ZERO-SHOT COMPOSITIONAL EVENT DETECTION VIA GRAPH MODULAR NETWORK

A Dissertation
Presented to
The Academic Faculty

By

Yuchen Zhuang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2021

ZERO-SHOT COMPOSITIONAL EVENT DETECTION VIA GRAPH MODULAR NETWORK

Thesis committee:

Dr. Matthieu Bloch, Co-advisor
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Biing Hwang Juang
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Chao Zhang, Advisor
Computer Science and Engineering
Georgia Institute of Technology

Dr. Chin-Hui Lee
Electrical and Computer Engineering
Georgia Institute of 54esTechnology

Dr. Mark Davenport
Electrical and Computer Engineering
Georgia Institute of Technology

Date approved: April 28, 2021

To my great parents and those who always stay with me in my life journey.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Chao Zhang, who gave me the opportunity to conduct research and developed my interest in machine learning and natural language processing research through his guidance on my master thesis research. I would also like to thank my co-advisor, Dr. Matthieu Bloch, who encourages me and reminds me to keep forward in my research project. Many thanks for their kind guidance and offering computational resources. Moreover, I would like to thank Dr. Mark Davenport, Dr. Biing Hwang Juang, and Dr. Chin-Hui Lee, for their willingness to become my thesis committee members and spend their precious time reviewing this thesis. Besides, I want to thank my friends, Lingkai Kong, Yinghao Li, Rui Feng, and Junyang Zhang, for their help introducing me to the natural language processing and their collaboration on the project. I want to thank the professors and friends at Georgia Tech who have helped and taught me a lot during the past two years. I want to thank my girl friend, Ms. Wenqi Shi, who has supported me a lot to overcome the difficulties during my master student life. Finally, I want to thank my dear parents, Mr. Bin Zhuang and Mrs. Lei Zhu, without whose love and support I could not reach my achievements today.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Humans are known to have the capability of understanding events by composing different atomic concepts, even for event types that have never been seen before. However, event detection has been so far treated as a sequence tagging problem in literature. Despite the increasing accuracy obtained on benchmarks such as ACE, current supervised sequence tagging models lack the compositional generalization ability. We present a model that is able to achieve zero-shot compositional generalization for event detection. Our model, named *compositional graph modular network* (CGMN), proposes two separate graph neural networks to obtain compositional semantic representations for sentences and events respectively. Meanwhile, it ties graph-based event representations with the weight parameters of an event matching layer, so that the semantic representations for sentences and events can be connected with each other, thereby achieving zero-shot recognition of new events using only their constituent atomic concepts. Our experiments on the ACE 2005 dataset as well as our collected Twitter event dataset show that, CGMN significantly outperforms state-of-the-art event detection methods on unseen classes and demonstrate strong zero-shot compositional generalization capabilities.

CHAPTER 1
INTRODUCTION

Event detection (ED) is an important natural language understanding problem with applications including document summarization [1], knowledge base population [2, 3], and question answering [4]. To date, ED has been treated as a sequence tagging problem—tagging event trigger words in a sentence and classifying them into different event types. The sequence tagging model is typically learned from a corpus (eg., ACE [5]) that consists of human-annotated sentences. Earlier methods use traditional statistical sequential models and pre-defined linguistic features, while recent works focus more on deep sequential models due to their representation power and feature learning capabilities, which can be further enhanced by large-scale pre-trained language models like BERT [6] and RoBERTa [7].

Zero-shot learning (ZSL) has been a long-standing tough problem in machine learning. In ZSL, a model learned from training data can face test samples that come from never-seen-before classes. Even with *zero* training experience for such unseen classes, the model still needs to recognize the samples from them, by using only certain semantic descriptions of the unseen classes. To achieve ZSL, the model needs to understand the associations between classes and transfers knowledge from the seen classes to the unseen ones. Various approaches have been proposed for zero-shot generalization for computer vision tasks, by leveraging prior knowledge about classes such as visual attributes [8], word embeddings of the labels [9], class hierarchy [10], and external knowledge graph [11]. In text mining, although supervised text classification has been widely studied, zero-shot text classification has been little explored and remains a challenging problem.

However, to accomplish the zero-shot event detection, a key limitation of current sequence tagging models is their *lack of compositional generalization ability for event de-*

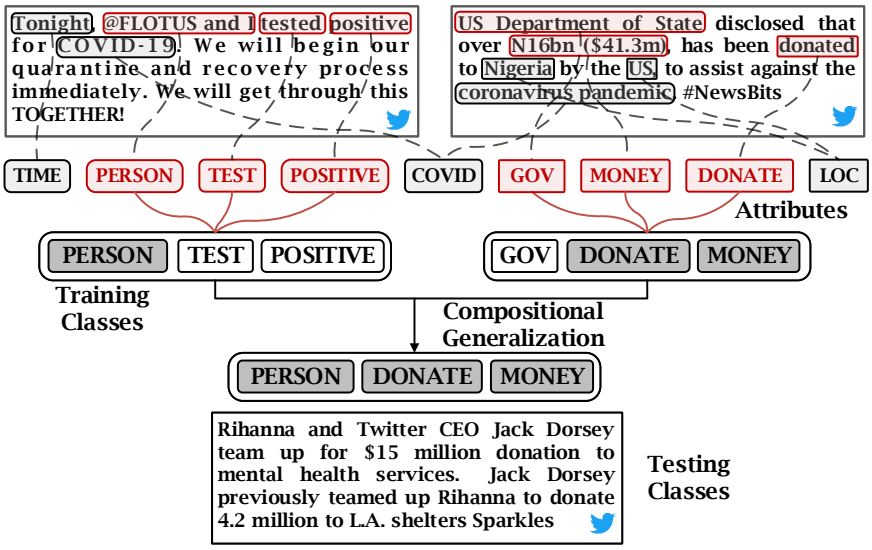

Figure 1.1: Given two categories PERSON TEST POSITIVE and GOVERNMENT DONATE MONEY, we can compose the attributes to understand new incoming class of PERSON DONATE MONEY during testing stage.

tection. Take the ongoing COVID-19 pandemic as an example. Every day, millions of tweets are created to discuss various events about this pandemic. When humans digest such tweets, we can easily recognize never-seen-before events types based on combinations of atomic concepts. Figure 1.1 shows a tweet describing an event about PER TEST POSITIVE. Even given a new event type we have never encountered before, taking PERSON DONATE MONEY as an example, we have no difficulty in understanding them because the constituent concepts (PERSON, DONATE, MONEY) may have well occurred in other event types we are familiar with, such as PERSON TEST POSITIVE and GOVERNMENT DONATE MONEY. Such compositional generalization is crucial for human's ability of quickly learning new concepts through the combination of atomic concepts, thus understanding an almost-infinite number of events from finite primitives. Unfortunately, this ability is missing in current sequence tagging models for ED because different event triggers are treated as independent symbols—the complex event semantics is treated as a whole label rather than the composition of several atomic concepts.

In this thesis, we propose a compositional graph modular network (CGMN) to enable

zero-shot compositional event detection. Unlike existing methods that treat event types as discrete labels, we model events using a graph of atomic concepts, which is constructed to model the relations among events and their primitives (including triggers and arguments). Such a atomic concept graph enables learning vector representations of events, so as to encode their compositional semantics. Specifically, CGMN uses a graph neural network (GNN) [12] over the concept graph to propagate information between events through their constituent atomic concepts. At inference time, the representation of a new event type can be obtained via GNN inference.

The event embeddings computed from the atomic concept graph will be tied to the weights of an event detector, thus enabling zero-shot detection of new events. The detector determines whether a sentence contains target events and is trained over seen events. It consists of: 1) a modular network that embeds atomic concepts into a latent semantic space and prepares for composition; 2) a modularized sentence encoder that obtains compositional semantic representations for sentences over the dependency parsing tree structure; and 3) a event matching layer using sentence and event embeddings for event detection. A key design for the event matching layer is to tie event embeddings (computed from the concept graph) to the classification layer's weights, so that the event embeddings act as weight parameters for combining sentence features and determining if the target event type is present. At inference time, the detection of any new event type can be naturally derived once its embedding has been computed.

To evaluate our model, we collected and annotated a COVID-19 event dataset from Twitter. Labels in our collected data are all compositional like exemplified in Figure 1.1. Our experimental results show that our model can improve accuracy from 39.40% to 68.44% on seen classes, and 18.88% to 25.90% on unseen classes. We also evaluated our method on the ACE 2005 benchmark under our compositional setting, and found that our method outperforms all the baseline methods, improving accuracy from 29.12% to 36.22% on seen classes and 18.73% to 25.94% on unseen classes.

Our main contributions include:

1. We exploit the compositionality of labels by the graph structure between labels and attributes for knowledge transfer, and we exploit the structure of sentences to embed both labels and sentences on the same semantic space;

2. We tie the label embeddings to classifier weights for efficient zero-shot learning;

3. We enrich the embeddings obtains by BERT with knowledge of atomic attributes and dependency structure for more efficient sentence-level information aggregation;

4. We conducted extensive experiments that evidence the effectivenss of our method compared with existing work. We also collect a new dataset named Twitter-COVID19 for evaluating compositional text classification.

CHAPTER 2
BACKGROUND

2.1 Related Works

2.1.1 Event Detection

Event detection is one of the important information extraction tasks and has been studied by the NLP community for years. Earlier approaches to event detection extracts linguistic features manually [13, 14, 15]. Later on, deep learning models have been recently dominating for event detection due to their better performances, including adapted versions of CNN [16, 17], RNN [18], and GNN [19, 20].

The dynamic multi-pooling convolutional neural networks (DMCNN) proposed in [16] and the event detection structure in [17] are both based on the convolutional neural network (CNN). Later, the joint recurrent neural networks in [18] introduce the recurrent neural networks (RNN) into the event extraction task. Furthermore, using an attention mechanism to model structured information has also been shown to benefit model performance. For example, [19] has proposed a supervised attention mechanism to encode argument information in event detection. Besides CNN and RNN, recent works [20] have shown that graph neural networks can be applied to extract the most relevant information of different entites in event detection as well.

2.1.2 Zero-shot Learning

Zero-shot learning (ZSL) is a challenging task testing models' ability of learning concepts about new unseen data with no corresponding labeled data. Most traditional works in Computer Vision aim to find implicit relations between categories [8], [21], [22], [23]. Besides, more complicated models like graphical convolutional network with semantic knowledge

graph [11] and compositional modular network [24]

Zero-shot text classification has also been studied by only a handful of works. Existing literature has learned relation information from deep neural network models based on a large amount of corpus data [25, 26]. [27, 28, 29] proposed models jointly learning information from the sentence and label semantic embeddings via methods like entailment learning. [30] take advantage of class labels, class descriptions, class hierarchy, and knowledge graph to thoroughly extract semantic information from the text data. The methods introduced above are more specific under circumstances that the labels have not much explicit relationship with the examples and will not test the model's compositional learning ability. To the best of our knowledge, our model is the first to realize compositional zero-shot learning in text classification.

2.1.3 Compositional Generalization

Zero-shot compositional generalization has been a trending topic in computer vision s [31, 32, 33, 34]. Some of the existing literature are based on embedding the object-attribute pair in image feature space [35, 36], while the others learn the joint compatibility between the features extracted from the images and the pairs defined as their labels [24, 11, 37].

The compositional generalization in natural language processing is defined in [38] as the intrinsic connection between the ability to produce and understand different sentences composed of the same component argument blocks, like *John loves Mary* and *Mary loves John*. The tasks or implementations details are different. A common trial is based on the SCAN task, which is a novel sequence modeling task, mapping word sequences to command sequences [39, 40]. Another common task is to compose different arguments representations to model compositional phenomena [41]. [42] also proposes an end-to-end decomposition and modular network with similar ideas tested both on image recognition and language modeling. However, although it is performing a compositional learning ability of the model, the modules are still meaningless while training and people can only

know how the model select and compose the modules with some post-training analysis to see their training paths, which we think is not that reasonable compared to the humans' compositional learning ability.

2.2 Related Techniques

2.2.1 Pre-trained Models

Pre-trained language models have recently brought the natural language processing (NLP) community into the transfer learning era. The transfer learning framework consists of two stages, where we first pre-train a large-scale language model (e.g., BERT [6], RoBERTa [7], ALBERT [43], T5 [44]) on a large text corpus in an unsupervised manner and then fine-tune it on downstream tasks.

In this thesis, we mainly utilize the BERT model architecture [6], which is based on a multi-layer bidirectional Transformer [45]. Instead of the traditional left-to-right language modeling objective, BERT is trained on two tasks: predicting randomly masked tokens and predicting whether two sentences follow each other.

2.2.2 Graph Neural Networks

Over the past few years, the success in neural networks have improved the researches in pattern recognition and the data mining. Many machine learning tasks that used to rely heavily on the manually extracted features, like object detection, machine translation, and speech recognition), have all been entirely changed by different kinds of end-to-end deep learning architectures, including Convolutional Neural Network (CNN) [46], Long Short Term Memory (LSTM) [47], and auto-encoder (AE) [48].

Although traditional deep learning methods have achieved great success in extracting features from Eucilidean Space data, many data in realistic application scenarios are generated from the non-Eucilidean space, and this makes the traditional deep learning methods perform not so well in handling these non-Eucilidean space data. Thus, recently, the re-

searchers become more interested in applying deep learning on graph. Inspired by this kind of thoughts, the Graph Neural Network (GNN) is generated, catering to people's needs.

Graph is a widely utilized data structure in algorithms. Many tasks and applications in real life can be described as or expressed by graph problems, like the social media [49, 50], protein architecture [51], transportation networks[52], the recent popular knowledge graph [53] and so on. As a unique non-Euclidean data structure, graph methods and analysis focus on the node classification, graph classification, edge prediction, and data clustering. With high validity in performance and very strong interpretability, GNN has widely aroused peoples' interest no matter from the academia and the industry. Graph Neural Network is to make full use of the graph data structure in neural networks to solve some graph-based data mining problems. Almost all the classic structures or models in natural language processing have the applications in graph neural network, like Graph Convolutional Network (GCN) [54], Graph Attention Networks [55], Graph Transformers Networks [56], Graph Recurrent Neural Network (RNN) [57]. Different GNN methods vary according to their different ways of building graphs, different information propagation methods, and different architectures. Although there exist many GNN methods, it is still a small proportion for GNN to be utilized to implement zero-shot classification problems. Only several methods combine GCN and knowledge graph to import label information into the model [11].

CHAPTER 3

METHODOLOGY

3.1 Problem Formulation

We formulate zero-shot compositional event detection as a sentence multi-label classification task. Given a sentence $\mathbf{x} = [x_1, \ldots, x_N]$, the entire sentence is associated with an one-hot event label vector $\mathbf{y}$. Different from existing works, we view events as compositions of *atomatic concepts*.

Formally, we assume a set of atomic concepts, denoted as $\mathcal{S}_c, |\mathcal{S}_c| = M$, include predicates, arguments, and entities that constitute events. Each event $\mathbf{y}$ is defined as a composition of several *atomic concepts*: $y = \{c_0, c_1, \cdots, c_a\}, c_i \in S_c$. Furthermore, each token is associated with an atomic concept label $\mathbf{C} = [c_1, \ldots, c_N]$. An event occurs only if all of its atomic concepts are present in the sentence. When necessary atomic concepts are present, though, whether the event occurs still depends on the context of the sentence.

Atomic concepts represent event predicates as well as the involved arguments in different events. For example, the atomic concepts can be predicates such as `ANNOUNCE` and `REOPEN`, or entities such as `SCHOOL` and `GOVERNMENT`; these atomic concepts are then composed into different events, such as `SCHOOL REOPEN`, and `GOVERNMENT ANNOUNCE`. A special concept named `NONE` is defined, used there's no other concept present.

The zero-shot event detection task is to train an event detector E from training samples belonging to a set of seen event types $\mathcal{Y}_{\text{seen}}$, namely $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}, y_i \in \mathcal{Y}_{\text{seen}}$. But at test time, E may encounter samples belonging to new event types $\mathcal{Y}_{\text{new}}$ that were never seen before during training. The new event types $\mathcal{Y}_{\text{new}}$ share the same set of atomic concepts with $\mathcal{Y}_{\text{seen}}$, but there are *zero* samples of the new event types during training. As such, the model must have the ability of composing previously seen atomic concepts into never-

seen-before event types with zero training. The zero-shot event detection problem has two settings: 1) conventional zero-shot learning, where all the test data are from the unseen classes ; 2) generalized zero-shot learning, where the test set includes data from both seen and unseen classes.

3.2 Model Architecture

At a high level, our CGMN model for zero-shot compositional event detection consists of three key components, as shown in Figure 3.1. First, it has a modular network (section 3.3) that embeds atomic concepts into a latent semantic space. Rephrasing needed: The concept embeddings will serve as building modules for learning composition semantic representations for both sentences (section 3.4) and events (section 3.5). Finally, CGMN determines whether a target event occurs in a sentence, while tying its weights to event representations to enable zero-shot detection for unseen events. In what follows, we detail these four components.

3.3 Modular Network for Embedding Atomic Concepts

CGMN features a modular network that learns embeddings for *atomic semantic concepts* of events. The concept embeddings output by the modular network will be fundamental building blocks for learning compositional representations of both events and sentences and realize compositional event detection. As shown in Figure 3.1, we design in CGMN corresponding modules that map atomic concepts into a shared embedding space $\boldsymbol{u}_i \in \mathbb{R}^D$ where D is the embedding dimension. The modules for embedding atomic concepts can be parameterized by a matrix $\mathbf{U} \in \mathbb{R}^{M \times D}$, which is end-to-end trainable during the learning of CGMN.

By treating such atomic concepts as basic units, we will later compose concept embeddings to obtain semantic representations for both sentences (section 3.4) and events (section 3.5). The reason of using such modularized concept embeddings, instead of directly

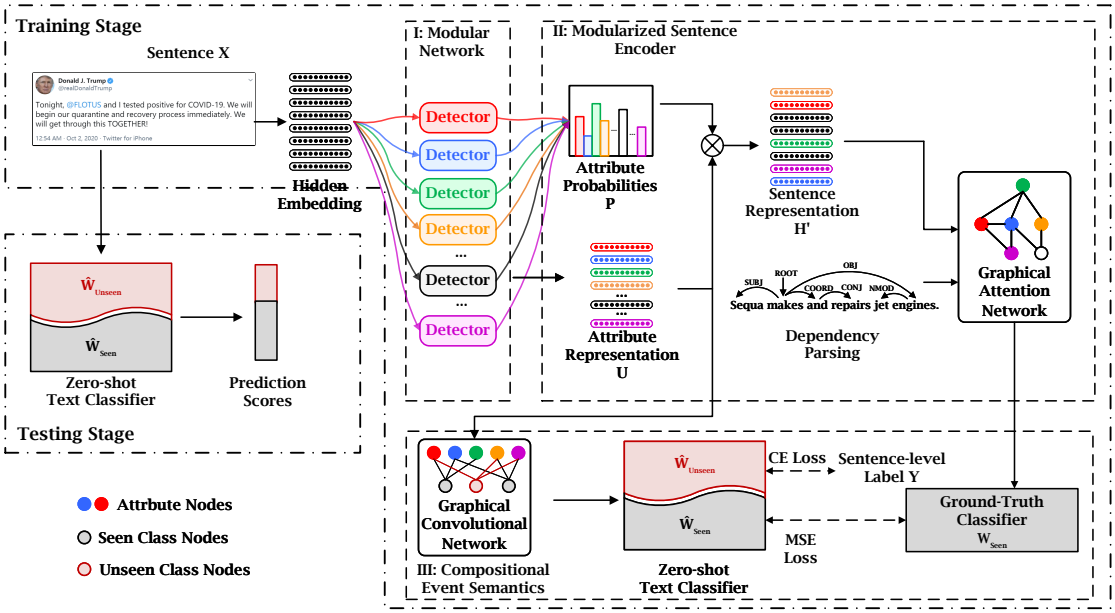

Figure 3.1: A sketch map of our proposed method frame, including 3 main components: 1) the modular network for embedding atomic concepts; 2) the modularized sentence encoder; and 3) compositional event semantics.

using BERT embeddings, is two-fold: 1) it maps atomic concepts, sentences, events into a shared semantic space, which facilitates matching events with sentences based on their compositional semantics; 2) it provides canonical and reusable abstractions, which enables connecting newly unseen event types with seen ones and obtaining their representations—this is critical to zero-shot compositional generalization.

3.4 Modularized Sentence Encoder

In this section, we want to learn *sentence-level concept representation*, i.e. representation of sentence semantics in terms of atomic concepts To this end, we propose an two-stage encoder framework. First, it detects probable atomic concepts on the token-level, and represents tokens in terms of atomic concepts, called *token-level concept representation*. Second, it aggregates token-level concept representations with dependency parse to produce concept representations on the sentence-level.

Token-level atomic concept detection. Given the BERT-encoded hidden representation of an N-token sentence $\{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_N\}$ as the input, CGMN utilizes multi-layer perceptrons as token classifiers to produce the probability distribution of each token over the atomic concepts. Namely, it computes a probability distribution $\mathbf{P} \in \mathbb{R}^{N \times M}$, denoting the classification probabilities of N tokens over M different atomic concepts. With $\mathbf{P}$, we can transform the BERT-encoded token representations into token-level concept representations by multiplying the probability matrix $\mathbf{P}$ with the universal atomic concept embedding matrix $\mathbf{U}$:

$$\mathbf{H}' = \mathbf{P}\mathbf{U} \in \mathbb{R}^{N \times D}, \tag{3.1}$$

where $\mathbf{H}'$ is an $N \times D$ matrix representing the the new token embeddings in the atomic concept space. In matrix $\mathbf{H}'$, each row vector represents its corresponding token in terms of atomic concepts.

Sentence-level atomic concept aggregation. Now, we need to construct sentence-level concept representations from the token-level. To better learn compositional sentence representations from the above token embeddings, it is important to model the compositional relations between tokens even for distant tokens in the sentences. Thus, we design our sentence encoder using Graph Attention Networks (GAT, [55]) and apply it over dependency parse to efficiently propagate information among tokens and focus on important parts of the sentence with syntactic guide. Formally, the dependency parsing tree is an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_{N_v}\}$ and $\mathcal{E} = \{e_{(v_i,v_j)} | v_i, v_j \in \mathcal{V}; i, j \in [0, N_e)\}$ are the nodes and edges. Each node $v_i \in \mathcal{V}$ represents a token w_i in the sentence, while each edge $e_{(v_i,v_j)}$ is a dependency parsing arc [58, 59]. We apply GAT to first compute the attention coefficients between a certain node i and its neighbours $j \in \mathcal{N}_i$:

$$e_{ij} = a([\mathbf{W}_{GAT}\mathbf{h}'_i \| \mathbf{W}_{GAT}\mathbf{h}'_j]),\ j \in \mathcal{N}_i, \tag{3.2}$$

where $\mathbf{W}_{GAT}$ is a learnable weight matrix of GAT, $a(\cdot)$ is the self-attention mechanism, and $[\cdot\|\cdot]$ means the concatenation operation. Then, we normalize the attention coefficients with softmax functions:

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(e_{ij}))}{\sum_{k\in\mathcal{N}_i}\exp(\text{LeakyReLU}(e_{ik}))}. \tag{3.3}$$

After computing the coefficients, we then aggregate them via concatenating the attention features:

$$\mathbf{h}''_i = \sigma(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\mathbf{W}_{GAT}\mathbf{h}'_j), \tag{3.4}$$

where $\mathbf{h}''_i$ is the new hidden feature of each node i (after fusing the neighbour domain information) and $\sigma(\cdot)$ is the activation function.

As the output of the sentence-encoding attention network is the node representations after propagation along the edges, we need to summarize the information to construct a graph representation. Hence, we add a Readout layer after the attention network for this purpose and derive the sentence representation $\mathbf{h}_s$ as:

$$\mathbf{h}_S = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\mathbf{h}''_v + \text{MaxPooling}(\mathbf{h}''_1, \mathbf{h}''_2, \cdots, \mathbf{h}''_{\mathcal{V}}). \tag{3.5}$$

As such, we've obtained sentence-level concept representations that can be conveniently compared with the event representations built from the same atomic concepts, enabling principled zero-shot learning and domain generalization. The remaining of this section is devoted to the construction of event representations.

3.5 Graph-Based Learning of Compositional Event Semantics

While we can directly train a classifier with weights $\mathbf{W}$ on the sentence-level concept representation for event detection, the learned weights on seen event types cannot naïvely generalize to unseen event types. Instead, we build event representations from atomic

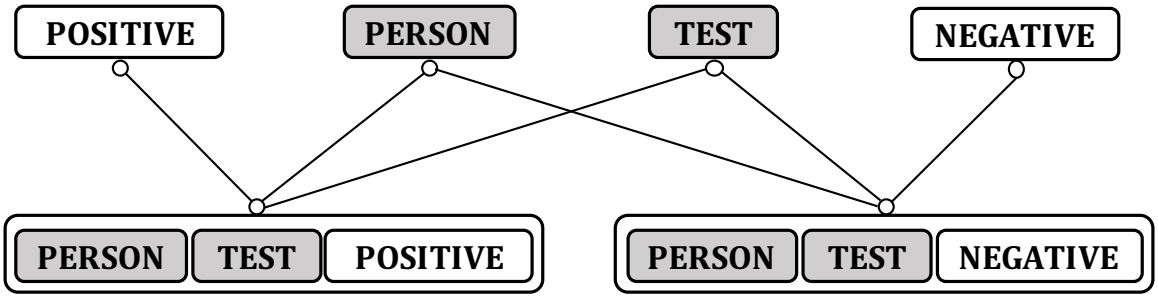

Figure 3.2: An example of event-concept schema graph. The first row includes example atomic concepts, and the second row event types consisting of three concepts each.

concepts and use them as classifier weights.

To generalize to unseen events in a principled manner, we construct an *event-concept schema graph* that encodes relations between events through common atomic concepts as intermediates, and we obtain representations for unseen events by propagating information from seen events to unseen events by this graph, thus enabling classification for unseen event types at test time.

Event-concept schema graph. The event-concept schema graph is a bipartite graph of events and atomic concepts. An edge exists between an event and an atomic concept if the latter is a constituent part of the former. Figure 3.2 shows an example schema graph.

Compositional event representations. After constructing the event-concept schema graph , we apply GCN to learn compositional event representations as follows:

$$\mathbf{Z}^{\ell+1} = \sigma(\hat{\mathbf{A}}\mathbf{Z}^{\ell}\mathbf{V}^{\ell}), \tag{3.6}$$

where $\hat{\mathbf{A}} \in \mathbb{R}^{(M+K)\times(M+K)}$ is the normalized adjaceny matrix of the graph, $\mathbf{V}^{\ell} \in \mathbb{R}^{D\times D}$ is the trainable weight matrix during the training process at l-th layer and $\mathbf{Z}^{\ell} \in \mathbb{R}^{(M+K)\times D}$ is the node feature representations at l-th layer.

To initialize node embeddings at the input layer, for atomic concept nodes we use embeddings learnt in section 3.3, and for event type nodes, we average the representations of

their constituent atomic concepts. We use the output node embeddings for event types as event representations and predicted event classifier weights $\hat{\mathbf{W}}$.

The event-concept schema graph encodes the second-order relation between events through common neighbors, i.e. atomic concepts, and how informative an atomic concept is to an event. By information propagation with GCN, similar event types according to informative atomic concepts would share similar internal representations in GCN.

Furthermore, we expect that the event representations can be effectively used for event detection. At training stage, we train a set of classifier weights $\mathbf{W}'|_{\text{seen}}$ directly on seen event types. We enforce that predicted weights for seen events, $\hat{\mathbf{W}}|_{\text{seen}}$ are close to the weights $\mathbf{W}'|_{\text{seen}}$ by the regularization loss (Equation 3.8). With all these combined, the GCN model should be able to predict classifier weights for seen and unseen event types according to their similarities.

3.6 Loss Function

For training CGMN, the overall objective function consists of three losses:

1. The atomic concept loss ℓ_1. For a token x in a sentence, the modularized sentence encoder in section 3.4 predicts probability distribution of atomic concepts for this token, $\hat{\mathbf{p}}$. The following atomic concept loss

$$\ell_1(x, c) = \text{cross\_entropy}(\hat{\mathbf{p}}, c) \tag{3.7}$$

where c is the ground-truth atomic concept label is used for training signals for automic concept modules.

2. The regularization loss ℓ_2. For sentences with event types seen in training, we define the regularization loss to make sure that the weights predicted with GCN in section 3.5, $\hat{\mathbf{W}}$, is close to the weights $\mathbf{W}'$, where $\mathbf{W}'$ is the classifier weights for seen event types trained

directly on the training examples with standard multiclass classification setting:

$$\ell_2 = \text{mean\_squared\_error}(\hat{\mathbf{W}}|_{\text{seen}}, \mathbf{W}'|_{\text{seen}}) \tag{3.8}$$

where $|_{\text{seen}}$ restricts the weight matrices to include columns of seen event types only.

3. The ground-truth classifier loss ℓ_3. Finally, we minimize the cross entropy loss for predicting events for sentence $\mathbf{x}$, where $\mathbf{y}'$ is the predicted event probabilities for seen event types $\mathbf{W}'|_{\text{seen}}$ and $\mathbf{y}$ is the ground-truth event type vector:

$$\ell_3(\mathbf{x}) = \text{cross\_entropy\_loss}(\mathbf{y}', \mathbf{y}) \tag{3.9}$$

The total loss during the training stage can be obtained once we compose the three kinds of loss together. To make the generated weights fit the ground-truth classifier weights faster than the speed of the ground-truth classifier changes, we apply a partition parameter before L_3:

$$\ell_{\text{total}} = \sum_{(\mathbf{x},\mathbf{y})} \sum_{x_i \in \mathbf{x}} \ell_1(x_i, c_i) + \ell_2 + \lambda \sum_{(\mathbf{x},\mathbf{y})_{\text{seen}}} \ell_3(\mathbf{y}', \mathbf{y}) \tag{3.10}$$

During testing stage, as the prediction scores for the seem classes are intended to be higher than the unseen classes, we employ a calibration bias on all the prediction scores for unseen classes according to the previous work [24, 28].

While in training, we use optimize both $\mathbf{W}'$ and predicted weights $\hat{\mathbf{W}}$, during inference at test time, we only use the weights $\hat{\mathbf{W}}$ predicted by our model to predict both seen and unseen event types.

CHAPTER 4
EXPERIMENTS

In this section, we empirically evaluate the performance of CGMN for zero-shot compositional text classification.

4.1 Datasets

4.1.1 Basic Information

To evaluate the zero-shot compositional generalization ability of different methods, we use two datasets on event classification.

• **Twitter-COVID19** is a dataset we collected during 2020/05/13 and 2020/07/06, which consists of 2,002 tweets discussing COVID-19 related events during the pandemic. We build this dataset to evaluate compositional zero-shot text classification because the semantics of the events are inherently compositional. Specifically, the event semantics is composed of atomic attributes, , "PERSON TEST POSITIVE" is composed of attributes "PERSON", "TEST", and "POSITIVE". In total, this dataset contains 127 different attributes and 163 different event labels. Among the 163 events, we use 72 as seen classes and the remaining 91 as unseen ones for testing the generalization ability of methods. The details of data collection and annotation process for this dataset are provided in sec:appendix.

• **ACE 2005** [5] is a popular benchmark for event detection. We treat the event labels as compositional labels whose semantics are composed from event predicates and arguments. The dataset originally contains 599 different documents. We preprocessed the dataset by excluding the instances that contain no event, and also segmenting the documents into sentences to perform sentence-level event classification. In this way, we obtain a total number of 3877 sentences. These sentences are annotated with 41 token-level argument

labels and 653 sentence-level event type labels. Among the 653 events, we use 400 as seen classes for training and the remaining 253 as unseen ones.

To extract compositional event information from the Twitter data about the COVID-19, we establish a dataset of Twitter-Covid19. For each tweet in the dataset, we offer a corresponding event/sentence-level label and a set of corresponding fine-grained token-level labels, so that it can be utilized either in sequence labelling tasks and sentence classification tasks. Table 4.1.1 shows the brief information about the dataset.

Table 4.1: Brief Information of Twitter-COVID19 dataset.

Data Set	# of Annotated Tweets	ZSL-Setting
Training	1200	S
Validation	400	S
Test	402	S(263) + U(149)
Total	2002	S(1863) + U(149)

The statistics of dataset partitions is shown in Table 4.2.

4.1.2 Data Collection

We have been continuously collecting the Twitter data since 2020/03/06 by tracking certain COVID-19 related keywords with the Twitter-API. For further annotation, we continue to filter the COVID-19 related Tweets with certain sets of keywords to get the required event type data. For example, we utilize "shortage", "lack" and "short of" to filter *SHORTAGE*-related Tweet events. Following these steps, we have annotated 2002 Tweets sampled from 2020/05/13 to 2020/07/06. We manually reduce the number of the duplicated Tweets and

Table 4.2: Statistics of the Twitter-COVID19 and ACE 2005 datasets.

Dataset	Seen Data			Unseen Data
	#Train	#Valid	#Test	#Test
Twitter-COVID19	1200	400	263	139
ACE 2005	2400	600	530	347

choose different topics for the Twitter data posted on the same day.

Token-level Annotations: For the token-level annotations, Tweets are required to be split into smaller-scale words or sentences, which is known as the tokenization operation. We utilize the Tweetokenizer in the NLTK toolkit to accomplish this issue and then utilize the AllenNLP's NER tool to accomplish a primary annotation towards the tokens. With the rough annotated Tweets, we then manually check the labels' validity and ensure them to follow the BIO format as is widely utilized in Named Entity Recognition (NER) tasks, where we add the suffix "B" (begin) to the first token of a mention and "I" (inside) to the tokens following it. And for the other tokens, we will assign an "O" (other) tagging to them. Following these steps, we generated 163 different token labels and annotated them on the previously obtained Twitter data. Besides the token labels popular in most NER tasks like *PER*, *ORG*, *LOC*, *TIME* and so on, we focus more on the fine-grained concepts related to COVID-19: *TEST*, *POSITIVE*, *VACCINE*, etc.

Sentence-level Annotations: According to the Token-level Annotations before, we set up the corresponding sentence-level annotation. We utilize 3-element tuples of the annotated token labels (without BIO schema) (c_0, c_1, c_2) to represent the brief information of the whole Tweet $\mathcal{C}$. For example, we utilize the tuple *(PER, TEST, POSITIVE)* to represent the events describing some person or people testing positive for COVID-19. If the tweets do not have as many main elements as 3, we pad the rest label components with "-", which has similar meanings with "O" in token-level annotations. For the Sentence-level annotations, we tagged 167 event types in all, covering events on $\{$*REOPEN*, *EDUCATION*, *SHORTAGE*, *CHARITY*, *VACCINE*, *PROTEST*, *DEATH*, *DELAY*, *CANCEL*, *BANKRUPT*, *FUNDAID*, *TEST POSITIVE*, *TEST NEGATIVE*, *DISEMPLOYMENT*$\}$.

4.2 Evaluation Protocols and Metrics

During the testing stage, we aim to test on both non-generalized and generalized zero-shot learning setting. For **non-generalized zero-shot learning setting**, we test model perfor-

mances on pure unseen data. As this is basically a text classification problem, we use multiple metrics including:

1. Accuracy on the unseen categories;
2. Macro precision over all testing the unseen classes;
3. Macro recall over all testing the unseen classes;
4. Macro F_1 score over all testing the unseen classes;
5. Besides, to measure the performance on the seen classes, we also utilize the accuracy on the seen classes.

For **generalized zero-shot learning setting**, we test on both seen data and unseen data. Thus, we use similar multiple metrics:

1. Overall accuracy on testing data, including both seen data and unseen data;
2. Accuracy on the unseen categories;
3. Accuracy on the seen classes;
4. Macro Precision over all the categories in testing set;
5. Macro Recall over all the categories in testing set;
6. Macro F_1 score over all the categories in testing set.

Given n samples in the testing set, we can assume $\{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_n\}$ as the model predictions and $\{y_1, y_2, \cdots, y_n\}$ as the ground-truth labels respectively. The testing set includes k categories in all.

Accuracy (Acc) measures the predictive accuracy on the categories appearing in the testing set. It only computes the cases where the sentence-level prediction equals the ground-truth label:

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{n} \mathbb{I}(y_i = \hat{y}_i). \tag{4.1}$$

Macro Precision is the average of the precision score computed over each category in the testing set, which is the proportion of correctly predicted samples in total predicted samples:

$$\text{macro precision} = \frac{1}{k}\sum_{i=1}^{k}\text{precision}_i\,. \tag{4.2}$$

Macro Recall is the average of the recall score computed over each category in the testing set, which is the proportion of correctly predicted samples in total gold samples in the dataset:

$$\text{macro recall} = \frac{1}{k}\sum_{i=1}^{k}\text{recall}_i\,. \tag{4.3}$$

Macro $\mathbf{F_1}$ score is the average of the $\mathrm{F_1}$-score computed over each category in the testing set. $\mathrm{F_1}$ score for each category can be computed via:

$$\mathrm{F_1}\text{-score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{4.4}$$

, while the macro average can be expressed as:

$$\text{macro } \mathrm{F_1}\text{-score} = \frac{1}{k}\sum_{i=1}^{k}\mathrm{F_1}\text{-score}_i\,. \tag{4.5}$$

4.3 Baseline Methods

• **BERT** is a popular model widely utilized in text classification. It is an uncased BERT-base model and we follow the sentence classification pipeline in [6]. With no adaptations to zero-shot learning settings, the model offers a benchmark on seen data accuracy.

• **BERT-M** is a basic zero-shot text classification approach, treating the task as a binary matching problem between sentences and labels. Considering the compositional labels as sequences, it uses the uncased BERT-base model to obtain representations for both sentences and labels. By computing the cosine similarity between a label and the sentence representations, the model measures how well a label matches the text,

• **TMN** [24] is a state-of-the-art compositional zero-shot learning model in computer vision area. It adopts a set of fully-connection-layer based modules with no explicit meanings and configure them with a gating mechanism in a task-driven way. The model can be generalized to unseen compositions of attributes via re-weighting the primitive modules to accomplish the zero-shot classification.

• **ZS-GCN** [11] is a zero-shot learning method also based on GCN. It uses semantic embeddings of labels and the categorical information in the knowledge graph as the node inputs and the graph structure of GCN to introduce the external relation information between seen classes and unseen classes. After iterations, the node representations of GCN will serve as the weights of the classifiers.

• **ZSTC-E** [29] is a benchmark of zero-shot text classification. It formulate the zero-shot text classification problem as an entailment learning problem. Considering the compositional sentence labels as sequences and also the definitions of the categories, it feeds both the sentence and labels to BERT, and predicts an event by deciding whether the constituent attributes are entailed by the sentence.

4.4 Implementation Details

The network structure of CGMN utilizes BERT-base-uncased [6] as the sentence encoder, a 1-layer single-head Graph Attention Network (GAT) in the sentence compositional semantics learning, and a L-layer Graph Convolutional Network (GCN) in graph-based learning of compositional label semantics. For training stage, we use ADAM [60] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ in our experiments for all the models. We use a learning rate of 0.001 and a batch size of 32 for the model across all datasets. The maximum number of epochs is 10. The max sequence length is set to 256. We select λ, the loss partition parameter during the model learning, from $\{0.1, 0.25, 0.5, 1, 2\}$ and the number of GCN layers L from $\{1, 2, 3, 4, 5\}$ based on the predictive accuracy on the development set.

Table 4.3: Predictive accuracy under non-generalized zero-shot learning setting. S Acc and U Acc are the predictive accuracy on the seen classes and unseen classes respectively. U Precision, U Recall, and U F_1-Score denotes the macro precision, recall, and F_1-Scores computed over unseen classes. The results are presented in percentage (%).

Models	Twitter-COVID19					ACE				
	U Acc	U Pre	U Rec	U F_1	S Acc	U Acc	U Pre	U Rec	U F_1	S Acc
BERT	0.72	1.06	0.30	0.72	64.26	0.58	0.03	0.79	0.05	35.09
BERT-M	15.11	10.23	18.10	13.07	41.14	5.08	4.02	5.62	5.03	22.31
TMN	17.99	10.98	17.40	12.01	66.54	15.32	13.82	18.48	14.81	34.71
ZS-GCN	22.97	15.35	25.09	16.81	67.18	23.22	16.69	20.75	16.85	35.16
ZSTC-E	18.88	11.27	18.45	12.63	39.40	18.73	11.15	17.99	12.48	29.12
Ours	**25.90**	**17.65**	**26.74**	**18.79**	**68.44**	**25.94**	**17.21**	**21.82**	**17.43**	**36.22**

4.5 Main Results

Non-Generalized Zero-Shot Learning: Table 4.3 shows the experimental results on both Twitter-COVID19 and ACE 2005 dataset in the non-generalized zero-shot setting. Compared with all the baselines, CGMN performs the best with a significant increase on both two datasets in terms of all the metrics. On average, our model not only achieves an improvement of 2.83% and 1.28% on unseen data accuracy and unseen macro F_1 score over the strongest baseline, but also outperforms the other models on seen data accuracy. This indicates that strong compositional capability will also make benefits on seen data classification.

Generalized Zero-Shot Learning: For generalized zero-shot learning setting, we can get similar results as is shown in Table 4.4. CGMN performs the best on all the metrics with only a little sacrifice of seen data accuracy. On average, CGMN achieves an improvement of 1.2% and 1.5% on overall accuracy and overall F_1 score over the strongest baseline, showing that CGMN possesses a strong zero-shot generalization ability. Specifically, the strongest baseline on seen data accuracy, BERT, does not possess a zero-shot learning ability. With still comparable seen data accuracy with BERT, CGMN does not sacrifice too much seen data accuracy to accomplish the generalized zero-shot learning problem.

Table 4.4: Predictive accuracy under generalized zero-shot learning setting. O Acc indicates the overall accuracy on testing set including both seen and unseen data. S Acc and U ACC are the predictive accuracy on the seen classes and unseen classes respectively. O Precision, O Recall, and O F_1-Score denotes the macro precision, recall, and F_1-Scores computed over all the testing classes. The results are presented in percentage (%).

Models	Twitter-COVID19					
	O Acc	S Acc	U Acc	O Pre	O Rec	O F_1
BERT	42.29	**64.26**	0.72	7.99	13.20	9.18
BERT-M	24.87	35.35	5.04	5.22	11.74	
TMN	42.29	59.70	9.35	14.19	16.68	13.98
ZS-GCN	44.62	61.36	12.95	15.00	19.19	15.05
ZSTC-E	27.70	33.04	17.60	10.38	12.55	10.36
Ours	**45.22**	59.31	**18.55**	**16.68**	**23.04**	**17.62**
Models	ACE					
	O Acc	S Acc	U Acc	O Pre	O Rec	O F_1
BERT	21.44	**35.09**	0.58	1.88	3.97	2.08
BERT-M	6.81	13.79	20.84	3.02	1.12	2.56
1.16						
TMN	22.39	31.58	8.35	10.97	11.81	10.42
ZS-GCN	26.53	34.20	14.82	12.75	14.68	13.65
ZSTC-E	24.32	28.87	17.37	12.61	13.76	12.76
Ours	**28.32**	33.08	**21.04**	**14.02**	**16.61**	**14.08**

Table 4.5: Examples incorrectly predicted by TMN, but correctly predicted by CGMN. Green parts indicates the correct components, while red parts and blue parts indicate the attribute detection error and attribute selection errors.

Ground-Truth Label	Incorrect Examples	Wrong Predictions
COMPANY REOPEN LOC	Teesside Cannabis Club shop reopens to members after lockdown closure Teesside Cannabis Club has reopened their shop to members following the coronavirus pandemic. The Exhale shop, located on Norton Road in #cannabiscommunity	COMPANY REOPEN CLOSE
ORG SHORTAGE RESOURCE	Kenya is hoarding unprocessed COVID-19 samples according to the Kenya Medical Association (KMA), a medical agency, following a shortage of reagents to carry out the tests.	GOV SHORTAGE RESOURCE

More discussion about the baselines: For baseline methods, we also have some additional observations on their performances. Without any information propagation from the seen classes, BERT makes almost no correct predictions on the unseen classes. For TMN, the gating mechanism will choose the modules automatically without any semantics or syntactic information as restriction. With some interference information or redundant information in the sentence, the method will not perform as good as CGMN on evaluations over both seen and unseen data, which also proves the effectiveness of our GAT based sentence compositional semantics learning.

For ZS-GCN, although the model is also based on GCN, the sentence representation and label representation are individually obtained. CGMN outperforms this method on all experiments, proving the efficiency of our universal attribute semantics.

4.6 Ablation Study

We perform ablation studies to evaluate the effectiveness of our three components: 1) the universal attribute embeddings, 2) the GAT based compositional sentence-level attribute encoder, 3) the label-attribute schema graph based GCN, and 4) the GCN output based

Table 4.6: Ablation study on Twitter-COVID19 dataset. For each ablation model, the parameters remain default as $\lambda = 0.25$ and $L = 1$. The results are presented in percentages (%).

Models	Non-generalized		Generalized		
	S Acc	U Acc	S Acc	U Acc	O Acc
BERT	64.26	0.72	64.26	0.72	42.29
Ours w/o UACE	64.25	**26.62**	53.61	17.27	42.79
Ours w/o GAT	**68.82**	11.51	55.89	10.79	40.30
Ours w/o GCN	58.94	14.39	56.65	2.16	37.81
Ours w/o ZSC	62.74	0.72	**62.74**	0.72	41.30
Ours	68.44	25.90	59.31	**18.55**	**45.22**

zero-shot text classifier. Table 4.6 shows the results on the Twitter-COVID19 dataset under both the non-generalized and generalized zero-shot setting. Our findings are shown as followings:

• **Ours w/o UACE:** Without the universal atomic concept embeddings (UACE), the predictive accuracy for seen classes drops significantly by 5.13% on average. Removing this part will result in the input of the GAT based sentence representation not in the same space as the following compositional label representation. However, the relation information between the seen classes and unseen classes are preserved by GCN. Therefore, there is an obvious drop in seen classes, while the performance on unseen classes maintains.

• **Ours w/o GAT:** Without the GAT based compositional sentence-level attribute encoder , the predictive accuracy on unseen classes drops signifigantly by 11.13% on average. Removing this part and simply averaging all the token representations as the sentence representation will stop the model from pruning redundant information and selecting the main parts of the sentences. Generating some intereference in this way, there is a significant drop on the performance over the unseen data.

• **Ours w/o GCN:** Without the label-attribute schema graph based GCN, the accuracy for seen classes drops by 6.08% and the accuracy for unseen classes drops by 14.02% on average. With simple average of the constituent attribute embeddings as the compositional label representations serving as the classifier weights, the relationships between compositional

labels become ambiguous as the information propagation between labels is missing, which makes it more difficult for the model to link the label representations with the sentence representations.

• **Ours w/o ZSC:** Without the GCN output based zero-shot classifier (ZSC), the accuracy for unseen classes drops to almost 0, as this part takes the very important role of transferring the knowledge from the seen classes to the unseen classes. Removing this part will make our model lose the ability of zero-shot learning.

4.7 Parameter Study

We investigate the effects of the loss partition weight λ and the number of GCN layers L, whose default values are $\lambda = 0.25$ and $L = 1$. All the experiments are conducted on Twitter-COVID19 dataset under both the non-generalized and the generalized zero-shot learning setting. Figure 4.1 summarizes and presents the results. Besides the study on influence of model parameters, we also want to learn how the partitions of seen and unseen classes will effect the model performance. Figure 4.2 displays the results.

Effect of GCN Depth: Along with the increment of the number of GCN layers L, the performances decrease. As our attribute-concept schema graph has limited size, too many iterations of information propagation along the edges results in a loss of structure information. Another reason might be that the optimization becomes harder with the network going deeper.

Effect of Loss Proportions: While increasing λ from 0.1 to 2, the performances improves at first and then begins to drop. If λ is too small, the sentence classifier is trained slowly; if λ is too large, the model will update the sentence classifier first, which might cause greater difference between weights of the ground-truth regularization classifier and the GCN based zero-shot classifier. Both circumstances harm the performance.

Model Sensitivity to Unseen Data Proportions: In this part, we investigate how the data partition between seen data and unseen data influences the model performance. There are

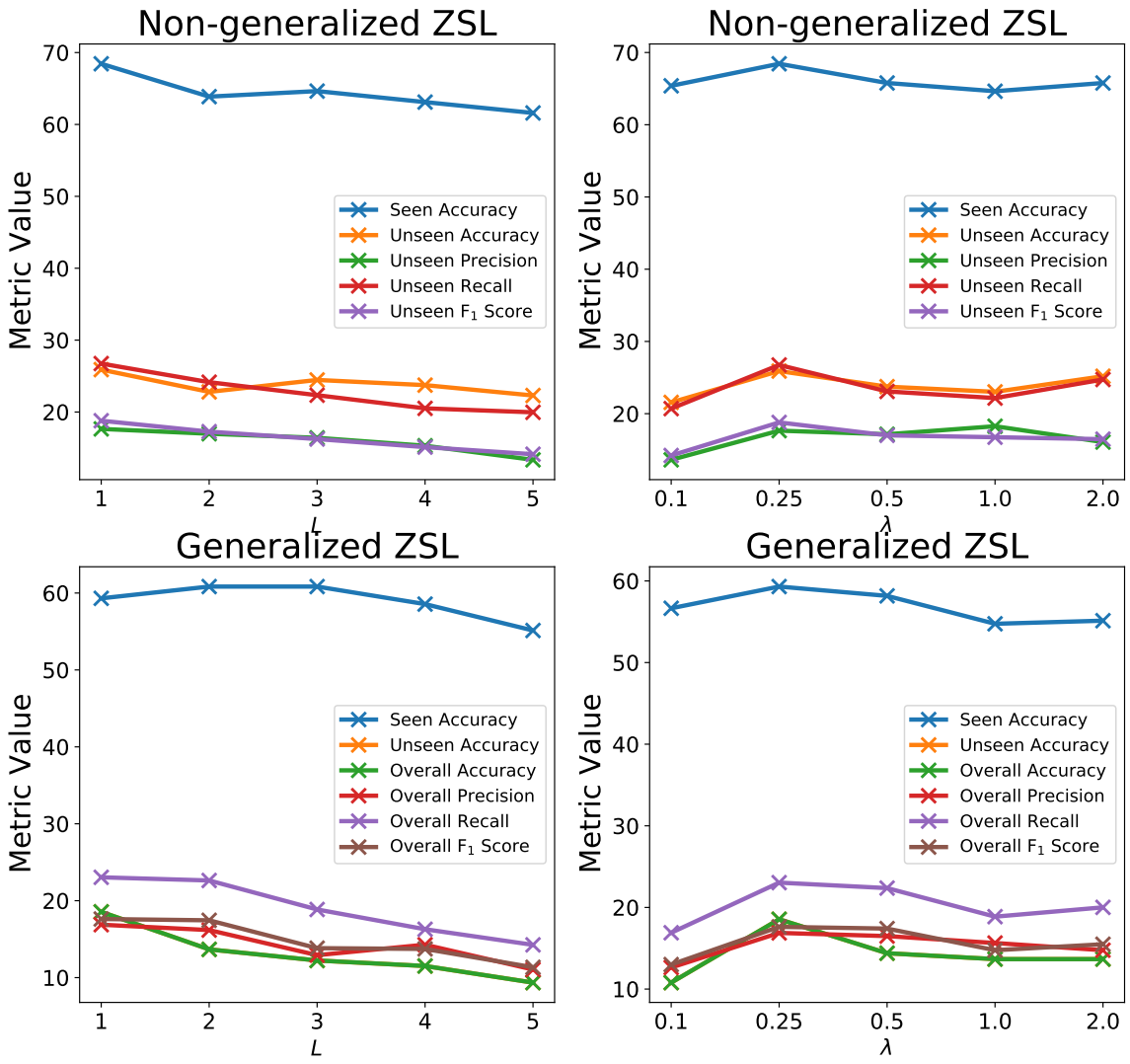

Figure 4.1: Parameter study investigating the influences of the loss partition weight λ and the number of the layers L. The first row and the second row show the results obtained under the non-generalized and generalized zero-shot learning setting respectively. The results are presented in percentages.

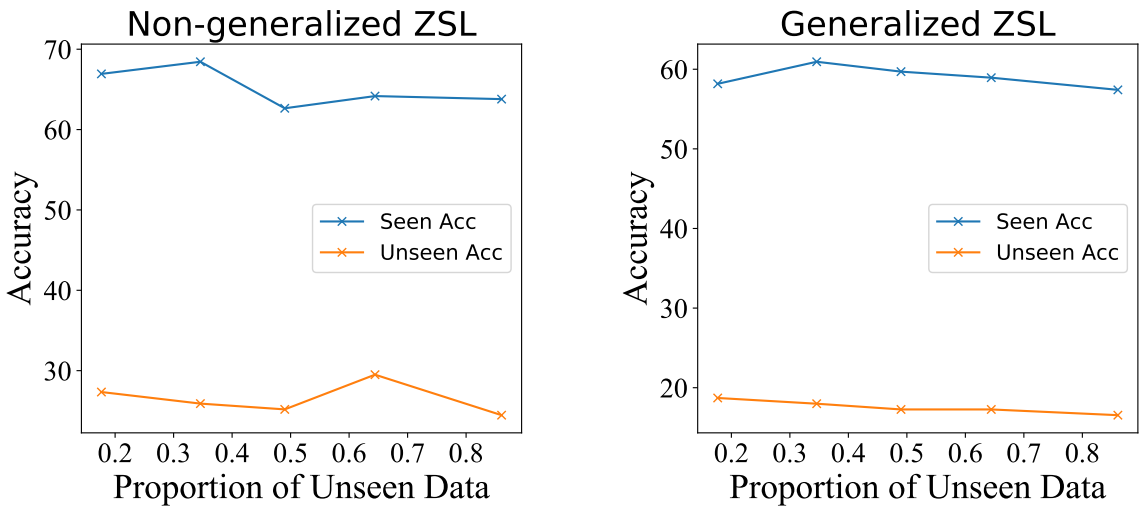

Figure 4.2: The curve of seen accuracy and unseen accuracy on different partition of data.

two changes in this set of experiments: 1) the proportion of the unseen classes to all classes; 2) the proportion of the unseen data in the test set. The first variable is implicit and is the cause of the second variable. As the number of unseen categories increases, the number of unseen examples rises accordingly.

Here, we select $\{63, 91, 103, 113, 123\}$ as the number of the unseen categories, leading to the number of unseen examples in the test set varying in $\{71, 139, 197, 259, 346\}$ from a constant size of the test set. In Figure 4.2, the x grid indicates the proportion of unseen data to testing data, while the y grid shows the accuracy of the model. From the figure, we can see that CGMN performs quite stable with these two variables increasing, which means that CGMN relies more on its capability of composing different atomic attributes than training on enough number of examples.

4.8 Case Study

We present the t-SNE visualization [61] of the label representations generated by GCN associated with the constituent attributes in the compositonal labels as shown in Figure 4.3. This visualization shows that the label representations are organized by label similarities, indicating that the compositional labels with common constituent attributes tend to

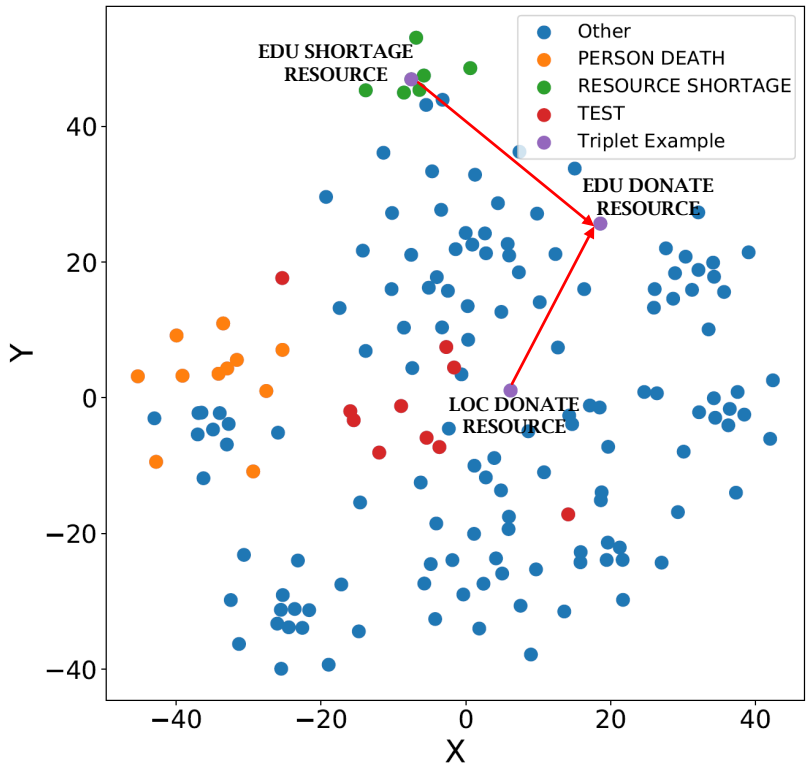

Figure 4.3: t-SNE visualization for our GCN output classifier. We focus on the "PERSON DEATH", "RESOURCE SHORTAGE", and "TEST" related categories. A triplet set of example is given as well to show the compositional capability of the model.

have similar embeddings. Taking the categories with common attributes "PERSON" and "DEATH" as an example, no matter belonging to seen classes or unseen classes, they intend to cluster with each other in the figure. Similar conclusions can be obtained from the distribution of categories with common attributes "RESOURCE" and "SHORTAGE".

To test the distributions of the categories with fewer common attributes, we also plot the representations of labels with only one common attribute of "TEST" and observe that they distribute less clustering than "PERSON DEATH" related labels. As the label representations obtained from the GCN also serves as the weights of the zero-shot classifier $\hat{\mathbf{W}}$, CGMN is more intend to output similar prediction scores towards similar compositional labels.

To show that CGMN possesses the ability of composing attributes to accomplish zero-shot learning, we also offer a triplet example of categories {"EDU SHORTAGE RESOURCE", "LOC DONATE RESOURCE", "EDU DONATE RESOURCE"}. In the triplet, the category "EDU SHORTAGE RESOURCE" and "LOC DONATE RESOURCE" are among the seen classes while training, and the category "EDU DONATE RESOURCE" belongs to the unseen classes, only appearing during the testing stage. In Figure 4.3, the "EDU DONATE RESOURCE" category embeds in the middle of the other two categories, which proves that the "EDU DONATE RESOURCE" category can be compositionally obtained via CGMN.

4.9 Error Analysis

Challenges in Zero-shot Event Detection: To better understand the challenges in compositional zero-shot text classification task, we examine predictions made by baselines. From the misclassified examples, we observe the following major error types: 1) attribute detection error; and 2) attribute selection error. Table 4.5 shows related examples of the errors predicted by TMN.

• **Attribute Detection Error:** This kind of error occurs when some constituent attributes of the predicted labels do not appear in the sentence or match the ground-truth labels. The

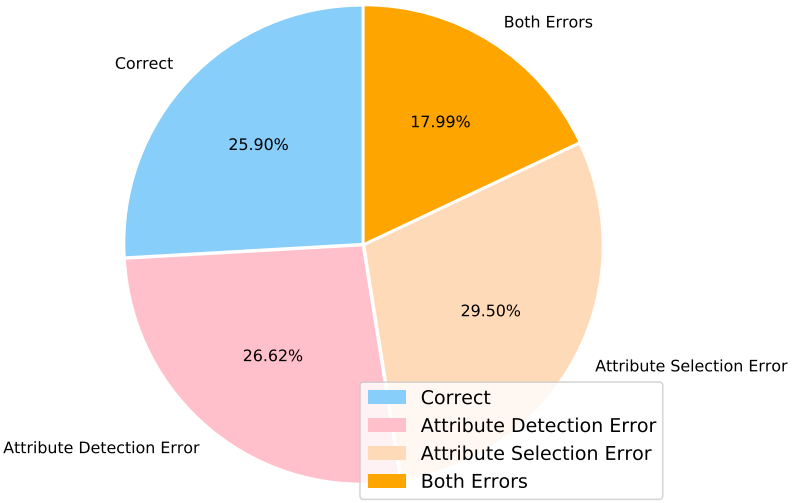

Figure 4.4: The proportions of error types predicted by CGMN.

error is usually caused by incorrect detection of attributes.

• **Attribute Selection Error:** Different from the previous one, some constituent attributes of predicted labels have appeared in the sentence instead of the ground-truth labels. The reason is usually that the information collected in the sentences are not in the correct direction or from the correct parts, leading to the model selecting incorrect attributes to compose.

Error Analysis for CGMN: Our model, to some extent, eliminates the error types mentioned above. We utilize GAT based compositional attribute encoder and dependency parsing tree to extract the information from the main parts in the sentence. However, these error types have not been totally removed. The statistics of error types occuring in the CGMN predictions are shown in Figure 4.4.

After the analysis, we found some possible reasons: 1) For attribute detection errors, due to the imbalanced appearance of the attributes during the training process and the existence of some ambiguous or noisy examples in the data, models sometimes annotate some tokens with incorrect attributes in the sentences. The evaluation of the results from the attribute detector in CGMN displayed in Table 4.7 also proves this. 2) For attribute selection errors, as the dependency parsing tree relies on the external tool—AllenNLP [1], it is more

[1]Obtained from website: https://allennlp.org/

Table 4.7: Evaluation of the results from the attribute detector in CGMN.

Metrics	Value(%)
Macro-Recall	79.61
Macro-Precision	89.79
Macro-F1 score	82.55
Macro-Accuracy	99.92
Overall Acccuracy	81.53
Off-O Accuracy	80.40
O Accuracy	81.64

likely to introduce error propagation stemming from dependency parsing errors.

CHAPTER 5
CONCLUSION

We proposed a compositional graph neural network for the zero-shot compositional event detection problem. To achieve zero-shot compositional generalization, we map both sentences and event labels into a shared embedding space of the atomic attributes, while using graph neural networks to capture compositional semantics. Moreover, we tie the compositional label embedding with the weights of the final classifier to achieve zero-shot learning. Extensive experiments on Twitter-COVID19 dataset and ACE 2005 dataset under both conventional and generalized zero-shot learning settings have demonstrated our model's strong ability of compositional learning and zero-shot generalization. For future work, it is interesting to model atomic attributes that are not given a prior, as well as incorporate compositional grammars as constraints into the model learning process.

REFERENCES

[1] E. Filatova and V. Hatzivassiloglou, "Event-based extractive summarization," in *Text Summarization Branches Out*, 2004, pp. 104–111.

[2] H. Ji and R. Grishman, "Knowledge base population: Successful approaches and challenges," in *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, 2011, pp. 1148–1158.

[3] T. Mitamura, Z. Liu, and E. H. Hovy, "Events detection, coreference and sequencing: What's next? overview of the tac kbp 2017 event track.," in *TAC*, 2017.

[4] J. Berant, V. Srikumar, P.-C. Chen, A. Vander Linden, B. Harding, B. Huang, P. Clark, and C. D. Manning, "Modeling biological processes for reading comprehension," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1499–1510.

[5] C. Walker, S. Strassel, J. Medero, and K. Maeda, "Ace 2005 multilingual training corpus," *Linguistic Data Consortium, Philadelphia*, vol. 57, p. 45, 2006.

[6] 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.

[7] 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.

[8] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 951–958.

[9] 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.

[10] R. Socher, M. Ganjoo, H. Sridhar, O. Bastani, C. D. Manning, and A. Y. Ng, "Zero-shot learning through cross-modal transfer," *arXiv preprint arXiv:1301.3666*, 2013.

[11] X. Wang, Y. Ye, and A. Gupta, "Zero-shot recognition via semantic embeddings and knowledge graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6857–6866.

[12] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[13] D. Ahn, "The stages of event extraction," in *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, 2006, pp. 1–8.

[14] G. A. Miller, *WordNet: An electronic lexical database*. MIT press, 1998.

[15] Q. Li, H. Ji, and L. Huang, "Joint event extraction via structured prediction with global features," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2013, pp. 73–82.

[16] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," 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. 167–176.

[17] T. H. Nguyen and R. Grishman, "Event detection and domain adaptation with convolutional neural networks," 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. 365–371.

[18] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 300–309.

[19] S. Liu, Y. Chen, K. Liu, and J. Zhao, "Exploiting argument information to improve event detection via supervised attention mechanisms," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1789–1798.

[20] T. H. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection.," in *AAAI*, vol. 18, 2018, pp. 5900–5907.

[21] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "Devise: A deep visual-semantic embedding model," in *Advances in neural information processing systems*, 2013, pp. 2121–2129.

[22] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3174–3183.

[23] Y. Li, J. Zhang, J. Zhang, and K. Huang, "Discriminative learning of latent features for zero-shot recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7463–7471.

[24] S. Purushwalkam, M. Nickel, A. Gupta, and M. Ranzato, "Task-driven modular networks for zero-shot compositional learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3593–3602.

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

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

[27] J. Nam, E. L. Mencıa, and J. Fürnkranz, "All-in text: Learning document, label, and word representations jointly," in *Proceedings of the thirtieth AAAI conference on artificial intelligence*, 2016, pp. 1948–1954.

[28] W. Yin, J. Hay, and D. Roth, "Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach," 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. 3905–3914.

[29] Z. Ye, Y. Geng, J. Chen, J. Chen, X. Xu, S. Zheng, F. Wang, J. Zhang, and H. Chen, "Zero-shot text classification via reinforced self-training," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3014–3024.

[30] J. Zhang, P. Lertvittayakumjorn, and Y. Guo, "Integrating semantic knowledge to tackle zero-shot text classification," 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. 1031–1040.

[31] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *Advances in neural information processing systems*, 2009, pp. 1410–1418.

[32] 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.

[33] V. Kumar Verma, G. Arora, A. Mishra, and P. Rai, "Generalized zero-shot learning via synthesized examples," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4281–4289.

[34] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5542–5551.

[35] I. Misra, A. Gupta, and M. Hebert, "From red wine to red tomato: Composition with context," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1792–1801.

[36] T. Nagarajan and K. Grauman, "Attributes as operators: Factorizing unseen attribute-object compositions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 169–185.

[37] E. Akyürek, A. F. Akyürek, and J. Andreas, "Learning to recombine and resample data for compositional generalization," *arXiv preprint arXiv:2010.03706*, 2020.

[38] J. A. Fodor, Z. W. Pylyshyn, *et al.*, "Connectionism and cognitive architecture: A critical analysis," *Cognition*, vol. 28, no. 1-2, pp. 3–71, 1988.

[39] Y. Li, L. Zhao, J. Wang, and J. Hestness, "Compositional generalization for primitive substitutions," 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. 4284–4293.

[40] J. Gordon, D. Lopez-Paz, M. Baroni, and D. Bouchacourt, "Permutation equivariant models for compositional generalization in language," in *International Conference on Learning Representations*, 2019.

[41] J. Andreas, "Measuring compositionality in representation learning," in *International Conference on Learning Representations*, 2018.

[42] L. Kirsch, J. Kunze, and D. Barber, "Modular networks: Learning to decompose neural computation," in *Advances in Neural Information Processing Systems*, 2018, pp. 2408–2418.

[43] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2019.

[44] 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.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

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

[48] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[49] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[50] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[51] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur, "Protein interface prediction using graph convolutional networks," in *Advances in neural information processing systems*, 2017, pp. 6530–6539.

[52] B. Shahsavari and P. Abbeel, "Short-term traffic forecasting: Modeling and learning spatio-temporal relations in transportation networks using graph neural networks," *University of California at Berkeley, Technical Report No. UCB/EECS-2015-243*, 2015.

[53] T. Hamaguchi, H. Oiwa, M. Shimbo, and Y. Matsumoto, "Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 1802–1808.

[54] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 7370–7377.

[55] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[56] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 983–11 993.

[57] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, "Graphrnn: Generating realistic graphs with deep auto-regressive models," in *ICML*, 2018.

[58] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1506–1515.

[59] X. Liu, Z. Luo, and H.-Y. Huang, "Jointly multiple events extraction via attention-based graph information aggregation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 1247–1256.

[60] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[61] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.