweet datasets. For example, *Sentiment* is easier to transfer than *Factoid* because there is more linguistic overlap in how sentiments are expressed in different domains. However, it is not immediately clear that Factoid phrases such as ‘*Indian army names aid mission*’ shown in Fig. 4.3 are in fact finding transferable representations. This calls for methods that leverage external knowledge to link entities [206] and then learn heterogeneous graph representations [207] for nodes and relations. Although, this is a growing field, tweet-based implementations (for the crisis domain in particular) are still scarce.

9.2.2 Interpretability

Interpretable nature should be an essential component of machine learning models that are deployed during emergency scenarios. This brings accountability and connects the emergency managers or any downstream users of such complex systems in a more meaningful way and can avoid ambiguities. With the advent of transformer models such as BERT

[8], where self-attention captures more contextual and complex features than the previous attention models [67], where the word-level weights have naturally interpretable meaning, an intuitive interpolation of interpretability is a challenging task [184]. This also includes the directions of studying transformers for symbolic reasoning [208]. Addressing this in the context of domain adaptation with low-resource tweet datasets is an impactful future direction.

9.2.3 Language Families

We showed the impact of analyzing language families in studying linguistic dependencies with respect to the underlying language model (e.g., training in Turkic language group helps Japanese; Fig. 7.2). An in-depth analysis with the help of linguistic researchers may lead to uncover more interesting connections between languages in the context of pre-trained embeddings (such as mBERT or XLM-R). This is particularly beneficial when training models that can be generalized to unseen low-resource languages where enough training data is not available.

9.2.4 Aligning Multilingual Knowledge Graphs for Crisis

Multilingual KGs such as DBpedia [209] and YAGO [23] have been successfully utilized in several cross-lingual applications. Among numerous ideas proposed for this alignment problem, embedding-based approaches such as MTransE [24], JAPE [25], a joint embedding method [210], and an iterative entity alignment method [211] stand out due to their strong performance without requiring machine translation or feature engineering. A newer approach [212] that showed to outperform the embedding-based methods utilizes graph convolution network (GCN) to train entity embeddings of each language into the same vector space, given a set of pre-aligned entities. However, crisis management still remains a challenging domain that could benefit from such multilingual KGs. A promising future direction is to incorporate such methods to build multilingual crisis KGs.

9.2.5 Transliteration

In our work, we acknowledge the limitations posed by the existing Hindi and Malayalam translation and transliteration tools. For example, the Indic transliteration tool adds an extra ‘*a*’ for some Hindi words which is unnecessary (eg., ‘*sandarbh*’ vs ‘*sandarbha*’) which may not reflect how native users tend to write. We expect that an improved transliteration system would further improve the downstream accuracy. This is also the case for many Indic languages where extensive datasets are not available. Expansion of our model to other Indic languages such that their translations and transliterations are aligned in state-of-the-art language models like mBERT/XLM-R is left as future work. Another future direction is to perform a detailed sensitivity analysis over β_2 and β_3 parameter that tunes the unsupervised loss (L_u) in Eq. 8.2, which might also address some of the exceptions shown in Tables 8.3 and 8.4 when, in some cases, the model does not preserve performance on English or non-romanized target data. To summarize, cross-lingual transliteration problem still has many areas of improvement on both the process of transliteration as well as language modeling.

Bibliography

Bibliography

- [1] J. W. Tukey, *The collected works of John W. Tukey*. Taylor & Francis, 1984, vol. 1.
- [2] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [3] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
- [4] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Gated feedback recurrent neural networks,” in *International Conference on Machine Learning*, 2015, pp. 2067–2075.
- [5] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *arXiv preprint arXiv:1910.10683*, 2019.
- [6] G. Lample and A. Conneau, “Cross-lingual language model pretraining,” *arXiv preprint arXiv:1901.07291*, 2019.
- [7] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, “Unsupervised cross-lingual representation learning at scale,” *arXiv preprint arXiv:1911.02116*, 2020.
- [8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.
- [9] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou, “Word translation without parallel data,” *arXiv preprint arXiv:1710.04087*, 2017.
- [10] M. Artetxe, G. Labaka, and E. Agirre, “A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 789–798.
- [11] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

- [12] M. Chen, Z. Xu, K. Weinberger, and F. Sha, “Marginalized denoising autoencoders for domain adaptation,” *arXiv preprint arXiv:1206.4683*, 2012.
- [13] A. Conneau, R. Rinott, G. Lample, A. Williams, S. Bowman, H. Schwenk, and V. Stoyanov, “Xnli: Evaluating cross-lingual sentence representations,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2475–2485.
- [14] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *EMNLP*, 2014, pp. 1532–1543.
- [15] S. Ruder and B. Plank, “Strong baselines for neural semi-supervised learning under domain shift,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 1044–1054.
- [16] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by backpropagation,” in *International conference on machine learning*. PMLR, 2015, pp. 1180–1189.
- [17] Z. Li, Y. Zhang, Y. Wei, Y. Wu, and Q. Yang, “End-to-end adversarial memory network for cross-domain sentiment classification,” in *IJCAI*, 2017, pp. 2237–2243.
- [18] Z. Li, Y. Wei, Y. Zhang, and Q. Yang, “Hierarchical attention transfer network for cross-domain sentiment classification,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [19] K. Zhang, H. Zhang, Q. Liu, H. Zhao, H. Zhu, and E. Chen, “Interactive attention transfer network for cross-domain sentiment classification,” 2019.
- [20] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, “End-to-end memory networks,” *arXiv preprint arXiv:1503.08895*, 2015.
- [21] T.-C. Liu, Y.-H. Wu, and H.-Y. Lee, “Query-based attention cnn for text similarity map,” *arXiv preprint arXiv:1709.05036*, 2017.
- [22] X. Yue, Z. Yao, S. Lin, H. Sun *et al.*, “Cliniqg4qa: Generating diverse questions for domain adaptation of clinical question answering,” *arXiv preprint arXiv:2010.16021*, 2020.
- [23] F. M. Suchanek, G. Kasneci, and G. Weikum, “Yago: A large ontology from wikipedia and wordnet,” *Journal of Web Semantics*, vol. 6, no. 3, pp. 203–217, 2008.
- [24] M. Chen, Y. Tian, M. Yang, and C. Zaniolo, “Multilingual knowledge graph embeddings for cross-lingual knowledge alignment,” *arXiv preprint arXiv:1611.03954*, 2016.
- [25] Z. Sun, W. Hu, and C. Li, “Cross-lingual entity alignment via joint attribute-preserving embedding,” in *International Semantic Web Conference*. Springer, 2017, pp. 628–644.
- [26] G. Hinton and S. T. Roweis, “Stochastic neighbor embedding,” in *NIPS*, vol. 15. Citeseer, 2002, pp. 833–840.

- [27] J. Blitzer, M. Dredze, and F. Pereira, “Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification,” in *Proceedings of the 45th annual meeting of the association of computational linguistics*, 2007, pp. 440–447.
- [28] L. Qin, M. Ni, Y. Zhang, and W. Che, “Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual nlp,” *arXiv preprint arXiv:2006.06402*, 2020.
- [29] J. Yang, S. Ma, D. Zhang, S. Wu, Z. Li, and M. Zhou, “Alternating language modeling for cross-lingual pre-training,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 9386–9393.
- [30] W. Xu, B. Haider, and S. Mansour, “End-to-end slot alignment and recognition for cross-lingual nlu,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 5052–5063.
- [31] D. Yarowsky, G. Ngai, and R. Wicentowski, “Inducing multilingual text analysis tools via robust projection across aligned corpora,” Johns Hopkins Univ Baltimore MD Dept of Computer Science, Tech. Rep., 2001.
- [32] R. Shah, B. Lin, A. Gershman, and R. Frederking, “Synergy: a named entity recognition system for resource-scarce languages such as swahili using online machine translation,” in *Proceedings of the Second Workshop on African Language Technology (AfLaT 2010)*, 2010, pp. 21–26.
- [33] J. Ni, G. Dinu, and R. Florian, “Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1470–1480.
- [34] J. Krishnan, H. Purohit, and H. Rangwala, “Diversity-based generalization for unsupervised text classification under domain shift,” in *Proceedings of the 19th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2020.
- [35] —, “Unsupervised and interpretable domain adaptation to rapidly filter tweets for emergency services,” in *Proceedings of the 2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2020.
- [36] —, “Attention realignment and pseudo-labelling for interpretable cross-lingual classification of crisis tweets,” in *Proceedings of KDD Workshop on Knowledge-infused Mining and Learning*, 2020.
- [37] J. Krishnan, A. Anastasopoulos, H. Purohit, and H. Rangwala, “Multilingual code-switching for zero-shot cross-lingual intent prediction and slot filling,” *arXiv preprint arXiv:2103.07792*, 2021.
- [38] T. Mikolov, E. Grave, P. Bojanowski, C. Puhersch, and A. Joulin, “Advances in pre-training distributed word representations,” in *LREC*, 2018.
- [39] F. Alam, S. Joty, and M. Imran, “Domain adaptation with adversarial training and graph embeddings,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 1077–1087.

- [40] J. Blitzer, R. McDonald, and F. Pereira, “Domain adaptation with structural correspondence learning,” in *Proceedings of the 2006 conference on empirical methods in natural language processing*, 2006, pp. 120–128.
- [41] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, “Analysis of representations for domain adaptation,” in *Advances in neural information processing systems*, 2007, pp. 137–144.
- [42] 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 19th international conference on World wide web*. ACM, 2010, pp. 751–760.
- [43] X. Glorot, A. Bordes, and Y. Bengio, “Domain adaptation for large-scale sentiment classification: A deep learning approach,” in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 513–520.
- [44] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [45] Z.-H. Zhou and M. Li, “Tri-training: Exploiting unlabeled data using three classifiers,” *IEEE Transactions on Knowledge & Data Engineering*, no. 11, pp. 1529–1541, 2005.
- [46] K. Saito, Y. Ushiku, and T. Harada, “Asymmetric tri-training for unsupervised domain adaptation,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2988–2997.
- [47] W. Cui, G. Zheng, Z. Shen, S. Jiang, and W. Wang, “Transfer learning for sequences via learning to collocate,” *arXiv preprint arXiv:1902.09092*, 2019.
- [48] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
- [49] C. Castillo, “Big crisis data: Social media in disasters and time-critical situations.” Cambridge University Press, 2016.
- [50] I. Varga, M. Sano, K. Torisawa, C. Hashimoto, K. Ohtake, T. Kawai, J.-H. Oh, and S. De Saeger, “Aid is out there: Looking for help from tweets during a large scale disaster,” in *ACL*, 2013, pp. 1619–1629.
- [51] B. Takahashi, E. C. Tandoc Jr, and C. Carmichael, “Communicating on twitter during a disaster: An analysis of tweets during typhoon haiyan in the philippines,” *Computers in Human Behavior*, vol. 50, pp. 392–398, 2015.
- [52] N. Pourebrahim, S. Sultana, J. Edwards, A. Gochanour, and S. Mohanty, “Understanding communication dynamics on twitter during natural disasters: A case study of hurricane sandy,” *International Journal of Disaster Risk Reduction*, vol. 37, p. 101176, 2019.
- [53] M. Imran, P. Mitra, and C. Castillo, “Twitter as a lifeline: Human-annotated twitter corpora for nlp of crisis-related messages,” *arXiv preprint arXiv:1605.05894*, 2016.

- [54] H. Li, D. Caragea, C. Caragea, and N. Herndon, “Disaster response aided by tweet classification with a domain adaptation approach,” *Journal of Contingencies and Crisis Management*, vol. 26, no. 1, pp. 16–27, 2018.
- [55] K. Lee, A. Agrawal, and A. Choudhary, “Real-time disease surveillance using twitter data: Demonstration on flu and cancer,” in *ACM SIGKDD*, 2013, pp. 1474–1477.
- [56] A. Acar and Y. Muraki, “Twitter for crisis communication: Lessons learned from japan’s tsunami disaster,” *International Journal of Web Based Communities*, vol. 7, no. 3, pp. 392–402, 2011.
- [57] S. Vieweg, C. Castillo, and M. Imran, “Integrating social media communications into the rapid assessment of sudden onset disasters,” in *International Conference on Social Informatics*. Springer, 2014, pp. 444–461.
- [58] D. T. Nguyen, K. A. Al Mannai, S. Joty, H. Sajjad, M. Imran, and P. Mitra, “Robust classification of crisis-related data on social networks using convolutional neural networks,” in *ICWSM*, 2017, pp. 632–635.
- [59] R. Mazloom, H. Li, D. Caragea, C. Caragea, and M. Imran, “A hybrid domain adaptation approach for identifying crisis-relevant tweets,” *IJISCRAM*, vol. 11, no. 2, pp. 1–19, 2019.
- [60] M. Imran, C. Castillo, J. Lucas, P. Meier, and S. Vieweg, “Aidr: Artificial intelligence for disaster response,” in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 159–162.
- [61] F. Ofli, P. Meier, M. Imran, C. Castillo, D. Tuia, N. Rey, J. Briant, P. Millet, F. Reinhard, M. Parkan *et al.*, “Combining human computing and machine learning to make sense of big (aerial) data for disaster response,” *Big Data*, vol. 4, no. 1, pp. 47–59, 2016.
- [62] R. Caruana, “Multitask learning,” *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.
- [63] S. Ruder, “An overview of multi-task learning in deep neural networks,” *arXiv preprint arXiv:1706.05098*, 2017.
- [64] M. Long and J. Wang, “Learning multiple tasks with deep relationship networks,” *arXiv preprint arXiv:1506.02117*, vol. 2, p. 1, 2015.
- [65] S. Ruder¹², J. Bingel, I. Augenstein, and A. Søgaard, “Sluice networks: Learning what to share between loosely related tasks,” *STAT*, vol. 1050, p. 23, 2017.
- [66] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-y. Wang, “Representation learning using multi-task deep neural networks for semantic classification and information retrieval,” in *NAACL: Human Language Technologies*, 2015, pp. 912–921.
- [67] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [68] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *NeurIPS*, 2014, pp. 3104–3112.

- [69] D. Gunning, “Explainable artificial intelligence (xai),” *Defense Advanced Research Projects Agency (DARPA), and Web*, vol. 2, 2017.
- [70] A. Nguyen, J. Yosinski, and J. Clune, “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 427–436.
- [71] R. Jia and P. Liang, “Adversarial examples for evaluating reading comprehension systems,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2021–2031.
- [72] M. T. Ribeiro, S. Singh, and C. Guestrin, “Why should i trust you? explaining the predictions of any classifier,” in *ACM SIGKDD*, 2016, pp. 1135–1144.
- [73] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [74] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” in *NeurIPS*, 2016, pp. 2172–2180.
- [75] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, “Explaining explanations: An overview of interpretability of machine learning,” in *DSAA*. IEEE, 2018, pp. 80–89.
- [76] A. S. Ross, M. C. Hughes, and F. Doshi-Velez, “Right for the right reasons: Training differentiable models by constraining their explanations,” *arXiv preprint arXiv:1703.03717*, 2017.
- [77] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” *arXiv preprint arXiv:1906.08237*, 2019.
- [78] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019.
- [79] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, “Unsupervised cross-lingual representation learning at scale,” *arXiv preprint arXiv:1911.02116*, 2019.
- [80] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100,000+ questions for machine comprehension of text,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 2383–2392.
- [81] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. Van Durme, “Record: Bridging the gap between human and machine commonsense reading comprehension,” *arXiv preprint arXiv:1810.12885*, 2018.

- [82] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, “Boolq: Exploring the surprising difficulty of natural yes/no questions,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 2924–2936.
- [83] S. Garg, T. Vu, and A. Moschitti, “Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7780–7788.
- [84] X. Yao and B. Van Durme, “Information extraction over structured data: Question answering with freebase,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, pp. 956–966.
- [85] D. Chen, A. Fisch, J. Weston, and A. Bordes, “Reading wikipedia to answer open-domain questions,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1870–1879.
- [86] S. Wang, M. Yu, J. Jiang, W. Zhang, X. Guo, S. Chang, Z. Wang, T. Klinger, G. Tesauero, and M. Campbell, “Evidence aggregation for answer re-ranking in open-domain question answering,” *arXiv preprint arXiv:1711.05116*, 2017.
- [87] A. Roberts, C. Raffel, and N. Shazeer, “How much knowledge can you pack into the parameters of a language model?” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 5418–5426.
- [88] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by backpropagation,” in *International conference on machine learning*. PMLR, 2015, pp. 1180–1189.
- [89] H. Wang, Z. Gan, X. Liu, J. Liu, J. Gao, and H. Wang, “Adversarial domain adaptation for machine reading comprehension,” *arXiv preprint arXiv:1908.09209*, 2019.
- [90] Y. Cao, M. Fang, B. Yu, and J. T. Zhou, “Unsupervised domain adaptation on reading comprehension,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7480–7487.
- [91] R. Zhang, R. G. Reddy, M. A. Sultan, V. Castelli, A. Ferritto, R. Florian, E. S. Kayi, S. Roukos, A. Sil, and T. Ward, “Multi-stage pretraining for low-resource domain adaptation,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 5461–5468.
- [92] T. Takahashi, M. Taniguchi, T. Taniguchi, and T. Ohkuma, “Cler: Cross-task learning with expert representation to generalize reading and understanding,” in *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, 2019, pp. 183–190.
- [93] Y.-A. Chung, H.-Y. Lee, and J. Glass, “Supervised and unsupervised transfer learning for question answering,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1585–1594.

- [94] E. Alsentzer, J. Murphy, W. Boag, W.-H. Weng, D. Jindi, T. Naumann, and M. McDermott, “Publicly available clinical bert embeddings,” in *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, 2019, pp. 72–78.
- [95] C. Alberti, D. Andor, E. Pitler, J. Devlin, and M. Collins, “Synthetic qa corpora generation with roundtrip consistency,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 6168–6173.
- [96] S. Shakeri, C. dos Santos, H. Zhu, P. Ng, F. Nan, Z. Wang, R. Nallapati, and B. Xiang, “End-to-end synthetic data generation for domain adaptation of question answering systems,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 5445–5460.
- [97] M. Xiao and Y. Guo, “Semi-supervised representation learning for cross-lingual text classification,” in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1465–1475.
- [98] A. O. Muis, N. Otani, N. Vyas, R. Xu, Y. Yang, T. Mitamura, and E. Hovy, “Low-resource cross-lingual event type detection via distant supervision with minimal effort,” in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 70–82.
- [99] A. Eriguchi, M. Johnson, O. Firat, H. Kazawa, and W. Macherey, “Zero-shot cross-lingual classification using multilingual neural machine translation,” *arXiv preprint arXiv:1809.04686*, 2018.
- [100] S. Srivastava, I. Labutov, and T. Mitchell, “Zero-shot learning of classifiers from natural language quantification,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 306–316.
- [101] D. T. Nguyen, K. Al-Mannai, S. R. Joty, H. Sajjad, M. Imran, and P. Mitra, “Robust classification of crisis-related data on social networks using convolutional neural networks.” *ICWSM*, vol. 31, no. 3, pp. 632–635, 2017.
- [102] E. W. Pamungkas and V. Patti, “Cross-domain and cross-lingual abusive language detection: A hybrid approach with deep learning and a multilingual lexicon,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, 2019, pp. 363–370.
- [103] L. Stappen, F. Brunn, and B. Schuller, “Cross-lingual zero-and few-shot hate speech detection utilising frozen transformer language models and axel,” *arXiv preprint arXiv:2004.13850*, 2020.
- [104] H. Purohit, C. Castillo, F. Diaz, A. Sheth, and P. Meier, “Emergency-relief coordination on social media: Automatically matching resource requests and offers,” *First Monday*, vol. 19, no. 1, Dec. 2013.
- [105] B. Pedrood and H. Purohit, “Mining help intent on twitter during disasters via transfer learning with sparse coding,” in *International Conference on Social Computing*,

Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation. Springer, 2018, pp. 141–153.

- [106] J. R. Chowdhury, C. Caragea, and D. Caragea, “Cross-lingual disaster-related multi-label tweet classification with manifold mixup,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, 2020, pp. 292–298.
- [107] G. Ma, “Tweets classification with bert in the field of disaster management,” <https://pdfs.semanticscholar.org/d226/185fa1e14118d746cf0b04dc5be8f545ec24.pdf>, 2019.
- [108] M. Imran, C. Castillo, F. Diaz, and S. Vieweg, “Processing social media messages in mass emergency: A survey,” *ACM Computing Surveys (CSUR)*, vol. 47, no. 4, pp. 1–38, 2015.
- [109] A. Joulin, P. Bojanowski, T. Mikolov, H. Jégou, and E. Grave, “Loss in translation: Learning bilingual word mapping with a retrieval criterion,” *arXiv preprint arXiv:1804.07745*, 2018.
- [110] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” *arXiv preprint arXiv:1508.04025*, 2015.
- [111] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril *et al.*, “Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces,” *arXiv preprint arXiv:1805.10190*, 2018.
- [112] P. Price, “Evaluation of spoken language systems: The atis domain,” in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.
- [113] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen, “Slot-gated modeling for joint slot filling and intent prediction,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, pp. 753–757.
- [114] P. Niu, Z. Chen, M. Song *et al.*, “A novel bi-directional interrelated model for joint intent detection and slot filling,” *arXiv preprint arXiv:1907.00390*, 2019.
- [115] M. Hardalov, I. Koychev, and P. Nakov, “Enriched pre-trained transformers for joint slot filling and intent detection,” *arXiv preprint arXiv:2004.14848*, 2020.
- [116] Q. Chen, Z. Zhuo, and W. Wang, “Bert for joint intent classification and slot filling,” *arXiv preprint arXiv:1902.10909*, 2019.
- [117] S. Upadhyay, M. Faruqui, G. Tür, H.-T. Dilek, and L. Heck, “(almost) zero-shot cross-lingual spoken language understanding,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6034–6038.
- [118] S. Schuster, S. Gupta, R. Shah, and M. Lewis, “Cross-lingual transfer learning for multilingual task oriented dialog,” *arXiv preprint arXiv:1810.13327*, 2018.

- [119] T. Pires, E. Schlinger, and D. Garrette, “How multilingual is multilingual bert?” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4996–5001.
- [120] X. Wan, “Co-training for cross-lingual sentiment classification,” in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009, pp. 235–243.
- [121] K. Yu, H. Li, and B. Oguz, “Multilingual seq2seq training with similarity loss for cross-lingual document classification,” in *Proceedings of The Third Workshop on Representation Learning for NLP*, 2018, pp. 175–179.
- [122] A. Zirikly and M. Hagiwara, “Cross-lingual transfer of named entity recognizers without parallel corpora,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2015, pp. 390–396.
- [123] C.-T. Tsai, S. Mayhew, and D. Roth, “Cross-lingual named entity recognition via wikification,” in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 2016, pp. 219–228.
- [124] J. Xie, Z. Yang, G. Neubig, N. A. Smith, and J. G. Carbonell, “Neural cross-lingual named entity recognition with minimal resources,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 369–379.
- [125] O. Täckström, D. Das, S. Petrov, R. McDonald, and J. Nivre, “Token and type constraints for cross-lingual part-of-speech tagging,” *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 1–12, 2013.
- [126] B. Plank and Ž. Agić, “Distant supervision from disparate sources for low-resource part-of-speech tagging,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 614–620.
- [127] X. He, L. Deng, D. Hakkani-Tur, and G. Tur, “Multi-style adaptive training for robust cross-lingual spoken language understanding,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8342–8346.
- [128] X. Chen, Y. Sun, B. Athiwaratkun, C. Cardie, and K. Weinberger, “Adversarial deep averaging networks for cross-lingual sentiment classification,” *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 557–570, 2018.
- [129] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.
- [130] A. Conneau, S. Wu, H. Li, L. Zettlemoyer, and V. Stoyanov, “Emerging cross-lingual structure in pretrained language models,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 6022–6034. [Online]. Available: <https://www.aclweb.org/anthology/2020.acl-main.536>

- [131] B. Liu and I. Lane, “Attention-based recurrent neural network models for joint intent detection and slot filling,” *arXiv preprint arXiv:1609.01454*, 2016.
- [132] Y. Xian, B. Schiele, and Z. Akata, “Zero-shot learning-the good, the bad and the ugly,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4582–4591.
- [133] B. Romera-Paredes and P. Torr, “An embarrassingly simple approach to zero-shot learning,” in *International Conference on Machine Learning*, 2015, pp. 2152–2161.
- [134] G. Aguilar and T. Solorio, “From english to code-switching: Transfer learning with strong morphological clues,” *arXiv preprint arXiv:1909.05158*, 2019.
- [135] V. Soto and J. Hirschberg, “Joint part-of-speech and language id tagging for code-switched data,” in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, 2018, pp. 1–10.
- [136] K. Ball and D. Garrette, “Part-of-speech tagging for code-switched, transliterated texts without explicit language identification,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 3084–3089.
- [137] A. Joshi, A. Prabhu, M. Shrivastava, and V. Varma, “Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text,” in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2482–2491.
- [138] D. Mave, S. Maharjan, and T. Solorio, “Language identification and analysis of code-switched social media text,” in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, 2018, pp. 51–61.
- [139] Z. Yirmibeşoğlu and G. Eryigit, “Detecting code-switching between turkish-english language pair,” in *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, 2018, pp. 110–115.
- [140] M. Mager, Ö. Çetinoğlu, and K. Kann, “Subword-level language identification for intra-word code-switching,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 2005–2011.
- [141] A. R. KhudaBukhsh, S. Palakodety, and J. G. Carbonell, “Harnessing code switching to transcend the linguistic barrier,” *arXiv preprint arXiv:2001.11258*, 2020.
- [142] Z. Jiang, A. Anastasopoulos, J. Araki, H. Ding, and G. Neubig, “Multilingual factual knowledge retrieval from pretrained language models,” *arXiv preprint arXiv:2010.06189*, 2020.
- [143] S. Tan and S. Joty, “Code-mixing on sesame street: Dawn of the adversarial polyglots,” *arXiv preprint arXiv:2103.09593*, 2021.
- [144] A. Chaudhary, K. Raman, K. Srinivasan, and J. Chen, “Dict-mlm: Improved multilingual pre-training using bilingual dictionaries,” *arXiv preprint arXiv:2010.12566*, 2020.

- [145] M. Kale and A. Siddhant, “Mixout: A simple yet effective data augmentation scheme for slot-filling,” in *Conversational Dialogue Systems for the Next Decade*. Springer, 2021, pp. 279–288.
- [146] P. Dufter and H. Schütze, “Identifying elements essential for bert’s multilinguality,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4423–4437.
- [147] X. Bi, B. A. Smith, and S. Zhai, “Multilingual touchscreen keyboard design and optimization,” *Human–Computer Interaction*, vol. 27, no. 4, pp. 352–382, 2012.
- [148] ClusterDev, “Malayalam keyboard,” *clusterdev.com*, 2020.
- [149] S. L. Smith, D. H. Turban, S. Hamblin, and N. Y. Hammerla, “Offline bilingual word vectors, orthogonal transformations and the inverted softmax,” *arXiv preprint arXiv:1702.03859*, 2017.
- [150] H. Aldarmaki and M. Diab, “Context-aware cross-lingual mapping,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 3906–3911.
- [151] T. Schuster, O. Ram, R. Barzilay, and A. Globerson, “Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 1599–1613.
- [152] S. Cao, N. Kitaev, and D. Klein, “Multilingual alignment of contextual word representations,” in *International Conference on Learning Representations*, 2019.
- [153] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
- [154] B. Shin, H. Yang, and J. D. Choi, “The pupil has become the master: teacher-student model-based word embedding distillation with ensemble learning,” *arXiv preprint arXiv:1906.00095*, 2019.
- [155] Y.-C. Chen, Z. Gan, Y. Cheng, J. Liu, and J. Liu, “Distilling the knowledge of bert for text generation,” *arXiv preprint arXiv:1911.03829*, 2019.
- [156] S. Hahn and H. Choi, “Self-knowledge distillation in natural language processing,” in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, 2019, pp. 423–430.
- [157] W. Lewis, R. Munro, and S. Vogel, “Crisis mt: Developing a cookbook for mt in crisis situations,” in *Proceedings of the Sixth Workshop on Statistical Machine Translation*, 2011, pp. 501–511.
- [158] U. N. O. F. T. C. O. H. A. (Unocha), “Global humanitarian response plan covid-19,” 2020.

- [159] C. Nanda, M. Dua, and G. Nanda, "Sentiment analysis of movie reviews in hindi language using machine learning," in *2018 International Conference on Communication and Signal Processing (ICCSP)*. IEEE, 2018, pp. 1069–1072.
- [160] P. Arora, "Sentiment analysis for hindi language," Ph.D. dissertation, International Institute of Information Technology Hyderabad, 2013.
- [161] Y. Sharma, V. Mangat, and M. Kaur, "A practical approach to sentiment analysis of hindi tweets," in *2015 1st International Conference on Next Generation Computing Technologies (NGCT)*. IEEE, 2015, pp. 677–680.
- [162] M. S. Akhtar, A. Ekbal, and P. Bhattacharyya, "Aspect based sentiment analysis in hindi: resource creation and evaluation," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 2703–2709.
- [163] P. Sharma and T.-S. Moh, "Prediction of indian election using sentiment analysis on hindi twitter," in *2016 IEEE international conference on big data (big data)*. IEEE, 2016, pp. 1966–1971.
- [164] M. A. Ansari and S. Govilkar, "Sentiment analysis of mixed code for the transliterated hindi and marathi texts," *International Journal on Natural Language Computing (IJNLC) Vol*, vol. 7, 2018.
- [165] A. Balamurali, A. Joshi, and P. Bhattacharyya, "Cross-lingual sentiment analysis for indian languages using linked wordnets," in *Proceedings of COLING 2012: Posters*, 2012, pp. 73–82.
- [166] P. Singh and E. Lefever, "Sentiment analysis for hinglish code-mixed tweets by means of cross-lingual word embeddings," in *Proceedings of the The 4th Workshop on Computational Approaches to Code Switching*, 2020, pp. 45–51.
- [167] D. S. Nair, J. P. Jayan, R. Rajeev, and E. Sherly, "Sentiment analysis of malayalam film review using machine learning techniques," in *2015 international conference on advances in computing, communications and informatics (ICACCI)*. IEEE, 2015, pp. 2381–2384.
- [168] S. S. Kumar, M. A. Kumar, and K. Soman, "Sentiment analysis of tweets in malayalam using long short-term memory units and convolutional neural nets," in *International Conference on Mining Intelligence and Knowledge Exploration*. Springer, 2017, pp. 320–334.
- [169] D. S. Nair, J. P. Jayan, E. Sherly *et al.*, "Sentima-sentiment extraction for malayalam," in *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2014, pp. 1719–1723.
- [170] M. Ashna and A. K. Sunny, "Lexicon based sentiment analysis system for malayalam language," in *2017 International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE, 2017, pp. 777–783.
- [171] B. R. Chakravarthi, N. Jose, S. Suryawanshi, E. Sherly, and J. P. McCrae, "A sentiment analysis dataset for code-mixed malayalam-english," *arXiv preprint arXiv:2006.00210*, 2020.

- [172] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation,” *arXiv preprint arXiv:1609.08144*, 2016.
- [173] M. Junczys-Dowmunt, R. Grundkiewicz, T. Dwojak, H. Hoang, K. Heafield, T. Neckermann, F. Seide, U. Germann, A. Fikri Aji, N. Bogoychev, A. F. T. Martins, and A. Birch, “Marian: Fast neural machine translation in C++,” in *Proceedings of ACL 2018, System Demonstrations*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 116–121. [Online]. Available: <http://www.aclweb.org/anthology/P18-4020>
- [174] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, “Attention-based bidirectional long short-term memory networks for relation classification,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 207–212.
- [175] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [176] P. Karuna, M. Rana, and H. Purohit, “Citizenhelper: A streaming analytics system to mine citizen and web data for humanitarian organizations,” in *Eleventh International AAAI Conference on Web and Social Media*, 2017, pp. 729–730.
- [177] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.
- [178] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in *The 49th Annual Meeting of the Association for Computational Linguistics*, 2011.
- [179] R. Pandey and H. Purohit, “Citizenhelper-adaptive: Expert-augmented streaming analytics system for emergency services and humanitarian organizations,” in *ASONAM*. IEEE, 2018, pp. 630–633.
- [180] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [181] Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1408.5882*, 2014.
- [182] M. Imran, P. Mitra, and C. Castillo, “Twitter as a lifeline: Human-annotated twitter corpora for nlp of crisis-related messages,” in *LREC*, 2016.

- [183] R. Řehůřek and P. Sojka, “Software Framework for Topic Modelling with Large Corpora,” in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, 2010, pp. 45–50.
- [184] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, “What does bert look at? an analysis of bert’s attention,” in *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019, pp. 276–286.
- [185] M. Wan and J. McAuley, “Modeling ambiguity, subjectivity, and diverging viewpoints in opinion question answering systems,” in *2016 IEEE 16th international conference on data mining (ICDM)*. IEEE, 2016, pp. 489–498.
- [186] J. McAuley and A. Yang, “Addressing complex and subjective product-related queries with customer reviews,” in *Proceedings of the 25th International Conference on World Wide Web*, 2016, pp. 625–635.
- [187] J. Ni, J. Li, and J. McAuley, “Justifying recommendations using distantly-labeled reviews and fine-grained aspects,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 188–197.
- [188] W. Xiong, J. Wu, H. Wang, V. Kulkarni, M. Yu, S. Chang, X. Guo, and W. Y. Wang, “Tweetqa: A social media focused question answering dataset,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5020–5031.
- [189] S. Zong, A. Baheti, W. Xu, and A. Ritter, “Extracting covid-19 events from twitter,” *arXiv preprint arXiv:2006.02567*, 2020.
- [190] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>
- [191] T. Wolf, J. Chaumond, L. Debut, V. Sanh, C. Delangue, A. Moi, P. Cistac, M. Funtowicz, J. Davison, S. Shleifer *et al.*, “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45.
- [192] L. Biewald, “Experiment tracking with weights and biases,” 2020, software available from wandb.com. [Online]. Available: <https://www.wandb.com/>
- [193] D.-H. Lee, “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in *Workshop on challenges in representation learning, ICML*, vol. 3, no. 2, 2013.

- [194] L. A. Ramshaw and M. P. Marcus, “Text chunking using transformation-based learning,” in *Natural language processing using very large corpora*. Springer, 1999, pp. 157–176.
- [195] C. Dyer, V. Chahuneau, and N. A. Smith, “A simple, fast, and effective reparameterization of ibm model 2,” in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 644–648.
- [196] B. Rowe and D. Levine, “A concise introduction to linguistics,” *Routledge*. pp. 340–341, 2017.
- [197] C. F. Voegelin and F. M. Voegelin, “Classification and index of the world’s languages.” 1976.
- [198] W. Harbert, *The Germanic Languages*. Cambridge University Press, 2006.
- [199] C. P. Masica, *The indo-aryan languages*. Cambridge University Press, 1993.
- [200] W. D. Elcock and J. N. Green, *The romance languages*. Faber & Faber London, 1960.
- [201] R. Shafer, “Classification of the sino-tibetan languages,” *Word*, vol. 11, no. 1, pp. 94–111, 1955.
- [202] R. A. Miller, *The Japanese Language*. University of Chicago Press Chicago, 1967.
- [203] L. Johanson and É. Á. C. Johanson, *The Turkic Languages*. Routledge, 2015.
- [204] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [205] A. Kunchukuttan, “The IndicNLP Library,” https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf, 2020.
- [206] B. Harandizadeh and S. Singh, “Tweeki: Linking named entities on twitter to a knowledge graph,” in *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, 2020, pp. 222–231.
- [207] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, “Heterogeneous graph neural network,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 793–803.
- [208] P. Clark, O. Tafjord, and K. Richardson, “Transformers as soft reasoners over language,” *arXiv preprint arXiv:2002.05867*, 2020.
- [209] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, “Dbpedia-a crystallization point for the web of data,” *Journal of web semantics*, vol. 7, no. 3, pp. 154–165, 2009.
- [210] Y. Hao, Y. Zhang, S. He, K. Liu, and J. Zhao, “A joint embedding method for entity alignment of knowledge bases,” in *China Conference on Knowledge Graph and Semantic Computing*. Springer, 2016, pp. 3–14.

- [211] H. Zhu, R. Xie, Z. Liu, and M. Sun, “Iterative entity alignment via joint knowledge embeddings.” in *IJCAI*, 2017, pp. 4258–4264.
- [212] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, “Cross-lingual knowledge graph alignment via graph convolutional networks,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 349–357.

Curriculum Vitae

Jitin Krishnan received his Bachelor's degree in Computer Science and minor in Economics from the University of Virginia, USA in 2012. He received his PhD degree in Computer Science from George Mason University, USA in 2021. He worked closely under the guidance of Dr. Huzefa Rangwala and Dr. Hemant Purohit. During the course of his doctoral study, he worked as a Graduate Teaching Assistant, Graduate Research Assistant, and Graduate Lecturer. As an instructor, he taught CS112 Introduction to Programming, CS211 Object Oriented Programming, and CS310 Data Structures. During the summer terms, he did internships with NASA, Facebook, and Amazon. After graduation, he plans to join Facebook (Seattle WA) as a Research Scientist.