# Incorporating External Knowledge in Domain Specific Conversation Systems

*A THESIS*

*submitted by*

**MOGHE NIKITA VINAY SANGEETA**

*for the award of the degree*

*of*

**MASTER OF SCIENCE**

(by Research)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**
**INDIAN INSTITUTE OF TECHNOLOGY MADRAS.**

**June 2019**

# THESIS CERTIFICATE

This is to certify that the thesis titled **Incorporating External Knowledge in Domain Specific Conversation Systems**, submitted by **Moghe Nikita Vinay Sangeeta**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Science (by Research)**, is a bona fide record of the research work done by her under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

**Prof. Balaraman Ravindran**
Research Guide
Professor
Dept. of CSE
IIT-Madras, 600 036

**Prof. Mitesh. M. Khapra**
Research Guide
Assistant Professor
Dept. of CSE
IIT-Madras, 600 036

Place: Chennai

Date:

# ACKNOWLEDGEMENTS

First of all, I express my sincere gratitude to my advisors Prof. Balaraman Ravindran and Prof. Mitesh M. Khapra for their guidance throughout my degree. Their vast expanse of knowledge amazes me. I have enjoyed listening to them during lectures, meetings, or even during the leisure walks on the campus. They have been supportive, co-operative, and have allowed me to explore different areas during my program. They ensured that I had both academic and industrial exposure. I have been fortunate to not only learn the finer aspects of the subject matter from them, but their mentorship has also shaped me into a stronger individual - personally and professionally. I can't thank them enough for standing by me during my personal crises and moments when I felt the sense of direction from a parental standpoint. I am grateful to them forever.

I would like to thank the members of my GTC - Prof. N.S. Narayanaswamy, Prof. Hema A. Murthy, and Prof. Lata Dyaram for their valuable feedback and encouragement. I would also extend my gratitude towards Prof. Sutanu Chakraborti who helped me understand the field of Natural Language Processing.

A majority of the work in this thesis was done in collaboration with Suman Banerjee, Siddhartha Arora, and Priyesh Vijayan. Suman and I ventured into dialogue systems from our first semester and he has been my first reviewer for every stupid/amazing idea. Siddhartha helped me to code better, and his detail-oriented attitude has helped in refining various sections in this work. Priyesh introduced (dragged) me to the network representation literature and was optimistic about the merger of NLP and Graphs which finally worked! Preksha Nema has played devil's advocate right from the dataset collection process to the best performing model. She has been my personal StackOverflow and has been the best senior I could have ever hoped for.

Most of the experiments done in this work were compute-heavy and required several GPUs. These experiments would not have been successful without the help of Karthik Thiagarajan, Sankaran Vaidyanathan, Rahul Ramesh, and Sai Kiran Narayanaswamy who ensured a smooth functioning even when they were chasing their respective academic deadlines. The work in this thesis involved crowdsourcing and several experiments required human evaluation. I thank all the AMT workers around the world and our evaluators who provided with honest judgment. I would like to thank RISE Lab, AIDB Lab, RBC-DSAI, and the Department of CSE for providing me with a productive work environment. I thank the CS office staff, Ramya Akka, Robert Sir, and Shaiju Sir for all the administrative help.

I would deeply cherish the banter with my lab-mates - Aakash, Ananya, Arjun, Avinash, Beethika, Deepaks (Maurya, Mittal), Jaya, Karthik, Nitesh, Pavan, Preksha, Pritha, Priyesh, Sankaran, Sapana, Shreyas, Shubham, Shweta, and Tarun. They have been my tutors, critics, the audience for mock presentations as well as food providers during submissions. Special mention to Tarun who was brave enough to be the first one to befriend me even after my disastrous introduction and has been a great support in all my endeavours. Beyond the ML folks, I have enjoyed discussing my work with Ayesha, Devi, Gargi, and PK. Ayesha mentored me during my NLP course project and has been a patient listener to all my academic and non-academic concerns. Kanika, Shikha, and Stanley ensured we had non-work discussions over a cup of *chai*.

I thank Himanshi, Indrani, Kanika, and Shweta for showering affection through critical and happier times. Deepti, Geeta, Namrata, Neha, Nithya, Preetika, Rupam, Shraddha, and Sivashankri ensured that the meals in mess were always filled with interesting conversations. I have had a fair share of exciting experiences with my batchmates Akshay, Aravind, Nishank, Puneet, Sayantan, Saptakatha, Suman, and Shom. The undergrad *junta* - Aakash, Abhishek, Akshat, Gurneet, Ishu, Kavya, Mohan, Rakesh, Sachin, Sagar, Sanchit, Soham, and Ujjawal enriched my student experience. Abhishek helped me to adjust to Insti-life and has been my go-to person for senior's advice.

My parents Sangeeta Moghe and Vinay Moghe and my sister Ankita Moghe have been my strength and support throughout my life. They always had the right words for every situation and encouraged me to pursue my dreams. My extended family has been encouraging, especially, my cousins Radhika and Rajas, who were enthusiastic to listen to my experiences. Surbhi and Darshan always had a listening ear even though we were miles apart. I would also like to thank my teachers beyond IIT Madras.

I am grateful to the Sabarmati Hostel Office, Neelkesh Caterers, Sharav SFC, Suprabha, CC (First Floor), and indeed the beautiful campus of IIT Madras.

# ABSTRACT

KEYWORDS: Natural Language Processing, Deep Learning, dialogue systems, crowdsourcing, structural information, graph convolutional networks

Existing data-driven dialogue systems treat conversations as a sequence modelling problem (*i.e.*, given a sequence of utterances that generate the response sequence). This is not only an overly simplistic view of conversation but it is also emphatically different from the way humans converse by heavily relying on their background knowledge about the topic. For example, it is common for humans to (involuntarily) produce utterances that are copied or suitably modified from background articles they have read about the topic. To facilitate the development of such natural conversation models that mimic the human process of conversing, we create a new dataset containing movie chats wherein each response is explicitly generated by copying and/or modifying sentences from unstructured background knowledge such as plots, comments, and reviews about the movie. We establish baseline results on this dataset (90K utterances from 9K conversations) using three different models. We observe that none of the baselines incorporate the rich structural information present in the background knowledge such as dependency relations, entity-entity co-occurrence relations, entity co-reference relations. Hence, we further focus on improving one of the generation based models by using structural information present in the background knowledge. Recently, there is a trend of incorporating structural information by using Graph Convolutional Networks (GCNs) to improve performance for several Natural Language Processing tasks. We, thus, investigate different ways to effectively leverage structural information using GCNs along with LSTM based contextual information. From our experimental study, we observe that the performance of using GCN for our task depends on factors such as the combination of structural and contextual information as well as the underlying graph designed for the task. While using GCN is one way to incorporate structural information explicitly, there have been efforts in learning deep contextualized word representations that learn these properties implicitly through unsupervised pre-training on a large amount of unstructured data. We observe that, though architectures using deep contextualized word embeddings achieve state-of-the-art performance, simpler non-contextualized word representation based architectures also achieve good performance on our dataset. Additionally, we find that explicitly incorporating structural information also benefits deep contextualized word embeddings.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# ABBREVIATIONS

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **AMT** | Amazon Mechanical Turk |
| **BiDAF** | BiDirectional Attention Flow |
| **BERT** | Bidirectional Encoder Representations from Transformers |
| **BLEU** | BiLingual Evaluation Understudy |
| **DL** | Deep Learning |
| **DSTC** | Dialogue State Tracking Challenge |
| **ELMo** | Embeddings from Language Models |
| **GCNs** | Graph Convolutional Networks |
| **GloVe** | Global Vectors for Word Representation |
| **GRU** | Gated Recurrent Unit |
| **GTTP** | Get To The Point |
| **HRED** | Hierarchical Recurrent Encoder Decoder |
| **LSTM** | Long Short Term Memory Network |
| **NLG** | Natural Language Generation |
| **NLP** | Natural Language Processing |
| **NLU** | Natural Language Understanding |
| **RNNs** | Recurrent Neural Networks |
| **ROUGE** | Recall-Oriented Understudy for Gisting Evaluation |
| **Seq2Seq** | Sequence-to-Sequence |
| **SSS** | Semantics-Sequences-Structures |

# CHAPTER 1

# Introduction

One of the key traits that separate human intelligence from other animals is our ability to understand and communicate in natural language. Thus, enabling a machine to communicate like human beings is one of the fundamental requirements of any system that exhibits Artificial Intelligence (AI). Beyond this alluring quest, conversational AI has also attracted commercial interest. Personal assistants such as Alexa, Siri, Cortana have gained popularity in carrying out simple tasks like setting up an alarm, creating a music playlist, or providing a weather update by using voice commands. Enterprises have also benefited by deploying troubleshooting chat-bots that handle simple queries, thus enhancing customer engagement. Hence, as a pursuit of the elusive goal or as an effort towards improving user experience, conducting research in conversational AI is indeed rewarding.

A dialogue system is a computer program that can produce meaningful responses to a given sequence of words. The excitement in this field began with Eliza (Weizenbaum, 1966), the psychotherapist bot, which was able to communicate using fixed pattern matching rules. Over the years, the field evolved from the development of systems based on hand-crafted rules to complex pipeline based approaches. With the emergence of data-driven approaches, there has been an active interest in building datasets (Vlad Serban *et al.*, 2015) for training dialog systems. Some of these datasets contain transcripts of human-bot conversations (Williams *et al.*, 2013; Henderson *et al.*, 2014*a*,*b*) while others are created using a fixed set of natural language patterns (Bordes *et al.*, 2017; Dodge *et al.*, 2016). The advent of Deep Learning (DL) created interest in the construction of large-scale dialogue datasets (Lowe *et al.*, 2015*b*; Ritter *et al.*, 2010; Sordoni *et al.*, 2015) leading to the development of several end-to-end conversation systems (Shang *et al.*, 2015; Vinyals and Le, 2015; Serban *et al.*, 2016).

However, most existing large scale datasets (Lowe *et al.*, 2015*b*; Ritter *et al.*, 2010; Serban *et al.*, 2016) simply contain a sequence of utterances and responses without any *explicit* background knowledge associated with them. This has led to the development of models that treat conversation as a simple sequence generation task and often produce output which is both syntactically incorrect and incoherent (off topic). To make conversations more coherent, there is an increasing interest in integrating structured and unstructured knowledge sources with neural conversation models. For example, to have a meaningful conversation about a movie, one uses their knowledge about the plot, reviews, comments and facts about the movie. A typical conversation involves recalling important points from this external knowledge and producing them appropriately in the context of the conversation.

There are already some works in this direction that try to integrate external knowledge resources with existing datasets(Rojas-Barahona *et al.*, 2017; Williams *et al.*, 2016; Lowe *et al.*, 2015*a*; Ghazvininejad *et al.*, 2018). But, we believe that building datasets where the utterances are *explicitly* linked to background knowledge will further facilitate the development of such background aware conversation models. For example, consider the conversation shown in Figure 1.1. To appreciate the performance of Russel Crowe, an actor, Speaker 2 refers to the Review of the movie. Similarly, to understand the opinion of Mae about the main character James Braddock in the fourth utterance, Speaker 2 needs to know about the Plot of the movie. This conver-

**Plot**

. . . At this point James Braddock was a light heavyweight boxer, who was forced to retired from the ring after breaking his hand in his last fight. His wife Mae had prayed for years that he would quit boxing, before becoming permanently injured. . . .

**Review**

. . . Geez, another boxing movie! . . . Russell Crowe owns the character of James Braddock, the unlikely hero who makes the most of his second chance. He's a good fighter turned hack. And then there's Joe Gould, played by Paul Giamatti. . . . The story, while familiar, is executed brilliantly. . . .

**Movie: Cinderella Man**

Speaker 1 (N): Yes very true, this is a real rags to riches story. Russell Crowe was excellent as usual

Speaker 2 (R): Russell Crowe owns the character of James Bradock, the unlikely hero who makes the most of his second chance. He's a good fighter turned hack.

Speaker 1 (N): Totally! Oh by the way do you remember his wife ... how she wished he would stop

Speaker 2 (P): His wife Mae had prayed for years that he would quit boxing, before becoming permanently injured.

Speaker 1 (N): Right. She is in a real dilemma there because she has to choose between her husbands dream and his safety.

Figure 1.1: A sample dialogue which produces responses using background knowledge. The snippets from the background knowledge used in the conversation are shown in blue.

sation strategy which produces responses from background knowledge would be useful in various domains. For example, a troubleshooting bot could exploit the information available in manuals, reviews, and previous bug reports about the software. Similarly, an e-commerce bot could exploit the information available in product descriptions, fact tables, *etc.*, about the product. In this thesis, we will describe the development of one such background aware conversation system for the domain of movies.

A majority of neural conversation systems are based on the Sequence-to-Sequence (Seq2Seq) architecture (Sutskever *et al.*, 2014). Such Seq2Seq architectures primarily rely on Recurrent Neural Networks (RNNs) to capture essential linguistic information. RNNs operate by scanning text in a serial manner and often have difficulties in capturing long-range dependencies. Continuing with the example shown in Figure 1.1, we produce a response in the fourth utterance using a sentence from the Plot. However, while reading the Plot resource sequentially using an RNN, it will have difficulty in retaining the information that "James Braddock", the subject in the sentence, has relation to the word "boxer" and "retired" which occur further in the sentence. Similarly, the RNN may have troubles in understanding that the word "His" in the second sentence refers to the entity "James Braddock" from the first sentence. These word connections are important to generate the given response. Though RNNs and their variants have been shown to implicitly capture linguistic properties to some extent (Linzen *et al.*, 2016; Kuncoro *et al.*, 2018), the explicit modelling of structural information with RNNs has shown improved results on several tasks. These properties within textual units can be represented in terms of graphs such as dependency parse or tree-based structures such as the constituency parse. As a result, there have been developments that account for this incorporation of structural information based on graphs (Marcheggiani and Titov, 2017) or trees (Tai *et al.*, 2015). Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017) have been one of the popular architectures to incorporate linguistic information that can be represented as graphs. GCNs can "teleport" across words/sentences which are far apart from each other along the edges of the underlying graphs (Marcheggiani and Titov, 2017). We use these abilities of GCNs to connect entities across different sentences and identify syntactic neighbours within a sentence. In our chat in Figure 1.1, we can now connect "James Braddock" to "boxer" and "forced" directly using a dependency graph. Similarly, "His" in the second sentence also refers to "James Braddock" from the first sentence. We can retain this information by using a co-reference graph. We hypothesize that using such structural information explicitly helps in finding the relevant pieces of information within the external knowledge to produce meaningful responses.

## 1.1 Contributions

The contributions of this thesis are:

(1) We built a new background knowledge aware movie conversations dataset in English using crowdsourcing. We asked workers to chat about a movie using structured and unstructured resources about the movie such as plots, reviews, comments, and fact tables. The setup for the dataset collection was such that every alternate response in the conversation was explicitly linked to specific background knowledge. We collected around 9K such conversations containing a total of 90K utterances pertaining to 921 movies. While the proposed dataset is domain specific, it serves as a good benchmark for developing creative background knowledge aware models which can then be ported to different domains by building similar datasets for other domains. We establish some initial baselines on this dataset using three different paradigms to demonstrate the various models that can be developed and evaluated using this dataset.

(2) We propose a simple encoder framework titled Semantics-Sequences-Structures (SSS) framework. This combines semantic information through word embeddings, sequential word order information using LSTMs[1], and structural information by using GCNs to improve the representation of background knowledge. We find that the conventional adaptation of GCNs for Natural Language Processing (NLP) tasks, where contextualized embeddings obtained through LSTM are fed as input to a GCN, exhibit poor performance on our dataset. Hence, we investigate better ways of combining structural and sequential information. We exploit word relations within a sentence using the dependency graph and across sentences through entity and co-reference graphs. We also evaluate our method based on the explicit incorporation of structural information with GloVe embeddings (Pennington *et al.*, 2014) against the popular deep contextualized word representations (Peters *et al.*, 2018; Devlin *et al.*, 2019). Such pre-trained language encoder models are hypothesized to capture these properties *implicitly* by virtue

[1]an RNN variant

of the training data, language modelling objective, and neural architecture. We show that though the models using deep contextualized representations are powerful, our explicit addition of structural information significantly boosts their performance resulting in state-of-the-art performance on our dataset.

## 1.2 Outline

We will now give a brief description of the contents of each chapter in this thesis. **Chapter 2:** This chapter contains the fundamental concepts used in this thesis. Anyone with a basic introduction to DL and NLP is free to skip this chapter. We give concise explanations about the terminologies used in DL in section 2.1. We explain the neural architectures used for NLP in section 2.2. We then explain architectures designed to incorporate graph-based structures in section 2.3.

**Chapter 3:** This chapter covers the various directions explored by the research community related to the problems addressed in this work. We discuss popular dialogue datasets in section 3.1. We list the existing approaches that incorporate background knowledge for conversation systems in section 3.2. We identify that our task is closely related to some problems in NLP in section 3.3. Finally, we describe existing methods for incorporating structural information in various NLP tasks in section 3.4.

**Chapter 4:** This chapter describes our dataset creation process in four stages in section 4.1. We evaluate our dataset on three paradigms which are described in detail in section 4.2. We describe our experiments in section 4.3 and comment on the performance in section 4.4

**Chapter 5:** In this chapter, we illustrate the need for incorporating structural information with an example from the dataset. We describe the various existing and proposed architectures that utilize structural information under the `SSS` Framework in section 5.1. We explain the experimental setup in section 5.2 and discuss the results in section 5.3

**Chapter 6:** This chapter summarizes the work done in this thesis. We conclude this chapter with some directions for further exploration.

# CHAPTER 2

# Background

This chapter serves as an introduction to the various terminologies used in this thesis. We introduce the reader to the area of Deep Learning and the application of Deep Learning for NLP tasks. We then demonstrate the importance of structural information in NLP.

## 2.1 Deep Learning Preliminaries

Neural network is a paradigm inspired by the structure and functioning of the animal brain. The first effort in building neural networks can be traced back to the development of McCulloch Pitts neuron (McCulloch and Pitts, 1988) which aggregates the input and the neuron "fires" if this aggregation produces a value above a pre-defined logic. The McCulloch Pitts neuron worked only for Boolean functions which are linearly separable. This model was followed by the development of the perceptron (Rosenblatt, 1958) model which suggested that the inputs need to be combined using "weights" and these weights need to be "learned" using the perceptron learning algorithm.

A single perceptron only had the representation power to accommodate linearly separable functions. To represent more complicated functions, a network of perceptrons was designed. This network consisted of layers of perceptrons which now forms the basis for designing "neural networks".

A neural network takes an input, passes it through several layers of neurons, and then produces an output that is conditioned on the combined input of the neurons. A typical neural network has three layers: input, hidden, and output. A neural network with more than one hidden layer is called a "deep" neural network. The term "Deep Learning" refers to the paradigm that uses these deep neural networks.

The inputs to any neural network can be real-valued and are preferred to lie in the same scale *i.e;* if the input feature $x_1$ lies in the range (1,100) and the input feature $x_2$

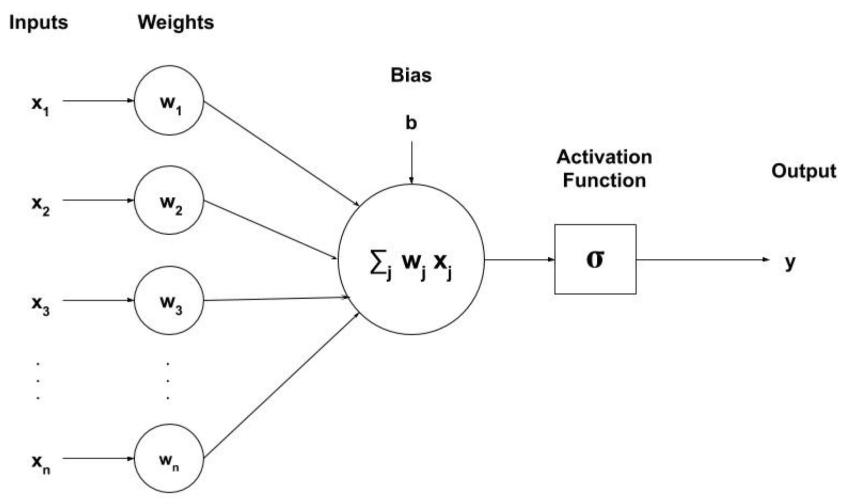


Figure 2.1: The components of a neuron.

lies in the range (1,1000), they need to be scaled appropriately before feeding into the network.

The weights are connections from one layer of neurons to the other layer of neurons. They also include connections between the input features and the first layer of neurons. They are represented using matrices: $W_{i,i+1} \in \mathbb{R}^{n \times k}$ where n is the number of neurons in the $i^{th}$ layer and k is the number of neurons in the $i + 1^{th}$ layer. We will refer to weights using superscripts and subscripts along with the letter *W* unless specified otherwise.

Sometimes, a "bias" vector is also added to this weighted representation. The role of the bias vector is to represent patterns that do not necessarily pass through the origin. For example, consider a case where all the input features are zero. Just as the weight matrices, the bias vectors are also "learned". We will refer to bias vectors using subscripts and superscripts along with the letter *b*. This weighted combination of input features and subsequent addition of the bias vector is referred to as "pre-activation".

An activation function modifies this pre-activation. In most cases, the activation functions are non-linear and act as key components in learning the complex mapping of the input data to the output. Several variants of activation functions have been proposed in the literature of which only the ones used in producing this work are explained below:

**(1) sigmoid:** The sigmoid is a non-linear activation function that produces output in the range (0,1). They give the probability of a neuron "firing" given the inputs. The sigmoid function is formulated as follows and will be referred using $\sigma$.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

**(2) tanh:** The tanh function is another non-linear activation function that produces outputs in the range (-1,1). In fact, it is a scaled version of the sigmoid function.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.2}$$

**(3) ReLU:** Rectifier Linear Unit (ReLU) takes an input and outputs the value directly if it is greater than 0 or returns a 0, i,e, it has a threshold at 0.

$$f(x) = max(0, x) \tag{2.3}$$

**(4) softmax**: The softmax activation function takes a K-dimensional input and provides a normalized K-dimensional output which gives a probability distribution. Larger values correspond to larger probabilities.

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{2.4}$$

The choice of using a particular activation function is determined by the problem at hand. The sigmoid seems a natural choice for binary classification problems while the softmax is used in multi-class classification problems. The use of ReLU promotes faster training in larger networks.

All the basic building components of neural networks can be seen in Figure 2.1.

Neural networks need to be trained so that we learn the weights which learn the complex mapping of inputs to the outputs. Consider a training set $\{(X_i, Y_i)_{i=1}^{N}\}$ where $X$ is the input and $Y$ is the output, $i$ refers to individual training samples and $N$ is the total number of samples. $X$ can be a $k$-dimensional input where $k$ is the number of input features. The task for any neural network is to map the input $X$ to $Y$ by learning the parameters $\theta$, the weights and bias vectors, of the neural network. The goodness of fit for $\hat{Y} = f(X)$ where $f(X)$ is our neural network is given by a loss function. The choice of the loss function is dependent on the problem. Some of the popular loss functions are:

(1) Mean Squared error $= \frac{1}{N}\sum_i (Y_i - \hat{Y_i})^2$. This is used when $Y$ is real-valued. (2) Cross entropy loss $= \sum_i - \sum_{c=1}^{M} Y_{i,c}\, log(\hat{Y_{i,c}})$. This is used for classification tasks with $M$ classes.

The passing of input features through the neural network and obtaining $\hat{Y}$ is known as the "forward pass". We then identify our prediction error using a loss function. The objective is to make $\hat{Y}$ as close to the target $Y$ as possible. Essentially, we want to learn the parameters that minimize the loss over the training data. This is done using gradient descent where the gradient of the loss function is calculated with respect to the parameters and the parameters are updated based on the gradient value. The training using gradient descent should guide the parameters to reach the local minima of the loss function. Refer to equation 2.5 which is the parameter update equation.

$$\theta_{new} \leftarrow \theta_{old} - \eta \nabla \theta_{old} \tag{2.5}$$

$\eta$ is the "learning rate". The learning rate decides the size of the steps taken in order to reach the local minima. This process continues in an iterative manner and we stop the training based on pre-defined stopping criteria.

If the entire training data is used to update the parameters, it is referred to as batch gradient descent. If the parameters are updated per example, it is called as stochastic gradient descent. As a best-of-both-worlds approach, in most cases, mini-batch gradient descent is used which updates the parameters every $n$ training examples where $n$ is known as the "batch size".

The mini-batch gradient descent also has several challenges associated with it. First, it introduces the batch size hyper-parameter. The trade-off between the right learning rate and the batch size needs to be identified. All the parameters are updated using the same learning rate. Thus, rarely occurring features and frequently occurring features get updated using the same rate. To overcome these drawbacks, variants over the mini-batch update equation have been proposed. Most of the work in this thesis uses the Adam (Kingma and Ba, 2015) update or the Adadelta (Zeiler, 2012) update.

In all the above variants of optimization algorithms, the backpropagation algorithm (Rumelhart *et al.*, 1986) is used to compute the gradients using the chain rule. In the forward pass, we compute the output at each layer using the input from the previous layer. The error is computed at the last layer and propagated backward through the network layers. The gradient is computed with respect to the error at the output layer and the gradients are backpropagated till the input layer using the chain rule of derivatives.

The training of neural networks is sensitive to many of the above components. We refer to these components as hyper-parameters. These hyper-parameters include dimensions and initialization techniques of the weights and the biases, number of layers, batch size, choice of optimization algorithms, learning rate, activation functions, and so on.

## 2.2 Deep Learning for NLP

Deep Learning has been widely used for several NLP tasks. The first reason for the popularity of using deep learning for NLP is the power of learning features. Traditional NLP techniques have relied a lot on using human prior knowledge through manually designing features for improving performance on a task. For example, to classify whether a document belongs to the 'sports'or 'economics'category, we need to curate a list of distinct words that appear either in sports or economics. DL techniques hope to learn these distinguishing features directly from raw textual data. Secondly, DL can provide hierarchical representations of the raw text. This is important as natural language inherently exhibits a compositional nature: characters form words, words form concepts, and so on. Lastly, unsupervised DL techniques can benefit from the availability of a large amount of textual data.

We now look at the basic components used to build neural NLP models.

### 2.2.1 Word Embeddings

As seen before, neural networks require real-valued inputs. How do we convert words from a language into numbers? A simple way would be to create a fixed length vector of the size of vocabulary $V$ and assign "one-hot" embedding for every word. Thus, for any word $w_k$, the $k^{th}$ component in the vector will be $1$ and the remaining components will be $0$. This method cannot capture the similarity between two words as every word differs from another word only in terms of a single bit.

To overcome this, a family of techniques that map (embed) words to continuous space where semantically similar words are assigned points which are close to each other. These techniques rely on the distributional hypothesis: words that appear in similar contexts share similar semantic meaning. They can be count-based methods (Latent Semantic Analysis (Deerwester *et al.*, 1990)), prediction based methods (word2vec (Mikolov *et al.*, 2013)) or a hybrid of both the techniques (GloVe embeddings (Pennington *et al.*, 2014)). Refer to Figure 2.2 to understand the word embedding space.[1] We observe that certain directions in the vector space actually give interesting semantic relationship between the words such as gender, verb-tense, or country-capital relationship.

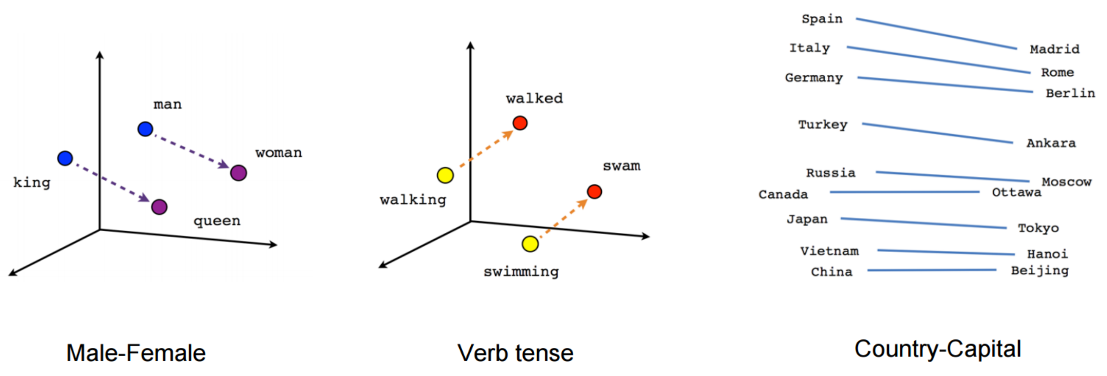


Figure 2.2: Illustration for the word embedding space.

### 2.2.2 Recurrent Neural Networks

In the previous section, we mention techniques that convert words to vectors. However, most of the problems in NLP use linguistic units larger than words such as phrases, sentences, and documents. In order to obtain representations for these larger textual units, Recurrent Neural Networks (RNNs) are used. RNN is a family of neural networks that are used for problems that deal with sequential data. They were initially proposed to solve problems on time series data (Elman, 1990) but have been extensively used in neural NLP techniques (Vinyals and Le, 2015; Bahdanau *et al.*, 2015; Seo *et al.*, 2017; Gu *et al.*, 2016).

Consider a sequence $\{w_1, w_2, \ldots, w_N\}$ where each token $w_t$, $t \in [1, N]$ is mapped to $x_t$. $x_t$ can be a word embedding or obtained from any other function that gives a fixed

[1] `https://www.tensorflow.org/images/linear-relationships.png`

length vector. RNN provides a representation for every token as $h_t$ which is dependent on representation of the previous token $h_{t-1}$ and the current input $x_t$. Conceptually, $h_t$ captures information of the sequence till the current time-step. Thus, conceptually, $h_N$ can be considered as the representation of the entire sequence.

$$h_t = g(h_{t-1}, x_t) \tag{2.6}$$

The choice of g typically is a feed forward neural network as shown below:

$$g(h_{t-1}, x_t) = \sigma(W h_{t-1} + U x_t + b) \tag{2.7}$$

where, $\sigma$ is the sigmoid activation function, $x_i \in \mathbb{R}^n$ (n-dimensional input), $h_i \in \mathbb{R}^d$ (d-dimensional state), $U \in \mathbb{R}^{n \times d}$ , $W \in \mathbb{R}^{d \times d}$, $b \in \mathbb{R}^d$. The parameters $U$, $W$, and $b$ are shared across time steps. The network is trained using backpropagation through time (Werbos, 1990).

### Long Short Term Memory Networks

Vanilla RNNs shown in equation 2.6 suffer from the problems of vanishing and exploding gradients. In the case of vanishing gradients, the values of gradients diminish exponentially (reaching zero) and in the case of exploding gradients, the gradients become very large when the error is backpropagated through several time-steps (Pascanu *et al.*, 2013). Thus, they are ineffective in capturing long-range dependencies. As a solution for the vanishing gradients problem in RNNs, Long Short Term Memory Networks (LSTMs) were proposed which consist of three gates namely the *input* ($i_t$), *forget* ($f_t$), and *output* ($o_t$) gate, that effectively regulate the information flow across the time steps. It also has a memory unit ($c_t$) to compute an intermediate representation. Finally,

$h_t$ is computed as follows:

$$\begin{aligned}
i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\
f_t &= \sigma(W_f h_{t-1} + U_f x_t + b_f) \\
\tilde{c}_t &= \sigma(W_c h_{t-1} + U_c x_t + b_c) \\
c_t &= i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \\
o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\
h_t &= o_t \, tanh(c_t)
\end{aligned} \tag{2.8}$$

The gates have same dimensions as that of the hidden state ($d-$dimensional), All the $U$, $W$, and $b$ matrices have $\mathbb{R}^{n\times d}$, $\mathbb{R}^{d\times d}$, $\mathbb{R}^{d}$ dimensions respectively.

#### Gated Recurrent Units

Gated Recurrent Unit (GRU) is a variant of the LSTM architecture. Instead of having three different gates as explained in LSTMs, the input gate and the forget gate are coupled as one gate. This architecture also removes the necessity to maintain a separate memory unit ($c_t$).

$$\begin{aligned}
i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\
o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\
\tilde{c}_t &= \sigma(W_c (o_t \odot h_{t-1}) + U_c x_t + b_c) \\
h_t &= i_t \odot \tilde{c}_t + (1 - i_t) \odot h_{t-1}
\end{aligned} \tag{2.9}$$

The components in GRU have the same dimensions as explained in the LSTM equation (2.8)

All these variants of RNNs also have their corresponding Bidirectional RNNs (BiRNNs) configurations. In case of BiRNN, two RNNs are considered which process the text left-to-right and right-to-left respectively. The hidden states produced by them are concatenated.

### 2.2.3 Sequence-to-Sequence Models

Many problems in NLP such as Machine Translation, Transliteration, Abstractive Summarization require generation of sequence of words which are based on a sequence of input words. For example, in machine translation, the task is to convert a sequence of words in some language to a sequence of words in another language. All of these problems essentially involve Language Modelling as a sub problem. Language Modelling is a task of predicting the $k^{th}$ word given previous $k-1$ words. Thus, the above tasks require a modified language modelling problem where it is conditioned on an input sequence of tokens as well as the previous $k-1$ words. More formally, given an input sequence of tokens $\{x_1, x_2, \ldots, x_{N_x}\}$, the task is to produce an output sequence of tokens $\{y_1, y_2, \ldots, y_{N_y}\}$. The tokens are predicted sequentially based on the probability distribution as follows:

$$p(y|x) = \prod_{t=1}^{N_y} p(y_t|y_1, y_2, \ldots, y_{t-1}, x) \tag{2.10}$$

Typically, a softmax function, conditioned on the input sequence and the previously generated tokens acts as the probability distribution.

The Sequence-to-Sequence (Seq2Seq) models (Sutskever *et al.*, 2014) are typically used for the problems which follow the above setup. Following are the important components of a Seq2Seq model:

#### Encoder

The encoder is used to obtain the representation of the input sequence using an RNN-based architecture. Layers of RNN can also be stacked together and the outputs from the last layer are used as the token representations. The $h_t$ produced per token are referred to as the "encoder states".

#### Decoder

The decoder is also an RNN based architecture that predicts the probability of output sequence conditioned on the final encoder state $(h_{x_N})$. The final encoder state acts as the first state of the decoder. Every decoder RNN state $s_t$ is dependent on the previous decoder state $s_{t-1}$ and the previous predicted token $\hat{y}_{t-1}$. Then, the prediction distribution is computed by softmax over the vocabulary by passing the new decoder state $s_t$ through a feed-forward neural network. The weights of this network are shared across

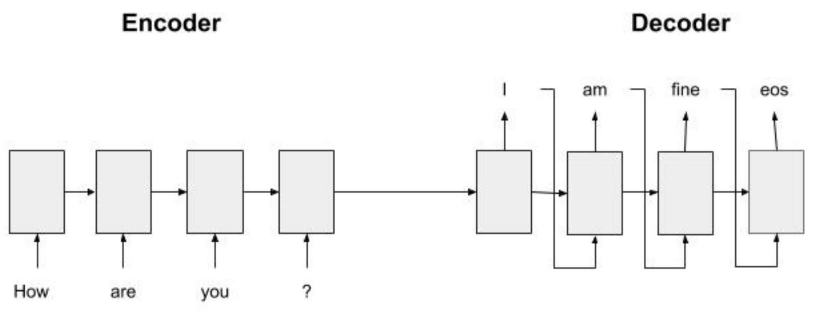


Figure 2.3: Illustration of the Seq2Seq architecture.

time-steps. This is summarized in equation 2.11.

$$\begin{aligned} s_t &= \sigma(W_d s_{t-1} + U_d \hat{y}_{t-1} + b_d) \\ \hat{y}_t &= softmax(W_s s_t + b_s) \end{aligned} \tag{2.11}$$

$\sigma$ is the sigmoid activation function. $y_i \in \mathbb{R}^V$, $s_i \in \mathbb{R}^d$ (d-dimensional state), $U_d \in \mathbb{R}^{d \times V}$, $W_d \in \mathbb{R}^{d \times d}$, $W_s \in \mathbb{R}^{V \times d}$ where $V$ is the size of the vocabulary. Please refer to Figure 2.3 for an illustration. The training objective is to minimize the negative log-likelihood of predicting each word at every time step. This is also known as a sequence loss and is defined as follows:

$$J = -\sum_{t=1}^{N_y} logp(y_t | y_1, y_2, \ldots, y_{t-1}, x) \tag{2.12}$$

During test time, every decoder state predicts a word using a decoding technique and the tokens are considered till a special marker such as <EOS> is predicted. The two popular decoding techniques to predict the word from the vocabulary distribution are as follows:

**Greedy Decoding**: During test time, at every time step, the model chooses the word with the largest probability. This predicted word is then fed as input to the next decoder step.

**Beam Search Decoding**: In beam search decoding, top-K hypotheses, where K is the beam width, are maintained at every time-step. These K hypotheses are fed into the next state of the decoder and thus providing K $\times$ K sequences. From these K $\times$ K sequences, only the top scoring K sequences are retained for further predictions. This process continues until the special marker is predicted.

### 2.2.4 Attention Mechanisms

In the previous section, we observe that only the first state of the decoder is conditioned on the last hidden state of the encoder which is supposed to capture the entire representation of the textual input. The subsequent states of the decoder are conditioned on the previous decoder state by the virtue of RNN and hence, there is no direct connection of the encoder representations on the remaining decoder states. However, while generating individual target words, it would be essential to look back at some of the hidden states *i.e.* we would want to pay "attention" to the hidden states. The $\alpha$ defined in equation 2.13 gives the relative importance of the encoder states.

$$\begin{aligned} \alpha_t^i &= \frac{exp(score(h_i, s_{t-1}))}{\sum_j exp(score(h_j, s_{t-1}))} \\ c_t &= \sum_{i=1}^{N_x} \alpha_t^i h_i \end{aligned} \tag{2.13}$$

There are variants of the scoring functions proposed in the literature. The method described in Bahdanau *et al.* (2015) uses the following scoring function:

$$score(h_i, s_{t-1}) = V_a^T tanh(W_a h_i + U_a s_{t-1} + b_a) \tag{2.14}$$

This attention vector $c_t$ is unique per time-step. The decoder is now conditioned on this attention vector and the new update equation is:

$$\begin{aligned} s_t &= \sigma(W_d s_{t_1} + U_d \hat{y}_{t-1} + V_d c_t + b_d) \\ \hat{y}_t &= softmax(W_s s_t + b_s) \end{aligned} \tag{2.15}$$

$V_d$ lies in $\mathbb{R}^{d \times d}$.

The variants also include self-attention (Wang *et al.* (2017)) where every word in a contiguous piece of text such as a document is computed as a weighted representation of all the words in the document, cross-attention (Seo *et al.* (2017); Xiong *et al.* (2017)) where every word in the text such as a document is computed as a weighted representation of influence of individual words in a different piece of text such as query. The cross-attention is referred to as m-aware-n attention where *n* is a text sequence whose representation is influenced by the representation of the words from the text sequence *m*. The m-aware-n attention can be extended to different modalities. Although the attention mechanism is explained in the context of the encoder-decoder paradigm, it can be applied to non-NLG tasks as well.

## 2.3 Graph-based Neural Architectures

The architectures explained in the previous subsection 2.2.2 can only process the textual data sequentially *i.e;* either from right-to-left or left-to-right. However, natural language can be sequential or compositional or even without any ordering information. A word in a particular sentence/paragraph/document may not be dependent on the immediate words surrounding it but can depend on a word/concept that occurs much earlier or later in a sentence/paragraph/document.

Every language has a set of grammar rules that govern the structure of a sentence including the word order. This is termed as syntax. For example, "The cat sat on the mat." is acceptable than "cat The on mat on sat the". Dependency grammar describes the syntactic structure of the sentence in terms of a graph where the nodes are individual words in a sentence and the edges are binary, asymmetric and labelled. See Figure 2.4 for an example. Thus, dependency graph structures can provide this explicit connection across words in a sentence that does not appear sequentially. Similarly, constituency

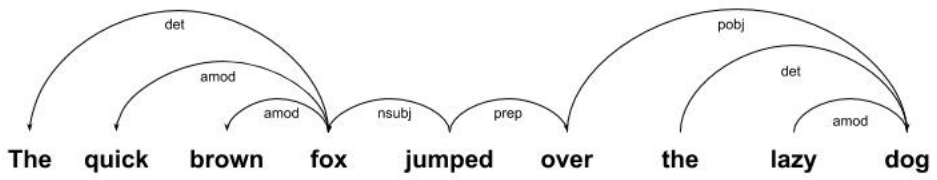


Figure 2.4: Dependency parse for the sentence *The quick brown fox jumps over the lazy fox*.

grammar describes the syntactic structure by constructing a tree where the non-terminal nodes are phrases and the terminal nodes are words.

This non-sequential information is also present at the discourse level. Consider the statements, "Sita has a dog. She likes it very much." In this example, "she" and "it" corresponds to "Sita" and "dog" respectively from the previous sentence. During the course of long documents, such connections may occur beyond consecutive sentences as well. Thus, a technique known as co-reference resolution is used which finds such expressions that refer to the same entity.

These are some of the examples of structural information present in NLP. We limit our discussion only to structural information that can be represented in terms of graphs in this work.

### 2.3.1 Graph Convolutional Networks

As described in the previous section, structural information is ubiquitous in NLP and as RNNs are not designed for representing such structured data, we resort to techniques from network representation learning. Specifically, we discuss the adaption of GCN (Kipf and Welling, 2017), a neural network that provides the representation of a node in terms of its k-th ordered neighbourhood, for NLP tasks.

A GCN takes in two inputs; a feature input matrix and an adjacency matrix that represents the underlying graph structure. Consider an undirected graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of $n$ nodes, and $\mathcal{E}$ is a set of edges. Every node can also be connected to itself, i.e. $\forall v \in \mathcal{V} : (v, v) \in \mathcal{E}$. The graph structure is contained in the adjacency matrix $A$. Now, let $X \in \mathbb{R}^{d\times n}$ be a matrix containing all $n$ nodes with their features.

$$h_v = \rho\left(\sum_{u\in\mathcal{N}(v)} W x_u + b\right) \tag{2.16}$$

where $W \in \mathbb{R}^{d\times d}$ is a weight matrix, $b \in \mathbb{R}^d$ is a bias vector and $\rho$ is an activation function, e.g. ReLU. $\mathcal{N}(v)$ is the set of neighbors of $v$. The $h_v$ term is the representation of the node $v$ in terms of its neighbours. Similar to the standard convolutional networks, we can incorporate higher degree neighbourhood in GCNs.

$$h_v^{(k+1)} = \rho\left(\sum_{u\in\mathcal{N}(v)} W^{(k)} h_u^{(k)} + b^{(k)}\right) \tag{2.17}$$

where $k$ is the index of the layer, and $h_v^{(0)} = x_v$, the input features to the GCN. Please note this adaptation does not include the normalization component as discussed in the original implementation(Kipf and Welling, 2017).

### 2.3.2 Labelled and Directed Graph Convolutional Networks

The wide acceptance of GCNs for NLP tasks can be attributed to the introduction of labels and directions in the existing GCN equation. When a dependency parse structure is used as the underlying graph with labels being the grammatical relations amongst the two connected words, this modification is commonly referred to as Syntactic Graph Convolutional Networks (Syntactic GCNs)(Marcheggiani and Titov, 2017). The input features to the Syntactic GCN can be the word embeddings or contextual representations of the input text obtained from any RNN based layer.

$$h_v^{(k+1)} = \rho\left(\sum_{u \in \mathcal{N}(v)} W_{L(u,v)}^{(k)}\, h_u^{(k)} + b_{L(u,v)}^{(k)}\right) \tag{2.18}$$

Unlike equation 2.17, the Syntactic GCN equation 2.18 has weights and biases which are specific to individual labels. As suggested in Marcheggiani and Titov (2017), to avoid over-parameterization, the weight matrices are shared across directions.

$$W_{L(u,v)}^{(k)} = V_{dir(u,v)}^{(k)} \tag{2.19}$$

Three directions are considered: (1) incoming edge in the dependency parse (2) outgoing edge in the dependency parse, and (3) self-loop.

#### Edge-Gating

As Syntactic GCNs rely on computing the representation of individual words based on the representation of its neighbouring words, it is important to weigh this information than accepting information from every neighbouring word equally. Hence, edge-gating is introduced which computes a score for every edge-node pair. Moreover, GCNs highly depend on a correct dependency parse while existing parsing algorithms are still far from perfect. Thus, the use of edge gating also helps in reducing the noise from such incorrect parses.

$$g_{(u,v)}^{(k)} = \rho\left(h_{u,v}^{(k)} \cdot \hat{v}_{dir(u,v)}^{(k)} + \hat{b}_{L(u,v)}^{(k)}\right) \tag{2.20}$$

where $\rho$ is the sigmoid function, and $\hat{v}^{(k)}_{dir(u,v)} \in R^d$ and $\hat{b}^{(k)}_{L(u,v)} \in R$ are learned parameters for the gate. After adding the edge-gating component in Syntactic GCNs, we obtain the following GCN equation which is used in the remainder of this work.

$$h_v^{(k+1)} = \rho\left( \sum_{u \in \mathcal{N}(v)} g^{(k)}_{(u,v)} (W^{(k)}_{dir(u,v)} \, h_u^{(k)} + b^{(k)}_{L(u,v)}) \right) \tag{2.21}$$

## Summary

In this chapter, we gave concise explanations of the concepts used in DL. We described the basic building blocks of any neural NLP architecture and the popular Seq2Seq architecture. We pointed out that as the RNN architecture processes language sequentially, it has some difficulties in capturing important linguistic properties. Hence, we discussed the development of Syntactic GCNs, which can explicitly model these important properties using graphs along with the contextual information obtained from RNNs.

# CHAPTER 3

# Related Work

In this chapter, we conduct a literature survey leading to the development of our work. We first give an overview of data-driven conversation systems followed by the a discussion on dialogue systems that rely on external information. We then compare the task in this thesis to other NLP tasks and finally, we emphasize on the importance of structural information in NLP.

## 3.1 Data-driven Conversation Systems

Conversation systems have been of interest to the community for a very long time starting with early rule-based systems such as (Weizenbaum, 1966; Colby, 1981) which used a fixed set of rules for matching patterns and rephrasing the input to generate responses. These systems required domain experts to come up with an exhaustive set of rules and hence it was difficult to scale them to multiple domains. This eventually led to data-driven approaches (Young, 2000; Williams and Young, 2007; Metallinou *et al.*, 2013) which decomposed the task into four stages, *viz.*, Natural Language Interpreter, Dialogue State Tracker, Dialogue Response Selector, and Natural Language Generator. One limitation of these approaches was that they required annotations for the intermediate stages which was indeed expensive. With the advent of deep learning, this pipeline process was replaced by an end-to-end system, which learns directly from utterance response pairs without any stage-wise annotation. Thus, there has been an active interest in the community to build corpora for training dialogue systems.

Vlad Serban *et al.* (2015) maintain an excellent and up-to-date survey on the corpora collected for building data-driven dialogue systems that can be organized into different categories based on several criteria. The first category is whether the dataset contains goal-oriented conversations or non-goal-oriented conversations. A goal-oriented conversation typically involves chatting for the sake of completing a task such as the Dialog

State Tracking Challenge (DSTC) datasets which involve tasks for reserving a restaurant (Henderson *et al.*, 2014*a*), checking bus schedules (Williams *et al.*, 2013), collecting tourist information (Henderson *et al.*, 2014*b*) and so on (Eric *et al.*, 2017; Asri *et al.*, 2017). Non-goal-oriented conversations are generally categorized as "chit-chat" but can also be informative. Some examples of these "chit-chat" datasets are the Ritel Corpus (Rosset and Petel, 2006), NPS Chat Corpus (Forsythand and Martell, 2007), Twitter Corpus (Ritter *et al.*, 2010), *etc*. The second criteria is whether the dataset is open-domain (Ritter *et al.*, 2010; Forsythand and Martell, 2007; Dinan *et al.*, 2019) or domain specific *i.e,* conversations revolve around only a particular topic. Most of the goal-oriented conversation systems are domain specific (Henderson *et al.*, 2014*a*; Asri *et al.*, 2017; Lowe *et al.*, 2015*b*; Uthus and Aha, 2013) whereas the datasets introduced by (Dodge *et al.*, 2016; Banchs, 2012; Zhou *et al.*, 2018) contain conversations only about movies only. Similarly, the Ubuntu Corpus(Lowe *et al.*, 2015*b*) or the IT-troubleshooting corpus in (Vinyals and Le, 2015) are limited to troubleshooting conversations. The third criteria is whether the dataset contains human-human conversations or human-bot conversations. The human-bot conversation datasets contain conversations between humans and an existing conversation system (Williams *et al.*, 2013; Henderson *et al.*, 2014*a*,*b*). Human-human conversations contain spontaneous conversations between humans, as are typically observed in discussion forums (Walker *et al.*, 2012), chat rooms (Forsythand and Martell, 2007), SMS messages (Chen and Kan, 2013) and so on.

We now mention some of the popular data-driven conversation systems built using the above datasets and refer the reader to Vlad Serban *et al.* (2015) for a review of the field.

Seq2Seq which contains an RNN based encoder and an RNN based decoder has been the most popular framework to develop neural dialogue systems where the response is generated as a sequence of words (Ritter *et al.*, 2011; Sordoni *et al.*, 2015; Shang *et al.*, 2015; Vinyals and Le, 2015). The Seq2Seq architecture has been suitably modified depending on the dataset and the task. In terms of architectural improvement, Seq2Seq has been modified to introduce a hierarchical variant (Serban *et al.*, 2016), a hierarchical variant using latent variables to improve the diversity of responses (Serban *et al.*, 2017*c*). Further, Li *et al.* (2016*a*) improved the loss function to encourage diverse responses while Li *et al.* (2016*c*) used reinforcement learning to avoid getting stuck in common responses such as "I don't know". In the Persona-based model (Li *et al.*, 2016*b*), a persona vector was added to a vanilla Seq2Seq model for making responses dependent on a particular user profile. In the MrRNN model (Serban *et al.*, 2017*a*), an additional layer was incorporated in the architecture that improves the topicality of responses.

In the next section, we will focus on the dialogue systems which are closely related to our problem.

## 3.2 Conversation Systems using External Knowledge

There is an increasing effort in integrating external knowledge with the existing models. This is because human beings rely on using background knowledge for conversations as well as other tasks (Schallert, 2002). One of the motivations to incorporate external knowledge in conversation systems is to improve the informativeness of the response. In the following subsections, we discuss some works in the literature which use external knowledge for both goal-oriented and non-goal-oriented systems.

### 3.2.1 Goal-oriented conversation systems

There has been an interest in using external knowledge for improving dialogue systems even before the advent of neural conversation systems. Consider *Let's Go!* (Raux *et al.*, 2005), a spoken dialogue system that provides bus schedules and route information by using bus timetables as external knowledge. Similarly, the *MIT ATIS* system (Seneff *et al.*, 1991) used a pipeline procedure to convert natural language statements to SQL queries to query the OAG database.

Over the years, the Dialogue State Tracking Challenge (DSTC) community has been curating goal-oriented datasets with a focus on the dialogue state tracker component of the dialogue manager. The first DSTC challenge (Williams *et al.*, 2013) used data from *Let's Go!* and used the bus time table as external knowledge. The second and the third challenge (Henderson *et al.*, 2014*a*,*b*) used human-computer chats in the restaurant domain and tourism domain. The modified DSTC 2 dataset (Bordes *et al.*, 2017) converts the DSTC 2 dataset into a sequence of utterance-response pairs to promote end-to-end learning of dialogue systems. They also added a Knowledge Base (KB) and appended the resultant KB triples to each dialogue. They use an API call that uses the information of all the constraints specified by the user so far and then receives all triples from the restaurant KB which match the user's requirements. The work in Rojas-Barahona *et al.* (2017) is also a variant of the DSTC 2 dataset where the conversations are crowdsourced using the Wizard-of-Oz technique where the wizard has access to the restaurant database. Banerjee *et al.* (2018) convert the modified DSTC 2 dataset (Bordes *et al.*, 2017) into four code-mixed languages to facilitate the development of code-mixed conversation systems in multilingual regions.

The Maluuba Frames Corpus (Asri *et al.*, 2017) was also collected using the Wizard-of-Oz technique, and the task was to book a trip for a user given some constraints. The "bots" had access to a database of vacation packages containing round-trip flights and a hotel. The In-Car dataset (Eric *et al.*, 2017) is a multi-turn, multi-domain dataset with three end tasks - calendar scheduling, weather information retrieval, and point-of-interest navigation. Each of these tasks was supported by their respective knowledge bases.

There have been instances of augmenting the background knowledge after the dataset has been collected. For example, the work in (Lowe *et al.*, 2015*a*) uses Ubuntu Manual pages as external information on the Ubuntu dataset (Lowe *et al.*, 2015*b*). They use information retrieval techniques to first obtain the relevant manual page according to the context of the conversation and then they condition the response generation on the history of the conversation and background information.

### 3.2.2 Non-goal-oriented conversation systems

In non-goal-oriented neural conversation systems, the work in Ghazvininejad *et al.* (2018) can be considered as the initial work in incorporating external knowledge. They used 1.1M FourSquare Tips as external knowledge for the 23M general-domain conversations from Twitter. They collect these Tips or Facts, which are comments left by customers about restaurants and commercial establishments and use the entities within these comments as the keys for retrieval of these facts. The training data now consists of the context of the conversation, the associated facts, and the task is to then generate the appropriate response. They improve the response generation of the Seq2Seq model by adding a "Facts" encoder. The "Facts" encoder uses a soft attention mechanism over all the facts in the training sample, and the decoder is now conditioned on this attention vector and the previous context of the conversation to produce the response.

The entries in the annual Alexa Prize Contest (Ram *et al.*, 2017; Khatri *et al.*, 2018) used background knowledge for improving informativeness in their responses. From the 2017 submissions, Milabot (Serban *et al.*, 2017*b*) and even the winning entry SoundingBoard (Fang *et al.*, 2018) used Reddit pages, Amazon's Evi Service, and large databases like OMDB, Google Knowledge Graph, and Wikidata as external knowledge. The submission named Eigen (Guss *et al.*, 2017) used several dialogue datasets and corpora belonging to related Natural Language Processing tasks to make their responses more informative. The teams from the 2018 Challenge used open sourced datasets or APIs for their response modules such as Reddit, News API, EVI, Wikidata, IMDB, ESPN, Washington Post, DuckDuckGo, Rotten Tomatoes, Spotify, and Bing. Knowledge graphs were also used by the submissions from the teams Gunrock, Iris, Slugbot, Fantom, and Alana, to either add information to the existing context of the conversation or traverse along the path in the knowledge graph to provide information about related interesting entities (Khatri *et al.*, 2018).

Concurrently with our dataset, Zhou *et al.* (2018) released the Document Grounded Conversations dataset which also uses background knowledge for movie conversations. The background knowledge is limited only to the Wikipedia page and the number of movies to 30. Their chats include setups, where either one of the workers had access to the background knowledge or both of them had access to the Wikipedia page. They evaluate their dataset in two settings: (1) using the Seq2Seq model with attention and copy-mechanism without the background knowledge and (2) by adding an encoder in the Seq2Seq model to obtain the representation for background knowledge.

Facebook AI Research (FAIR) also released Wizard of Wikipedia, a knowledge-powered conversations dataset with 1365 diverse conversation topics (open-domain). The background knowledge here is also Wikipedia pages. They follow a wizard-apprentice setup, where the apprentice is encouraged to ask the wizard about a particular topic, and the wizard is shown the relevant information to create the response. The baselines for this task use the following two paradigms: (1) retrieval models that produce an output among a set of candidate responses using an extension of Memory Networks (Sukhbaatar *et al.*, 2015) and (2) generation based models using the Transformer architecture (Vaswani *et al.*, 2017).

## 3.3 Related NLP Problems

In section 4.2, we use baseline methods from tasks beyond neural conversation system. In this section, we describe these tasks that are closely related to our work and the approaches to solve them.

### 3.3.1 Reading Comprehension

Our problem is partly analogous to the problem of reading comprehension. The task of reading comprehension involves answering the question given a resource passage. In our work, we construct a response (answer) based on the previous history of the conversation (question) and using appropriate background knowledge (passage). Though this field is rapidly advancing, we will restrict ourselves only to the literature up to the development of the BiDAF model (Seo *et al.*, 2017).

The interest in the development of neural reading comprehension systems began with the availability of large scale reading comprehension datasets (Rajpurkar *et al.*, 2016; Hermann *et al.*, 2015; Onishi *et al.*, 2016; Hill *et al.*, 2016; Nguyen *et al.*, 2016; Joshi *et al.*, 2017). Of the various reading comprehension models that have been developed in the last few years, the models based on SQuAD (Rajpurkar *et al.*, 2016) and MS MARCO(Nguyen *et al.*, 2016) are most relevant to us as these require the answer to be a span of contiguous words present in the document just as our responses are a span of contiguous words from the background resource.

We list some of the popular works in Table 3.1 and describe the common layers within these architectures below.

### Encoder Layer

This layer provides representation for the query and the document. The first step is to construct an embedding for the individual words in the query and passage using the following procedure:

**(1) Obtaining character-level embeddings**: Here an embedding for the word is obtained by combining the embeddings for the characters in the word. The character-level embeddings are typically used to solve the out-of-vocabulary problem for pre-trained word embeddings or to obtain similar representations for words that share the same lemma *e.g;* plays and play. Most of the works pass the characters from individual tokens through an RNN or a Convolutional Neural Network (CNN) like the work in Kim (2014).

**(2) Obtaining word-level embeddings**: The models obtain the representation of every word by doing a look-up of the pre-trained word embeddings. As the works in Table 3.1 rely only on GloVe embeddings (Pennington *et al.*, 2014), we do not maintain a separate column for the same.

**(3) Obtaining other word features**: To refine the word embeddings further, certain features such as Parts-of-Speech tags, Named Entity Recognition tags, frequency are also added to the pre-trained embeddings. In Table 3.1, only the Fine-Grained gating architecture uses word features.

**(4) Fusing all the embeddings:** In most of the works, the character-level and word-level embeddings are merged together by using the concatenation operation. In the case of the BiDAF model, after the concatenation operation, these embeddings are passed through Highway connections (Srivastava *et al.*, 2015). Some works also use fine-grained gating which obtains a weighted representation of the character-level and word-level embeddings.

Finally, these token/word representations are passed through some variant of RNN to obtain their contextual representations.

### Interaction Layer

The Encoder Layer obtains refined representations of the query and the document individually. To incorporate query information into the document representation, an Interaction Layer is used. This infusion of information is necessary to improve the answer prediction. We divide the Interaction Layer into two layers: Cross-Attention Layer and Merging Layer. The various methods that are used in these layers are described in this section.

Note that, the use of Interaction Layer is computationally expensive. We discuss the limitations of such models using such expensive layers in section 4.3.

**Cross-Attention Layer**: This layer computes the importance of every passage word with respect to every query word. In some models, the relative importance of every query word is also computed with respect to every passage word.

The simplest way to compute this relative importance is by computing the pair-wise dot product of the individual document token representations with the individual query representations followed by a non-linearity. This pair-wise similarity can also be computed using a trainable shared similarity matrix. Then, attention layers for query-aware-document representation as well as document-aware-query representation (for some models) are computed which are dependent on the similarity matrix.

The Match-LSTM method, borrowed from the textual entailment literature (Wang and Jiang, 2016) is another way to compute the importance of document words with respect to the query words sequentially. For every document word, attention mechanism is used to obtain a weighted representation of the query. This weighted query representation is then combined with the current document representation and fed into an LSTM, which is known as Match-LSTM. This Match-LSTM thus refines the passage representation sequentially.

Some models use a modification of Match-LSTM referred to as gated attention-based recurrent networks that add an additional gate in the Match-LSTM to refine the individual token representation in the document based on the attended query representation.

**Merging Layer**: After computing the cross-attention layer, the document words have been weighted according to the input query. However, these representations may not be able to capture important hints in the document beyond its neighbouring words. The presence of these hints across various sections in the document is necessary to answer the question. Hence, a Merging Layer is used to further refine the document representation. In some models, an RNN layer is used as a Merging Layer. In some cases, a Self-Matching Layer is used where every document word is attended by every other document word *i.e,* both m,n in the m-aware-n representation refer to the document words.

### Answer Layer

Depending on the problem at hand, the answer can be a span of contiguous words in the document or needs to be generated as a sequence of words which are conditioned on the query and the document.

Typically, for span-based answers, layers based on Pointer Networks (Vinyals *et al.*, 2015) are used. In this method, first, a start position is obtained by using a Softmax layer over the passage representation from the previous layer, typically, the Merging Layer. Then, an additional step is included to improve the document representation based on the position of the start pointer. Finally, using another layer of Pointer Networks, the end position is computed based on this new document representation. All the models in Table 3.1 use some variant of Pointer Networks.

For generation based answers, first an answer is predicted using any of the span prediction techniques and then it is passed through an RNN based decoder. Of the entries in Table 3.1, S-Net uses a generation based approach.

| Model Name | Character Level Embedding | Embedding Fusion | Cross Attention Layer | Merging Layer |
|---|---|---|---|---|
| Fine-Grained Gating | RNN | Fine-Grained Gating | Pair-wise dot product | No |
| Match-LSTM | No | No | Match-LSTM | No |
| R-Net | RNN | Concat | Gated attention-based recurrent network | Self-Matching Network |
| S-Net | RNN | Concat | Gated attention-based recurrent network | No |
| BiDAF | CNN | Concat + Highway | Shared similarity matrix | RNN over document |

Table 3.1: Different Question Answering Models - Fine-Grained Gating (Yang *et al.*, 2017), Match-LSTM (Wang and Jiang, 2017), R-Net (Wang *et al.*, 2017), S-Net (Tan *et al.*, 2018), and BiDAF (Seo *et al.*, 2017).

We describe BiDAF (Seo *et al.*, 2017) architecture in section 4.2.3. We refer the reader to the respective papers for detailed descriptions of the components in these models. We also refer the reader to the SQuAD[1] and MSMARCO[2] leader boards for the newer reading comprehension models.

### 3.3.2 NLP tasks with Copying Mechanism

We lay emphasis on the architectures that use a copy-or-generate decoder as our task involves a span based dialogue response generation where contiguous information needs to be copied. In most of the NLG problems, words are produced based on the probability distribution over the vocabulary. A special marker such as <UNK> is used to generate words that are not in the vocabulary. In order to improve the informativeness of the generated words, networks are designed to exhibit "Copying Mechanism"

[1] `https://rajpurkar.github.io/SQuAD-explorer/`
[2] `http://www.msmarco.org/leaders.aspx`

**Source Document** ... norway delivered a diplomatic protest to russia on monday after three norwegian fisheries research expeditions were barred from russian waters . the norwegian research ships were to continue an annual program of charting fish resources shared by the two countries in the barents sea region ...

**Summary:** norway protests russia barring fisheries research ships

---

**Source Document**...i , of course , was directing my remarks at a legacy of issues and not the workers in ASC whom I consider to be world class. with fears the $ 40 billion plus program will be lost to south australia in favour of submarines built overseas , opposition leader bill shorten moved a censure motion ...

**Summary:** there are fears the $ 40 billion program will be taken away from south australia and submarines bought from overseas

Figure 3.1: Examples from the CNN-Daily Mail Summarization dataset. (Nallapati *et al.*, 2016). The words in blue are most likely to be copied from the source document during the summary generation.

where a decoder is enabled to copy from the words from the encoder side. For example, consider the task of abstractive summarization which is different from extractive summarization (Nallapati *et al.*, 2017; Qazvinian *et al.*, 2014). It requires the network to produce coherent and concise text pieces of long documents where the sentences produced should contain novel phrases or words. The copying mechanism is crucial here as the facts need to be copied as it is while the information surrounding the facts can be abstractive. These facts are typically named-entities or numbers; the tokens which do not occur in the normal vocabulary cutoff. Consider the articles and their summary shown in Figure 3.1. In the first summary, "nigeria" and "russia" need to be copied from the source document as they will not appear in the vocabulary. Similarly, in the second summary, the fact "$ 40 billion" needs to be copied as it is.

The first application of the copy mechanism was demonstrated for text summarization and synthetic dialog dataset (Gu *et al.*, 2016). The synthetic dialog dataset used simple scenarios that required copying from the first user's utterance. A follow-up work (He *et al.*, 2017) proposed a hybrid model with the copying and retrieval mechanism for question answering. CopyNet (Gu *et al.*, 2016) has been adapted to code-generation (Lin *et al.*, 2018), goal-oriented dialogue systems (Eric and Manning, 2017). One of our baselines uses a model from the abstractive summarization literature that uses a per time step copying mechanism. In this, the decoder decides to copy a word from the encoder side or generate a word from the fixed vocabulary at every time step. The architecture of most interest to our work is the one in See *et al.* (2017). We discuss it in detail in subsection 4.2.2.

## 3.4 Approaches to Model Structural Information

The neural architectures for NLP described so far heavily rely on the family of RNNs which treat natural language as a sequential problem. In section 2.3, we emphasize the importance of using structural information in NLP. We briefly discuss improvements over the RNN architecture or new techniques that use such structural information.

Socher *et al.* (2011) proposed bottom-up recursive neural networks that can predict binary syntax parse trees. Later, gated LSTM variants were proposed to model constituency parse structures (Tai *et al.*, 2015; Le and Zuidema, 2015; Zhu *et al.*, 2015*a*) for sentiment classification and sentence matching tasks. Tai *et al.* (2015) also proposed a generic child sum tree that could incorporate dependency parse structures. These models were commonly called as Tree-LSTMs.

Peng *et al.* (2017) further generalized LSTMs to model more generic graph structures over specific sequence or tree structures. Peng *et al.* (2017)'s Graph-LSTM was designed to model the document graph that incorporated adjacent, syntax, and discourse dependencies among words both at inter-sentence and intra-sentence level. They showed that Graph-LSTMs achieved superior performance over Tree-LSTMs on n-ary relation extraction across sentences. Not just Peng *et al.* (2017) but numerous other recent works have resorted to graph models to capture structural information. This was driven by the success of Graph Convolution Networks(GCNs) for network representation learning, primarily the work in Kipf and Welling (2017).

GCNs have been extended to incorporate multiple relations with gating mechanisms that re-weigh the importance of each neighbor based on the edge features (Bastings *et al.*, 2017). These models are more flexible to model structural prior and can compute multi-level word features in parallel unlike with trees that require sequential level-wise processing. Some works relevant to our work include the use of semantic graphs for neural machine translation (Marcheggiani *et al.*, 2018), the use of syntactic information using dependency graphs for semantic role labelling (Marcheggiani and Titov, 2017), neural machine translation (Bastings *et al.*, 2017), relation extraction (Zhang *et al.*, 2018), entity information in question answering (Song *et al.*, 2018; Cao *et al.*, 2018) and temporal information for neural dating of documents (Vashishth *et al.*, 2018).

## Summary

In this chapter, we discussed various datasets and techniques for building data-driven conversation systems. As the task in this thesis is similar to certain NLP tasks such as reading comprehension or the tasks which involve production of words as is from a resource document, we briefly described the architectures for these tasks. We also discussed certain drawbacks of the existing data-driven conversation systems. In chapter 4, we will focus on improving neural conversation systems by explicitly incorporating external knowledge during the production of a response. We also mentioned that structural information is ubiquitous in NLP and hence, we listed the existing methods that incorporate structural information in neural NLP architectures. In chapter 5, we will discuss the importance of exploiting this structural information present in the background knowledge to improve the dialogue response generation.

# CHAPTER 4

# Building Domain Specific Background Aware Conversation Systems

In chapter 1, we establish the importance of using background knowledge in conversation systems. From chapter 3, we observe that though there is an increasing effort in incorporating background knowledge in neural conversation systems, most of the systems follow a post-facto process making the existing chats not well coupled with the background resources.

In this chapter, we first discuss the process to build a background knowledge aware conversations dataset for the domain of movies. We then evaluate our dataset on three different baselines. Finally, we present our analysis of the outputs produced by these baselines.

## 4.1 Dataset Collection

In the following subsections we describe the various stages involved in collecting our dataset: (1) curating a list of popular movies, (2) collecting/crawling background information such as plots, reviews, comments, and facts for each of these movies, (3) collecting conversations starters (or opening statements) specific to these movies, and (4) collecting crowdsourced conversations for these movies starting with the collected opening statements. We describe each of these stages in detail below followed by a discussion of the method used for verifying the collected chats and the statistics of the collected data.

### 4.1.1 Curating a list of popular movies

While curating a list of movies, we had two main factors in mind. First, the movies should be popular so that (1) resources (plots, reviews, *etc.*) about the movie would be easily available online and (2) it would be easy for an average worker to chat about the movie. Second, we wanted these movies to correspond to a wide range of genres so that our dataset was not tied to a specific genre.

We created a list of 921 movies containing (1) top 10 popular movies within the past five years, (2) top 250 movies as per IMDb rankings, (3) top 10 movies in popular genres, and (4) other popular movie lists made available elsewhere on the Internet. These movies belonged to 22 different genres such as sci-fi, action, horror, fantasy, adventure, romance, *etc.* thereby ensuring that our dataset is not limited to a specific genre. We considered those movies for which enough background information such as plots, reviews, comments, facts, *etc.* was available on the Internet irrespective of whether they were box-office successes or not. Following are the Universal Resource Locators (URLs) used to curate the popular movies list:

- IMDb top 250: `https://www.imdb.com/chart/top`
- Popular movies by genre: `https://www.imdb.com/feature/genre/`
- Other popular movie lists:
  - `https://www.imdb.com/feature/genre/`
  - `https://www.thetoptens.com/best-movie-genres/`
  - `http://1001films.wikia.com/wiki/The_List`

### 4.1.2 Collecting background knowledge

For each movie, we collected the following background knowledge:

**1. Review (R):** For each movie, we also had the corresponding IMDb id of the movie since most of the movie names were curated from IMDb itself. Once this id is known, we can create a unique URL that takes us to the IMDb page of the corresponding movie. We gave this unique URL to some in-house workers and asked them to fetch the top 2 most popular reviews for this movie. Specifically, IMDb allows the reviews to be sorted by "Total Votes" received by the review, so we asked the workers to fetch the reviews which had the maximum votes. We also instructed them to avoid choosing reviews which were less than 50 words but this was typically never the case with popular reviews.

**2. Plot (P):** For each movie, we identified the corresponding Wikipedia page using a semi-automatic process. We found out that the movie meta-data on the IMDb movie page also included the corresponding Wikipedia page ID. In most cases, we were able to construct the Wikipedia URL but in some cases, we needed manual intervention. All the URLs constructed in this manner were then manually verified by in-house workers. Once we had the Wikipedia page corresponding to the movie, we extracted information about the "Plot" of the movie from the Wikipedia page of the movie. Wikipedia pages of movies have an explicit section on "Plot" making it easy to extract this information using scripts. However, showing the entire plot to the users was hampering the speed of the task. Hence, the plot was then divided into snippets of 10 sentences by adding a breakpoint after every 10 sentences. During the chat collection, a snippet was chosen randomly and shown to the user.

**3. Comments (C):** Websites like *Reddit* have a segment called "official discussion page about X", where X is a movie name, containing small comments about various aspects of the movie. We identified such pages and extracted the first comment on every thread on this page. We bundled all these comments into a single text file and refer to it as the resource containing "Comments". For a few movies, the official discussion page was not present in which case we used the review titles of all the IMDb reviews of the movie as comments. The difference between Reviews and Comments is that a Review is an opinion piece given by one person thus typically exhibiting one sentiment throughout while Comments include opinions of several people about the same movie ensuring that positive, negative, and factual aspects of the movie are captured as well as some banter is present in the conversation.

**4. Metadata or Fact Table (F):** For each movie, we also collected factual details about the movie from the corresponding IMDb pages and Wikipedia Infoboxes. Such information would be useful for inserting facts in the conversation, for example, *"Did you know that the movie won an Oscar?*". We included only 4 fields (box office collection, awards, taglines, and similar movies) in our fact table instead of showing the entire Wikipedia Infobox to reduce the cognitive load on workers who already had to read the Plot, Review, and Comments of the movie.

### 4.1.3 Collecting conversation starters

During our initial pilots, we observed that if we asked the workers to converse for at least 8 turns, they used a lot of the initial turns in greetings and general chit-chat before actually chatting about a movie. To avoid this, we collected opening statements using Amazon Mechanical Turk (AMT) where the task for the workers was to answer the following questions "*What is your favorite scene from the movie X?*", "*What is your favorite character from the movie X?*" and "*What is your opinion about the movie X?*" where X is the movie name. We paid the workers 0.04$ per movie and showed the same movie to 3 different workers, thereby collecting 9 different opening statements for every movie. By using these statements as conversation starters in our data collection, the workers could now directly start conversing about the movie.

### 4.1.4 Collecting background knowledge aware conversations via crowd-sourcing

We wanted to create a conversation dataset wherein every response is explicitly linked to some structured or unstructured background knowledge. Creating such a dataset using dedicated in-house workers would obviously be expensive and time-consuming and so we decided to use crowdsourcing. However, unlike other NLP and Vision tasks, where crowdsourcing has been very successful, collecting conversations via crowdsourcing is a bit challenging. The main difficulty arises from the fact that conversation is inherently a task involving two persons but it is hard to get two workers to synchronize and chat on AMT. We did try a few pilot experiments where we set up a server to connect two AMT workers but we found that the probability of two workers simultaneously logging in was very low. Thus, most workers logged in and left in a few seconds because no other worker joined simultaneously. Finally, we took inspiration from the idea of self chats (Krause *et al.*, 2017) in which, the same worker plays the role of both Speaker 1 and Speaker 2 to create the chat thus chatting with herself/himself. We also had several discussions with expert workers on Turkopticon (`http://turkopticon.ucsd.edu/`) and TurkerNation (`http://turkernation.com/`) to arrive at the final setup used for collecting conversations. Their suggestions helped us make the task more acceptable and conducive to workers on AMT.

**Movie: Spider-Man**

**Plot**

... The lab works on spiders and has even managed to create new species of spiders through genetic manipulation. While Peter is taking photographs of Mary Jane for the school newspaper, one of these new spiders lands on his hand and bites him Peter comes home feeling ill and immediately goes to bed. ...

**Review**

... I thoroughly enjoyed "Spider-Man" which I saw in a screening. I thought the movie was very engrossing. Director Sam Raimi kept the action quotient high, but also emphasized the human element of the story. Tobey was brilliant as a gawky teenager...

Speaker 1(N): Which is your favourite character?

Speaker 2 (C): My favorite character was Tobey Maguire.

Speaker 1 (N): I thought he did an excellent job as Peter Parker, I didn't see what it was that turned him into Spider-Man though.

Speaker 2 (P): Well this happens while Peter is taking photographs of Mary Jane for the school newspaper, one of these new spiders lands on his hand and bites him.

Speaker 1 (N): I see. I was very excited to see this film and it did not disappoint!

Speaker 2 (R): I agree, I thoroughly enjoyed "Spider-Man"

Speaker 1 (N): I loved that they stayed true to the comic.

Speaker 2 (C): Yeah, it was a really great comic book adaptation

Speaker 1 (N): The movie is a great life lesson on balancing power.

Speaker 2 (F): That is my most favorite line in the movie, "With great power comes great responsibility."

**Comments**

... Crazy attention to detail. My favorite character was Tobey Maguire. I can't get over the "I'm gonna kill you dead" line. It was too heavily reliant on constant light-hearted humor. However the constant joking around kinda bogged it down for me. A really great comic book adaptation. ....

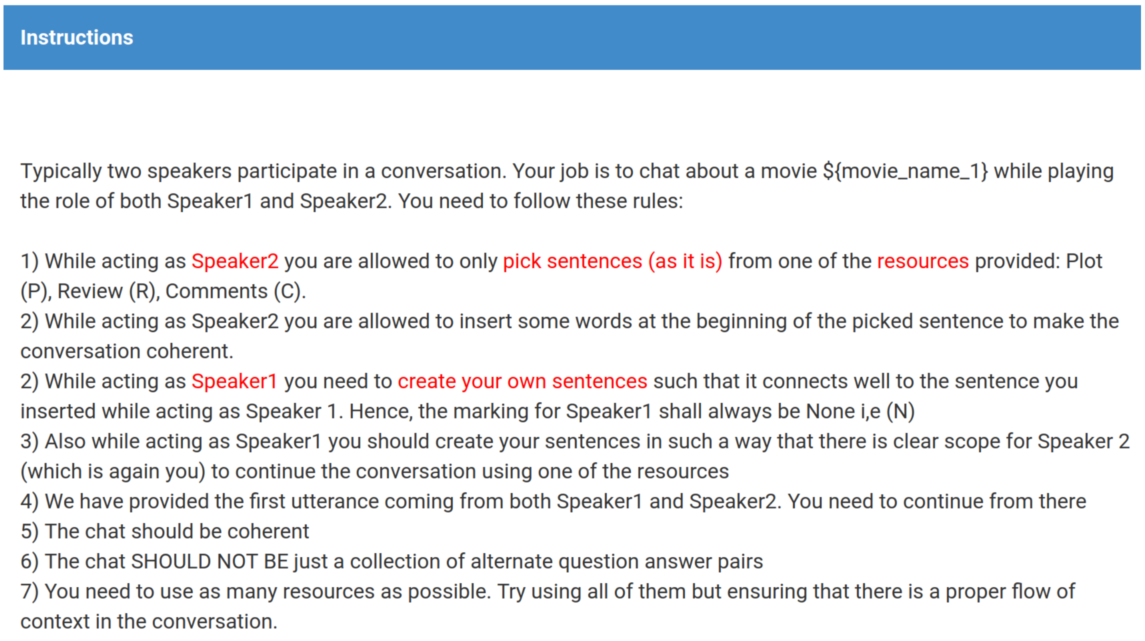
**Fact Table**

| | |
|---|---|
| Awards | Golden Trailer Awards 2002 |
| Taglines | With great power comes great responsibility. Get Ready For Spidey ! |
| Similar Movies | Iron Man Spider-Man 2 |

Figure 4.1: A sample chat from our dataset which uses background resources. The chosen spans used in the conversation are shown in blue. The letters in the brackets denote the type of resource that was chosen - P, C ,R, F and N indicate **P**lot, **C**omments, **R**eview, **F**act Table and **N**one respectively.

In the above self chat setup, we showed every worker 3 to 4 resources related to the movie, *viz.*, plot (P), review (R), comments (C) and fact table (F). We also showed them a randomly selected opening statement from the 9 opening statements that we had collected for each movie and requested them to continue the conversation from that point. The workers were asked to add at least 8 utterances to this initial chat.

While playing the role of Speaker 1, the worker was not restricted to copy/modify sentences from the background resources but was given the freedom to create (write) original sentences. However, when playing the role of Speaker 2, the worker was strictly instructed to copy/modify sentences from the shown resources such that they were relevant in the current context of the conversation. The reason for not imposing any restrictions on Speaker 1 was to ensure that the chats look more natural and coherent. Further, Speaker 2 was allowed to add words at the beginning or end of the span selected from the resources to make the chats more coherent and natural (for example, see the prefix in utterance 2 of Speaker 2 in Figure 4.1). We paid the workers 40 cents for every chat. We will now list the screenshots for the procedure in Figures 4.2, 4.3, and 4.4.


Instructions

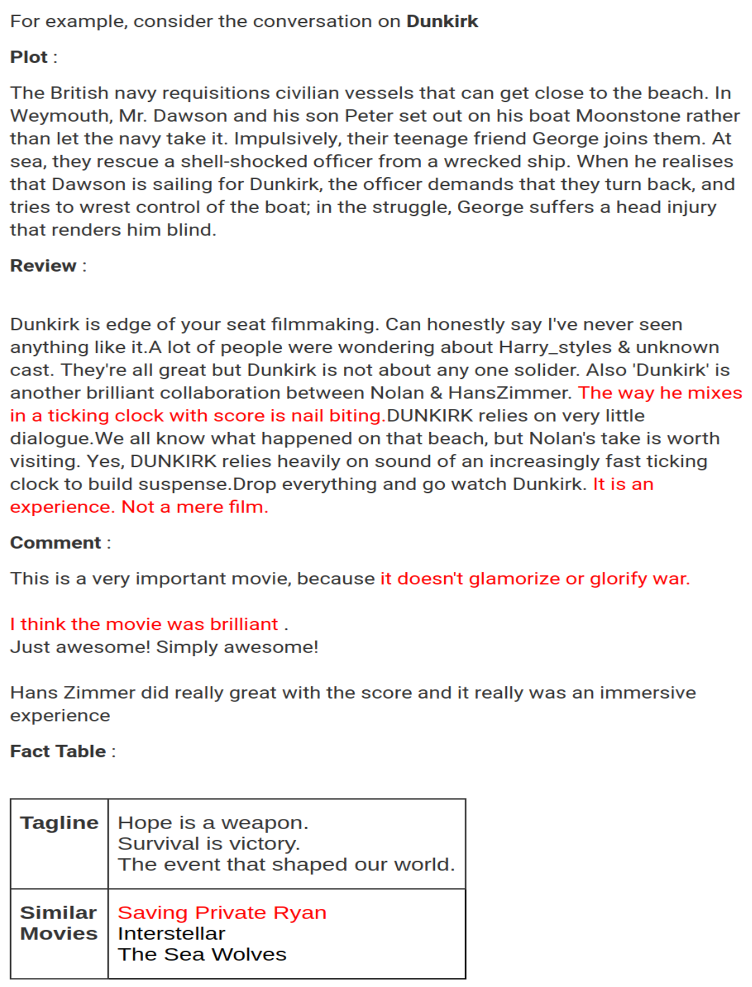
Typically two speakers participate in a conversation. Your job is to chat about a movie ${movie_name_1} while playing the role of both Speaker1 and Speaker2. You need to follow these rules:

1) While acting as Speaker2 you are allowed to only pick sentences (as it is) from one of the resources provided: Plot (P), Review (R), Comments (C).
2) While acting as Speaker2 you are allowed to insert some words at the beginning of the picked sentence to make the conversation coherent.
2) While acting as Speaker1 you need to create your own sentences such that it connects well to the sentence you inserted while acting as Speaker 1. Hence, the marking for Speaker1 shall always be None i,e (N)
3) Also while acting as Speaker1 you should create your sentences in such a way that there is clear scope for Speaker 2 (which is again you) to continue the conversation using one of the resources
4) We have provided the first utterance coming from both Speaker1 and Speaker2. You need to continue from there
5) The chat should be coherent
6) The chat SHOULD NOT BE just a collection of alternate question answer pairs
7) You need to use as many resources as possible. Try using all of them but ensuring that there is a proper flow of context in the conversation.

Figure 4.2: Instruction screen for the chat data collection.

### 4.1.5 Verification of the collected chats

Every chat that was collected by the above process was verified by an in-house evaluator to check if the workers adhered to the instructions and produced coherent chats. Since humans typically tend to paraphrase the background knowledge acquired by reading articles, one could argue that such conversations may not look natural because of this restriction to copy/modify content from the provided resources. To verify this, we conducted a separate human evaluation wherein we asked 15 in-house evaluators to read conversations (without the background resources) from our dataset and rate them on five different parameters. Specifically, they were asked to check if the conversations were (1) **intelligible:** *i.e.*, an average reader could understand the conversation, (2) **coherent:** *i.e.*, there were no abrupt context switches, (3) **grammatically correct**, (4) **on-topic:** *i.e.*, the chat revolved around the concerned movie with digression limited to related movies/characters/actors, and (5) **natural two-person chats:** *i.e.*, the chat looked like a two-person conversation. These evaluators were post-graduate students who were fluent in English and had watched at least 100 Hollywood movies. We did



For example, consider the conversation on **Dunkirk**

**Plot** :

The British navy requisitions civilian vessels that can get close to the beach. In Weymouth, Mr. Dawson and his son Peter set out on his boat Moonstone rather than let the navy take it. Impulsively, their teenage friend George joins them. At sea, they rescue a shell-shocked officer from a wrecked ship. When he realises that Dawson is sailing for Dunkirk, the officer demands that they turn back, and tries to wrest control of the boat; in the struggle, George suffers a head injury that renders him blind.

**Review** :

Dunkirk is edge of your seat filmmaking. Can honestly say I've never seen anything like it.A lot of people were wondering about Harry_styles & unknown cast. They're all great but Dunkirk is not about any one solider. Also 'Dunkirk' is another brilliant collaboration between Nolan & HansZimmer. The way he mixes in a ticking clock with score is nail biting.DUNKIRK relies on very little dialogue.We all know what happened on that beach, but Nolan's take is worth visiting. Yes, DUNKIRK relies heavily on sound of an increasingly fast ticking clock to build suspense.Drop everything and go watch Dunkirk. It is an experience. Not a mere film.

**Comment** :

This is a very important movie, because it doesn't glamorize or glorify war.

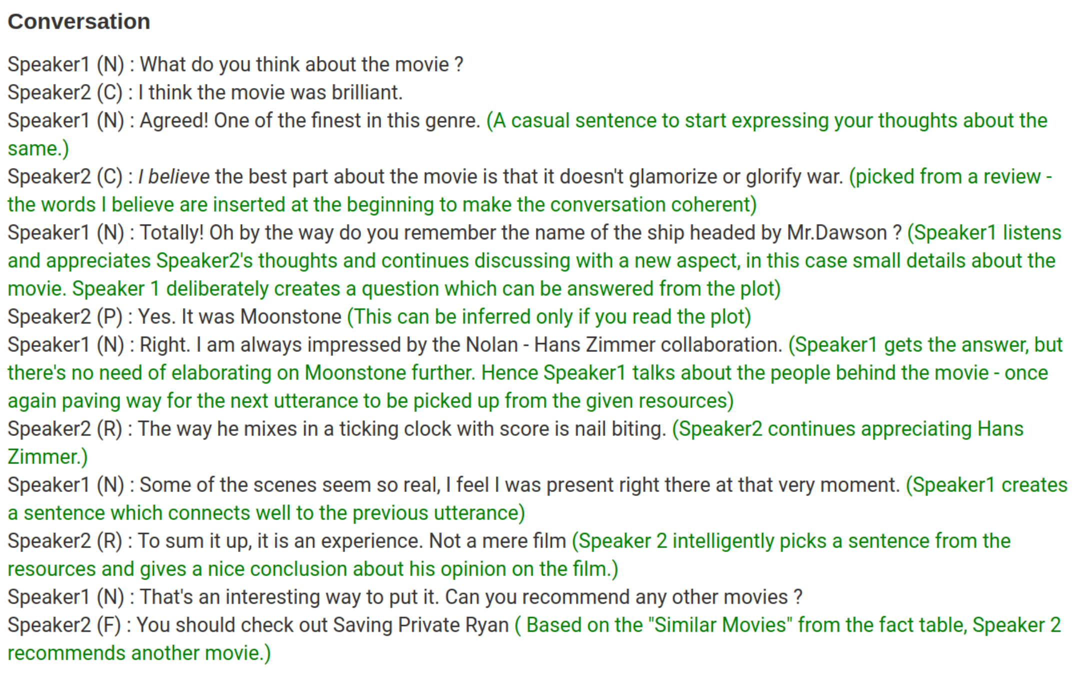
I think the movie was brilliant .
Just awesome! Simply awesome!

Hans Zimmer did really great with the score and it really was an immersive experience

**Fact Table** :

| | |
|---|---|
| **Tagline** | Hope is a weapon.<br>Survival is victory.<br>The event that shaped our world. |
| **Similar Movies** | Saving Private Ryan<br>Interstellar<br>The Sea Wolves |

Figure 4.3: Background resources for the example chat shown to the workers on AMT.



**Conversation**

Speaker1 (N) : What do you think about the movie ?
Speaker2 (C) : I think the movie was brilliant.
Speaker1 (N) : Agreed! One of the finest in this genre. (A casual sentence to start expressing your thoughts about the same.)
Speaker2 (C) : *I believe* the best part about the movie is that it doesn't glamorize or glorify war. (picked from a review - the words I believe are inserted at the beginning to make the conversation coherent)
Speaker1 (N) : Totally! Oh by the way do you remember the name of the ship headed by Mr.Dawson ? (Speaker1 listens and appreciates Speaker2's thoughts and continues discussing with a new aspect, in this case small details about the movie. Speaker 1 deliberately creates a question which can be answered from the plot)
Speaker2 (P) : Yes. It was Moonstone (This can be inferred only if you read the plot)
Speaker1 (N) : Right. I am always impressed by the Nolan - Hans Zimmer collaboration. (Speaker1 gets the answer, but there's no need of elaborating on Moonstone further. Hence Speaker1 talks about the people behind the movie - once again paving way for the next utterance to be picked up from the given resources)
Speaker2 (R) : The way he mixes in a ticking clock with score is nail biting. (Speaker2 continues appreciating Hans Zimmer.)
Speaker1 (N) : Some of the scenes seem so real, I feel I was present right there at that very moment. (Speaker1 creates a sentence which connects well to the previous utterance)
Speaker2 (R) : To sum it up, it is an experience. Not a mere film (Speaker 2 intelligently picks a sentence from the resources and gives a nice conclusion about his opinion on the film.)
Speaker1 (N) : That's an interesting way to put it. Can you recommend any other movies ?
Speaker2 (F) : You should check out Saving Private Ryan ( Based on the "Similar Movies" from the fact table, Speaker 2 recommends another movie.)

Figure 4.4: Example chat shown to the workers on AMT.

| Metric | Rating |
|---|---|
| Intelligible | 4.47 ± 0.52 |
| Coherent | 4.33 ± 0.93 |
| Grammar | 4.41 ± 0.56 |
| Two-person-chat | 4.47 ± 0.46 |
| On Topic | 4.57 ± 0.43 |

Table 4.1: Average human evaluation scores with standard deviations for conversations (scale 1-5).

not give them any information about the data creation process. Instead, we asked them to consider these chats as an informal conversation between two friends.

We used a total of 500 chats for the evaluation and every chat was shown to 3 different evaluators. The evaluators rated the conversations on a scale of 1 (very poor) to 5 (very good). The average rating for each of the 5 parameters is reported in Table 4.1 and is very encouraging.

Human evaluation is one way to ensure that using the self-chats method produces chats on par with a natural dialogue. For future corpora creation that use the self-chats technique, we would recommend collecting self-chats and natural chats as a pilot study and provide possible hints to the workers to improve the naturalness of the conversation.

## Statistics

In Table 4.2, we show different statistics about the dataset collected using the above process. These include the average number of utterances per chat, the average number of words per utterance, and so on followed by the statistics of the different resources which were used as background knowledge. Please note that the # unique Plots and # unique Reviews correspond to unique paragraphs while the # unique Comments is the count of unique sentences. We observed that 41.2%, 34.6%, 16.1%, and 8.1% of Speaker 2 responses came from Reviews, Comments, Plots, and Fact Table respectively.

We also compare the size of our dataset against other popular domain specific dialogue datasets in Table 4.3. The Frames dataset is goal-oriented and thus has limited conversation scenarios. Our and CMU_DoG datasets are based on movie conversations, an informal domain setting and hence have more examples than the Frames dataset.

| | |
|---|---|
| #chats | 9071 |
| #movies | 921 |
| #utterances | 90810 |
| Average # of utterances per chat | 10.01 |
| Average # of words per utterance | 15.29 |
| Average # of words per chat | 153.07 |
| Average # of words in Plot | 186.10 |
| Average # of words in Review | 384.44 |
| Average # of words in Comments | 123.81 |
| Average # of words in Fact Table | 33.47 |
| # unique Plots | 5157 |
| # unique Reviews | 1817 |
| # unique Comments | 12740 |

Table 4.2: Statistics of the dataset.

Finally, Wizard of Wikipedia which comprises of 1365 diverse topics has the largest corpus size. It is interesting to note that the corpus size increases as more domains and diverse responses are included (Vlad Serban *et al.*, 2015).

| Dataset | #chats | #utterances |
|---|---|---|
| Frames (Asri *et al.*, 2017) | 1369 | 19,986 |
| Our | 9071 | 90810 |
| CMU_DoG (Zhou *et al.*, 2018) | 4112 | 130000 |
| Wizard of Wikipedia (Dinan *et al.*, 2019) | 22311 | 201999 |

Table 4.3: Comparison with other datasets.

## Collecting multiple reference responses

One common issue with evaluating dialogue systems is that existing datasets typically contain only one reference response whereas in practice several responses can be correct in a given context. To solve this problem to a certain extent, we collected three reference responses for every Speaker 2 utterance in our dataset (note that Speaker 2 is treated as the bot while training/testing our models). We show the previous utterances ending with Speaker 1's response and ask workers to provide three appropriate responses from the given resources. We found that in some cases there was only one appropriate response like factual response and the workers could not provide multiple references. In this way, we were able to create a multiple reference test set where 78.04% of the test instances have multiple responses. Refer to Table 4.4 for an example of the multi-reference test set.

| Movie Name | The Secret Life of Pets |
|---|---|
| Chat | Speaker 1: What do you think about the movie?<br>Speaker 2: I think it was comical and entertaining.<br>Speaker 1: It delivered what was promised. |
| Reference 1 | I agree! I'm surprised this film got such a low overall score by users. |
| Reference 2 | My favorite character was Gidget! She was so much fun and so loyal to her friends! |
| Reference 3 | Yes! As a Great Dane owner, I often wonder what my dogs are thinking.<br>It was fun to see this take on it. |
| Reference 4 | It was full of cliches with a predictable story, but with some really funny moments. |

Table 4.4: Multiple references for the given chat.

## 4.2 Baselines

We evaluate our dataset on three different types of models, *viz.*, (1) generation based models which completely ignore the background knowledge, (2) generate-or-copy models which learn to copy from the background knowledge whenever required and generate text otherwise, and (3) copy based models or span prediction models which learn to predict the correct span from the background resources (as opposed to generating the response). We evaluate one such state of the art model in each of these three categories as described below. Note that, in this work, we treat all the background resources as a single document. In other words, we merge the comments, reviews, plots and facts into one single document. So when we refer to a document or resource in the rest of the thesis we mean this single document which is a merger of all the resources. Eventually, we will need to design appropriate models to first select the right resource (plot, reviews, comments, *etc.*) and then generate or copy the response from the selected resource.

### 4.2.1 Generation based Models

We evaluate the Hierarchical Recurrent Encoder Decoder (HRED) model Serban *et al.* (2016) which is one of the state-of-the-art conversation models. One of its variants Serban *et al.* (2017*c*) has been shown to perform slightly better on some datasets but the original model is much easier to implement and hence we chose this model. As the name suggests, the model uses a hierarchical RNN to encode the context of the conversation. The lower level RNN computes the representation of every utterance in the conversation by treating it as a sequence of words. The representation of every utterance thus computed is fed as input to another RNN *i.e.*, the sequence of utterance representations is fed as input to the second level RNN. The final state of this RNN is taken as the representation of the complete context *i.e.*, all previous utterances. This context representation is then fed to the decoder which is again an RNN and generates the response one word at a time. Note that this model does not take into account any background information and simply treats the conversation as a sequence of utterances.

### 4.2.2 Generate-or-Copy Models

Here, we consider a model inspired from the literature on abstractive summarization where the goal is to generate an abstractive summary of the document as opposed to the extractive summary which simply ranks the most important sentences in the document. Even though the summary is generated, it is quite natural that it will contain a few words which are copied from the document. Keeping this in mind, (See *et al.*, 2017) proposed a hybrid pointer generator network that learns to copy by pointing to the words in the source document when required and otherwise generates a word like any Seq2Seq model. We modified and adapted this architecture for our task. In the summarization task, the input is a *document* and the output is a *summary* whereas in our case the input is a *resource/document, context* pair and the output is a *response*. Note that the context includes the previous two utterances(dialog history) and the current utterance. Since, in both the tasks, the output is a sequence (*summary* v/s *response*) we don't need to change the decoder (*i.e.*, we can use the decoder from the original model as it is). However, we need to change the input fed to the decoder. Similar to the original model, we use an RNN to compute the representation of the document. Let $N$ be the length of the document then the RNN computes representations $h^r_1, h^r_2, ..., h^r_N$ for all the words in the resource (we use the superscript $r$ to indicate resource). The final representation of the resource is then the attention weighted sum of these word representations:

$$\begin{aligned} e^t_i &= v^T tanh(W_r h^r_i + U s_t + b_r) \\ a^t &= softmax(e^t) \\ r_t &= \sum_i a^t_i h^r_i \end{aligned} \tag{4.1}$$

where $s_t$ is the state of the decoder at the current time step. In addition, in our case, we also have the context of the conversation apart from the document (resource). Once again, we use an RNN to compute a representation of this context. Specifically, we consider the previous $k$ utterances as a single sequence of words and feed these to an RNN. Let $M$ be the total length of the context (*i.e.*, all the $k$ utterances taken together) then the RNN computes representations $h_1^c, h_2^c, ..., h_M^c$ for all the words in the context (we use the superscript $c$ to indicate context). The final representation of the context is then the attention weighted sum of these word representations:

$$\begin{aligned} f_i^t &= v^T tanh(W_c h_i^c + V s_t + b_c) \\ m^t &= softmax(f^t) \\ c_t &= \sum_i m_i^t h_i^c \end{aligned} \tag{4.2}$$

where $s_t$ is the state of the decoder at the current time step. Similarly, we also use the document representation to pay attention to important words in the context and compute a modified representation of the context in light of the document

The decoder then uses $r_t$ (document representation), $c_t$ (context representation) and $s_t$ (decoder's internal state) to compute a probability distribution over the vocabulary $P_{vocab}$. In addition the model also computes $p_{gen}$ which indicates that there is a probability $p_{gen}$ that the next word will be *generated* and a probability $(1-p_{gen})$ that the next word will be *copied*. We use the following modified equation to compute $p_{gen}$

$$p_{gen} = \sigma(W_r^T r_t + W_c^T c_t + W_s^T s_t + W_x^T x_t + b_g) \tag{4.3}$$

where $x_t$ is the previous word predicted by the decoder and fed as input to the decoder at the current time step. Similarly, $s_t$ is the current state of the decoder computed using this input $x_t$. All the $W$ and $b$ are trainable. The final probability of a word $w$ is then computed using a combination of two distributions, *viz.*, ($P_{vocab}$) as described above and the attention weights assigned to the document words as shown below.

We do not alter the equation to select words from the extended vocabulary distribution

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \tag{4.4}$$

where $a_i^t$ are the attention weights assigned to every word in the document as computed in equation 4.1. Thus, effectively, the model could learn to copy a word $i$ if $p_{gen}$ is low and $a_i^t$ is high. Following the title of the original paper, we refer to this model as the GTTP (Get To The Point) model. The first term of the equation gives weight to generation while the second term of the equation gives weight to the copy by pointing. With our new modification, the $p_{gen}$ is computed using attended representation of the document $h_t^*$, attended representation of the context of the conversation $q_t^*$, decoder state $s_t$ and decoder input word $x_t$.

### 4.2.3 Span Prediction Models

The BiDirectional Attention Flow Model (BiDAF) (Seo *et al.*, 2017) model is a Question Answering model that was proposed in the context of the SQuAD dataset (Rajpurkar *et al.*, 2016). Given a *document* and a *question*, the task is to pick the answer from the document *i.e.*, to predict the span in the document which contains the answer. This has an analogy to our task wherein instead of *document* we have *resources* and instead of *question* we have the *context i.e.*, the previous $k$ utterances. Given the {resource, context} we want to predict the span in the resource from which the next response was generated. Given this direct analogy, we can use their model as it is for our task without any modifications by simply treating the context as the question. We briefly describe the layers of the architecture. The first and second layer provides character and word level representations for both query and document respectively. The third layer (contextual embedding layer) passes these representations through another RNN to capture temporal interaction between the words in the document and query respectively and individually. The fourth layer (Attention Flow Layer) provides query-aware-document representation and document-aware=query representation. These representations are obtained from a shared similarity matrix which computes the similarity between each representation from the third layer of the document word and each representation from the third layer of the query word. These two representations are then fused to get an enhanced query aware representation of every document word by passing them through a feed-forward network. The fifth layer (Modelling Layer) obtains the further enriched representation of every document word representation from the fourth layer by modeling interaction of every document word representation with every other word in the document representation using another set of LSTM. The final layer (Answer Layer) predicts the start and end position of the answer. Since we have not made any modifications to their model, we refer the reader to the paper (Seo *et al.*, 2017) for details of the model.

## 4.3 Experimental Setup

In this section, we describe the train-validation-test splits, the process used for creating training instances, the manner in which the models were trained using our data, hyper-parameter details and the evaluation metrics.

### Creating train/valid/test splits

On average we have 9.14 chats per movie. We divided the collected chats into train, validation, and test splits such that all the chats corresponding to a given movie are in exactly one of the splits. This ensured that a movie seen in the test or validation set is never seen at training time. We created the splits such that the percentage of chats in the train-validation-test set is roughly 80%-10%-10%.

### Creating training instances

For each chat in the training data, we constructed training instances of the form {*resource, context, response*} where the *context* is taken as previous two utterances and current utterance. We considered only the even-numbered utterances as training examples as they are generated from the background resources thus emulating a human-bot setup. If a chat has 10 turns, we will have 5 instances. The task then is to train a model that can predict these even-numbered responses. At test time the model is shown {*resource, context*} and predicts the response. Note that, HRED will ignore the *resource* and only use {*context, response*} as input-output pairs. BiDAF and GTTP will use {*resource, context, response*} as training data with relevant *span* instead of *response* for BiDAF.

## Merging resources into a single document

We simply merged all the background information to create a single document that we collectively refer to as *resource*. For the BiDAF model, we had to restrict the length of the resource to 256 words because we found that even on a K80 GPU with 12GB RAM, this model gives an out of memory error for longer documents. We found this to be a severe limitation of this and other span based models (for example, R-Net (Wang *et al.*, 2017)). We experimented with three methods of creating this resource. The first method *oracle* used the actual resource from which the next response was generated as a resource. If that resource itself had more than 256 words then we truncated it from the beginning and the end such that the span containing the actual response is contained within the retained 256 words. The number of words that were discarded from the start or the end was chosen at random so that the correct spans did not end up in similar positions throughout the dataset. The next two methods *mixed-short* and *mixed-long* were created by merging the individual resources. We retained each resource in the merged document proportional to its length. *i.e,* if there are 400 words in the plot, 200 words in the review and 100 in the comments, the merged resource will contain contiguous sentences from these three resources in the ratio of 4:2:1. Further, we ensured that the merged resource contained the actual response span. In this way, we created *mixed-short* with 256 words and *mixed-long* with 1200 words (the maximum length of the merged resources). We will henceforth denote *oracle*, *mixed-long* and *mixed-short* using '(o) ', '(ms) 'and '(ml) 'respectively.

## Hyper-parameters

Following the original paper, we trained the HRED model using Adam (Kingma and Ba, 2015) optimizer with an initial learning rate of 0.001 on a minibatch of size 16. We used dropout (Srivastava *et al.*, 2014) with a rate of 0.25. We use pre-trained GloVe (Pennington *et al.*, 2014) embeddings of size 300. For all the encoders and decoders in the model, we used GRU with 300 as the size of the hidden state. We restricted our vocabulary size to 20,000 most frequent words.

We followed the hyperparameters mentioned in the original paper (See *et al.*, 2017) and trained GTTP using Adagrad (Duchi *et al.*, 2011) optimizer with an initial learning rate of 0.15 and an initial accumulator value of 0.1 on a minibatch of size 16. For the encoders and decoders, we used LSTM with 256 as the size of the hidden state. To avoid vanishing and exploding gradient problem we use gradient clipping with a maximum gradient norm of 2. We used early stopping based on the validation loss.

Again following the original paper, we trained BiDAF using AdaDelta (Zeiler, 2012) optimizer with an initial learning rate of 0.5 on a minibatch of size 32. For all encoders, we use LSTM with 256 as the size of the hidden state. We used a dropout (Srivastava *et al.*, 2014) rate of 0.2 across all LSTM layers, and for the linear transformation before the softmax for the answers. We use pre-trained GloVe embeddings of size 100. For both GTTP and BiDAF, we had to restrict context length to 65 tokens for a fair comparison. Note that GTTP can scale beyond 65 tokens but BiDAF cannot.

### Evaluation metrics

As HRED and GTTP models are generation based models we use BLEU-4(Papineni *et al.*, 2002), ROUGE-1, ROUGE-2 and ROUGE-L (Lin, 2004) as the evaluation metrics. For BiDAF, we use these metrics by comparing the predicted span with the reference span. For BiDAF, we also report F1 as stated in (Rajpurkar *et al.*, 2016).

In addition to the automatic evaluation, we also collected human judgments using 100 test responses generated for every model for every setup (oracle, mixed-short, mixed-long). These evaluators had the same qualifications as the evaluators who earlier helped us evaluate our dataset. They were asked to rate the response on scale of 1 to 5 (with 1 being the least) on the following four metrics: (1) Fluency(Flu), (2) Appropriateness/relevance (apt) of the response in the current context language (3) Humanness (Hum) of the response, *i.e.,* whether the responses look as if they were generated by a human (4) and Specificity (spec) of the response, *i.e.*, whether the model produced movie-specific responses or generic responses such as “This movie is amazing”.

## 4.4 Results and Analysis

In this section, we analyze the results of our experiments as summarized in Tables 4.5 and 4.6.

| Model | F1 | | BLEU-4 | | ROUGE-1 | | ROUGE-2 | | ROUGE-L | |
|---|---|---|---|---|---|---|---|---|---|---|
| HRED | - | - | 05.23 | 05.38 | 24.55 | 25.38 | 07.61 | 08.35 | 18.87 | 19.67 |
| GTTP (o) | - | - | 13.92 | 16.46 | 30.32 | 31.6 | 17.78 | 21.21 | 25.67 | 27.83 |
| GTTP (ms) | - | - | 11.05 | 15.68 | 29.66 | 31.71 | 17.70 | 19.72 | 25.13 | 27.35 |
| GTTP (ml) | - | - | 07.51 | 08.73 | 23.20 | 21.55 | 09.91 | 10.42 | 17.35 | 18.12 |
| BiDAF (o) | 39.69 | 47.18 | 28.85 | 34.98 | 39.68 | 46.49 | 33.72 | 40.58 | 35.91 | 42.64 |
| BiDAF (ms) | 45.73 | 51.35 | 32.95 | 39.39 | 45.69 | 50.73 | 40.18 | 45.01 | 43.46 | 46.95 |

Table 4.5: Performance of the baseline models on our dataset. The figures on the left in each column indicate scores on single-reference test dataset while the figures on the right denote scores on multi-reference dataset.

| Model | Hum | Apt | Flu | Spec |
|---|---|---|---|---|
| HRED | 3.08 | 2.49 | 2.64 | 2.06 |
| GTTP (o) | 4.10 | 3.73 | 4.03 | 3.33 |
| GTTP (ml) | 2.93 | 2.97 | 3.42 | 2.60 |
| BiDAF (o) | 3.78 | 3.71 | 4.05 | 3.76 |
| BiDAF (ms) | 3.41 | 3.38 | 3.47 | 3.30 |

Table 4.6: Human evaluation results on the model performances.

### Generation based models v/s Span prediction models

We compare the generation based models and span prediction models based only on results in the *oracle* setting. Here, the span based model (BiDAF) outperforms the generation based models (HRED and GTTP). This confirms our belief that the NLG capabilities of current generation based models are far from being acceptable even in case of generate-or-copy modes. This also emphasizes the importance of this data which allows building models that can exploit well-formed sentences in the background knowledge and reproduce them with minor modifications instead of generating them from scratch. While the results for BiDAF are encouraging, we reiterate that it does not scale to longer documents as we were not able to run it in the *mixed-long* setting. We still need better models as BiDAF on the SQuAD dataset gives an F1 of 81.52 % which is much higher than the results on our dataset. Further, it should be noted that using the predicted span as a response is not natural. This is evident from the Human likeliness (Hum) score of GTTP (o) being higher than both the BiDAF models. We need models that can suitably alter the span to retain the coherence of the context.

### Effect of including background knowledge

We observe that there is not much difference between the performance of HRED which does not use any background knowledge when compared to GTTP (ml) which actually uses a lot of background knowledge. However, there is a substantial difference between the performance of HRED and GTTP (o) which uses only the relevant background knowledge. Further, without background knowledge, HRED learns to produce generic responses with Spec score = 2.06. This shows that through background knowledge is important, the models should learn to focus on the right background knowledge relevant to the current context. Alternately, we can have a two-stage network that first predicts the right resource from which the span should be selected and then selects the span from this chosen resource.

### *Oracle* v/s *mixed-short* resource

We observe that the performance of BiDAF (ms) is actually better than BiDAF (o) even when the resource length for both is 256 words. We would expect poor performance for BiDAF (ms) as the resource has more noise because of the sentences from irrelevant resources. However, we speculate that the model learns to regard irrelevant sentences as noise and learns to focus on sentences corresponding to the correct resource resulting in improved performance. However, this is only a hypothesis and it needs to be verified. We realize that this is clearly a poor baseline and we need better span prediction based models that can work with longer documents. At the same time, GTTP (o) and GTTP (ms) have comparable, yet poor, performance. There is no cross attention mechanism in this model which can effectively filter out noisy sentences.

### Observations from the copy-and-gen model:

We observed that this model produced sentences where on average of 82.18% (*oracle*) and 71.95% (*mixed-long*) of the tokens were copied. One interesting observation we made was that it easily learns to copy longer contiguous sequences one word at a time. However, in many cases, the “copied” spans are not relevant to the current context.

### Evaluating with multiple references

When considering multiple references, the performance numbers as reported in Table 4.5 indeed improve. This shows the importance of having multiple references and the need to develop metrics that account for multiple dissimilar references.

### Sample responses produced by the models

As seen from Table 4.7, HRED was not able to produce responses that correspond to the given movie or the given context as it lacks any notion of background knowledge associated with it. In Example 1, we can clearly see that only GTTP (o) matches the ground truth. The remaining three models produce varied outputs which are still relevant to the context. In Example 2, we observe that prediction based models produce appropriate recommendations because of better context-document mapping mechanisms. Both the GTTP models produce responses that are copied but irrelevant to the context. At the same time, just producing spans without any structure isn't natural. This explains the need for hybrid models. Example 3 asks for the backstory of a character that requires complex reasoning. The model has to first understand the plot of the movie to locate the sentences which talk about that character's past. As seen from the responses of the given models, all of them except GTTP (ml) pick sentences that are relevant to the character but do not answer the required question. These models rely on word-overlap and thus possess limited natural language understanding.

## Summary

In this chapter, we described in detail the procedure for creating our movie conversations dataset. To demonstrate the usefulness of our dataset, we evaluated our dataset on models based on three paradigms: (1) Generation based models, (2) Copy-or-generate models, and (3) Span prediction models. We observed that all these paradigms have certain limitations such as the span-prediction models are not scalable while the performance of generation-based models is far from acceptable. In the next chapter, we will improve the copy-or-generate baseline by incorporating structural information present in the external knowledge resource.

| Movie | Example 1 (Creed) | Example 2 (Ted) | Example 3 (Jaws) |
|---|---|---|---|
| Context | S2: I liked the one in which Rocky trains and runs.<br>S1: It was great when the match commences, and although Donnie takes a few hits, he defeats Leo.<br>S2: Creed goes the distance as an exceptional crowd pleaser.<br>S1: I agree with grit, style, and substance. There's no other way to say it; Creed is a knockout from start to finish. | S2: The scene where they're yelling at each other out the window was gold<br>S1: Thats a similar scene from family guy. I couldnt stop laughing.<br>S2: I thought every scene was gold. I also couldn't believe Liam Neeson was in the movie.<br>S1: I will love to pick something similar this evening. Do you have a recommendation? | S1: What do you think about the characters in this movie?<br>S2: My favorite character was Brody, he was such a fun character.<br>S1: I didn't get Quint's backstory. |
| Ground Truth | It wholeheartedly captures what was so brilliant about the first film: the characters. | Yes. You can try 22 Jump Street. | Quint, aboard that ship in World War II when it was sunk by torpedoes, had witnessed the deaths of roughly 800 men that day, many of whom were eaten by sharks as they struggled helplessly in the water. |
| HRED | yeah , but as the ark is launched , and you are right taxi driver , runs with the beauty , <UNK><UNK>and <UNK><UNK>. | the plot is a classic in the film , it 's not that it 's an original and cinematic experience . | i 'm not ashamed to admit it , mary poppens is bangin ! ! ! |
| BiDAF(o) | I can't say enough great things about writer / director Ryan Coogler. | The Hangover | Hooper notices the shark beginning to circle the boat, and Quint rushes out for a look. |
| BiDAF(ms) | I loved how they introduced each opponent with his record and accomplishments. | 21 Jump Street | The mood suddenly darkens as Quint admits that the former tattoo represented the US Navy cruiser |
| GTTP (o) | it wholeheartedly captures what was so brilliant about the first film : the characters . | $ 218,628,680 i think . | brody is stunned and alerts quint . |
| GTTP (ml) | i can't say enough great things about writer / director ryan coogler . | this movie had way too much product placement . | jaws' is the original summer blockbuster |

Table 4.7: Examples produced by various models. S1 denotes Speaker1 while S2 denotes Speaker2. 'o','ml' and 'ms' represent *oracle*, *mixed-long* and *mixed-short* versions of the dataset respectively.

# CHAPTER 5

# Incorporating Structural Information to Improve Dialogue Response Generation

In chapter 4, we established the importance of building conversation systems that can effectively incorporate external information. The baseline results on our dataset clearly indicate a scope for improving the generation based models. In this work, we explore the avenue of explicitly incorporating structural information within a document to improve the performance of generation-based models.

While using chat specific background knowledge eliminates the retrieval step from a global knowledge resource, finding relevant pieces of information within the document and then aggregating them to produce meaningful responses is indeed difficult. Intuitively, it is clear that any model for this task should exploit the structural and semantic information within documents and sentences. For illustration, consider the chat shown in Figure 5.1 from our dataset. In this example, Speaker 1 nudges Speaker 2 to talk about how James's wife was irritated because of his career. The right response to this conversation comes from the line beginning at "*His wife Mae* . . . ". However, to generate this response, it is essential to understand that *His* refers to James from the previous sentence; James Braddock, Russel Crowe, and boxer refer to the same person; and other linguistic properties such as *quit* and *he would stop* mean the same.

We can incorporate such structural information in the form of dependency, entity co-occurrence and co-reference resolution graphs using the popular adaption of GCNs for NLP tasks.

This chapter focuses on designing architectures that can exploit such structural information along with semantic and sequential word order information. To that end, we propose a simple encoder framework which combines semantic information from word embeddings, sequential word order information by using LSTMs and structural information by using GCNs. We conduct a series of experiments and arrive at interesting observations. We discuss these observations in section 5.3.

**Source Doc:**
. . . At this point James Braddock was a light heavyweight boxer, who was forced to retired from the ring after breaking his hand in his last fight. His wife Mae had prayed for years that he would quit boxing, before becoming permanently injured. . . .

**Conversation:**
**Speaker 1:** Yes very true, this is a real rags to riches story. Russell Crowe was excellent as usual
**Speaker 2:** Russell Crowe owns the character of James Bradock, the unlikely hero who makes the most of his second chance. He's a good fighter turned hack.
**Speaker 1:** Totally! Oh by the way do you remember his wife ... how she wished he would stop
**Speaker 2:** Yes! His wife Mae had prayed for years that he would quit boxing, before becoming permanently injured.

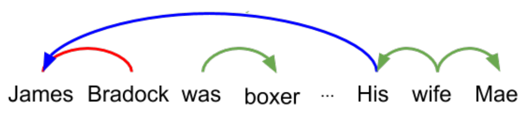


Figure 5.1: A sample conversation from our dataset. For simplicity, we show only a few of the edges. The edge in blue corresponds to the co-reference edge, the edges in green are dependency edges and the edge in red corresponds to the entity edge.

## 5.1 Semantics-Sequences-Structures (SSS) Encoder Framework

In this section, we describe a framework of hybrid encoder architectures in the SSS paradigm that combines our adaptation of GCN for multiple graphs - the Multi-Graph Convolutional Network (M-GCN) with existing models that capture semantics and sequence information. Our SSS encoder framework incorporates (1) semantic information with the usage of word embeddings such as GloVe (Pennington *et al.*, 2014), ELMo (Peters *et al.*, 2018), and BERT (Devlin *et al.*, 2019), (2) sequential word order with bidirectional LSTM, and (3) structural priors such as dependency, co-reference, and entity graphs with M-GCN. The SSS encoder framework aims at providing linguistically rich word representations for the document. We first describe the three components in SSS in detail:

### 5.1.1 Semantics

This component maps raw text to its semantic vector space representation. While we are free to use any word representation, we experiment with three variants of word embeddings described as follows:

**GloVe**: Global Vectors for Word Representation (GloVe) (Pennington *et al.*, 2014) are token level word representations that have been trained on a large token corpus. The model uses both the global matrix factorization and local context window methods. They first collect word co-occurrence statistics in a matrix and then learn this co-occurrence between a word and its surrounding context word. They also include a weighting function to avoid learning only from frequently occurring word pairs. This produces a vector space such that the resulting distance between the words is related to their semantic similarity. The model was trained on five corpora: (1) a 2010 Wikipedia dump with 1 billion tokens, (2) a 2014 Wikipedia dump with 1.6 billion tokens, (3) Gigaword which has 4.3 billion tokens, (4) the combination Gigaword5 + Wikipedia2014, which has 6 billion tokens, and (5) web data from Common Crawl which has 42 billion tokens.

**ELMo**: The ELMo model (Peters *et al.*, 2018) produces deep, contextualized word representations by training stacked convolutional, highway, and BiLSTM layers on a large corpus of text (1B Word Benchmark (Chelba *et al.*, 2014)) with an unsupervised Language Modelling (LM) objective. The first layer is the convolutional layer which computes the character level embeddings of the input tokens. These token representations are then passed through multiple BiLSTM layers, all of which are trained end-to-end to jointly optimize forward and backward LM objectives. The final ELMo embeddings consist of learnable task-specific parameters to obtain a linear combination of the vectors stacked above each input word. This provides the deep contextualized task-specific word representations. They hypothesize that the lower layer states model aspects of the syntax of a sentence while higher level states capture context-dependent aspects of the word meaning. The advantage of using ELMo embeddings over GloVe embeddings is that they provide a representation of single word with respect to the sentence in which it occurs as opposed to the single-sense word embeddings given by GloVe.

**BERT**: BERT (Devlin *et al.*, 2019) produces deep bidirectional representations from the unlabeled text by jointly conditioning on both left and right context in all layers. It uses a multi-layer bidirectional Transformer (Vaswani *et al.*, 2017) as its underlying architecture. BERT obtains bidirectional contextual representation by using a peculiar language modelling objective - masked language model. In this objective function, some percentage of the input tokens are masked at random, and then the task is to predict those masked tokens. In order to improve sentence relationships, the model is also trained in the next sentence prediction task. BERT is trained on BooksCorpus (Zhu *et al.*, 2015*b*) (800M words) and Wikipedia dump. BERT can act as a standalone architecture by just adding a task-specific output layer and fine-tuning. BERT also provides token level features which the authors dub as a "feature-based" approach for a task-specific architecture. We have used the "feature-based" approach in this work. As BERT provides context-dependent representations, it has the same advantages over GloVe listed in the previous sub-section. Moreover, BERT improves over ELMo embeddings by using the bidirectional language modelling objective and Transformer based architecture.

### 5.1.2 Sequences

This component captures sequential word order information. We have already discussed the working of RNN and its variants in section 2.2. We use BiLSTM so that both the left-to-right and right-to-left contextual information gets utilized. We refer to the representation obtained through the Sequences layer as sequential contextualized representations.

### 5.1.3 Structures

This component model structural information within a document/resource which is represented as graphs to enhance representation learning.

The GCN described in equation 2.21 is capable of leveraging information only from one graph. We generalize $G$ to denote a labelled multi-graph, i.e., a graph which can contain multiple (parallel) labelled edges between the same pair of nodes. Let $\mathcal{R}$ denote the set of different graphs (structures) considered and let $G = \{\mathcal{N}_1, \mathcal{N}_2 \dots \mathcal{N}_{|\mathcal{R}|}\}$ be a set of dictionary of neighbors from the $|\mathcal{R}|$ graphs. We propose Multi-Graph Convolutional Network (M−GCN) that extends the Syntactic GCN defined in equation 2.21 to multiple graphs by having $|\mathcal{R}|$ graph convolutions at each layer as given in equation 5.1. Here, $g\_conv(\mathcal{N})$ is the graph convolution defined in Eqn: 2.21 with $\sigma$ as the identity function. In a departure from Marcheggiani and Titov (2017), we remove the individual node $v$ from the neighbourhood list $\mathcal{N}(v)$ and model the node information with a separate weight matrix $W_{self}$.

$$h_v^{(k+1)} = ReLU\big((h_v^{(k)} W_{self}^{(k)} + \sum_{\mathcal{N} \in G} g\_conv(\mathcal{N})\big) \tag{5.1}$$

This formulation is advantageous over having $|\mathcal{R}|$ different GCNs as it can extract information from multi-hop pathways and can use information across different graphs with every GCN layer.For simplicity, we will refer equation as an operation $MGCN(h_i, neighbours(i))$ where $h_i$ is the input to the GCN and $neighbours(i)$ consist of all the neighbours of the word $i$.

### 5.1.4 Configurations

We now describe the three hybrid architectural variants that fit in the SSS Framework. These variants can be well understood in terms of their design combinations which are best illustrated in Figure 5.2.

#### Sequence Contextualized GCN (**Seq−C−GCN**)

Seq−C−GCN (Bastings *et al.*, 2017; Marcheggiani and Titov, 2017; Zhang *et al.*, 2018) is the popular hybrid variant in the literature where M−GCN, originally GCN, is fed with sequential word context features obtained from BiLSTM which were fed with word embeddings $x_i^r$ and hence we dub it as 'Sequence Contextualized GCN'( Seq−C−GCN). Let the encoder states obtained from BiLSTM be denoted by $h_{i_{seq}}^r$ where i denotes the $i^{th}$ token of the resource $r$. We then feed this representation $h_{i_{seq}}^r$ along with the neighbours of the token $i$ to the M−GCN defined in equation 5.1. Thus, this encoder first captures the sequential relationship among the words through BiLSTM and then the structural relationship through M−GCN which is finally represented as $h_i^r$ as shown in

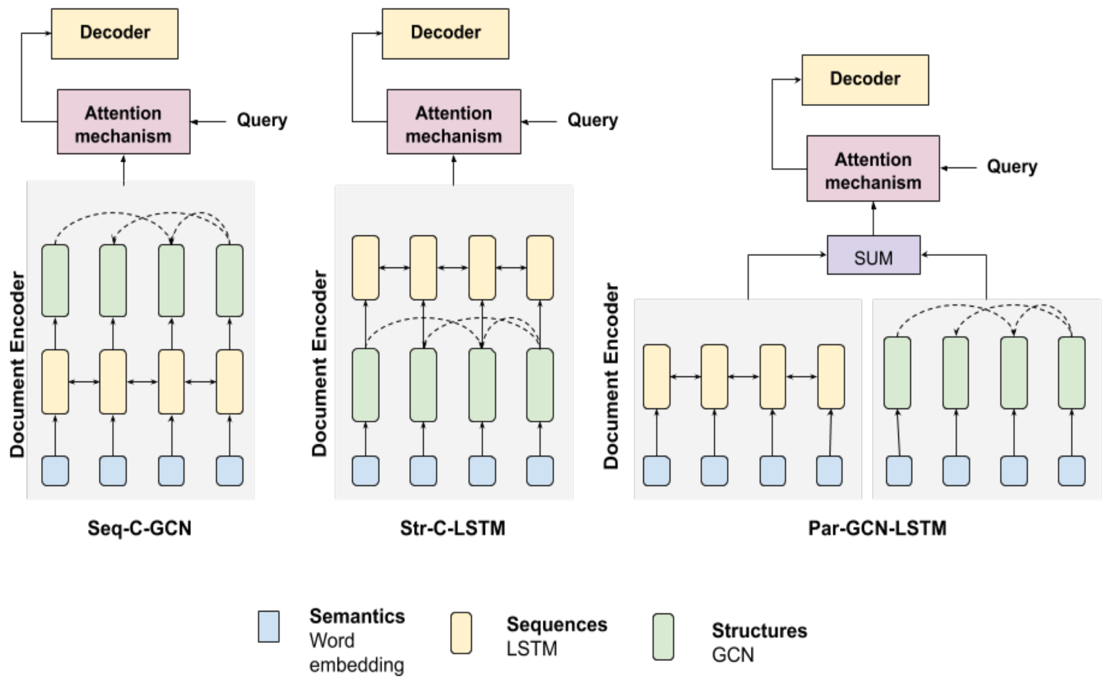


Figure 5.2: Variants in the SSS Framework.

equation 5.2. This feeding of LSTM features to GCN allows 'teleporting'over parts of the sentence through the edges beyond its immediate neighbouring words as explained in Bastings *et al.* (2017).

$$\begin{aligned} h^r_{i_{seq}} &= BiLSTM(h^r_{i-1_{seq}}, x^r_i) \\ h^r_i &= MGCN(h^r_{i_{seq}}, Neighbours(i^r)) \end{aligned} \tag{5.2}$$

### Structure Contextualized LSTM (`Str-C-LSTM`)

Most of the literature in the development of GCN based encoders tries to augment the structural information after obtaining the sequential information through LSTM. `Str-C-LSTM` is a variant where structural features are first learned with `M-GCN` which has semantically rich word embeddings $x^r_i$ as an input, and then this structure contextualized word information $h^r_{i_{str}}$ is passed on to the BiLSTM layer. Hence, we call this model 'Structure Contextualized LSTM'(`Str-C-LSTM`). We obtain the resource token representation as $h^r_i$ as shown in equation 5.3.

$$\begin{aligned} h^r_{i_{str}} &= MGCN(x^r_i, Neighbours(i^r)) \\ h^r_i &= BiLSTM(h^r_{i-1}, h^r_{i_{str}}) \end{aligned} \tag{5.3}$$

**Parallel GCN-LSTM (`Par-GCN-LSTM`)**

`Par-GCN-LSTM` is a variant where structural word features from `M-GCN` and sequential word features from BiLSTM are combined together using a weighted representation. Both `M-GCN` and BiLSTM are fed with word embeddings $x^r_i$ as input. We obtain features from BiLSTM as $h^r_{i_{seq}}$ and features from `M-GCN` as $h^r_{i_{str}}$ respectively. These are finally combined together $h^r_i$ as shown in equation 5.4.

$$\begin{aligned} h^r_{i_{seq}} &= BiLSTM(h^r_{i-1_{seq}}, x^r_i) \\ h^r_{i_{str}} &= MGCN(x^r_i, Neighbours(i^r)) \\ h^r_i &= W_{str}\, h^r_{i_{str}} + W_{seq}\, h^r_{i_{seq}} \end{aligned} \tag{5.4}$$

The final resource representation $h^r_i$ obtained from equations 5.2, 5.3, and 5.4 respectively is further used in equation 5.5.

## 5.2 Experimental setup

In this section, we first describe the pre-processing steps we carried out to obtain different graphs. We then describe the different models evaluated on our dataset along with specific implementation details.

### 5.2.1 Construction of graphs to incorporate structural information

We considered leveraging three different graph-based structural priors for our task. We use dependency graph (`Dep`), entity co-reference graph (`Coref`) and entity co-occurrence graph (`Ent`). The `Ent` and `Coref` graphs, unlike the `Dep` graph, can provide relations that may span across sentences in a document. We used the dependency parser provided by SpaCy[1] to obtain the `Dep` graph. For the construction of the `Coref` graph, we use the NeuralCoref model[2] integrated with SpaCy and connect every word that occurs in the co-reference mention with its antecedent mention. We retain only the nouns in the antecedent mention and connect every word in the cluster with every word in the antecedent mention. For the construction of the entity co-occurrence graph, we first perform named-entity recognition using SpaCy and connect all the entities that lie in the window of $k = 20$ words as stated in Song *et al.* (2018).

### 5.2.2 Models

We consider the GTTP model in section 4.2.2 as our base architecture. All the models in this section provide different ways to obtain $h_i^r$ *i.e.*, the resource representation. The context representation and the decoder remain the same for all the setups. We modify the attention equation described in equation 4.1 to:

$$\begin{aligned} e_i^t &= v^T tanh(W_r h_i^r + U s_t + V c_t + b_r) \\ a^t &= softmax(e^t) \\ r_t &= \sum_i a_i^t h_i^r \end{aligned} \tag{5.5}$$

where $c_t$ is the attended context representation. Thus, at every decoder time-step, the attention on the document words is also based on the currently attended context representation. Additionally, $c_t$ was removed from the $p_{gen}$ equation. The new equation is

$$p_{gen} = \sigma(W_r^T r_t + W_s^T s_t + W_x^T x_t + b_g) \tag{5.6}$$

We study the performances of the following models on the oracle setup:

**(1) Semantics (`Sem`):** We provide a baseline *Sem* that uses only the semantic information. This model conditions the decoder only on the word embeddings. We experi-

[1]https://spacy.io/
[2]https://github.com/huggingface/neuralcoref

ment with GloVe, ELMo, and BERT embeddings.

**(2) Semantics+Sequences (`Sem+Seq`):** To capture semantics and sequence information together, we use the standard LSTM based encoder along with word embeddings.

**(3) Semantics+Sequences+Structures (`Sem+Seq+Str`):** We experiment on all the three variants of the Semantics-Sequences-Structures framework described in section 5.1, *i.e.*, `Seq-C-GCN`, `Str-C-LSTM`, and `Par-GCN-LSTM`. We evaluate these family of models on all four sets of graphs, viz: (1) dependency graph `Dep`, (2) dependency and entity co-occurrence graph `Dep+Ent`, (3) dependency graph and co-reference graph `Dep+Coref`, and (4) all three graphs `Dep+Ent+Coref`.

### 5.2.3 Implementation details

To incorporate ELMo[3] and BERT embeddings[4] in our architecture, we used the implementation provided in Tensorflow hub. The GloVe embeddings were obtained directly from the website[5]. We used the BERT-base model for obtaining the BERT embeddings. We used the feature based-approach mentioned in Devlin *et al.* (2019) that does not require fine-tuning of the encoder layers as we deal with a task specific architecture. We do a task-specific fine-tuning for the ELMo embeddings *i.e.*, we learn a linear combination of the encoding layers as well as fine-tuned GloVe embeddings during training.

For all the models which incorporate structural priors, we search for optimal number ($K$) of `M-GCN` layers to consider based on the performance on the validation set. We evaluated up to $K = 3$ hops for all the models and observed that performance either saturates or deteriorates after $K = 3$ hops. We also developed a multi-GPU version of the code using Horovod (Sergeev and Del Balso, 2018) as all the experiments that used ELMo embeddings and some of the experiments with BERT embeddings required computation beyond a single GPU (1080Ti) with batch size 64.

We use BLEU-4 (Papineni *et al.*, 2002) and ROUGE-L (Lin, 2004) as our evaluation metrics.

[3]`https://tfhub.dev/google/elmo/2`
[4]`https://bit.ly/2JVd9Jl`
[5]`https://stanford.io/2Gdv8uo`

#### Hyper-parameters

We selected the hyper-parameters using the validation set. In particular, we used Adam optimizer with a learning rate of 0.0004 and a batch size of 64. We used GloVe embeddings of size 100. For the RNN-based encoders and decoders, we used LSTMs with a hidden state of size 256. To avoid the problem of exploding gradients, we used gradient clipping with a maximum gradient norm of 2. We used a hidden state of size 512 for `Seq-C-GCN` and 128 for the remaining GCN-based encoders. We ran all the experiments for 15 epochs and we used the checkpoint with the least validation loss for testing.

For models using ELMo embeddings, a learning rate of 0.004 was most effective. For the BERT-based models, a learning rate of 0.0004 was suitable. Rest of the hyper-parameters and other setup details remain the same for experiments with BERT and ELMo.

## 5.3 Results and Discussion

In this section, we organize our observations under distinct headings. We then discuss the merits of the `SSS` paradigm followed by the evaluation of different hybrid architectures that efficiently materialize the `SSS` paradigm. Then, we analyze the importance of incorporating various linguistic structures in these models.

### 5.3.1 Quantitative Evaluation

We compare the results of the `SSS` Framework against the popular dialogue generation baselines in Table 5.1. S2S refers to the vanilla Sequence-to-Sequence model discussed in section 2.2.3 and does not use any background knowledge for response generation. We use the results of HRED and GTTP, discussed in detail in section 4.2, directly, from table 4.5. For a fair comparison with the dialogue generation baselines, we evaluate the BiDAF model predictions against the ground truth responses instead of spans within the responses. As it can be seen from Table 5.1, our `SSS` framework outperforms the popular dialogue generation baselines.

| Model | BLEU-4 | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|---|
| S2S | 4.63 | 26.91 | 9.34 | 21.58 |
| HRED | 5.23 | 24.55 | 7.61 | 18.87 |
| GTTP | 13.97 | 36.17 | 24.84 | 31.07 |
| BiDAF | 16.79 | 26.73 | 18.82 | 23.58 |
| SSS (GloVe) | 18.96 | 38.61 | 26.92 | 33.77 |
| SSS (ELMo) | 19.32 | 39.65 | 27.37 | 34.86 |
| SSS (BERT) | **22.78** | **40.09** | **27.83** | **35.20** |

Table 5.1: Comparing the performance of `SSS` Framework against dialogue generation baselines.

### 5.3.2 Building the `SSS` paradigm

We report the benefits of incorporating sequential and structural information along with different semantic information in the `SSS` paradigm in Table 5.2. The best *Sem+Seq+Str* results across different graph combinations as well as different contextual and structural infusion are reported.

| Emb | Paradigm | BLEU-4 | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|---|---|
| GloVe | Sem | 4.4 | 29.72 | 11.72 | 22.99 |
|  | Sem+Seq | 14.83 | 36.17 | 24.84 | 31.07 |
|  | SSS | 18.96 | 38.61 | 26.92 | 33.77 |
| ELMo | Sem | 14.36 | 32.04 | 18.75 | 26.71 |
|  | Sem+Seq | 14.61 | 35.54 | 24.58 | 30.71 |
|  | SSS | 19.32 | 39.65 | 27.37 | 34.86 |
| BERT | Sem | 11.26 | 33.86 | 16.73 | 26.44 |
|  | Sem+Seq | 18.49 | 37.85 | 25.32 | 32.58 |
|  | SSS | 22.78 | 40.09 | 27.83 | 35.2 |

Table 5.2: Performance of components within the SSS Framework.

#### Comparison of different semantic information (ELMo v/s BERT v/s GloVe)

We observe that the performance using only the *Sem* model for ELMo and BERT is far superior than the GloVe embeddings. This confirms the original hypothesis of the respective works that deep contextualized word representations are better than non contextualized word representations. Just as ELMo and BERT have shown empirical improvements over lexicalized word representations for several NLU tasks, our experiments suggest a similar trends for tasks with a NLG component.

### Incorporating sequential information with semantic information

We now add sequential contextual information to the semantic information in the *Sem+Seq* models. As expected providing contextual information through LSTM helps in improving performance for all the three models. The gain in ELMo model is not significant because the underlying ELMo architecture already has two BiLSTM layers. This suggests that the addition of one more LSTM layer may not contribute to learning any new sequential word information that has not already been captured by the previous layers.

### Incorporation of structural information is indeed useful

It is clear from Table 5.2 that models that explicitly incorporate structural priors obtain a significant boost in performance. All the three embeddings seem to benefit from the incorporation of structural priors - we obtain an improvement of 4.13%, 4.71%, and 4.29%, on GloVe, ELMo, and BERT over their respective *Sem+Seq* baselines on BLEU-4 metric and an improvement of 2.7%, 4.15% and 2.62% on the ROUGE-L metric. This confirms our base hypothesis that explicitly incorporating structural information with sequential and semantic information is indeed essential.

## 5.3.3 Combining structural and sequential information

Having established that using structural priors can provide us with richer representations, we now examine which hybrid architectures discussed in Figure 5.2 provides the best way to incorporate structural information with sequence information. Table 5.3 reports the best performance of each hybrid architecture across multiple linguistic structures with different semantic information.

| Sem | Seq-C-GCN | | | | Str-C-LSTM | | | | Parl-GCN-LSTM | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BLEU | ROUGE | | | BLEU | ROUGE | | | BLEU | ROUGE | | |
| | | 1 | 2 | L | | 1 | 2 | L | | 1 | 2 | L |
| GloVe | 15.61 | 36.6 | 24.54 | 31.68 | 18.96 | 38.61 | 26.92 | 33.77 | 17.1 | 37.04 | 25.70 | 32.2 |
| ELMo | 18.44 | 37.92 | 26.62 | 33.05 | 19.32 | 39.65 | 27.37 | 34.86 | 16.35 | 37.28 | 25.67 | 32.12 |
| BERT | 20.43 | 40.04 | 26.94 | 34.85 | 22.78 | 40.09 | 27.83 | 35.20 | 21.32 | 39.9 | 27.60 | 34.87 |

Table 5.3: Performance of different hybrid architectures to combine structural information with sequence information.

**Use of LSTM layer at the end**:

The response generation task of our dataset is a span based generation task where phrases of text are expected to be copied or generated as is. Sequential information is thus crucial to reproduce these long phrases from external resources. In such cases where sequence information from LSTM seems to be necessary, having a GCN layer on top of that may modify the input information.

We suspect that this issue is strongly reflected in the results in Table 5.3. From the table, it is clear that Structure Contextualized LSTM, `Str-C-LSTM` which has LSTM layer on top of GCN layers performs the best on all the three embeddings followed by `Par-GCN-LSTM` which has both GCN and LSTM layers at the top obtaining the second-best performance on GloVe and BERT. `Seq-C-GCN` model performs the worst among all the three as its GCN layer at the top is likely to modify the sequence information from the LSTM.

### 5.3.4 Understanding the effect of structural information

We have established that the incorporation of structural information is beneficial and we have discussed ways to effectively combine structural and sequential information. We now explore the effect of structural priors in Table 5.4.

| Graph | GloVe | | | | ELMo | | | | BERT | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BLEU | ROUGE | | | BLEU | ROUGE | | | BLEU | ROUGE | | |
| | | 1 | 2 | L | | 1 | 2 | L | | 1 | 2 | L |
| Dep | 16.79 | 37.77 | 25.89 | 32.88 | 17.00 | 37.56 | 26.14 | 32.77 | 22.78 | 40.09 | 27.83 | 35.2 |
| Dep+Ent | 14.44 | 35.14 | 24.61 | 30.43 | 18.34 | 39.55 | 28.00 | 34.76 | 19.33 | 39.37 | 27.52 | 34.33 |
| Dep+Coref | 16.58 | 37.60 | 25.72 | 32.63 | 18.56 | 40.08 | 28.42 | 35.06 | 20.99 | 40.10 | 28.66 | 35.11 |
| Dep+Ent +Coref | 18.96 | 38.61 | 26.92 | 33.77 | 19.32 | 39.65 | 27.37 | 34.86 | 20.37 | 39.11 | 27.2 | 34.19 |

Table 5.4: Comparing performance of different structural priors across different semantic information on the `Str-C-LSTM` architecture.

**Which structural priors are important?**

While a combination of intra-sentence and inter-sentence graphs is helpful across all the models, the best performing model with BERT embeddings relies only on the dependency graph as seen in Table 5.4. In the case of GloVe based experiments, the entity and co-reference relations were not independently useful but when used together gave a significant performance boost. However, most of the BERT based and ELMo based models achieved competitive performance with the individual entity and co-reference graphs. There is no clear trend across the models. Hence, probing these embedding models is essential to identify which structural information is captured implicitly by the embeddings and which structural information needs to be added explicitly.

## Do we need linguistic priors?

We analyze if the structural priors defined by dependency, entity, and co-reference graphs actually contribute to understanding natural language. We conducted two sets of experiments using the `SSS` encoders with GloVe embeddings. In the first experiment, we shuffled the dependency edges within the sentences for all the documents. The results in Table 5.5 suggest that these arbitrary relations amount to adding noise and thus reduce the performance as compared to the respective experiments with the dependency graph. Thus, using syntax information is crucial.

| Model | BLEU-4 | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|---|
| Seq-C-GCN | 14.04 | 33.66 | 21.22 | 28.83 |
| Str-C-LSTM | 15.21 | 35.29 | 22.77 | 30.54 |
| Parl-GCN-LSTM | 13.19 | 32.28 | 20.2 | 27.58 |

Table 5.5: Effect of using arbitrary intra-sentence edges.

In the second experiment, we retained the correct dependency edges but added an equal number of inter-sentence arbitrary edges to check if specific inter-sentence edges were crucial to the performance improvement. From Table 5.6, we observe that the performance is poorer than the corresponding `Dep+Ent+Coref` experiments. Thus, even the inter-sentence priors need to possess interesting language properties. The slight performance improvement for `Str-C-LSTM` and `Par-GCN-LSTM` over Table 5.5 indicates improved learning in the presence of correct syntax information.

| Model | BLEU-4 | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|---|
| Seq-C-GCN | 14.04 | 33.66 | 21.22 | 28.83 |
| Str-C-LSTM | 15.84 | 36.25 | 24.89 | 31.28 |
| Parl-GCN-LSTM | 14.62 | 35.21 | 24.63 | 30.49 |

Table 5.6: Effect of using inter-sentence arbitrary edges along with dependency edges.

### 5.3.5 Effectiveness of deep contextualized word representations in modelling structural information

The authors of ELMo (Peters *et al.*, 2018) suggest that the lower level states can capture syntactic information while the higher level states capture context dependent relational information amongst words. Thus, the *Sem+Seq* ELMo model can be considered as a baseline that can implicitly capture structural information. When compared to our best performing *Sem+Seq+Str* model using GloVe embeddings, it is clear that ELMo is not sufficient to capture these structural priors implicitly. A similar trend was observed in Strubell and McCallum (2018) where ELMo outperformed its comparative GloVe baseline, but the model which incorporated structural information explicitly for semantic role-labelling performed even better.

Similarly, BERT embeddings which are based on the Transformer architecture (Vaswani *et al.*, 2017) where every token is represented by attending to other tokens also implicitly captures contextual relational information. Follow-up work on BERT embeddings (Goldberg, 2019) suggests that these embeddings can also capture syntactic information to a certain extent. Hence, we can consider the *Sem+Seq* BERT model to implicitly capture structural information. Our best performing model using GloVe embeddings performs slightly better than this model suggesting that BERT embeddings could still benefit from explicit structural information. Further, our best performing model with GloVe word embeddings for incorporating semantic information has lesser memory footprint owing to fewer model parameters (layers) in comparison to both ELMo based and BERT based models.

To the best of our knowledge, the only work that suggests a similar hypothesis for ELMo embeddings is Strubell and McCallum (2018). Ours is the first work to report gains for a natural language generation task for ELMo and BERT embeddings by using explicit structural information. This opens up for interesting avenues to probe and understand the extent to which the deep contextualized word representations learn the syntax and other linguistic properties. Also, this calls for an improved design for language modelling objectives during the pre-training of these embeddings that can incorporate structural information.

### 5.3.6 Qualitative Evaluation

We conducted human evaluation for the *Sem+Seq+Str* by comparing the respective generated outputs against the outputs generated by the baseline. We use the results of GTTP(o) from Table 4.5 as our baseline.We randomly sampled 100 outputs and showed them to three different annotators. The annotators were allowed to pick which response was most appropriate for the given context of the conversation by choosing from four options: A, B, both, and none. The evaluators were not given any information about the nature of the experiments and were asked to consider these conversations as conversations between friends. From the options given by the three evaluators, majority of vote was taken and ties were broken randomly. We report these results in Table 5.7. As seen from the table, only the outputs by ELMo based model are qualitatively similar to the baseline model.

| Sem | Win | Loss | Same | None |
|---|---|---|---|---|
| SSS (GloVe) | 24 | 17 | 47 | 12 |
| SSS (ELMo) | 22 | 23 | 41 | 14 |
| SSS (BERT) | 29 | 25 | 29 | 17 |

Table 5.7: Human evaluation of models within SSS Framework against GTTP baseline.

We supported our analysis in the previous section through quantitative evaluation. We discuss some examples to identify the strengths and weaknesses of our approach.

We find that the `SSS` Framework improves over the GTTP baseline for the cases of opening statements discussed in section 4.1.3. GTTP baseline had confusion in picking opening statements and often mixed the responses for “Which is your favorite character?", “Which is your favorite scene” and “What do you think about the movie?”. These account for roughly 20% of the examples in the dataset. The responses to these questions have different syntactic structures - “My favorite character is XYZ”, “I liked the one in which XYZ”, and “ I think this movie is XYZ” where XYZ was the crowd-sourced phrase respectively. We believe the presence of dependency graphs over the respective sentences helps to alleviate the confusion. Please refer to *Rocky V* example in Table 5.8.

To illustrate the importance of presence of inter-sentence level graphs, we refer the reader the example under *Hannibal* in Table 5.8. We find that the presence of a co-reference graph between “Anthony Hopkins” in the first sentence and “he” in the second sentence can help in continuing the conversation on the actor “Anthony Hopkins”. Moreover, connecting tokens in “Anthony Hopkins” to refer to “he” in the second sentence is possible because of the explicit entity-entity connection between the two tokens.

There is a limited diversity of responses generated by the `SSS` Framework as it often resorts to the patterns seen during training while it is not copying from the background knowledge. We also identify that `SSS` Framework cannot handle the cases where Speaker2 initiates a context switch, *i,e;* when Speaker2 introduces a topic that has not been discussed in the conversation so far. In the chat on *The Road Warrior* in Table 5.8, we find that *Mad Max: Fury Road* has been used to initiate a discussion that compares the themes of both the movies. All the models produce irrelevant responses.

We speculate that the `SSS` Framework helps in improving the informativeness of the response by providing more suitable snippets from the background knowledge than other dialogue generation baselines. While we observe a performance improvement due to the explicit addition of structural information, aspects of human creativity such as diversity, context switch, and common-sense reasoning require more sophisticated approaches.

## Summary

In this chapter, we developed the `SSS` Framework that can combine semantic, sequential, and structural information present in the resource. To capture the structural information from multiple graphs simultaneously, we proposed the Multi-GCN architecture. We conducted experiments using the variants of the `SSS` Framework. We empirically observed that using structural information explicitly indeed improves the dialogue response generation.

| Movie | Rocky V | Hannibal | The Road Warrior |
|---|---|---|---|
| Resource | Rocky V, terribly underrated. I liked the one in which Rocky .... . My favorite character was Adrian because she was ...I think it was enjoyable, though slightly less than th. | Anthony Hopkins gave .... However, the material he was given to work .... almost seemed like an excuse to present us | Box Office: $ 9,003,011 Similar Movies: Lethal Weapon, Mad Max: Fury Road ... |
| Chat | Speaker 1: Which is your favourite character in this? | Speaker 1: What is your opinion about the movie?<br>Speaker 2: I think it was a fair sequel.<br>Speaker 1: Anthony Hopkins gave an impeccable performance. | Speaker 1: Which is your favourite character in this?<br>Speaker 2: My favorite character was Dog<br>Speaker 1: I liked the feral kid even after he's found in the car and Max evicts him without ceremony he doesn't give up. |
| Ground Truth | my favorite character was adrian because she was strong and did her best to keep her family together through tough times . | however , the material he was given to work with was not as good as silence of the lambs . | Have you seen the new Mad Max: Fury Road? |
| Baseline | i liked the one in which rocky and gunn fight in the street . | in fairness , perhaps there was no way it could be . | lethal weapon . |
| SSS(GloVe) | my favorite character was adrian because she was strong and did her best to keep her together through tough times . | however , the material he was given to work with was not as good as silence of the lambs . | it made $ 9,003,011 |
| SSS(ELMo) | my favorite character was adrian because she was strong and did her best to keep her family together through tough times . | the material he was given to work with was not as good as silence of the lambs . | [UNK] only one man can make the difference in the future . |
| SSS(BERT) | my favorite character was adrian because she was strong and did her best to keep her family together through tough times . | the material he was given to work with was not as good as silence of the lambs . | yes .[UNK] only one man can make the difference in the future . |

Table 5.8: Sample outputs from the SSS Framework compared with baseline and ground truth responses.

# CHAPTER 6

# Conclusion and Future Work

We contributed to the development of domain specific conversation systems by introducing a new dataset for building dialog systems which would hopefully allow the community to take a fresh look at this task. Unlike existing datasets that only contain a sequence of utterances, in our dataset, each response is explicitly linked to some background knowledge. This mimics how humans converse by recalling information from their background knowledge and use it appropriately in the context of the conversation. Using this dataset, we evaluated models belonging to three different paradigms, *viz.*, generation based models, generate-or-copy models and span prediction models. Our results suggest that the NLG capabilities of existing Seq2Seq models are far from desirable, while the span based models that completely bypass the process of NLG show some promise but with a clear scope for improvement.

We demonstrated the usefulness of incorporating structural information for the task of background aware dialogue response generation. We combined the structural information explicitly in the standard semantic+sequential model and observed a performance boost. We studied different structural priors and reported that both intra-sentence and inter sentence-level structures are highly beneficial when used with the right combination of sequential and structural information (in our task, structural followed by sequential). We also observed that the explicit incorporation of structural information improves the performance of architectures which already use richer deep contextualized representations.

## 6.1 Future Work

In terms of architectural development, we would like to build models that are a hybrid of span prediction models and generation models. Specifically, we would like to build models that can learn to copy a large sequence from the input instead of one word at a time. Another important aspect is to build less complex models that can handle longer documents. For example, the BiDAF model has an expensive outer product between two large matrices which makes it infeasible for long documents (because the size of these matrices grows with the length of the document). Alternatively, we would like to build two-stage models that first select the correct resource from which the next response is to be generated and then generate or copy the response from the resource.

We observed certain interesting phenomena in the collected chats such as Speaker2 generating context switch, diversity while generating the response for movie recommendation, use of commonsense and world knowledge beyond the information present in the background knowledge and so on. We believe our and other corpora can be studied for such interesting cases and work in the direction of incorporating these aspects of human creativity in existing architectures.

Our dataset collection was performed in a restricted setting wherein all the responses were taken from background resources. Going further, we would like to build datasets that have a mix of generic responses and responses obtained from background knowledge. Additionally, we would like to encourage paraphrasing of the background knowledge instead of copying it verbatim. Further, we would also like to have situations where a single response is generated using multiple background resources.

We evaluated the quality of our corpus on five different metrics - intelligible, coherent, grammatical, on topic and natural two person chats. We also evaluated our baseline models on aspects such as fluency, appropriateness, human-likeliness, and specificity of responses. Existing metrics at the chat-level as well as response level still rely on word-overlap and fail to capture these important characteristics. Hence, there is a necessity in developing better evaluation metrics.

We observed that extensive experimentation and architecture search is required to obtain state-of-the-art performance on our dataset and other tasks. Instead, it would be interesting to develop new neural building blocks for NLP that would implicitly capture the structural information. A promising direction is mentioned in the Ordered Neuron LSTM (Shen *et al.*, 2019) which introduces inductive bias in the standard LSTM architecture to capture the hierarchical structure implicitly.

The goal of building intelligent machines which can learn to converse is indeed fascinating. This work attempts to contribute to the advancement of this elusive goal.

# REFERENCES

1. **Asri, L. E.**, **H. Schulz**, **S. Sharma**, **J. Zumer**, **J. Harris**, **E. Fine**, **R. Mehrotra**, and **K. Suleman**, Frames: a corpus for adding memory to goal-oriented dialogue systems. *In Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, 207–219. 2017.

2. **Bahdanau, D.**, **K. Cho**, and **Y. Bengio**, Neural machine translation by jointly learning to align and translate. *In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015.

3. **Banchs, R. E.**, Movie-dic: a movie dialogue corpus for research and development. *In The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 2: Short Papers*, 203–207. 2012.

4. **Banerjee, S.**, **N. Moghe**, **S. Arora**, and **M. M. Khapra**, A dataset for building code-mixed goal oriented conversation systems. *In Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2018.

5. **Bastings, J.**, **I. Titov**, **W. Aziz**, **D. Marcheggiani**, and **K. Sima'an**, Graph convolutional encoders for syntax-aware neural machine translation. *In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, 1957–1967. 2017.

6. **Bordes, A.**, **Y. Boureau**, and **J. Weston**, Learning end-to-end goal-oriented dialog. *In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

7. **Cao, N. D.**, **W. Aziz**, and **I. Titov**, Question answering by reasoning across documents with graph convolutional networks. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 2306–2317. 2018.

8. **Chelba, C.**, **T. Mikolov**, **M. Schuster**, **Q. Ge**, **T. Brants**, **P. Koehn**, and **T. Robinson**, One billion word benchmark for measuring progress in statistical language modeling. *In INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, 2635–2639. 2014.

9. **Chen, T.** and **M. Kan** (2013). Creating a live, public short message service corpus: the NUS SMS corpus. *Language Resources and Evaluation*, **47**(2), 299–335.

10. **Colby, K. M.** (1981). Modeling a paranoid mind. *Behavioral and Brain Sciences*, **4**(4), 515–534.

11. **Deerwester, S. C.**, **S. T. Dumais**, **T. K. Landauer**, **G. W. Furnas**, and **R. A. Harshman** (1990). Indexing by latent semantic analysis. *JASIS*, **41**(6), 391–407.

12. **Devlin, J.**, **M. 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 4171–4186. 2019.

13. **Dinan, E.**, **S. Roller**, **K. Shuster**, **A. Fan**, **M. Auli**, and **J. Weston** (2019). Wizard of wikipedia: Knowledge-powered conversational agents. *International Conference on Learning Representations*.

14. **Dodge, J.**, **A. Gane**, **X. Zhang**, **A. Bordes**, **S. Chopra**, **A. H. Miller**, **A. Szlam**, and **J. Weston**, Evaluating prerequisite qualities for learning end-to-end dialog systems. *In 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. 2016.

15. **Duchi, J.**, **E. Hazan**, and **Y. Singer** (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, **12**(Jul), 2121–2159.

16. **Elman, J. L.** (1990). Finding structure in time. *Cognitive Science*, **14**(2), 179–211.

17. **Eric, M.**, **L. Krishnan**, **F. Charette**, and **C. D. Manning**, Key-value retrieval networks for task-oriented dialogue. *In Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, 37–49. 2017.

18. **Eric, M.** and **C. Manning**, A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 468–473. Association for Computational Linguistics, Valencia, Spain, 2017.

19. **Fang, H.**, **H. Cheng**, **M. Sap**, **E. Clark**, **A. Holtzman**, **Y. Choi**, **N. A. Smith**, and **M. Ostendorf**, Sounding board: A user-centric and content-driven social chatbot. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. 2018.

20. **Forsythand, E. N.** and **C. H. Martell**, Lexical and discourse analysis of online chat dialog. *In Proceedings of the First IEEE International Conference on Semantic Computing (ICSC 2007), September 17-19, 2007, Irvine, California, USA*, 19–26. 2007.

21. **Ghazvininejad, M.**, **C. Brockett**, **M. Chang**, **B. Dolan**, **J. Gao**, **W. Yih**, and **M. Galley**, A knowledge-grounded neural conversation model. *In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 5110–5117. 2018.

22. **Goldberg, Y.** (2019). Assessing bert's syntactic abilities. *Technical Report*.

23. **Gu, J.**, **Z. Lu**, **H. Li**, and **V. O. K. Li**, Incorporating copying mechanism in sequence-to-sequence learning. *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. 2016.

24. **Guss, W. H.**, **J. Bartlett**, **P. Kuznetsov**, and **P. Patil** (2017). Eigen: A step towards conversational ai. *Alexa Prize Proceedings*.

25. **He, S.**, **C. Liu**, **K. Liu**, and **J. Zhao**, Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, 199–208. 2017.

26. **Henderson, M.**, **B. Thomson**, and **J. D. Williams**, The second dialog state tracking challenge. *In Proceedings of the SIGDIAL 2014 Conference, The 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 18-20 June 2014, Philadelphia, PA, USA*, 263–272. 2014*a*.

27. **Henderson, M.**, **B. Thomson**, and **J. D. Williams**, The third dialog state tracking challenge. *In 2014 IEEE Spoken Language Technology Workshop, SLT 2014, South Lake Tahoe, NV, USA, December 7-10, 2014*, 324–329. 2014*b*.

28. **Hermann, K. M.**, **T. Kociský**, **E. Grefenstette**, **L. Espeholt**, **W. Kay**, **M. Suleyman**, and **P. Blunsom**, Teaching machines to read and comprehend. *In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 1693–1701. 2015.

29. **Hill, F.**, **A. Bordes**, **S. Chopra**, and **J. Weston**, The goldilocks principle: Reading children's books with explicit memory representations. *In 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. 2016.

30. **Joshi, M.**, **E. Choi**, **D. S. Weld**, and **L. Zettlemoyer**, Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, 1601–1611. 2017.

31. **Khatri, C.**, **B. Hedayatnia**, **A. Venkatesh**, **J. Nunn**, **Y. Pan**, **Q. Liu**, **H. Song**, **A. Gottardi**, **S. Kwatra**, and **S. Pancholi** (2018). Advancing the State of the Art in Open Domain Dialog Systems through the Alexa Prize. *arXiv e-prints*, arXiv:1812.10757.

32. **Kim, Y.**, Convolutional neural networks for sentence classification. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751. Association for Computational Linguistics, Doha, Qatar, 2014.

33. **Kingma, D. P.** and **J. Ba**, Adam: A method for stochastic optimization. *In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015.

34. **Kipf, T. N.** and **M. Welling**, Semi-supervised classification with graph convolutional networks. *In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

35. **Krause, B.**, **M. Damonte**, **M. Dobre**, **D. Duma**, **J. Fainberg**, **F. Fancellu**, **E. Kahembwe**, **J. Cheng**, and **B. L. Webber** (2017). Edina: Building an open domain socialbot with self-dialogues. *Alexa Prize Proceedings*.

36. **Kuncoro, A.**, **C. Dyer**, **J. Hale**, **D. Yogatama**, **S. Clark**, and **P. Blunsom**, Lstms can learn syntax-sensitive dependencies well, but modeling structure makes them better. *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, 1426–1436. 2018.

37. **Le, P.** and **W. Zuidema**, Compositional distributional semantics with long short term memory. *In Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, 10–19. 2015.

38. **Li, J.**, **M. Galley**, **C. Brockett**, **J. Gao**, and **B. Dolan**, A diversity-promoting objective function for neural conversation models. *In NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, 110–119. 2016*a*.

39. **Li, J.**, **M. Galley**, **C. Brockett**, **G. P. Spithourakis**, **J. Gao**, and **W. B. Dolan**, A persona-based neural conversation model. *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. 2016*b*.

40. **Li, J.**, **W. Monroe**, **A. Ritter**, **D. Jurafsky**, **M. Galley**, and **J. Gao**, Deep reinforcement learning for dialogue generation. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, 1192–1202. 2016*c*.

41. **Lin, C.-Y.**, Rouge: A package for automatic evaluation of summaries. *In Proc. ACL workshop on Text Summarization Branches Out*, 10. 2004.

42. **Lin, X. V.**, **C. Wang**, **L. Zettlemoyer**, and **M. D. Ernst**, NL2Bash: A corpus and semantic parser for natural language interface to the linux operating system. *In Proceedings of the 11th Language Resources and Evaluation Conference*. European Language Resource Association, Miyazaki, Japan, 2018.

43. **Linzen, T.**, **E. Dupoux**, and **Y. Goldberg** (2016). Assessing the ability of lstms to learn syntax-sensitive dependencies. *TACL*, **4**, 521–535.

44. **Lowe, R.**, **N. Pow**, **I. Serban**, **L. Charlin**, and **J. Pineau**, Incorporating unstructured textual knowledge sources into neural dialogue systems. *In Neural Information Processing Systems Workshop on Machine Learning for Spoken Language Understanding*. 2015*a*.

45. **Lowe, R.**, **N. Pow**, **I. Serban**, and **J. Pineau**, The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *In Proceedings of the SIGDIAL 2015 Conference, The 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 2-4 September 2015, Prague, Czech Republic*, 285–294. 2015*b*.

46. **Marcheggiani, D.**, **J. Bastings**, and **I. Titov**, Exploiting semantics in neural machine translation with graph convolutional networks. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, 486–492. 2018.

47. **Marcheggiani, D.** 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*, 1506–1515. 2017.

48. **McCulloch, W. S.** and **W. Pitts**, Neurocomputing: Foundations of research. chapter A Logical Calculus of the Ideas Immanent in Nervous Activity. MIT Press, Cambridge, MA, USA, 1988. ISBN 0-262-01097-6, 15–27.

49. **Metallinou, A.**, **D. Bohus**, and **J. D. Williams**, Discriminative state tracking for spoken dialog systems. *In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, 466–475. 2013.

50. **Mikolov, T.**, **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 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 3111–3119. 2013.

51. **Nallapati, R.**, **F. Zhai**, and **B. Zhou**, Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. *In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, 3075–3081. 2017.

52. **Nallapati, R.**, **B. Zhou**, **C. dos Santos**, **Ç. Gulçehre**, and **B. Xiang**, Abstractive text summarization using sequence-to-sequence RNNs and beyond. *In Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 280–290. Association for Computational Linguistics, Berlin, Germany, 2016.

53. **Nguyen, T.**, **M. Rosenberg**, **X. Song**, **J. Gao**, **S. Tiwary**, **R. Majumder**, and **L. Deng**, MS MARCO: A human generated machine reading comprehension dataset. *In Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016.*. 2016.

54. **Onishi, T.**, **H. Wang**, **M. Bansal**, **K. Gimpel**, and **D. A. McAllester**, Who did what: A large-scale person-centered cloze dataset. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, 2230–2235. 2016.

55. **Papineni, K.**, **S. Roukos**, **T. Ward**, and **W. Zhu**, Bleu: a method for automatic evaluation of machine translation. *In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA.*, 311–318. 2002.

56. **Pascanu, R.**, **T. Mikolov**, and **Y. Bengio**, On the difficulty of training recurrent neural networks. *In Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, 1310–1318. 2013.

57. **Peng, N.**, **H. Poon**, **C. Quirk**, **K. Toutanova**, and **W. Yih** (2017). Cross-sentence n-ary relation extraction with graph lstms. *TACL*, **5**, 101–115.

58. **Pennington, J.**, **R. Socher**, and **C. D. Manning**, Glove: Global vectors for word representation. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, 1532–1543. 2014.

59. **Peters, M.**, **M. Neumann**, **M. Iyyer**, **M. Gardner**, **C. Clark**, **K. Lee**, and **L. Zettlemoyer**, Deep contextualized word representations. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237. Association for Computational Linguistics, 2018.

60. **Qazvinian, V.**, **D. R. Radev**, **S. M. Mohammad**, **B. Dorr**, **D. Zajic**, **M. Whidby**, and **T. Moon** (2014). Generating Extractive Summaries of Scientific Paradigms. *arXiv e-prints*, arXiv:1402.0556.

61. **Rajpurkar, P.**, **J. Zhang**, **K. Lopyrev**, and **P. Liang**, Squad: 100, 000+ questions for machine comprehension of text. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, 2383–2392. 2016.

62. **Ram, A.**, **R. Prasad**, **C. Khatri**, **A. Venkatesh**, **R. Gabriel**, **Q. Liu**, **J. Nunn**, **B. Hedayatnia**, **M. Cheng**, **A. Nagar**, **E. King**, **K. Bland**, **A. Wartick**, **Y. Pan**, **H. Song**, **S. Jayadevan**, **G. Hwang**, and **A. Pettigrue** (2017). Conversational AI: the science behind the alexa prize. *Alexa Prize Proceedings*.

63. **Raux, A.**, **B. Langner**, **D. Bohus**, **A. W. Black**, and **M. Eskénazi**, Let's go public! taking a spoken dialog system to the real world. *In INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, September 4-8, 2005*, 885–888. 2005.

64. **Ritter, A.**, **C. Cherry**, and **B. Dolan**, Unsupervised modeling of twitter conversations. *In Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, 172–180. 2010.

65. **Ritter, A.**, **C. Cherry**, and **W. B. Dolan**, Data-driven response generation in social media. *In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, 583–593. 2011.

66. **Rojas-Barahona, L. M.**, **M. Gasic**, **N. Mrksic**, **P. Su**, **S. Ultes**, **T. Wen**, **S. J. Young**, and **D. Vandyke**, A network-based end-to-end trainable task-oriented dialogue system. *In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, 438–449. 2017.

67. **Rosenblatt, F.** (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65–386.

68. **Rosset, S.** and **S. Petel**, The ritel corpus - an annotated human-machine open-domain question answering spoken dialog corpus. *In Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC 2006, Genoa, Italy, May 22-28, 2006.*, 1640–1643. 2006.

69. **Rumelhart, D. E.**, **G. E. Hinton**, and **R. J. Williams** (1986). Learning representations by back-propagating errors. *Nature*, **323**, 533–536.

70. **Schallert, D. L.** (2002). Schema theory. *Literacy in America: An encyclopedia of history, theory, and practice Santa Barbara, CA*, 556–558.

71. **See, A.**, **P. J. Liu**, and **C. D. Manning**, Get to the point: Summarization with pointer-generator networks. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, 1073–1083. 2017.

72. **Seneff, S.**, **J. R. Glass**, **D. Goddeau**, **D. Goodine**, **L. Hirschman**, **H. C. Leung**, **M. S. Phillips**, **J. Polifroni**, and **V. Zue**, Development and preliminary evaluation of the MIT ATIS system. *In Speech and Natural Language, Proceedings of a Workshop held at Pacific Grove, California, USA, February 19-22. 1991*. 1991.

73. **Seo, M. J.**, **A. Kembhavi**, **A. Farhadi**, and **H. Hajishirzi**, Bidirectional attention flow for machine comprehension. *In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

74. **Serban, I. V.**, **T. Klinger**, **G. Tesauro**, **K. Talamadupula**, **B. Zhou**, **Y. Bengio**, and **A. C. Courville**, Multiresolution recurrent neural networks: An application to dialogue response generation. *In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, 3288–3294. 2017*a*.

75. **Serban, I. V.**, **C. Sankar**, **M. Germain**, **S. Zhang**, **Z. Lin**, **S. Subramanian**, **T. Kim**, **M. Pieper**, **S. Chandar**, **N. R. Ke**, **S. Mudumba**, **A. de Brébisson**, **J. Sotelo**, **D. Suhubdy**, **V. Michalski**, **A. Nguyen**, **J. Pineau**, and **Y. Bengio** (2017*b*). The octopus approach to the alexa competition: A deep ensemble-based socialbot. *Alexa Prize Proceedings*.

76. **Serban, I. V.**, **A. Sordoni**, **Y. Bengio**, **A. C. Courville**, and **J. Pineau**, Building end-to-end dialogue systems using generative hierarchical neural network models. *In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, 3776–3784. 2016.

77. **Serban, I. V.**, **A. Sordoni**, **R. Lowe**, **L. Charlin**, **J. Pineau**, **A. C. Courville**, and **Y. Bengio**, A hierarchical latent variable encoder-decoder model for generating dialogues. *In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, 3295–3301. 2017*c*.

78. **Sergeev, A.** and **M. Del Balso** (2018). Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv e-prints*, arXiv:1802.05799.

79. **Shang, L.**, **Z. Lu**, and **H. Li**, Neural responding machine for short-text conversation. *In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, 1577–1586. 2015.

80. **Shen, Y.**, **S. Tan**, **A. Sordoni**, and **A. Courville**, Ordered neurons: Integrating tree structures into recurrent neural networks. *In International Conference on Learning Representations*. 2019.

81. **Socher, R.**, **C. C. Lin**, **C. Manning**, and **A. Y. Ng**, Parsing natural scenes and natural language with recursive neural networks. *In Proceedings of the 28th international conference on machine learning (ICML-11)*, 129–136. 2011.

82. **Song, L.**, **Z. Wang**, **M. Yu**, **Y. Zhang**, **R. Florian**, and **D. Gildea** (2018). Exploring Graph-structured Passage Representation for Multi-hop Reading Comprehension with Graph Neural Networks. *arXiv e-prints*, arXiv:1809.02040.

83. **Sordoni, A.**, **M. Galley**, **M. Auli**, **C. Brockett**, **Y. Ji**, **M. Mitchell**, **J. Nie**, **J. Gao**, and **B. Dolan**, A neural network approach to context-sensitive generation of conversational responses. *In NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, 196–205. 2015.

84. **Srivastava, N.**, **G. E. Hinton**, **A. Krizhevsky**, **I. Sutskever**, and **R. Salakhutdinov** (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, **15**(1), 1929–1958.

85. **Srivastava, R. K.**, **K. Greff**, and **J. Schmidhuber** (2015). Highway Networks. *arXiv e-prints*, arXiv:1505.00387.

86. **Strubell, E.** and **A. McCallum**, Syntax helps elmo understand semantics: Is syntax still relevant in a deep neural architecture for srl? *In Proceedings of the Workshop on the Relevance of Linguistic Structure in Neural Architectures for NLP*, 19–27. 2018.

87. **Sukhbaatar, S.**, **A. Szlam**, **J. Weston**, and **R. Fergus**, End-to-end memory networks. *In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2440–2448. 2015.

88. **Sutskever, I.**, **O. Vinyals**, and **Q. V. Le**, Sequence to sequence learning with neural networks. *In Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 3104–3112. 2014.

89. **Tai, K. S.**, **R. Socher**, and **C. D. Manning**, Improved semantic representations from tree-structured long short-term memory 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)*, volume 1, 1556–1566. 2015.

90. **Tan, C.**, **F. Wei**, **N. Yang**, **B. Du**, **W. Lv**, and **M. Zhou**, S-net: From answer extraction to answer synthesis for machine reading comprehension. *In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 5940–5947. 2018.

91. **Uthus, D. C.** and **D. W. Aha**, The ubuntu chat corpus for multiparticipant chat analysis. *In Analyzing Microtext, Papers from the 2013 AAAI Spring Symposium, Palo Alto, California, USA, March 25-27, 2013*. 2013.

92. **Vashishth, S.**, **S. S. Dasgupta**, **S. N. Ray**, and **P. Talukdar**, Dating documents using graph convolution networks. *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1605–1615. Association for Computational Linguistics, 2018.

93. **Vaswani, A.**, **N. Shazeer**, **N. Parmar**, **J. Uszkoreit**, **L. Jones**, **A. N. Gomez**, **L. Kaiser**, and **I. Polosukhin**, Attention is all you need. *In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 6000–6010. 2017.

94. **Vinyals, O.**, **M. Fortunato**, and **N. Jaitly**, Pointer networks. *In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2692–2700. 2015.

95. **Vinyals, O.** and **Q. V. Le**, A neural conversational model. *In ICML Deep Learning Workshop*. 2015.

96. **Vlad Serban, I.**, **R. Lowe**, **P. Henderson**, **L. Charlin**, and **J. Pineau** (2015). A Survey of Available Corpora for Building Data-Driven Dialogue Systems. *arXiv e-prints*, arXiv:1512.05742.

97. **Walker, M. A.**, **J. E. F. Tree**, **P. Anand**, **R. Abbott**, and **J. King**, A corpus for research on deliberation and debate. *In Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, 812–817. 2012.

98. **Wang, S.** and **J. Jiang**, Learning natural language inference with LSTM. *In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1442–1451. Association for Computational Linguistics, San Diego, California, 2016.

99. **Wang, S.** and **J. Jiang**, Machine comprehension using match-lstm and answer pointer. *In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

100. **Wang, W.**, **N. Yang**, **F. Wei**, **B. Chang**, and **M. Zhou**, Gated self-matching networks for reading comprehension and question answering. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, 189–198. 2017.

101. **Weizenbaum, J.** (1966). Eliza- a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, **9**(1), 36–45.

102. **Werbos, P. J.** (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, **78**(10), 1550–1560.

103. **Williams, J. D.**, **A. Raux**, and **M. Henderson** (2016). The dialog state tracking challenge series: A review. *D&D*, **7**(3), 4–33.

104. **Williams, J. D.**, **A. Raux**, **D. Ramachandran**, and **A. W. Black**, The dialog state tracking challenge. *In Proceedings of the SIGDIAL 2013 Conference, The 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 22-24 August 2013, SUPELEC, Metz, France*, 404–413. 2013.

105. **Williams, J. D.** and **S. J. Young** (2007). Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, **21**(2), 393–422.

106. **Xiong, C.**, **V. Zhong**, and **R. Socher**, Dynamic coattention networks for question answering. *In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

107. **Yang, Z.**, **B. Dhingra**, **Y. Yuan**, **J. Hu**, **W. W. Cohen**, and **R. Salakhutdinov**, Words or characters? fine-grained gating for reading comprehension. *In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017.

108. **Young, S. J.** (2000). Probabilistic methods in spoken–dialogue systems. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, **358**(1769), 1389–1402.

109. **Zeiler, M. D.** (2012). ADADELTA: An Adaptive Learning Rate Method. *arXiv e-prints*, arXiv:1212.5701.

110. **Zhang, Y.**, **P. Qi**, and **C. D. Manning**, Graph convolution over pruned dependency trees improves relation extraction. *In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, 2205–2215. 2018.

111. **Zhou, K.**, **S. Prabhumoye**, and **A. W. Black**, A dataset for document grounded conversations. *In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, 708–713. 2018.

112. **Zhu, X.**, **P. Sobhani**, and **H. Guo**, Long short-term memory over recursive structures. *In Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 1604–1612. 2015*a*.

113. **Zhu, Y.**, **R. Kiros**, **R. S. Zemel**, **R. Salakhutdinov**, **R. Urtasun**, **A. Torralba**, and **S. Fidler**, Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *In 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, 19–27. 2015*b*.

# LIST OF PAPERS BASED ON THESIS

**Published**

1. Nikita Moghe, Siddhartha Arora, Suman Banerjee and Mitesh M. Khapra, *Towards Exploiting Background Knowledge for Building Conversation Systems.* In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018.

**Communicated**

1. Nikita Moghe, Priyesh Vijayan, Balaraman Ravindran and Mitesh M. Khapra, *On Incorporating Structural Information to Improve Dialogue Response Generation.* Under review at AAAI 2020.