CLASS REPRESENTATIVE PROJECTION FOR TEXT-BASED ZERO-SHOT LEARNING

A Thesis
IN
Computer Science

Presented to the Faculty of the University
of Missouri–Kansas City in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

by
SAI SRI NARNE

RVR & JC College of Engineering, Andhra Pradesh, India, 2018

Kansas City, Missouri
2020

© 2020
SAI SRI NARNE
ALL RIGHTS RESERVED

CLASS REPRESENTATIVE PROJECTION FOR TEXT-BASED ZERO-SHOT LEARNING

Sai Sri Narne, Candidate for the Master of Science Degree

University of Missouri–Kansas City, 2020

ABSTRACT

There have been significant advances in supervised machine learning and enormous benefits from deep learning for a range of diverse applications. Despite the success of deep learning, in reality, very few works have shown progress in text classification. Transfer learning, known as the zero-shot learning (ZSL) or generalized zero-shot learning (G-ZSL), is receiving much attention due to its ability to transfer knowledge learned from a known (seen) domain to unknown (unseen) domains. But most of the ZSL works are relying on large training corpus and external semantic knowledge. Thus, there are very few studies that have investigated the improvement of text classification performance in sorely text-based ZSL/G-ZSL.

In this thesis, a class representative framework was proposed for text-based ZSL by designing the novel projection method, learned from the seen classes, and applying it to transfer the knowledge to the unseen classes effectively. We designed a three-step

approach, which consists of (1) sentence-based embeddings, (2) deep neural networks, and (3) class-based representative classifiers. Experimental results show that the proposed projection framework achieves the best classification results in text-based ZSL/G-ZSL compared with the state-of-the-art approaches investigated with three benchmark datasets including large newsgroup post of 20 classes called 20 Newsgroup Dataset and DBpedia dataset on various topics.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Computing and Engineering, have examined a thesis titled " Class Representative Projection for Text-based Zero-Shot Learning ," presented by Sai Sri Narne, candidate for the Master of Science degree, and hereby certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Yugyung Lee, Ph.D. Committee Chair
Department of Computer Science & Electrical Engineering

Roozmehr Safi, Ph.D.
Department of Management

Ye Wang, Ph.D.
Department of Communication Studies

CONTENTS

ILLUSTRATIONS

TABLES

ACKNOWLEDGEMENTS

I feel extremely blessed for working under the supervision of Dr. Yugyung Lee and might want to express gratitude towards her for the valuable and constructive suggestions throughout the research work as my advisor. Her knowledge, inspiring experience and immediate response helped me to work with all the enthusiasm and think bound the scope to achieve in the whole work. She is very patient in listening to all the new ideas, pragmatic in giving suggestions and always helps me in doing the reality check. Her astonishing vitality and eagerness consistently propel me to go the extra mile. It has been a huge honor to work with her on different projects besides the thesis.

I would like to thank my committee: Dr. Ye Wang and Dr. Roozmehr Safi for their insightful comments and encouragement. I also wish to thank the PhD students Mayanka Chandrasekhar and Vijaya Kumari Yeruva for their continuous support and guidance. I thank my colleague, Priyanka Gaikwad, for her valuable insight and comments all through my thesis.

I would also like to thank the University of Missouri-Kansas City, for providing me the perfect environment to research. It provided me with many opportunities to support myself and world-class facilities to conduct research with the finest machines available without which the thesis work could not have been accomplished.

Finally, I would like to thank my family and friends who always encourage me, gave me valuable suggestions throughout the research and made sure that I pursue my dream without any problems.

CHAPTER 1

INTRODUCTION

Natural language processing (NLP) is a branch of artificial intelligence that supports computers to understand, infer, and engineer human language. NLP covers many domains, including computer science and computational linguistics, to fill the space between human communication and computer understanding. By extending the capabilities of Natural Language Processing, we can classify or categories the contextual meaning in the sentence to a specific class.

The cause of technology invention is to make lives more comfortable. It is hard to go through significant texts and understand what exactly meant to be. But what if there is a classification model that classifies, standardize the data and make it simple to navigate through. Consider, for example, an image classification technique that does a good job explaining what the picture contains. If the image resembles a cat, the model says it's a cat. Likewise, there is a large text referring to science and technology; the text classification model should classify this under science and technology. The so-called text classification assigns one or more labels to a data point. In practice, semantic analysis and sentiment classification does a similar job. Semantic analysis tends to classify the textual data under positive, negative, and strength of being positive or negative based on a word embedding. This job is done quite quickly with available data for both the classes, but then it is quite hard if there are a more significant number of classes and accessible

information.

The text classification technique becomes more complicated when dealing with multi-label classification. The usage of resources like training data and model inferencing is quite an issue when it comes to multi-label tasking. Then came the Zero-shot learning into focus, ZSL setting shows promising results in achieving the goals for inferencing the unseen class during the training. Most of the ZSL algorithms use some connection between the available information or seen data to the unseen data.

1.1 Problem Statement

There are numerous species and items without named information and new visual classifications, such as the most recent contraptions or vehicle models presented regularly. The problem for computation of multi-label classification has become more complex for a few labeled data; more techniques are proposed to deal with lesser training data. As research started to advance, more and more techniques came up. One of them is ”Zero-shot learning,” it has been a very active research area over recent years. The strategy of ZSL includes preparing a model on an enormous corpus of sentences to gain proficiency with the connection between a sentence and embedding of sentence's labels. Learning the relationships between the data and labels helps to understand the unseen data for the model. Though the technique has gained more popularity in recent times, there are few assumptions made are the text information was auxiliary information used for rendering a better representation of data. We are inspired by the works in ZSL and the use of more sophisticated classifiers and auxiliary information for creating better representations for

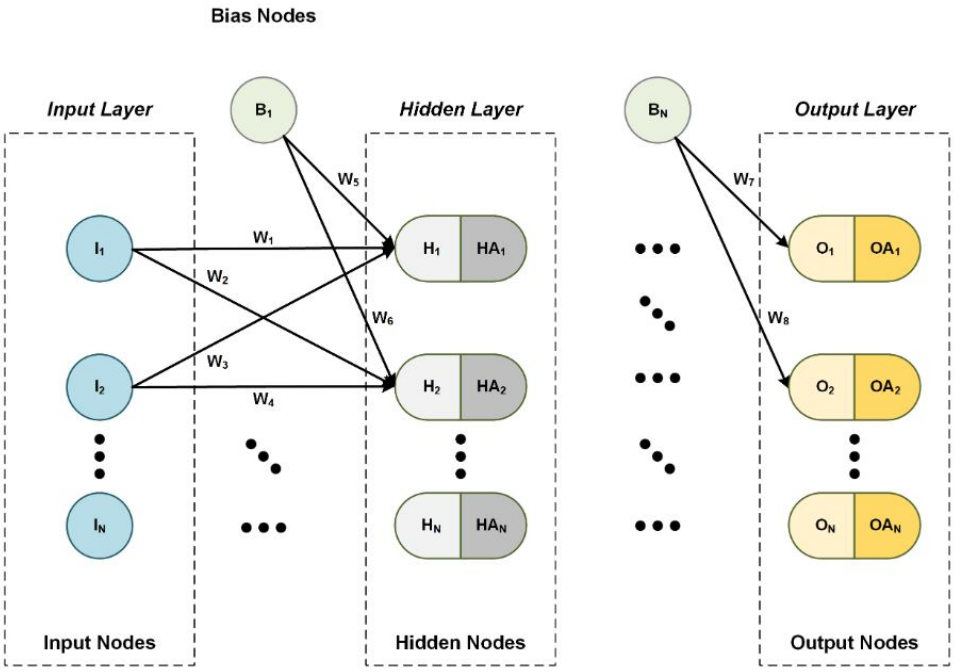

Figure 1: Traditional classification Architecture

text classification. We propose a novel ZSL framework based on an embedding space and ZSL projection method to classify unseen categories in text data by learning on seen classes in the text.

1.2 Proposed Solution

As mentioned in the problem statement, in our work, we proposed a new representation in zero-shot learning. The contributions of our work can be summarised as follows: We propose a new framework for Zero-Shot text classification, a framework for the design of abstract representatives as classifiers, and suggest innovation in ZSL research. Unlike previous ZSL works [1]–[3] in which they require extra information besides text data, our

framework does not request any external or auxiliary data. We use the productive relationship between embeddings of text and represent them as inputs for the classifier. We create prototypes for each class as classifiers, called Class Representatives, independent of other categories. These class representatives hold on the rich information for each class label and try to connect or drag some relationship patterns with the unseen label data.

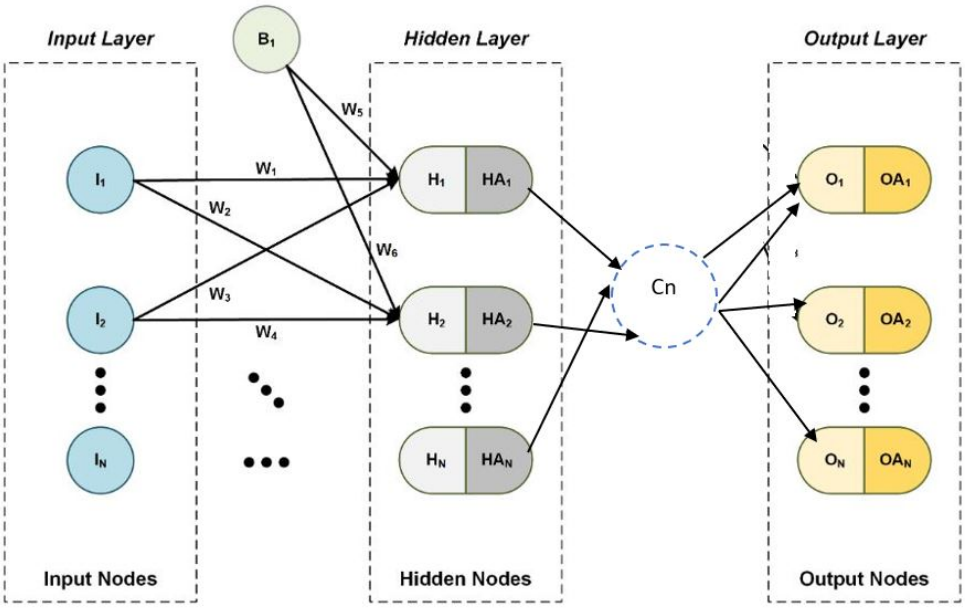

Figure 2: Proposed CRL Architecture

In the Figure 1 shows an illustration of the traditional classification architecture where the I (1 to N) is referred to show the inputs text documents, the weights and bias are added on for the hidden layer information. The softmax output is shown as O (1 to n); in this case, n is the number of classes. This is the traditional classifier architecture. Our CRL architecture is shown in 2; this architecture differs from the main one by creating a sum of hidden layers, i.e., show as Cn, where n is the number of classes. We have class

representative(CR) C for n classes, i.e., each class is represented by a CR.

We evaluate the Class Representative Learning (CRL) framework using metrics established by [4]. Our evaluation is done in two steps: focusing the performance CRL architecture and others with CRL architecture with zero-shot learning.

CHAPTER 2

BACKGROUND AND RELATED WORK

This chapter gives background information of various components used in the thesis and provides an overview of related work that will help understand this work better.

2.1 Related Work

2.1.1 Embedding Techniques

Natural Language processing techniques become famous for dealing with text abstractions. Learning of distributed representation has been successful for symbolic data and is first introduced by Hinton(1986)[5] later on, this representation has given a name as word embeddings.

BOW known for bag-of-words[6], this model is a way of representing the text in the form of a dictionary, so-called bag-of-words. It tries to count on the measure of the presence of words describing the frequency of the word in the dictionary document. This model is confined to size, space, and neglects the meaning of a sentence. In contrast, Word2Vec is a skip-gram neural network model[7] designed by a team of Google Inc., tries to learn the weights of the hidden layers in a neural net referred to as word vectors. These vectors represent the words in a dimensional space.

Therefore, words can be distinguished based on the distance between the vectors. Tomas Mikolov., who developed Word2Vec in 2013, came up with FastText[8] with a

Table 1: Text Embeddings and NLP

Embedding	Word Level	Sentence Level	Position of words/phrases	Segmentation	Sequence of phrases	Neural Network
Word2Vec [9]	✓	✗	✗	✗	✗	Dense Network
GloVe [10]	✓	✗	✗	✗	✗	Dense Network
BERT [11]	✓	✗	✓	✓	✓	Transformer Encoder
USE [12]	✓	✓	✓	✓	✗	Transformers, DAN
S-BERT [13]	✗	✓	✓	✓	✓	Transformers

group from Facebook AI Research. FastText[8], a super-fast model includes generating words representation that didn't appear at the time of training, also known as the out-of-vocabulary phenomenon.

Several works came up with the intuition of unsupervised learning with a super vision of sentiment and semantic knowledge. The state-of-art-model ElMo**Peters:2018**, is a bi-directional language models compute embedding in two states. ElMo (Embedding for language model) takes characters as inputs instead of a word, thus increase the out-of-vocabulary as FastText by considering sub-words. Later on, a novel approach of multi-tasking came into existence i.e which can do both unsupervised and supervised learning. Universal sentence embedding[12] computes the embedding vector for an entire sentence not just a word or a character, this helps to improve the model to include the context of the sentence. Recently [14] has published an Evaluation of sentence embeddings in downstream and linguistic probing tasks, this nice work by them compares on different techniques and pre-trained model details.

Talking about the text classifiers, there are a lot of text classifiers used widely. Among them are Deep Neural Network classifier(DNN Classifier), support vector machines, nearest neighbors, naive-Bayes, decision trees, etc. While each has its advantage over others, choosing a better classifier among them is a challenging task.

Text Classification Using Machine Learning Techniques[15] has an apt distinguish between different text classifiers. It concludes a combined approach of any classifiers has superior performance when compared with an individual classifier.

Doc2vec is one of the first sentence embedding techniques, [16] this is an unsupervised algorithm.

2.1.2 Zero-Shot Learning

In [17], a method for word embedding with LSTM network and aspect-based LSTM was proposed for Zero-Shot multiclass classification by learning the relationship between a sentence and embedding of sentence's tags and applying it for inferencing with unseen sentences and tags into the same embedding space.

In [18], a Discriminative and Generative LSTM model was proposed for ZSL with label-embedding space as an auxiliary task.

In [19], a neural architecture was proposed for handling a few- and zero-shot labels in the multi-label setting in the form of a DAG for tags with their natural language descriptor.

In [20], a zero-shot text classification framework was designed using data augmentation and feature augmentation. For an efficient ZSL, semantic knowledge, including word embeddings, class descriptions, class hierarchy, and a general knowledge graph, were incorporated into the proposed framework.

In [21], natural language descriptions were mapped to probabilistic assertions grounded in latent class labels. A classifier has been trained with quantitative constraints for guiding predictions from the learned models. A ZSL method was proposed for semantic utterance classification (SUC) by linking categories and utterances through a semantic space [22]. The discriminative semantic features were learned without supervision and will guide the learning of the semantic features.

A latent feature generation framework was proposed for generalized zero-shot learning (GZSL) that aims at improving the prediction on codes for the diagnoses of diseases [23]. For improved semantic consistency between the generated features and real features of the International Classification of Diseases (ICD), an adversarial generative model was designed for the GZSL on multi-label text classification.

A joint space of embedding documents and labels was designed for multi-label text and ZSL classification [24]. The zero-shot learning algorithm has been applied to the multi-label classification task in Medical Subject Headings (MeSH) assignment for biomedical publications.

CHAPTER 3

LITERATURE ON DOCUMENT EMBEDDINGS

3.1 Supervised Embedding Techniques

Supervised techniques make full use of the labeled data available to create the representation of its learning. The quality and heartiness of these techniques in this way depend intensely on the learning structure, yet additionally on how well the misleadingly planned learning objective requires or achieves the learning of significant highlights or information that would demonstrate valuable in different downstream undertakings. Expectations are reached by capturing the semantic and synthetic meaning of the words or content in a text document. Thus, some evaluations proved these techniques focus on rich learning information. Below are some of the existing embedding approaches that follow supervised learning architecture. These techniques are designed to make use of the in-depth neural network information in generating the mathematical embedded information.

3.1.1 Learning Text Representations from Labeled Data

This one of the first techniques developed to generate the sentence embedded information from the labeled set of data. [25] and [26] are the first one's made use of the different neural network fashion in generating the text representations overcoming the more general techniques of representation words using Word2Vec. [7] developed recurrent neural network encoder-decoder based representations where the encoder tries to map

the sequence between the uneven length of input with a specific size of vector and decoder maps the vector to the desired target sequence. The RNN-encoder-decoder has some hidden layers in between to enhance the training time and capacity of the memory. Later works focus on predicting the next series occurring using long-short term methodology. The ideology of using an LSTM is to handle the sequence to sequence problems. The input is read word by word and make a high fixed dimensional vector and also to extract the output from it.

3.1.2 Supervised Sentence Embedding using task-specific Data

A typical directed strategy to create record embeddings utilizes different neural system models, learning composition patterns that map word vectors to document vectors, and are passed to a supervised task that depends on a label of the class back-propagate through the composition weights. Therefore, most of all hidden layers of a network can be considered to generate a vector representation embedding for input document data. [27] have rigors ways of learning sentence vector embeddings based on word vectors and a supervised learning task Several other approaches have been developed for learning composition patterns/operators that map word vectors to sentence vectors, including recursive neural networks, recurrent networks, convolutional networks, and recursive-convolutional architecture others. All of these architectures produce sentence representations that pass on to a supervised task, and they depend on a class label to regenerate through the composition weights. Consequently, the methods learn high-quality sentence representations but tuned in such a way that they can only use them for a respective task. The paragraph

vector is proposed as an alternative to the above models because it can learn unsupervised sentence representations by introducing a distributed sentence indicator as part of a neural language model.

Skip-thought proposes a new architecture of learning the embeddings without the specific contest, and it is entirely unsupervised. Means, instead of making use of a word to predict its surrounding meaning, the authors in [28] encoded a sentence to predict the other sentences around it. Thus, any composition operator can act as a sentence encoder, and only the objective function becomes modified. Hence came in the name skip-thought vectors, more information can be seen in the following sections.

3.1.3 Transfer learning for sentence Representations

Distributed representations of words, also called word embedding, have appeared to give valuable highlights to different natural language processing phenomena and computer vision tasks. While there is an agreement concerning the support of word embeddings and how to learn them, this isn't yet clear to portrayals that convey the importance of a full sentence. That is, the way to catch the connections among different words and phrases in a single vector stays an inquiry to illuminate. In this [29], the authors proposed a new technique of learning the sentence representation in a supervised way, i.e., a sentence encoder model trained on a large corpus and subsequently transferred to other tasks. To perform this, the immediate two questions to be solved, build such an encoder, namely: what is the preferable neural network architecture, and how and on what task should such a network be trained. The authors then investigated whether supervised learning can be

leveraged instead of the pre-trained model before being transferred. The evaluations show that the sentence embeddings, which trained on various supervised tasks, have best transfer accuracy results from that of the sentence embeddings generated from models trained on a natural language inference (NLI) task.

3.1.4 Universal Sentence Encoder

Universal sentence encoder[12] works on two main concepts: the Transformer model and the Deep Averaging Network. Both of the designed models allow multi-task learning, with supported tasks including a skip-thought like a task for unsupervised learning, a conversational input-response task for the inclusion of parsed conversational data; and classification tasks for training on supervised data. The authors focus on different experiments with transfer learning tasks and benchmark their models versus simple Convolutional neural networks and deep averaging network baselines.

Universal sentence encoder converts the sentence into a 512-dimensional vector, which internally transforms the input document into chunks of words, so it can at times work as sentence-to-vector and also word-to-vector. It considers the whole document's data but doesn't concentrate on the order of the sentence, though it focuses on the sentences' segmentation.

The design model creates embeddings for sentences using the encoding sub-graph architecture from the transformer. The encoder uses an attention technique to compute context-aware representations of words in a sentence that considers both the order and identity of other words. The context-aware word representations are averaged along to

obtain a sentence-level embedding. Whereas in the DAN model, presented in [30], the input of words and bi-grams were first averaged together and then moved through a feed-forward deep neural network (DNN) to generate sentence embeddings for the given data.

3.2 Unsupervised Embedding Techniques

3.2.1 N-grams based Embedding Mechanism

Inspiration from pre-trained word to vector concept leads to training bi-grams and tri-grams embedding. N-grams are defined as a combination of 1,2....N words. This technique has extended word2vec's skip gram-model to handle the small number of sentences or short phrases. This n-gram based embedding technique tries to capture the context of a single word with others in the sentence, unlike the bag-of-words approach. Since the meaning is already achieved, it makes things easy to work with unlabeled data.

3.2.2 Shared representation of words and phrases

There is an exceptionally natural approach to build document embeddings from meaningful word embeddings: Given a sequence of words, play out some vector mathematics on all the vectors relating to the expressions of the document, to sum up, them into a single large vector in the equivalent installing space; two such standard outline administrators are sum and aggregation.

[31] describe a Siamese CBOW (continuous bag of words) architecture, an efficient neural network model for obtaining high-quality word embeddings to get directly optimized sentence representations from it. This setting of averaging words embeddings

and making better optimized to representation out of it has shown better evaluation results. [31] had evaluated this technique on 20 different data sets and has proven the sharing of word representation outperforms the concept, naturally choosing the representation for of single word.

After a series of years [32] have performed unsupervised evaluations on CBOW and skip-gram based embeddings architecture. His evaluations have shown good results for sentence representations using shared words concept.

3.2.3 Sentence to vector

This technique is pretty much a combination of section 3.2.1 and 3.2.2 approaches: The classic C-BOW model of word2vec is extended to consider word n-grams and explore the options to optimize the word (and n-grams) embeddings to aggregate them to yield sentence or phrase vectors. Sentence to vector, in short, also called sent2vec, can also be described as an unsupervised model of fastText [8] where the single text sentence or phrase is the context, and all the possible label words are determined to be vocabulary. In addition to this, the input sub-sampling technique removed; instead, the architecture considers the entire phrase in context. This implies both that (a) the utilization of regular word sub-sampling is disposed of so as not to forestall the age of n-grams highlights and (b) the dynamic setting windows utilized by word2vec are carried off. The whole sentence considers as the setting window, rather than testing the setting window size for each sub-sampled word consistently among one and the length of the current sentence.

3.2.4 Paragraph vector Technique

Sometimes referred to as doc2vec or paragraph to vector representations, the method presented in [16] is mostly the first attempt to generalize word2vec to work with word sequences. The authors introduce two variants of the paragraph vectors model: Distributed Memory and Distributed Bag-of-Words.

The distributed memory(DM) model uses the standard encoder-decoder model, which most embedding techniques are built on. DM arguments encoder-decoder by adding a memory vector to capture the topic of the paragraph is focused on, or the context from the input data. The training task here is quite the same as that of any continuous bag of word architecture. Expecting the context words are the other words, not the surrounding words in document or paragraph.

The second proposed variant of paragraph vectors, distributed bag-of-words, as from its name, this is parallel to word2vec's skip-gram architecture. The work of the classifier is to predict a single context word using only the paragraph vector. At each step of stochastic gradient descent, a text window is sampled, then a single random word is obtained from that window. Apart from this rest of the architecture and training are similar, except that word vectors are not jointly learned along with paragraph vectors, which makes both memory and run-time performance of the distributed bag of word architecture better.

3.2.5 Document vector (Doc2Vec)

Unline the paragraph vectors, Doc2Vec tries to use both averaging the words and directly send the context of the words to the classifier. Doc2Vec represents each document as an average of the embeddings of words randomly sampled from the text. The fusion layer between the document vector and the word vector tries to catch in the information required by a classifier to generate the next words' prediction in the sequence of the sentence.

3.2.6 Skip-Thought Vectors

The authors in [28] presented another early attempt to generalize word2vec. The research works mostly extend word2vec â especially the skip-gram architecture in an intuitive way. The base unit in their innovation is now sentencing, and an encoded sentence is used to predict the sentences around it. The vector representations are learned using an encoder-decoder model trained on the above phenomena. This architecture is built on the use of an RNN encoder with GRU activations and RNN decoders with a conditional GRU. Two different decoders are trained for the previous and next sentences since the input is a sentence, and predictions around are the following sentence.

The skip-thought encoder [28] uses a word embedding layer for the conversion of each input word of the sentence to its corresponding word embedding, effectively turning the input sentence of a document into a sequence of continuous word embeddings. This embedding layer has also been shared with both of the decoders.

3.2.7 Bidirectional Encoder Text Representations

Bidirectional encoder Text Representations(BERT)[11] is built of the transformer. Transformers are mainly used for an attention mechanism that tries to learn the contextual relations between words (or sub-words) in a text. The nature of a transformer includes two separate mechanisms, an encoder that takes and reads the text input and a decoder that, in general, produces a prediction for the task. Since BERT's goal will be likely to generate a language model, only the encoder mechanism is necessary.

Rather than directional models, which read the content information successively from left-to-right or right-to-left, the transformer encoder concentrates on the entire sequence of words. Along these lines, it is viewed as bidirectional; however, it would be more exact to state that it's non-directional. This trademark permits the model to become familiar with the setting of a word dependent on the entirety of its environmental factors (left and right of the word). Also proved to BERT is a sequential embedding technique since tracks of the sequence in terms.

3.2.8 Sentence BERT

While a vanilla BERT [11], also called the traditional BERT, can be used for encoding the sentences, the embeddings generated by it are not robust. The samples deemed similar by the model are often more lexically than semantically related. Small perturbations in input samples result in significant changes in predicted similarity. Hence a new concept is created to generate sentence embeddings using BERT, i.e., a Siamese network

with BERT designed under Sentence-BERT[13] also referred to as S-BERT. Sentence-BERT is a fine-tuned pre-trained BERT using the siamese and triplet system. It is obtained by adding a pooling layer to the BERT model's output to extract some semantical similarity comparison within a vector space that can be used to compare with cosine similarity function. While comparing the Bert with SBERT following things are the major drawbacks of BERT, which overcame the introduction of the SBERT. BERT uses cross-encoder networks that take two sentences as input for the transformer network and then predict a target value. BERT can achieve a state of the art performance on semantic textual similarity tasks, but to do this, both sentences must be passed through the full network. For example, in a corpus consisting of 10000 sentences, finding similar pairs of sentences requires about 50 million inference computations taking approximately 65 hours. Sentence-BERT impressively can provide the encodings for 10000 sentences in about 5 seconds. SBERT is a so-called twin network which allows it to process two sentences in the same way, simultaneously. These two twins are identical down to every parameter (their weight is tied), which allows us to think about this architecture as a single model used multiple times.

CHAPTER 4

PROPOSED FRAMEWORK

The Framework we proposed is a combination of the feature space, model network, and the class representative. The CRL is the signature of the Class that uses to make for the network do make decisions. CR symbols the Class and the unseen perspectives try to find combinations from the signatures to validate the data. The input to the learner network is rich in content; it focuses on the data that is more important in the entire data point and is carry free about the other content. Figure 3 is an example from a class data point(i.e., one email document from 20 Newsgroup dataset) and the colored words are the ones that the embedder tries to focus on to learn the content of a text document that is useful enough to represent the entire context. The learner network tries to degrade the information to the usability and prompts for the class representative learning.

4.1 Framework Architecture

Class Representative Learning (CRL) Framework consists of three main steps (as shown in Figure 4). The first step is Sentence Encoder (SE), which transforms given documents to feature matrix. The second step is Learner Network (LN), which learns better intermediate feature vectors by taking the document feature matrix from SE as the input and mapping them to seen classes S_c. The third step is Class Representative Generation (CRG), which takes both the seen S_c and unseen U_c classes through the pre-trained SE

From: mathew |mathew@mantis.co.uk|
Subject: Re: STRONG | weak Atheism
Organization: Mantis Consultants, Cambridge. UK.
X-Newsreader: rusnews v1.02
Lines: 9

acooper@mac.cc.macalstr.edu (Turin Turambar, ME Department of Utter Misery) writes:
| Did that FAQ ever got modified to re-define strong atheists as not those who
| assert the nonexistence of God, but as those who assert that they BELIEVE in
| the nonexistence of God?

In a word, yes.

mathew

Figure 3: Example of CRL based Classification

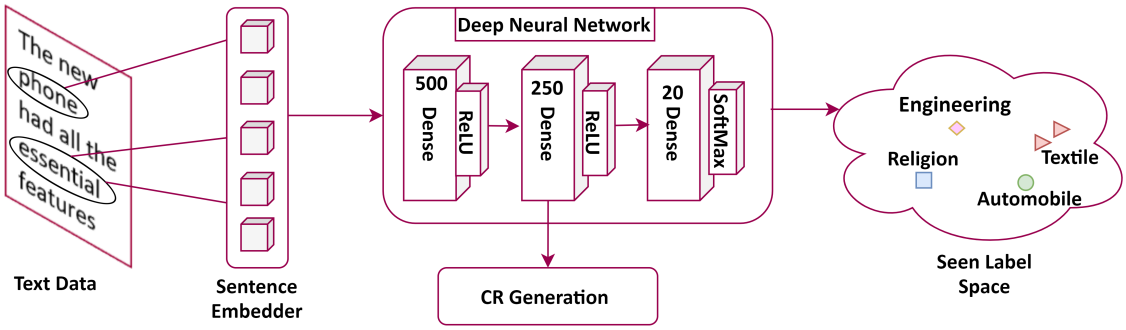

Figure 4: Class Representative - Text Classification

and LN and extracts an intermediate feature vector to create Class Representatives CR.

$$\begin{aligned} E_k^i &= SE(T_i) \\ CR_i &= LN(E_k^i)y^* \end{aligned} \tag{4.1}$$

The equation 4.1 describes the method of generating the class representatives for each class from using the information from sentence encoder and learner network, where E_k^i denotes the embedding matrix obtained from T documents processed by sentence encoder $SE(T_i)$. The CR_i consists of seen and unseen class information $S_c \cup U_c$ retrieved from the learner network $LN(E_k^i)y^*$.

Notation	Description
T	n Input Documents
S_c	Seen Classes
U_c	Unseen Classes
E_k	Document Embedding Matrix with k dimensions
$Sf(.)$	Sentence Encoder
$Lf(.)$	Learner network
CR	class representatives

Table 2: Description of mathematical notations used

4.1.1 Sentence Encoder

Sentence Encoder (SE) is the first step of our framework, which focuses on converting sentences to encoded vectors. We consider a set of n text documents $T = T^1, T^2, .., T^n$ as an input set, and convert the text document T into Document Embedded Matrix E_k Equation 4.2 shows the transformation of each text T^i, where $i \in n$ to

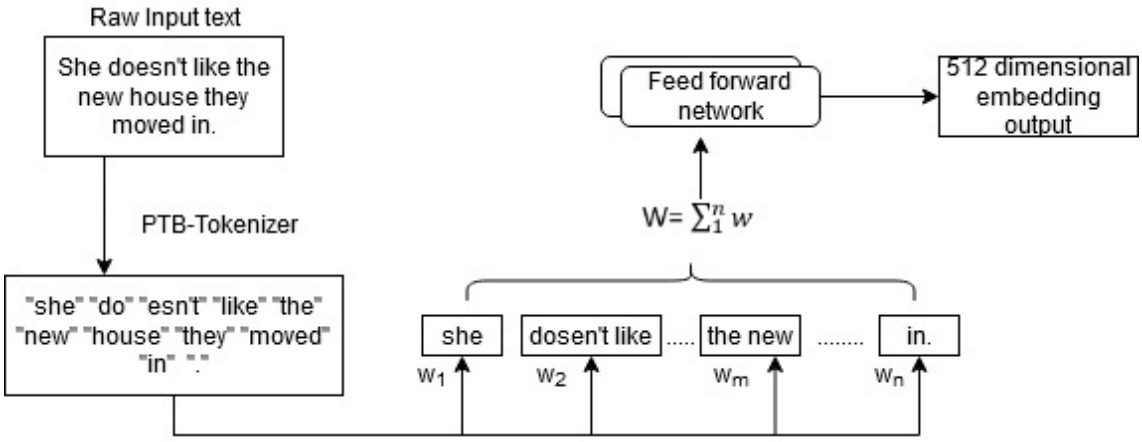

Figure 5: Universal sentence encoder based on DAN

Document Embedded Vector E_k^i through Sentence Encoder Function $Sf(.)$

$$E_k^i = SE(T^i) \tag{4.2}$$

The Document Embedded Matrix E_k consists of n Document Embedded Vector E_k^i, where $i \in n$ with the corresponding class label y^i where $y^i \in S$. The dimension of the Document Embedded Matrix k is dependent on which a sentence encoder is used for transformation. In this paper, we have carefully considered one of the existing framework, Universal Sentence Encoder (USE), as the sentence encoder. USE encodes text into high-dimensional vectors, where the model is trained and optimized for sentences, phrases, and short paragraphs. [12] has introduced two mechanisms for USE, one is with using the transformer[33] and other is with Deep Averaging network [30]. The USE's Deep Averaging Network model is used as part of our CRL Model.

Deep Averaging Network averages the input embedding of the words and bi-grams together, and then the average embedding is passed through a feed-forward network [30].

4.1.2 Learner Network

Learner Network (LN) can be on any neural network model aiming to build a ZSL model with the seen classes S for predicting class labels of the documents for unseen classes U. In this paper, we introduce a Three-Layer Deep Neural Network(DNN) for the learner network, as shown in Figure 4. The DNN consists of two dense layers coupled with the rectified linear unit (ReLU) activation function and the final layer for the seen classes.

Two dense layers are based on the non-linear activation function. Equation 4.3 shows the non-linear mapping function $DL(.)$, which incorporates both the dense layers. The ReLU activation function was implemented element-wise over the each feature vector D_k^i in Document Feature Matrix D_k. Each element in D_k is represented as x in Equation 4.3. The weights and biases of an dense layer 1 & 2 are represented as (W_1, b_1) & (W_2, b_2), respectively.

$$DL(D_k^i) = W_2^T \max\{0, W_1^T x + b_1\} + b_2 \tag{4.3}$$

The final layer is a multi-class probabilistic classifier that produces a S-dimensional vector of probabilities p for each feature vector D_k^i from Document Feature Matrix D_k, where $i \in n$, as shown in Equation 4.4.

$$p(D_k^i) = SoftMax(DL(.)) \tag{4.4}$$

The layer of SoftMax is calculated for the seen classes S. During the Learner Network training, the model is building based on the seen classes' data. The probability calculated in Equation 4.4 is used just to facilitate the binary cross-entropy loss function.

Unlike the general neural network model, in the CRL framework, the CRs play class-base discriminant roles in the final classification instead of the Softmax regression.

As shown in Equation 4.5, we aim at learning the non-linear mapping $DL(.)$, i.e., obtaining network weights W_1 and W_2 using Binary Cross-Entropy Loss. Binary Cross-Entropy Loss ($L(.)$) sets up a binary classification problem between $C' = 2$ classes for every class in Seen Class Set S. Equation 4.5 shows the minimization function across the weights W_1 and W_2 (shown as W) with the Loss function, which takes each Document Feature Vector Dk^i with the corresponding label y^i as the input. (Note: y_e^i is the one-hot encoded version of the label.)

$$\begin{aligned} \min_{W} \sum_{i=1}^{n} L(D_k^i, y_e^i) \\ L(D_k^i, y_e^i) = -\sum_{i=1}^{C'=2} y_e^i log(p(D_k^i)) \end{aligned} \tag{4.5}$$

4.2 CR Generation

Class Representatives (CRs) are generated using the nearest prototype strategy by aggregating feature vectors and is independent of the Learner Network. The closest mean feature vector with instances of the given Class (i.e., CRs) is computed Class by Class. To generate CRs, an average mean operation with feature maps was used to summarize the classes' instances. For each Class, the instances of features in the feature maps are aggregated into an abstract mean feature. The class representative(CR) is an aggregated vector of the mean features from all the elements in the feature maps. The feature maps are generated from a dense layer of the DNN network. The Class Representative is generated

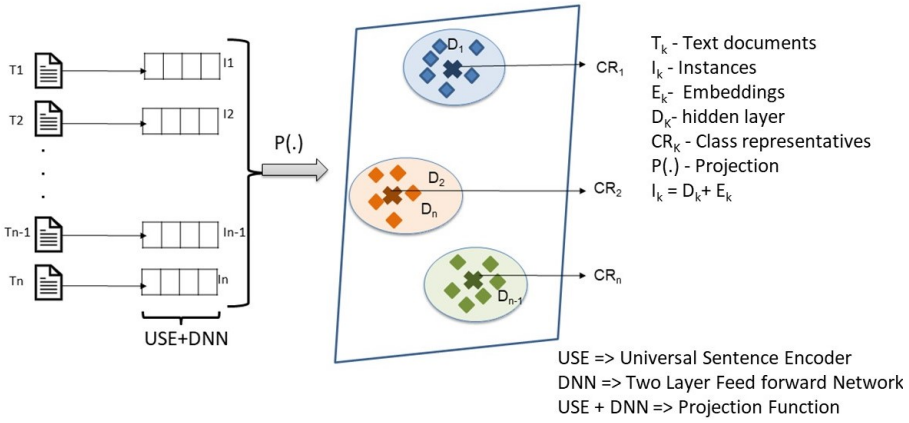

Figure 6: Workflow of class representative generation

for both Seen S and Unseen U classes using the pre-trained Sentence Encoder (SE) and pre-trained Learner Network (LN) from the previous steps. In this stage, we consider the non-linear mapping $DL(.)$ from Equation 4.3 as the intermediate feature layer.

More specifically, we computed an aggregative measure (i.e., the mean of the orientation of the vectors) using the high dimensional feature maps (e.g., 12K for a CNN layer) using the cosine similarity measure.

For the CR Generation, we considered the transformed Target Dataset $\hat{D}_t$ as the input. As we emphasis on the parallelism, we considered the individual activation vector $\hat{a}(x^{(i)})$ such that $y^i = c$, that will be used in formulating the CR as shown in Equation 4.6.

$$\begin{aligned} CR(c) &= \{CR^1, CR^2, ..., CR^l\} \\ CR^j &= \frac{1}{N_c} \sum_{j=1}^{l} DL(D_k^i) \end{aligned} \tag{4.6}$$

where j ranges from 1 to l representing the feature dimensions, c is the class of the input

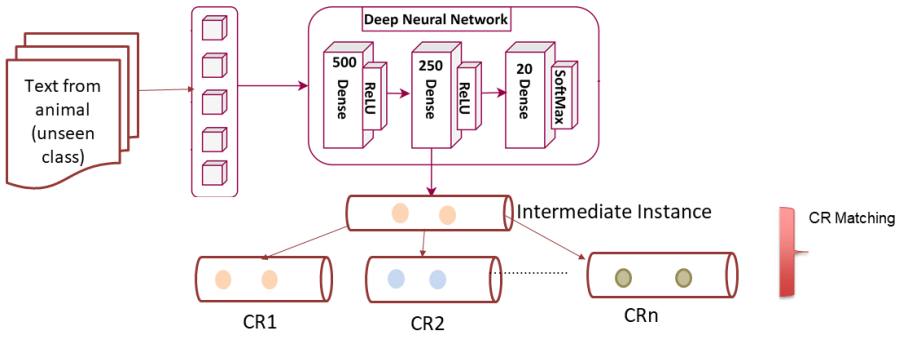

Figure 7: Architecture of class representative inferencing with Unseen class data

text and $c \in S \cup U$, and N_c is the number of data points for the class c. Class Representative of the given class c is represented as the group of CR features values CR^j where j ranges from 1 to l feature dimensions. CR^j is generated from the mean of $DL(.)$ with pre-trained weights of every input document D_k^i in a given class c as shown in Equation 4.6. Each class c in Seen and Unseen Classes ($S \cup U$) has a CR generated at this stage.

4.3 CR Inferencing

The CR-based inferencing is matching the input data into the Class Representatives (CRs) and classifying with the best matched CR to the input.The CR-based inferencing is parallel since the CRs are independent of each other.

$$\cos(CR(c), NI) = \frac{CR(c) \cdot NI}{\|CR(c)\| \, \|NI\|} \tag{4.7}$$

Steps for the CR-based inferencing. The input is vectorized using Equation 4.3: $NI = DL(D_k^i)$. The cosine similarity between the new text document (NI) and Class Representatives for class c ($CR(c)$), where $c \in C$ is computed using Equation 4.7. The

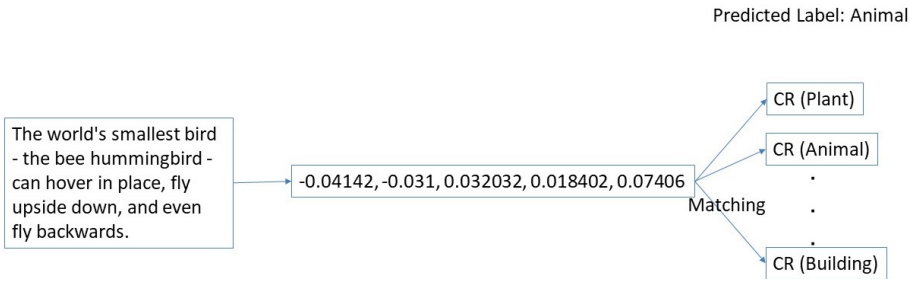

Figure 8: CR based inferencing example

CRL generated Model assigns the new inferencing input with the label associated with Class C that has the greater cosine similarity value. Higher the cosine similarity score, closer the Class Representative $CR(c)$ and the new input (NI) in the Class Representative Feature Space (CRFS).

$$\hat{c} =_{c \in S | c \in T} \{\cos(CR(c), NI)\} \tag{4.8}$$

As shown in Equation 4.8, the label for the input from CRL Model $\hat{c}$ is classified by selecting the Class from all seen (source) classes S or all unseen (target) classes T that has the highest cosine similarity to the new input. The CRL model will conduct inferring by matching the new input against all the extracted CRs and label them with a class having the highest cosine similarity score.

In figure 8 is an example of the CR based inferencing for an unseen class data point.

CHAPTER 5

RESULTS AND EVALUATIONS

5.1 Introduction

In this section, we discuss the results and evaluation of the proposed framework. First, we describe the results of text classification with CRL setting using different embedding techniques. Second, we show the accuracy results of zero-shot learning with CRL on three benchmark datasets.

5.2 Data Preparation

Table 3 gives in an excellent idea about all the datasets which are used to evaluate all different experiments. The more detailed information about datasets is mentioned below.

5.2.1 IMDB Dataset

It is a large dataset representing a binary (positive or negative) sentiment for each movie review. The IMDB dataset [34] sentiment analysis dataset consists of 100,000 movie reviews taken from the IMDB large movie rating and review site. One main aspect of this dataset is that each movie review has several sentences. The IMDB dataset [35] contains lowercase English reviews. The reviews are originally released in 2002, but the cleaned and refined version was released in 2004. Author of the dataset named it Polarity dataset. Dataset [35] contains movie reviews in two folders, one for negative

Table 3: Statistics of Dataset used in Evaluation

Dataset	Training Samples	Testing Samples	No. of Classes	Experiments
IMDB	25000	25000	2	CRL
20 Newsgroup	11314	7352	20	ZSL, CRL
DBpedia	15410	9233	14	ZSL, CRL

reviews and positive reviews. The dataset provides 100,000 movie reviews, divided into 25,000 reviews for labeled training and testing, and 50,000 unlabeled instances. There are two types of labels: Positive and Negative, and these labels are balanced in both the training and the test sets. The dataset can be downloaded at **Maas2011**. This data is already preprocessed with NLP techniques, including lemmatization and stemming, segmentation, stop-word removal.

Table 4: IMDB dataset sample documents and class labels

Movie Review	Label
An idiotic dentist finds out that his wife has been unfaithful. So, no new storylines here. However, the authors managed to create a stupid, disgusting film.	Negative
Wesker has to be the best character in the RE series, in my opinion. The story amazed me and took many different twists that I wasn't expecting. The only rating this game deserves is great.	Positive

5.2.2 20 Newsgroup Dataset

It is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. These 20 different newsgroups are organized according to their topic so that each group corresponds to a specific topic. Some of them are very closely related to each other, for example, comp.sys.ibm.pc.hardware,

comp.sys.mac.hardware, while others are highly unrelated, for example, misc.forsale, soc.religion.christian. The dataset is divided into 11,314 training samples and 7,352 testing samples.

5.2.3 DBpedia Dataset

The DBPedia dataset contains hierarchical classes representing the structured information from Wikipedia [35]. The DBPedia dataset is constructed by picking 14 non-overlapping classes[36] from DBPedia 2014. From each of these 14 classes, the fields we used for this dataset contain the title and abstract of each Wikipedia article.

5.3 Experiments and Evaluation

5.3.1 CRL Experiments for Classification

We have conducted experiments with four widely used embeddings in Table 5, to know which embedding techniques are useful in building competent Class Representatives (CRs). In this experiment, we have used both sentence-level embedding (e.g., USE) and word-level embedding (e.g., GloVe [10], Word2Vec [9], NNLM) techniques to see how the model might have to perform in the novel method for class representations. As seen from Table 5, USE outperforms the word embedding techniques GloVe or Word2Vec for the Class Representation Learning.

For this experiment (as shown in Table 6), the IMDB dataset was used as the base model. The two benchmark datasets, 20Newsgroup, and IMDB, are used for training and testing. Class Representatives (CRs) were generated for positive and negative categories

Table 5: CRL Classification Testing Accuracy

Model	20 Newsgroup	IMDB
Word2Vec +CRL	64.9%	54.2%
NNLM+ CRL	68.6%	58.6%
USE + CRL	79.3%	65.0%
USE+ DNN+ CRL	76%	64.2%

Table 6: CRL Classification Setting

Dataset	Class	Base	Embedding
20NG	20	IMDB	USE, NNLM,
IMDB	2		Word2Vec

for the IMDB movie reviews.

In this experiment, the four kinds of the CRL model have been built and evaluated.

Word2Vec+CRL: Word2Vec [9] computes high dimensional word vectors from a very large corpus. High-quality word vectors were trained using a model architectures with the CBOW and Skip-gram models. This model has been efficiently encoded in embedding space to improve the semantic and syntactic generalizations by increasing the volume of training data.

NNLM+CRL: A model architecture, called neural network language model (NNLM) [37], was designed to learn both the word vector representation and a statistical language model. The NNLM architecture is composed of a feed-forward neural network with a linear projection layer and a non-linear hidden layer.

GloVe+CRL: GloVe [10] is a unsupervised learning approach for word representations in a large corpus, rather than representing the entire sparse matrix or individual context windows. GloVe leverages a word-word co-occurrence matrix using a global log-bilinear

regression model and outperforms other models on word similarity and named entity detection. The CRL has conducted with CRs generated from the GloVe embedding. In our experiment, the classification with GloVe+CRL performed better than Word2Vec+CRL.

USE+CRL: Universal Sentence Encoder [12] provided sentence level embeddings for transfer learning in a deep averaging network for NLP tasks. USE showed a better performance in transfer learning with NLP tasks compared to word-level embeddings alone. In the text classification, the CRL with USE embeddings showed the best performance among the four different CRL models.

USE+DNN+CRL: For this experiment, we have also built a Deep neural network (DNN) that is composed of two hidden units with dimensions as [500, 100] along with the input and output layer. In this architecture (USE+DNN+CRL), (1) words classes were embedded using USE, (2) they were learned in the DNN network for the conditional probability of levels and words, (3) CR feature vectors were generated with the features extracted from DNN, and (4) CRL was used to estimate the class-based model. For the DNN training, 1000 epochs, i.e., 128,000 iterations with a batch size of 5 with 25000 training samples.

5.3.2 CRL Experiment for Zero-Shot Learning

The second set of experiments focuses on applying the class representation technique for Zero-shot learning. For these experiments, we divide our dataset into seen and unseen classes. Two different rates of unseen classes, 50%, and 25%, were chosen and the corresponding sizes of a set of the seen classes and unseen classes are shown in Table 7.

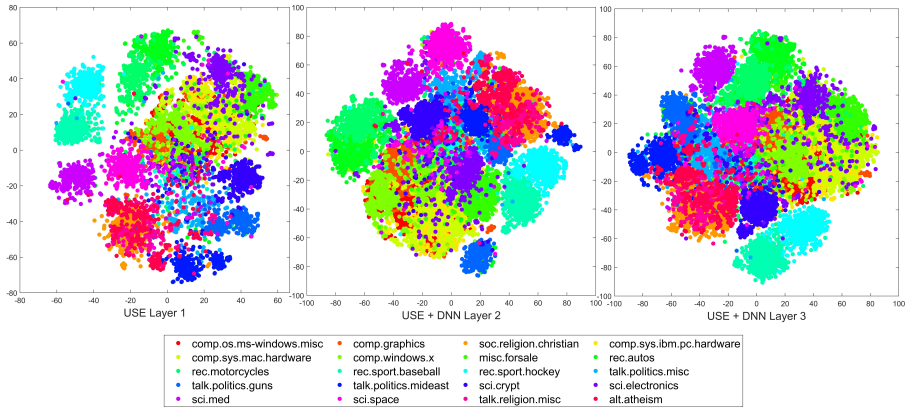

Figure 9: T-SNE Visualization for 20 Newsgroup Instances (USE+DNN+CRL)

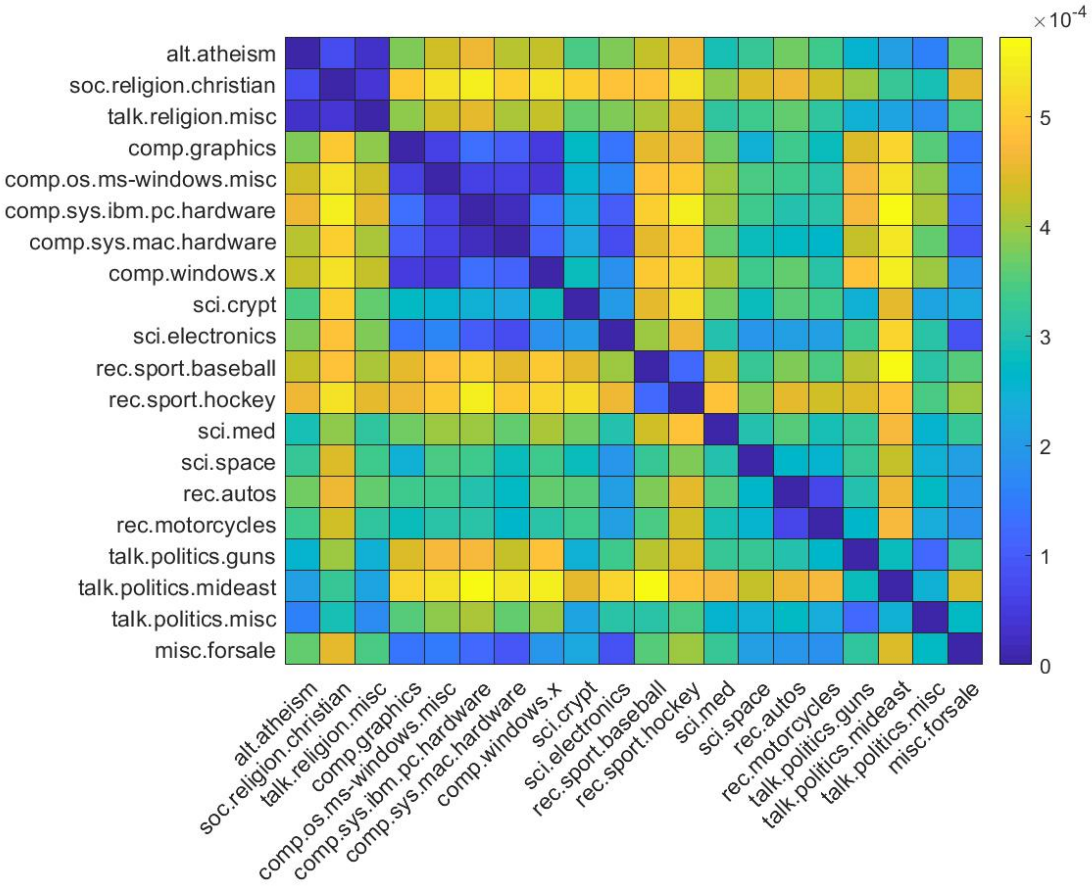

Figure 10: Heatmap CR Visualization for 20 Newsgroup (USE+DNN+CRL)

Table 7: Dataset for Zero-Shot Learning Evaluation

Dataset	Unseen Split	#Seen	#Unseen
20NG	25%	15	5
	50%	10	10
DBPedia	25%	11	3
	50%	7	7

Table 8: CNN Architecture

Layer	L1	L2	L3	L4	L5	L6
Name	Conv1D	Conv1D	Conv1D	Conv1D	Conv1D	Conv1D
Activation Shape	499 x 400	497 x 400	495 x 400	491 x 200	94 x 80	23 x 32
Activation Size	120400	160400	240400	400200	160160	8224

5.3.2.1 Comparison based on Learner Network

DNN Learner Network with Universal Sentence Encoder, as explained in Section 4.1.2. For CNN Learner Network, we use 5-Layer Convolution Network with GloVe based sentence encoder as the input. Table 8 and Table 9 show Layer-wise details of CNN and DNN Learner Network, respectively.

Figure 11 shows the layer-wise accuracy of the corresponding layer shown in Table 8 9. The evaluation is based on DBPedia and 20-NewsGroup Datasets. For evaluating CRL in Zero-Shot Learning Setting, we use Convolution Neural Network and Deep Neural Network as Learner Network to validate.

USE+DNN+CRL: This model is what we proposed for Zero-Shot learning in this paper. The framework of USE+DNN+CRL, as described in Section 4.1.2. The layer-wise details of USE+DNN+CRL are shown in Table 9.

Table 9: DNN Architecture

Layer	L1	L2	L3
Name	Embedding	Dense	Dense
Activation Shape	512	500	250
Activation Method	N/A	ReLU	ReLU
Activation Size	N/A	262656	134572

We have developed the CRL architecture with the three phases for Zero-Shot Learning (ZSL), as shown in Figure 4. First, we have applied Universal Sentence Encoder (USE), i.e., to extract features from our datasets. Second, after the embedding is generated, deep learning has been implemented with the embedding. As per our knowledge, there is no proper classification pre-trained model for text data. Thus, all the deep neural networks we presented in our experiments have been implemented using Keras [38] with the TensorFlow framework [39] as a back-end. Using the deep neural network models as a base environment, we have built class representatives (CRs) with the feature vectors using the source data for Zero-Shot Learning (ZSL). Our experiments in Table 10 show that the effectiveness of the transfer learning model, the pre-trained model of the source domain, has successfully applied to the target domain.

Figure 9(a) shows the t-SNE visualization of the 20Newsgroup dataset for the first, second, third layers of the CRL-ZSL architecture (USE+DNN+CRL) architecture, respectively. TSNE Visualization shows the clarity obtained through instance grouping, which corresponds to the Accuracy shown for each layer in Figure 11.

Figure 9(b) shows the similarity between Class Representatives are computed each other using cosine similarity for the 20Newsgroup dataset using a heatmap. The darker

blue group captured in the heatmap represents a similar CRs with a high similarity score.

GloVe+CNN+CRL:

GloVe [10] is an unsupervised learning approach for word representations in a large corpus, rather than representing the entire sparse matrix or individual context windows. GloVe leverages a word-word co-occurrence matrix using a global log-bilinear regression model and outperforms other models on word similarity and named entity detection. The CRL has conducted with CRs generated from the model architecture of GloVe embedding and Convolution Neural Network (CNN). The Convolution Neural Network (CNN) is of 6-Layers. The first convolution is built on GloVe. We have used Glove embedding here since the sentence level embedding doesn't support convolution. A pooling layer follows every two convolutional 1D layers, this model different pooling techniques have been used, and we tried different hyperparameters for the CNN model. Convolution expects to have a 3-dimensional embedding shape rather 2-dimensional for sentence level. In our experiment, the classification with GloVe+CRL performed better than Word2Vec+CRL. As shown in Figure 11, USE+DNN+CRL model clears out-performs GloVe+CNN+CRL Model on both the datasets. Interestingly, in Classification Accuracy we noted that USE+CRL performs better USE+DNN+CRL (See Table 5). Whereas in Zero-Shot Learning USE+CRL i.e. Layer 1 in USE+DNN+CRL Model has atleast 5-10% less accuracy compared than other layers on both the datasets.

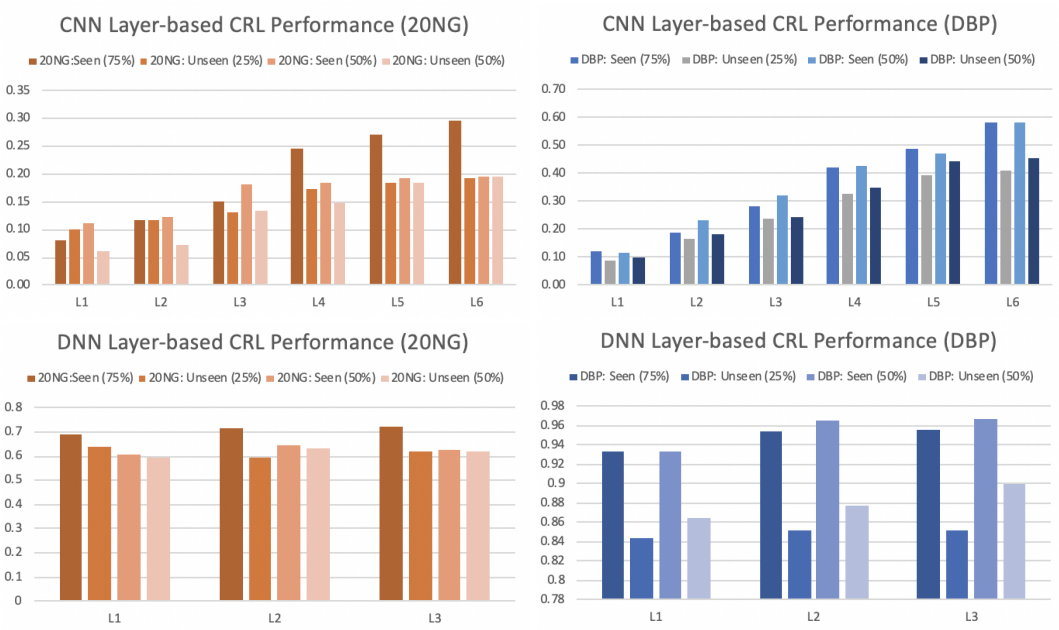

Figure 11: CNN/DNN Layer-based CRL Performance for ZSL with 20NG/DBP Datasets

5.3.2.2 Comparison with SOTA ZSL Model

GloVe+RNN+CRL: This Recurrent Neural Network (RNN) model is trained on GloVe and is a combination of convolution and Bidirectional RNN layers. This model is more potent since it is trained back and forth. The first dimension of the RNN is 128, which at the final, is reduced to 32.

RNN AutoEncoder: RNN AutoEncoder was built based on a Seq2Seq model with LSTM (512 hidden units), and is trained to encode documents and class labels on same latent space. The cosine similarity was applied to predict a class label closest to the text on the latent space.

USE+DNN+CRL: Unlike CNN or RNN, we trained the Dense Neural Network network with Universal sentence embedding. The DNN network is a deep neural network with

Table 10: Comparison of Zero-Shot Learning Models

Dataset	Split(%)	Category	Label Similarity [40]	RNN+FC [17]	CNN+FC [20]	CNN+ZSL [20]	CRL+ZSL (Ours)
20NG	75:25	Seen	0.279	0.614	**0.792**	0.745	0.713
		Unseen	0.287	0.065	0.134	0.280	**0.622**
		Overall	0.280	0.482	0.633	0.633	**0.649**
	50:50	Seen	0.293	0.709	0.684	**0.767**	0.646
		Unseen	0.266	0.052	0.126	0.168	**0.629**
		Overall	0.280	0.381	0.405	0.469	**0.616**
DBPedia	75:25	Seen	0.377	0.895	**0.985**	0.975	0.954
		Unseen	0.426	0.046	0.204	0.402	**0.852**
		Overall	0.386	0.713	0.818	0.852	**0.900**
	50:50	Seen	0.401	0.960	**0.991**	0.982	0.944
		Unseen	0.369	0.044	0.069	0.197	**0.878**
		Overall	0.386	0.052	0.530	0.590	**0.929**

three layers. The USE encoder is defined in 512 dimensions; these embeddings are further reduced into smaller dimensions as specified in Table 9.

Label Similarity: [40] The label similarity model was previously introduced for predicting unseen documents with labels using a semantic similarity between the label and the corpus [40]. For the class prediction, the cosine similarity measure was used to compute the similarity between class-based word embeddings and N-gram based word embeddings. The multi-label ZSL model [40] was revised to a single-label ZSL model and the revised model was used in comparative evaluation in [20].

They used semantic embedding of labels and document words, predicting previously unseen labels based on the similarity between the label name and the document words in this embedding.

Table 11: CRL-ZSL: CRL Models for Zero-Shot Learning

Dataset	Split	Category	GloVe+RNN+CRL	GloVe+CNN+CRL	USE+DNN+CRL
20NG	75:25	Seen	0.321	0.297	**0.713**
		Unseen	0.181	0.192	**0.622**
		Overall	0.236	0.233	**0.649**
	50:50	Seen	0.234	0.196	**0.646**
		Unseen	0.125	0.195	**0.629**
		Overall	0.162	0.195	**0.616**
DBPedia	75:25	Seen	0.719	0.582	**0.954**
		Unseen	0.601	0.412	**0.852**
		Overall	0.654	0.482	**0.900**
	50:50	Seen	0.762	0.582	**0.944**
		Unseen	0.631	0.456	**0.878**
		Overall	0.692	0.511	**0.929**

RNN+FC: [17] RNN+FC predicts unseen sentences by learning the relationship between the sentences and embedding of their tags using an RNN layer with LSTM [17]. The LSTM network is designed 512 hidden units together with two dense layers, having 400 and 100 units, respectively. This model is generalized to predict if a given sentence is related to a tag or not, rather than classifying the sentence with a label. Training model on a large group of sentences to learn the relationship between a sentence and embedding of sentence's tags. Learning such a relationship makes the model generalize to unseen sentences, tags, and even new datasets provided they can be put into the same embedding space. The model learns to predict whether a given sentence is related to a tag or not, unlike other classifiers that learn to classify it as one of the possible classes. It uses an RNN layer with LSTM(512 hidden units) followed by two dense layers with 400 and 100 units. Though this model training rate is high, the learning is very low, and also the seen Accuracy in Table 10 is small when compared with that of RNN+FC [17] and high unseen

Dataset	20NG		DBPedia	
	50:50	75:25	50:50	75:25
Acc_S	0.603	0.671	0.859	0.907
Acc_U	0.346	0.455	0.724	0.765
HM	0.439	0.544	0.785	0.829

Table 12: Generalized Zero-Shot Learning

rate.

CNN+FC: CNN+FC was revised in [20] by replacing the LSTM in the LSTM+FC model with a CNN for building the zero-shot classifier.The CNN+FC in the table has shown increase in the accuracy for unseen rate and the training accuracy is 98%. Table 8 implies layer-wise accuracy information for CNN+CRL. As the activation shape decreases the learning rate is seen to increasing. We choose the best layer class representatives for testing.

CNN+ZSL: [20] CNN+ZSL architecture has low learning rate for CR-based approach. We observed that CNN architecture 8 has a good learning rate. We have used the same architecture to train the model on seen, generated CR on top of it to see how they understand the unseen data. CNN+ZSL is a CNN-based two-phase framework using data augmentation and feature augmentation [20]. For ZSL, semantic knowledge, including word embeddings, class hierarchy, class descriptions, and a knowledge graph, are incorporated into the proposed framework.

As shown in Table 10, for comparison between SOTA Models, we report Seen Accuracy, i.e. ($S \rightarrow S$), Unseen Accuracy, i.e. ($U \rightarrow U$) and Overall Accuracy. The Unseen Accuracy of USE+DNN+CRL clearly outperforms with a minimum of 40% increase comparing to any other SOTA Model. The Seen Accuracy still lacks, especially

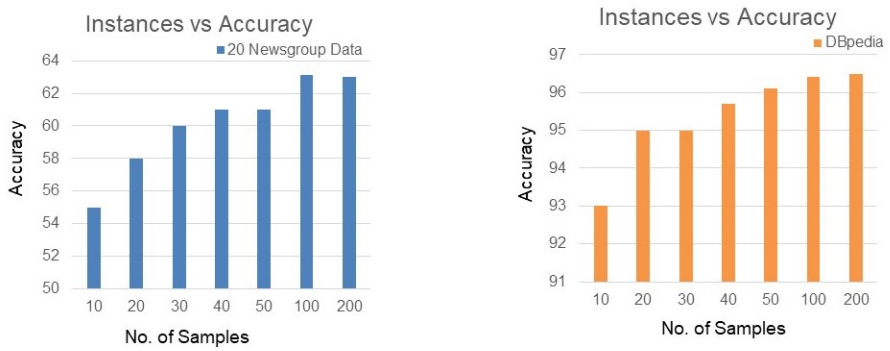

Figure 12: Increasing instances for CR generation and change in accuracy

comparing to CNN+FC and CNN+ZSL models, but with only a maximum of 7% accuracy drop.

DNN+GZSL: We have performed generalized zero-shot learning 12 evaluation along with the zero-shot learning. GZSL assessment is quite different from that of the ZSL, i.e., the seen Accuracy is generated by comparing the seen test data S to seen(S) and unseen (U) ($S \rightarrow S \cup U$). In contrast, the unseen Accuracy is generated by matching the CR produced to ($U \rightarrow S \cup U$). In our case, the ZSL had a rise in the Accuracy of 2% compared to GZSL.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this thesis, we proposed a novel class representative learning (CRL) framework for ZSL-based text classification using sentence-level word embeddings and deep neural network features. The CRL framework's performance has been evaluated with various embedding techniques such as Universal Sentence Encoder, Glove, Word2Vec for text classification. The CRL framework has been evaluated for Zero-Shot Learning and Generalized Zero-Shot Learning problems that aims for effective transferring knowledge from seen to unseen classes. We achieved the highest overall accuracy from the experiments on three benchmark datasets across various domains compared with the state-of-the-art Zero-Shot Learning and Generalized Zero-Short Learning algorithms.

6.2 Future Work

We are planning to extend our CRL framework to perform Zero-Shot Learning for multi-modal classification with a more significant amount of mixed text and image data.

Bibliography

[1] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 453–465, 2013.

[2] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 7, pp. 1425–1438, 2015.

[3] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean, "Zero-shot learning by convex combination of semantic embeddings," *arXiv preprint arXiv:1312.5650*, 2013.

[4] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning a comprehensive evaluation of the good, the bad and the ugly," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2251–2265, 2018.

[5] G. E. Hinton *et al.*, "Learning distributed representations of concepts," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, MA, vol. 1, 1986, p. 12.

[6] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: A statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 43–52, 2010.

[7] 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.

[8] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext.zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[10] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

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

[12] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.

[13] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[14] C. S. Perone, R. Silveira, and T. S. Paula, "Evaluation of sentence embeddings in downstream and linguistic probing tasks," *arXiv preprint arXiv:1806.06259*, 2018.

[15] M. Ikonomakis, S. Kotsiantis, and V. Tampakas, "Text classification using machine learning techniques.," *WSEAS Transactions on Computers*, vol. 4, no. 8, pp. 966–974, 2005.

[16] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning*, 2014, pp. 1188–1196.

[17] P. K. Pushp and M. M. Srivastava, "Train once, test anywhere: Zero-shot learning for text classification," *arXiv preprint arXiv:1712.05972*, 2017.

[18] D. Yogatama, C. Dyer, W. Ling, and P. Blunsom, "Generative and discriminative text classification with recurrent neural networks," *arXiv preprint arXiv:1703.01898*, 2017.

[19] A. Rios and R. Kavuluru, "Few-shot and zero-shot multi-label learning for structured label spaces," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, NIH Public Access, vol. 2018, 2018, p. 3132.

[20] J. Zhang, P. Lertvittayakumjorn, and Y. Guo, "Integrating semantic knowledge to tackle zero-shot text classification," *arXiv preprint arXiv:1903.12626*, 2019.

[21] 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.

[22] Y. N. Dauphin, G. Tur, D. Hakkani-Tur, and L. Heck, "Zero-shot learning for semantic utterance classification," *arXiv preprint arXiv:1401.0509*, 2013.

[23] C. Song, S. Zhang, N. Sadoughi, P. Xie, and E. Xing, "Generalized zero-shot icd coding," *arXiv preprint arXiv:1909.13154*, 2019.

[24] J. Nam, E. L. Mencía, and J. Fürnkranz, "All-in text: Learning document, label, and word representations jointly," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[25] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[26] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, 2014, pp. 3104–3112.

[27] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "Towards universal paraphrastic sentence embeddings," *arXiv preprint arXiv:1511.08198*, 2015.

[28] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in Neural Information Processing Systems*, 2015, pp. 3294–3302.

[29] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, 2017.

[30] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III, "Deep unordered composition rivals syntactic methods for text classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 1681–1691.

[31] T. Kenter, A. Borisov, and M. De Rijke, "Siamese cbow: Optimizing word embeddings for sentence representations," *arXiv preprint arXiv:1606.04640*, 2016.

[32] F. Hill, K. Cho, and A. Korhonen, "Learning distributed representations of sentences from unlabelled data," *arXiv preprint arXiv:1602.03483*, 2016.

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

[34] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 142–150.

[35] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, *et al.*, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[36] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[37] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[38] A. Gulli and S. Pal, *Deep learning with Keras*. Packt Publishing Ltd, 2017.

[39] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[40] S. P. Veeranna, J. Nam, E. L. Mencıa, and J. Fürnkranz, "Using semantic similarity for multi-label zero-shot classification of text documents," in *Proceeding of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges, Belgium: Elsevier*, 2016, pp. 423–428.

VITA

Sai Sri Narne completed her bachelors degree in Information Technology from RVR JC College of Engineering, Andhra Pradesh, India. She started her master s in computer science at the University of Missouri-Kansas City (UMKC) in August 2018, with an emphasis on Data Sciences and graduates in July 2020. While she was studying at UMKC, she has worked as a Graduate Teaching assistant for Advance Software Engineering, Knowledge Discovery Management, and Web Mobile Development Courses. Upon completion of her requirements for the masterâs Program, she plans to work as a Data Scientist/Machine Learning Engineer.