

Université de Montréal

Neural Approaches to Dialog Modeling.

par

Chinnadhurai Sankar

Département d'Informatique et de Recherche Opérationnelle

Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures et postdoctorales

en vue de l'obtention du grade de

Philosophiæ Doctor (Ph.D.)

en Informatique

août 2020

Sommaire

Cette thèse par article se compose de quatre articles qui contribuent au domaine de l'apprentissage profond, en particulier dans la compréhension et l'apprentissage des approches neuronales des systèmes de dialogue.

Le premier article fait un pas vers la compréhension si les architectures de dialogue neuronal couramment utilisées capturent efficacement les informations présentes dans l'historique des conversations. Grâce à une série d'expériences de perturbation sur des ensembles de données de dialogue populaires, nous constatons que les architectures de dialogue neuronal couramment utilisées comme les modèles seq2seq récurrents et basés sur des transformateurs sont rarement sensibles à la plupart des perturbations du contexte d'entrée telles que les énoncés manquants ou réorganisés, les mots mélangés, etc.

Le deuxième article propose d'améliorer la qualité de génération de réponse dans les systèmes de dialogue de domaine ouvert en modélisant conjointement les énoncés avec les attributs de dialogue de chaque énoncé. Les attributs de dialogue d'un énoncé se réfèrent à des caractéristiques ou des aspects discrets associés à un énoncé comme les actes de dialogue, le sentiment, l'émotion, l'identité du locuteur, la personnalité du locuteur, etc.

Le troisième article présente un moyen simple et économique de collecter des ensembles de données à grande échelle pour modéliser des systèmes de dialogue orientés tâche. Cette approche évite l'exigence d'un schéma d'annotation d'arguments complexes. La version initiale de l'ensemble de données comprend 13 215 dialogues basés sur des tâches comprenant six domaines et environ 8 000 entités nommées uniques, presque 8 fois plus que l'ensemble de données MultiWOZ populaire.

Le dernier article présente une méthode sans intégration pour calculer les représentations de mots à la volée. Cette approche réduit considérablement l'empreinte mémoire, ce qui facilite le déploiement sur les périphériques (contraintes de mémoire) sur l'appareil. En

plus d'être indépendante de la taille du vocabulaire, nous trouvons que cette approche est intrinsèquement résistante aux fautes d'orthographe courantes.

Mots-clés: systèmes de dialogue axés sur les tâches, actes de dialogue, hachage sensible à la localité, auto-attention, inférence en langage naturel, analyse des sentiments, graphique de calcul dynamique, réseaux récurrents, réseaux récursifs, réseaux de neurones, apprentissage profond, naturel traitement du langage, apprentissage par renforcement, apprentissage automatique.

Summary

This thesis by article consists of four articles which contribute to the field of deep learning, specifically in understanding and learning neural approaches to dialog systems.

The first article takes a step towards understanding if commonly used neural dialog architectures effectively capture the information present in the conversation history. Through a series of perturbation experiments on popular dialog datasets, we find that commonly used neural dialog architectures like recurrent and transformer-based seq2seq models are rarely sensitive to most input context perturbations such as missing or reordering utterances, shuffling words, etc.

The second article introduces a simple and cost-effective way to collect large scale datasets for modeling task-oriented dialog systems. This approach avoids the requirement of a complex argument annotation schema. The initial release of the dataset includes 13,215 task-based dialogs comprising six domains and around 8k unique named entities, almost 8 times more than the popular MultiWOZ dataset.

The third article proposes to improve response generation quality in open domain dialog systems by jointly modeling the utterances with the dialog attributes of each utterance. Dialog attributes of an utterance refer to discrete features or aspects associated with an utterance like dialog-acts, sentiment, emotion, speaker identity, speaker personality, etc.

The final article introduces an embedding-free method to compute word representations on-the-fly. This approach significantly reduces the memory footprint which facilitates deployment in on-device (memory constraints) devices. Apart from being independent of the vocabulary size, we find this approach to be inherently resilient to common misspellings.

Keywords: task-oriented dialog systems, dialog-acts, multiwoz, locality sensitive hashing, wizard-of-oz, self-attention, natural language inference, sentiment analysis, dynamic

computational graph, recurrent networks, recursive networks, neural networks, deep learning, natural language processing, reinforcement learning, machine learning.

Contents

Sommaire	iii
Summary	v
List of tables	xiii
List of figures	xv
Acknowledgement	xvii
Chapter 1. Introduction	1
Chapter 2. Background : Dialog	3
2.1. Types of Dialog systems	4
2.1.1. Task Oriented Dialog Systems.....	4
2.1.2. Open-Ended Dialog Systems	6
Chapter 3. Basic Neural Network Architectures	9
3.1. Feed Forward Networks	9
3.2. Recurrent Neural Architectures	10
3.2.1. Long-Short Term Memory units.....	10
3.2.2. Gated Recurrent Unit	11
3.3. Deep Learning For NLP.....	13
3.3.1. Language modeling	13
3.3.2. Sequence to Sequence Models	13
3.3.3. Attention Mechanism	14
3.3.4. Transformers.....	15

Chapter 4. Neural Approaches for End-to-End Dialog modeling	19
4.1. Sequence to Sequence approaches	19
4.2. The Hierarchical Recurrent Encoder-Decoder	20
Chapter 5. Background : Efficient Locality Sensitive Hashing (LSH)-based Text Representation	25
5.1. Introduction	25
5.2. Motivation.....	27
Chapter 6. Prologue to First Article	29
6.1. Article Details	29
6.2. Context.....	29
6.3. Contributions.....	30
6.4. Recent Developments	30
Chapter 7. Do Neural Dialog Systems Use the Conversation History Effectively? An Empirical Study	31
7.1. Abstract	31
7.2. Introduction	31
7.3. Related Work.....	33
7.4. Experimental Setup.....	34
7.4.1. Datasets	35
7.4.2. Types of Perturbations.....	35
7.4.3. Models	36
7.5. Results & Discussion.....	37
7.6. Conclusion.....	39

Chapter 8. Prologue to Second Article	41
8.1. Article Details	41
8.2. Context.....	41
8.3. Contributions.....	42
8.4. Recent Developments	42
Chapter 9. Taskmaster-1: Toward a Realistic and Diverse Dialog Dataset.	43
9.1. Abstract	43
9.2. Introduction	44
9.3. Related work	45
9.3.1. Human-machine vs. human-human dialog.....	45
9.3.2. The Wizard of Oz (WOz) Approach and MultiWOZ	46
9.4. The Taskmaster Corpus.....	47
9.4.1. Overview	47
9.4.2. Two-person, spoken dataset	50
9.4.2.1. WOz platform and data pipeline.....	50
9.4.2.2. Agents, workers and training.....	51
9.4.3. Self-dialogs (one-person written dataset)	54
9.4.3.1. Task scenarios and instructions	54
9.4.3.2. Pros and cons of self-dialogs	54
9.4.4. Annotation	55
9.5. Dataset Analysis.....	56
9.5.1. Self-dialogs vs MultiWOZ.....	56
9.5.2. Self-dialogs vs Two-person	57
9.5.3. Baseline Experiments: Response Generation.....	57
9.5.4. Baseline Experiments: Argument Prediction.....	59

9.6. Conclusion.....	59
Chapter 10. Prologue to third Article.....	61
10.1. Article Details.....	61
10.2. Context.....	61
10.3. Contributions.....	62
10.4. Recent Developments.....	62
Chapter 11. Deep Reinforcement Learning For Modeling Chit-Chat Dialog With Discrete Attributes.....	63
11.1. Abstract.....	63
11.2. Introduction.....	63
11.2.1. Contributions.....	65
11.3. Attribute Conditional HRED.....	66
11.3.1. Dialog Attribute Prediction.....	67
11.3.2. Conditional Response Generation.....	68
11.3.3. RL for Dialog Attribute Prediction.....	69
11.4. Training Setup.....	71
11.5. Experimental Results.....	72
11.5.1. Dialog Attribute Prediction.....	73
11.5.2. Utterance Evaluation.....	74
11.5.3. RL For Dialog Attribute Prediction.....	76
11.6. Related Work.....	78
11.7. Conclusion.....	80
Chapter 12. Prologue to Fourth Article.....	81
12.1. Article Details.....	81

12.2.	Context	81
12.3.	Contributions	82
12.4.	Recent Developments	82
Chapter 13.	Transferable Neural Projection Representations	83
13.1.	Abstract	83
13.2.	Introduction	83
13.3.	Neural Projection Model	85
13.3.1.	Vanilla Skip-Gram Model	85
13.3.2.	Neural Projection Skip-Gram (NP-SG)	85
13.3.3.	Training NP-SG Model	86
13.3.4.	Discriminative NP-SG Models	87
13.3.5.	Improved NP-SG Training	87
13.4.	Training Setup	88
13.4.1.	Dataset	88
13.4.2.	Implementation Details	88
13.5.	Experiments	89
13.5.1.	Qualitative Evaluation and Results	89
13.5.2.	Quantitative Evaluation and Results	90
13.5.2.1.	Similarity Task	90
13.5.2.2.	Language Modeling	91
13.5.2.3.	Text Classification	92
13.6.	Conclusion	92
Chapter 14.	Conclusion	93
Bibliography	97

List of tables

7.1	An example of an LSTM seq2seq model with attention’s insensitivity to shuffling of words in the dialog history on the DailyDialog dataset.....	32
7.2	Model performance across multiple datasets and sensitivity to different perturbations. Columns 1 & 2 report the test set perplexity (without perturbations) of different models. Columns 3-7 report the increase in perplexity ($\Delta PPL_{[\sigma]}$) when models are subjected to different perturbations. The mean (μ) and standard deviation $[\sigma]$ across 5 runs are reported. The model that exhibits the highest sensitivity (higher the better) to a particular perturbation on a dataset is in bold. <i>seq2seq_lstm_att</i> are the most sensitive models 24/40 times, while transformers are the least with 6/40 times.	36
7.3	Model performance across multiple datasets and sensitivity to different perturbations. Columns 1 & 2 report the test set perplexity (without perturbations) of different models. Columns 3-7 report the increase in perplexity when models are subjected to different perturbations. The mean (μ) and standard deviation $[\sigma]$ across 5 runs are reported. The <i>Only Last</i> column presents models with only the last utterance from the dialog history. The model that exhibits the highest sensitivity (higher the better) to a particular perturbation on a dataset is in bold. <i>seq2seq_lstm_att</i> are the most sensitive models 24/40 times, while transformers are the least with 6/40 times.	38
9.1	Statistics comparison: Self-dialogs vs MultiWOZ corpus both containing approximately 10k dialogues each.	46
9.2	Statistics comparison: Self-dialogs vs two person corpus both containing 5k dialogs. Perplexity and BLEU are reported for Transformer baseline. Joint-Perplexity and	

	Joint-BLEU are perplexity/BLEU scores from the joint training of self-dialogs and two-person but evaluated with their respective test sets.	56
9.3	Evaluation of various seq2seq architectures [156] on our self-dialog corpus using both automatic evaluation metrics and human judgments. Human evaluation ratings in the 1-5 LIKERT scale (higher the better), and human ranking are averaged over 500 x 3 ratings (3 crowdsourced workers per rating).	58
9.4	Inter-Annotator Reliability scores of seq2seq model responses computed for 500 self-dialogs from the test set, each annotated by 3 crowdsourced workers.	59
9.5	API Argument prediction accuracy for Self-dialogs. API arguments are annotated as spans in the utterances.	59
11.1	Dialog-acts prediction accuracy in Reddit validation set.	73
11.2	Dialog-acts prediction accuracy for classifiers trained on validation set of different datasets.	74
11.3	Perplexity and Embedding Metrics for the Reddit validation set.	74
11.4	Validation Perplexity for the Open-Subtitles dataset.	75
11.5	Human Evaluation results: <i>Seq2Seq+Attr vs Seq2Seq</i>	76
11.6	Sample conversations	77
11.7	Diversity scores on the Open-Subtitles validation set after RL fine-tuning	77
11.8	Human Evaluation results: <i>RL vs Seq2Seq+Attr</i>	78
11.9	Percentage of generic responses after RL fine-tuning.	78
11.10	Sample conversations	79
13.1	Similarity Tasks: # of params, 100k vocabulary size for skipgram baseline, 100 embedding size.	89
13.2	Sampled nearest neighbors for NP-SG.	90

List of figures

2.1	4
3.1	Deep fully-connected feed forward network.....	9
3.2	Recurrent Neural Network (RNN).....	10
3.3	Sequence to Sequence Encoder-Decoder Architecture.....	14
4.1	The computational graph of the HRED architecture [141] for a sample dialog composed of three turns. The token encoder provides a representation for each utterance and the context encoder provides the condensed representation for the previous utterances. The decoder predicts tokens conditioned additionally on this context embedding.	21
5.1	Memory for V look-up vectors for each token vs storing $K(\ll V)$ vectors and linearly combining them for token representation. We consider $K = 1120$ following [124] in this paper.	26
5.2	Binary Locality-Sensitive Hashing (LSH) projection representation for text.	27
7.1	The increase in perplexity for different models when only presented with the k most recent utterances from the dialog history for Dailydialog (left) and bAbI dialog (right) datasets. Recurrent models with attention fare better than transformers, since they use more of the conversation history.	34
9.1	Sample Taskmaster-1 two-person dialog.....	48
9.2	Sample instructions for agents playing "assistant" role.....	49
9.3	Sample instructions for crowdsourced workers playing "user" role.....	50
9.4	Sample instructions for written "self-dialogs".....	52

9.5	Sample one-person, written dialog.....	53
9.6	Indicating transaction status with “accept” or “reject”.....	55
11.1	Dialog attribute classification: We predict the dialog attribute of the next utterance based on the previous context and attributes corresponding to the previous utterances. Please note that we depict only a single attribute for convenience.....	67
11.2	Attribute Conditional HRED : Token generation is additionally conditioned on the predicted dialog attributes. The dialog attribute’s embedding is concatenated with the context vector.	68
13.1	Neural Projection Skip-gram (NP-SG) model.....	86

Acknowledgement

There are a lot of people who helped me in various aspects of the research I conducted, as well as being supportive during the past five years.

First, I would like to thank my advisor Yoshua Bengio for his great advice and support. His optimism, enthusiasm and research freedom helped me explore ambitious research plans during my PhD. I would like to thank the committee members for suggesting thoughtful experiments to analyze and improve the methods presented in the thesis. I would like to thank members of MILA: Sarath Chandar, Julian Serban, Sandeep Subramanian, Srinivas Venkattaramanujam, Zhouhan Lin, Alex Lamb, Taesup Kim, Sai Rajeswar, Saizheng Zhang, Prasanna Parthasarathi, Guillaume Alain, Hugo Larochelle, Aaron Courville from whom I have learned lot about research on a diverse set of topics. Special thanks to all the staff members at Mila who made my Ph.D. life much easier: Julie Mongeau, and Linda Peinthere, Frédéric Bastien, Myriam Cote, Jocelyne Etienne and others.

I spent a considerable amount of time during my PhD at Google Brain/AI where I had great mentors, Sujith Ravi and Zornitsa Kozareva during internships. I thank the Google Brain/AI team for providing an excellent atmosphere to learn about many topics in neural networks and different tasks. Discussing research with fellow internship friends: Arvind Neelakantan, Semih Yavuz, Bill Byrne, Da Cheng Juan, Gaurav Menghani and others was hugely beneficial. I would like to thank Layla El Asri for hosting me at Microsoft Research and support me with my dataset collection project.

I would like to thank my brother, Jeevan who introduced me to Deep Learning and encouraged me to pursue a PhD on the topic. My parents, Sankar, and Mala have made many sacrifices to make sure that I get the best education and they derive immense joy from my academic progress. Finally, I would like to thank my wife, Prathusha who has been incredibly supportive of me and has always believed in me.

Chapter 1

Introduction

Advances in deep learning-based speech recognition systems [10], with improved Automatic Speech Recognition (ASR) modules [135], resulted in a flurry of voice operated personal-assistant bots like Apple’s Siri, Amazon’s Alexa, Google Home and Microsoft’s Cortana, to name a few. Such systems have enjoyed reasonable success in replacing the traditional touch interface, used by most applications, with voice/speech signals. Over the past few years, "Chatbots" have been successfully commercialized in several areas, with the most success in e-commerce, agenda/scheduling, and event/travel reservations. This also led to the development of similar applications aimed at people with disabilities like autism, visually impairment, etc. Unlike traditional rule-based learning systems which are often restricted to specific domains and require handcrafted rules, deep learning approaches are data driven, scalable and don’t usually require domain experts to make commercialization viable.

Deep Learning [44] has been extremely effective in machine learning sub-fields like speech recognition & synthesis [10], and natural language processing (NLP) tasks like language modeling [100], machine translation [8, 184] and abstractive summarization [134], etc. Naturally, there have been several research efforts [127, 168, 83, 80, 141, 142] in deep learning research community in training end-to-end neural architectures for modeling conversations.

However, the success in applying end-to-end deep learning based approaches in the real world dialog tasks has been slightly underwhelming. While these models have demonstrated the ability to generate fluent responses, they still lack the ability to “understand” and produce incoherent responses. They often produce boring and repetitive responses like “Thank you.” [79, 136] or meander away from the topic of conversation. Fortunately, the research community has been very active in deeply understanding the root cause of such issues [81, 147],

organizing challenges [135, 38], releasing effective large scale datasets [84, 93, 20] and research frameworks like ParlAI [106], ConvLab [76], Plato [111], etc.

In this thesis, we present four articles exploring neural approaches to modeling dialog systems. Chapter 2 provides the basics of dialog and introduces the types of dialog systems. Chapters 3 - 5 introduces the basic neural architectures used in the current dialog systems research.

Chapter 2

Background : Dialog

This chapter covers the basics needed to understand the motivation and solutions proposed in this thesis. Dialog systems are conversational systems which respond to queries or messages posed by humans, in a meaningful natural language form. Entities take turns to answer to messages based on the previous context.

A dialog typically consists of multiple turns. Each dialog turn may contain multiple utterances spoken by the same user. Dialogs can be either dyadic - for example, conversations between a customer service representative and a customer - or may involve more than two people, like Reddit forums. Dialogs can involve users interacting through written language, spoken language, or in a multi-modal setting (e.g. using both speech and visual modalities). There is usually a very weak correlation between the word usage statistics of spoken and text-based conversations. Spoken language tends to be less formal, containing lower information content and many more pronouns than written language [21].

Dialog systems have been an active research topic from the early 60s [162, 25, 174, 7, 97, 146, 67, 17, 87, 167, 126, 149, 15, 68]. Predominately, the hand engineered rule-based approaches were the ones that were popular and practical, among the ideas put forward. Advances in machine learning systems and computing power, thanks to Moore's law, paved the way for successful statistical machine learning approaches, embedded with hand engineered rules, to gather necessary attention [155, 60, 16, 180]. Recent progress in the last years in both Automatic Speech Recognition (ASR) and Question Answering (QA) systems opens the way to a new generation of dialog systems allowing a human user to, through a phone or in front of a screen, ask a computer for information on any subject.

2.1. Types of Dialog systems

Dialog systems can be commonly grouped into two categories based on most uses-cases.

- *Task oriented* - conversational agents are expected to accomplish well defined tasks with quantifiable goals like movie or restaurant reservation, meeting scheduling, booking cabs, etc.
- *Open-Ended* - conversational agents capable of conversing with humans on popular topics, such as entertainment, fashion, politics, sports, technology, etc.

We introduce the nature of each dialog task along with commonly used approaches to solving each task in this section.

2.1.1. Task Oriented Dialog Systems

Traditionally, for goal oriented systems, the dialog task is framed as a sequential slot filling process [186, 171] where the dialog system interacts with users to extract the data needed to fill the slots and the task is completed when all the slots are filled. Typical domains of this task are telephone shopping, online banking, online trading, etc [97, 146, 7]. The user has his/her purpose and knows almost all information in order to achieve the task. As an example, for a restaurant reservation system, such slots can be the location, price range or type of cuisine served at the restaurant.

A typical task-oriented dialog system architecture as shown in Figure 2.1 contains four major components - a natural language interpreter/understanding (NLU) unit, a state tracker, a dialog manager, and a Natural Language Generation (NLG) unit. Spoken dialog systems additionally contain an Automatic speech recognition (ASR) module.

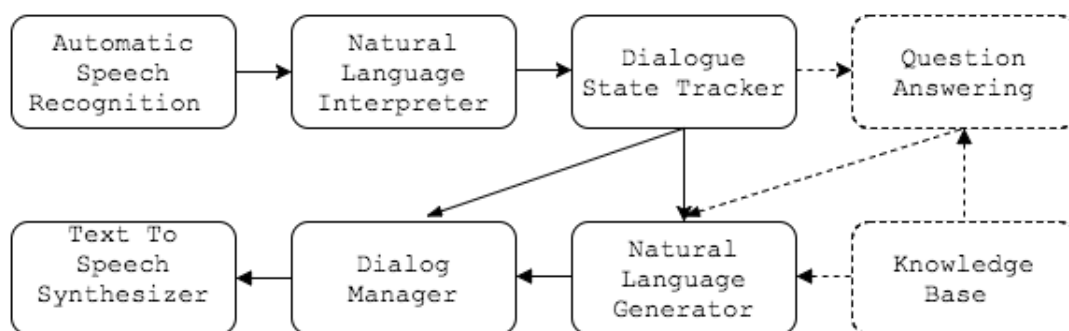


Fig. 2.1

The NLU unit. : parses the user utterance for extracting features like user intent, entities, etc., which are vital for dialog state tracking.

The state/user intent tracker. : computes a probability distribution over the dialog states, taking into account previous user utterances and states [179]. For instance, consider a restaurant reservation system where the dialog states in this case might be slot values like *time*, *city* and *number of people*. Each slot might be filled with one of the many valid discrete values. The task of the dialog state tracker is to output a distribution over all possible slot-value pairs.

The Dialog manager. : collects scores from ASR module, if available and gathers the dialog state features computed from the other submodules (practical implementations typically incorporate helper modules like Question-Answering (QA) modules, Memory/Knowledge base (KB) modules, etc.) in the system and chooses an action with the intention of increasing user satisfaction, both in the short term as well as in the long term. Actions here usually refer to *dialog acts* pertaining to the conversational domain (e.g., movie booking). Dialog acts can be used to suggest, inform, request certain information, etc. and they are unique to each dialog domain. Therefore, the set of dialog acts in the hotel domain may be different from the train domain although there could be some overlap among domains. For example, in the MultiWOZ [20], the hotel and train domain share the following dialog acts - *offer booking*, *inform booked*, *decline booking*. Apart from dialog acts, the NLG response selection problem can also be formulated as a planning problem and solved using reinforcement learning (RL) to optimize a dialog policy through interaction with users [77, 181, 61, 186].

The NLG unit. : converts the selected dialog manager actions to natural language responses. In case of a spoken system, the generated responses are passed to the Text To Speech module (TTS). There are, indeed, no standard rules as to what responsibilities should be assigned to each module. In recent years there has been a trend towards developing fully data driven systems by unifying these modules using a deep neural network that maps the user input to the agent output directly [175, 20, 150].

Although there are recent end-to-end trainable models using RNNs for slot filling [102], it is inherently hard to scale to new domains, i.e, it is will be a tedious task to manually list down

all the slots that users might refer to in a conversation. Nevertheless, sequence to sequence based models have inspired recent task oriented system designs [128, 176] where the models are end-to-end trainable but still modularly connected. [19], taking inspiration from the latest developments in memory networks [177] for question answering, propose a completely end-to-end trainable model from scratch, without any prior slot structure assumption. While it is easier to evaluate task oriented systems, it is difficult to gather large data-sets to train models as they are usually too domain specific. So, this makes it especially harder to train deep learning based architectures as they are known to be data hungry.

Evaluation. : Task oriented systems are usually evaluated on two types of metrics - task completion metrics like accuracy, precision, recall, F1 etc., and generated response quality metrics like F1 and BLEU scores. Although there are several recent research efforts [63, 88], human evaluation is still among best possible ways to judge the quality of the generated responses.

2.1.2. Open-Ended Dialog Systems

Open-Ended dialog systems belong to a class of unstructured dialogs where the agents are expected to converse with humans on pretty much any popular topic, such as entertainment, fashion, politics, sports, technology, etc. Open-Ended dialog systems are evaluated *only* on user satisfaction scores. Unlike task oriented systems, there are no task completion metrics like accuracy, precision, etc. Open-Ended conversations are more complex to model as they inherently exhibit high entropy i.e., there are numerous plausible responses or directions in which discussions can progress at each time step. Evaluation criteria for testing such systems are hard for the same reason, and we usually resort to human evaluation (mechanical turkers), like Amazon Mechanical Turk (AMT) [35], to rate the responses of Open-Ended dialog systems.

Early works in non-goal oriented systems like the popular ELIZA program [174] involved template (regular expression) matching and just simple text parsing rules. It soon gained popularity mostly by persistently rephrasing statements or asking questions. ELIZA's reputation led to similar works like [32], where the authors used simple text parsing rules, similar to ELIZA, to mimic behaviors of paranoid patients. However, such systems were merely viewed as an amusement and not commercialized for any real world applications.

Later, works like [55] employed data driven methods to statistically model dialog as a stochastic sequence of tokens using Markov chains. During inference, their model would first generate topic or intent-based tokens, which was then fed to another Markov chain to fill in the words surrounding the topic tokens.

Lack of low latency, reliable evaluation metrics, lack of large scale datasets and other related issues contributed to the slow pace of the Open-Ended dialog research until recently when promising works like [150, 141, 168, 83, 80] took inspiration from the success of deep neural network based sequence to sequence modeling for Neural Machine Translation (NMT) [29] and proposed end-to-end training of dialog systems by posing the response generation as a machine translation problem. We explain the end-to-end neural approaches to dialog modeling further in Chapter 4.

Chapter 3

Basic Neural Network Architectures

In this chapter we are going to introduce some important building bricks of modern neural architectures specifically in the context of natural language processing. These components forms the basis of the models we are going to introduce in the following articles.

3.1. Feed Forward Networks

The architecture of a shallow feed-forward neural network is typically defined as follows:

$$\begin{aligned} h_t &= f(Wx_t + b) \\ \hat{y}_t &= \text{softmax}(Vh_t + c) \end{aligned} \tag{3.1.1}$$

where f is some non-linearity function like *sigmoid* or *tanh*, and h_t is the hidden state of the network. Softmax gives a valid probability distribution over possible values y_t can take. Figure 3.1 depicts a deep fully-connected feed forward network.

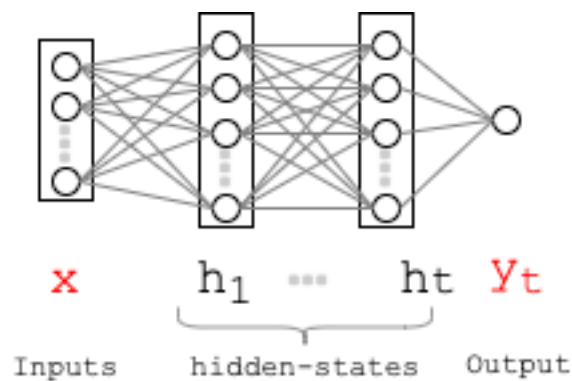


Fig. 3.1. Deep fully-connected feed forward network

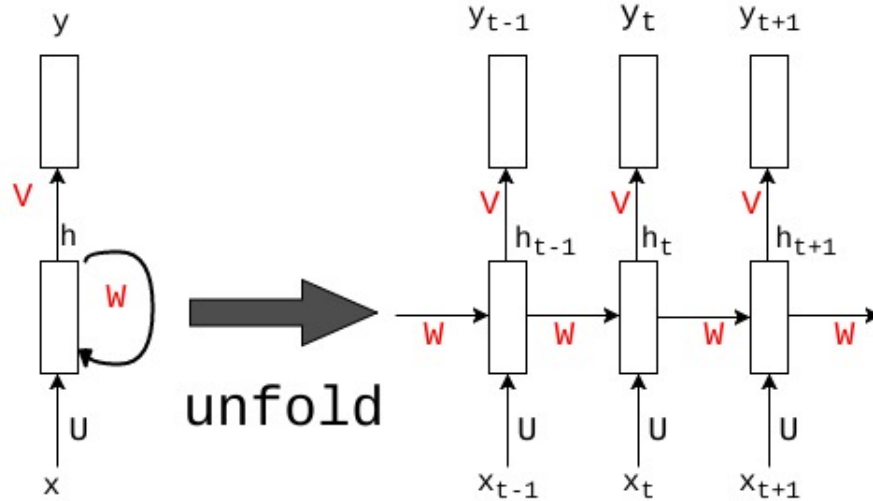


Fig. 3.2. Recurrent Neural Network (RNN)

3.2. Recurrent Neural Architectures

Recurrent neural networks(RNNs) [130] are a family of neural networks for processing sequential data, Figure 3.2. The typical architecture of a vanilla RNN is defined as follows:

$$h_t = f(Wx_t + Uh_{t-1} + b) \quad (3.2.1)$$

$$\hat{y}_t = \text{softmax}(Vh_t + c) \quad (3.2.2)$$

where the matrix U corresponds to hidden-to-hidden connections. RNNs can be used as a probabilistic model [103] over a sequence of tokens, as in eq 3.3.1. The hidden state acts as a sufficient statistic, which summarizes the past sequence and parametrizes the output distribution of the model: $P_\theta(w_t|w_{1:t-1}) = P_\theta(w_t|h_{t-1})$.

3.2.1. Long-Short Term Memory units

Vanilla RNNs suffer from exploding and vanishing gradients [14] for tasks involving longer sequences like language modeling. RNNs become very ineffective when the gap between the relevant information and the point where it is needed become very large. That is due to the fact that the information is passed at each step and the longer the chain is, the more probable the information is lost along the chain. While techniques like gradient clipping [14] can limit exploding gradients, vanishing gradients are harder to prevent and so limit the network's ability to learn long term dependencies [52]. A key development in addressing the

vanishing gradient issue in RNNs was the introduction of gated memory in the Long Short Term Memory (LSTM) network [53]. LSTM can be viewed as an extension to vanilla RNN by adding a series of gates and an internal cell state. The update equations at time step t for the LSTM are as follows

$$\begin{aligned}
i_t &= \sigma(U_i x_t + W_i s_{t-1} + b_i) \\
f_t &= \sigma(U_f x_t + W_f s_{t-1} + b_f) \\
o_t &= \sigma(U_o x_t + W_o s_{t-1} + b_o) \\
g_t &= \tanh(U_g x_t + W_g s_{t-1} + b_g) \\
c_t &= c_{t-1} \odot f + g \odot i \\
h_t &= \tanh(c_t) \odot o_t
\end{aligned} \tag{3.2.3}$$

Here \odot stands for element-wise multiplication, and $\sigma(\cdot)$ is the sigmoid activation function. W and b are the corresponding weights and biases. The i_t , f_t , o_t are the input gate, forget gate, and output gates, respectively. c_t is the internal cell state, and h_t is the output hidden state.

The three gates control the flow of information in the network. All of i_t , f_t , o_t are outputs of sigmoid functions, thus their values are in between 0 and 1. Input gate (i_t), forget gate (f_t), and output gate (o_t) control the inputs to the cell state (g), the cell state inherited from the last time step (c_{t-1}), and the cell state at the current time step (c_t), respectively. These gating mechanisms are explicitly designed to enable the LSTM to retain contents in the cell states for longer time steps.

3.2.2. Gated Recurrent Unit

The Gated Recurrent Unit (GRU) [29, 31] is a simplified version of the LSTM (with fewer gates) which works equally well. It can be viewed as a simplified version of LSTM without the cell state and output gate.

At time step t , GRU first computes two sets of gates, i.e., the update gate z_t and the reset gate r_t :

$$\begin{aligned}
z_t &= \sigma(W_{xz} x_t + W_{hz} h_{t-1} + b_z) \\
r_t &= \sigma(W_{xr} x_t + W_{hr} h_{t-1} + b_r)
\end{aligned} \tag{3.2.4}$$

where x_t is the input at time step t , W_* and b_* are the corresponding weights and biases. h_{t-1} is its hidden state at $t - 1$, which we will elaborate later. Then a candidate activation \tilde{h}_t is computed with the reset gate r_t and hidden state h_{t-1} :

$$\tilde{h}_t = \tanh(Wx_t + U(r_t \odot h_{t-1})) \quad (3.2.5)$$

The reset gate, r_t is used to control the amount of past information to erase, just like the forget gate in the LSTM. The final hidden state of GRU is then specified by

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (3.2.6)$$

where the hidden state h_t is a linear combination of previous activation h_{t-1} and the candidate activation \tilde{h}_t , where the interpolation is determined by the update gate z_t . The update gate helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future.

Note that there is another variant for the Equation 3.2.5, which was proposed in [31], where the order of multiplication between U and r_t is slightly different [28]:

$$\tilde{h}_t = \tanh(Wx_t + r_t \odot (Uh_{t-1})) \quad (3.2.7)$$

The GRU effectively removes one fourth of the parameters in the LSTM, thus allows more hidden states with a same budget of model size. This advantage allows the GRU to be able to yield better results on tasks with large datasets such a machine translation and dialog modeling. Although gates introduced in LSTMs and GRUs help model the long-term dependencies better, it often gets tricky to train the gates when they are saturated especially when they are expected to be saturated by design [23]. Also, it gets hard to parallelize the model for processing sentences due to their sequential computations requirements. Now, Transformers [166] - a self-attention-based non-recursive architecture, has become widely popular and has achieved state-of-the-art performance on several NLP tasks [36, 91]. It allows for parallelizable implementations as the self-attention-based architecture is non-recursive by design. We introduce the Transformer model in Section 3.3.4.

3.3. Deep Learning For NLP

3.3.1. Language modeling

We use language models to compute the probability distribution over the sequence of words w_1, \dots, w_T . One way to factorize the probability $P(w_1, \dots, w_T)$, is to represent it as the product of conditional probabilities of each word given all the previous ones, as in eq 3.3.1.

$$P(w_{1:T}) = \prod_{t=1}^T P(w_t | w_{1:t-1}) \quad (3.3.1)$$

where $w_{1:t-1}$ is the sequence w_1, \dots, w_{t-1} and w_1^0 is defined as null. *N-gram* models adopt a count-based approach towards statistical language modeling. *N-gram* models compute probability of a sequence of words by maintaining a look up table of conditional probabilities for each word given the previous $(N - 1)$ words. As for neural network based approaches, we can either use feed-forward neural networks or Recurrent Neural Networks (RNNs) to parameterize the probability of the next token given the previous tokens, for each time step, $1 \leq t \leq T$. We train such models with the standard maximum likelihood objective, by decreasing the cross-entropy loss [45]. Once trained, we can use this model for conditional language generation by sampling a word given the sequence of previously sampled words. [12] use feed-forward neural networks for language modeling, where they approximate the next word prediction (in eq 3.3.1) by conditioning with the previous N words instead of the entire history : $P(w_t | w_{1:t-1}) \approx P_\theta(w_t | w_{t-1-N:t-1})$, where θ refers to the parameters of the feed forward network. With the assumption that the outputs lie within in closed set V (vocabulary of the tokens), RNNs can also be considered as a simple generative model of discrete sequences. For probabilistic language models word perplexity is a well-established performance metric [13, 103]. Word perplexity of a language model over a sequence $W = w_1, \dots, w_t$ is defined as

$$PP(W) = 2^{-\frac{1}{N} \log_2 P(w_1, \dots, w_{t-1})} \quad (3.3.2)$$

3.3.2. Sequence to Sequence Models

We can extend the idea of predicting the next token given the previous tokens to predicting the output sequence given the input sequence directly. For instance, we can frame the problem of machine translation (translating text from one language to another) as a sequence

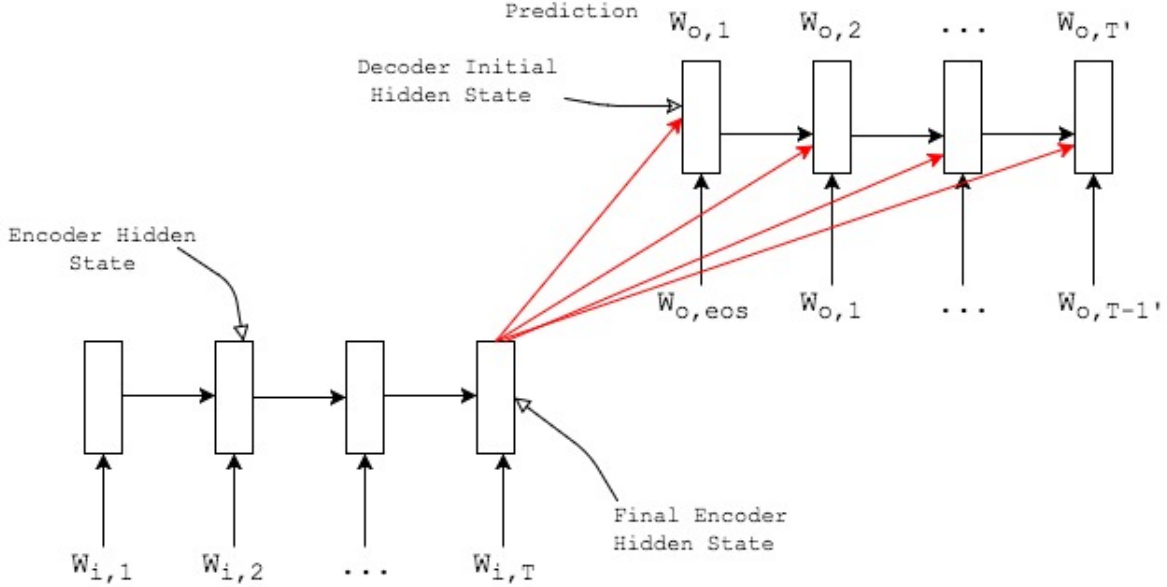


Fig. 3.3. Sequence to Sequence Encoder-Decoder Architecture

to sequence problem where the input sequence is a sentence belonging to the source language sentence and output sequence is the target language sentence. Sequence to Sequence models (Figure 3.3) mainly involve a Recurrent Neural Network (RNNs) based encoder to summarize the input sequence $w_{1:N}^i$ and a RNN based decoder which predicts the output sequence $w_{1:t-1}^o$ token by token conditioned on the encoder hidden state.

$$P_{\theta}(w_t^o | w_{1:t-1}^o, w_{1:N}^i) \propto \text{softmax}(W_t^o \odot f(h_{\text{decoder}_{t-1}}, h_{\text{encoder}_N})) \quad (3.3.3)$$

where W_t^o is the output token embedding of w_t^o , $h_{\text{decoder}_{t-1}}$, h_{encoder_N} are the hidden states of the decoder and encoder respectively. f is a non-linear transformation - usually a Multi-Layer Perceptron (MLP) network.

3.3.3. Attention Mechanism

In the Sequence to Sequence model described above, the basic premise of the Encoder RNN is to parse every item in an input series, one after the other, and keep updating it's hidden state vector every step of the way as shown in Figure 3.3. This hidden vector at the end of every step is understood to represent the context of all prior inputs. In other words, the last hidden state represents the context of the entire sequence and is fed to the decoder at every time step to generate the output sequence. *Attention* mechanism introduced in

[8] enables the decoder to attend to all the tokens in the input sequence directly instead of just conditioning on the last hidden state context vector. The attention mechanism can be described by the following equations

$$\begin{aligned}
 c_i &= \sum_{j=1}^M \alpha_{ij} h_j \\
 \alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{k=1}^M \exp(e_{ik})} \\
 e_{ij} &= f(s_{i-1}, h_j)
 \end{aligned}
 \tag{3.3.4}$$

where M is the input sequence length, c_i is the *attention* based context vector which is fed to decoder instead of h_M , f is a type of non-linear transformation (eg. a feedforward neural network) termed as the *alignment model*. The *attention* context vector, c_i is a linear combination of the input hidden states from all the time steps. As seen from the Equation 3.3.4, there is a shorter and direct path for the backpropagation training signals to propagate from the decoder equally to all the input time steps. α_{ij} are called as the *attention weights* corresponding to the attention placed by the decoder at the i^{th} time step over the j^{th} input token.

There have been different variants of attention mechanisms since they were first proposed, among which self-attention becomes the most influential variant, which was proposed under the name of self-attention [86]. Later on the Transformer model [166] significantly developed the self-attention mechanism, which resulted in a series of the state-of-the-art performances. Transformers [166] based architectures like BERT [36], XL-net [185], GPT-2 [118], MT-DNN [89], RoBERTA [92] reached state-of-the-art performance on tasks like machine translation [6], language modeling [118], text classification benchmarks like GLUE [170].

3.3.4. Transformers

The Transformer still follows the encoder-decoder framework as depicted in Section 3.3.2. The difference lies in the form of the encoder and decoder, and the attention mechanism between them. Following the notations from [166], the encoder maps an input sequence of symbol representations (x_1, \dots, x_n) to a sequence of continuous representations $\mathbf{z} = (z_1, \dots, z_n)$. Given \mathbf{z} , the decoder then generates an output sequence (y_1, \dots, y_m) of symbols one element at a time. Transformers consists of multiple layers (typically 6) of

encoder and decoder layers stacked upon each other. All encoder and decoder layers have similar structure.

Each encoder consists of two layers: *self-attention* and a *feed forward layer*.

Self-attention layer. : The encoder’s inputs first flow through the self-attention layer. The self-attention operation can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. In the first layer, the input tokens are converted to query, keys and values by a simple linear transformation and packed into matrices, $Q, K, V \in \mathbb{R}^{N \cdot d}$, where N is the input sequence length and d is the embedding dimension. We compute the matrix of outputs as:

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \quad (3.3.5)$$

The scaling factor, d is to avoid the softmax from saturating due to high values resulting the dot products. Please note that the dimension of the self-attention output matrix, A is $\mathbb{R}^{N \cdot d}$ as well.

Feed forward layer. : The output of the self-attention layer is fed into a full connected layer with ReLU activation [109]. The output of this feed-forward layer is fed into the subsequent encoder layers after which they are fed into the decoder.

Position encodings. : Since the transformer doesn’t have a recurrent mechanism, the token embeddings are added with *positional encodings* to inform the network about the token’s position in the sequence. [166] use the following functions to create a constant positional encoding matrix with position-specific values.

$$\begin{aligned} PE_{pos,2i} &= \sin\left(\frac{pos}{10000^{2i/d}}\right) \\ PE_{pos,2i+1} &= \cos\left(\frac{pos}{10000^{2i/d}}\right) \end{aligned} \quad (3.3.6)$$

where pos is the position in the sequence and i is the dimension in the embedding vector, d is the embedding dimension. The positional encoding matrix is simply added to the token embedding matrix before feeding to the encoder layer. The authors chose a *sinusoid* function as it would allow the model to easily learn to attend by relative positions, since for any fixed offset k , PE_{pos+k} can be represented as a linear function of PE_{pos} . Such embeddings are

typically referred to as absolute position embeddings, to usually depend on the absolute location of the token within the sequence. This differs from how recurrent neural networks operate as they model position in relative terms through their recurrence over the positions in the input sequence. [143] introduce relative position embeddings to allow attention to be informed by how far two tokens are apart in a sequence. This involves learning a separate relative position embedding, $PE_i \in \mathbb{R}^d$ for $0 \leq i \leq N-1$. which is added to the self-attention weights.

Cross-attention. : Similar to the encoder layers, the partially decoded outputs are fed into the decoder layers. In the final decoder layer, the attention is performed between the encoder and decoder layers where the query, Q is from the final decoder layer and key K & value V are the outputs from the last encoder layer. The output of this attention is fed to softmax layer to decode the next token. It is important to note that the Transformer is still an auto-regressive model which generates sequences token by token.

Chapter 4

Neural Approaches for End-to-End Dialog modeling

4.1. Sequence to Sequence approaches

Following the success of neural machine translation systems [8], there were multiple works [150, 141, 142] with the sequence to sequence paradigm adapted to the generative dialog framework. Early work on probabilistic end-to-end generative dialog systems done by [127, 168, 83, 80] pose the utterance generation problem as a machine translation problem. All the utterances within a conversation concatenated back-to-back (with appropriate sequence delimiters) in order to form a flattened dialog history serves as the input sequence and next utterance serves as the output sequence.

For instance, [168] feed the flattened dialog history into a single layer LSTM based encoder and generate the next utterance via another single layer LSTM based decoder conditioned on the dialog history as in Figure 3.3. Let $U_{1:m}$ be the sequence of m utterances in a conversation where each utterance, $U_i = w_{i1}, w_{i2} \dots w_{in}$ in turn can be seen as a sequence of n tokens. Now, the input and outputs to the sequence to sequence model are of the form

$$\begin{aligned} \text{Inputs} &= U_1 \langle eou \rangle U_2 \langle eou \rangle \dots \langle eou \rangle U_{m-1} \\ \text{Outputs} &= U_m \end{aligned} \tag{4.1.1}$$

During training, the true output sequence is given to the model, so learning can be done by backpropagation [130]. The model is trained to maximize the cross entropy of the ground truth utterance given its dialog context. During inference, given that the true output sequence is not observed, we simply feed the predicted output token as input to predict the next output token. This approach is referred to as the "greedy inference" approach. A less greedy approach would be to use beam search [156], by feeding several probable candidates

at the previous step to the next step. In the next section, we introduce a encoder-decoder architecture [141] (used in Article 3) which takes into the view that a dialog is a sequence of utterances which, in turn, are sequences of tokens.

4.2. The Hierarchical Recurrent Encoder-Decoder

For modeling dialog systems, we can concatenate all the utterances of a conversation into a single sequence and train the LSTMs (or GRU) as a conditional language model as in [167]. However, we lose the utterance level structure present in the dialog data while flattening out the utterances. Also, this throttles the gradient flow for efficient training, as sequence lengths get unnecessarily longer. It would be difficult to model the dependencies of later utterances of a dialog on the initial utterances due to lack of alternate shorter paths for information flow across utterances. The Hierarchical Recurrent Encoder-Decoder (HRED) [141] tries to address both these issues by adding another utterance level RNN encoder over the token level encoder, thus reducing the number of computational steps between two utterances facilitating gradient flow. The model architecture presented in [150] is as shown in Figure 4.1. The model primarily consists of three modules - token level encoder, utterance level encoder and output token decoder.

The m^{th} tokenized utterance, $U_m = w_{m_1}, w_{m_2} \dots w_{m_n}$ is fed through the token level RNN to obtain a utterance representation as in equation 4.2.1.

$$h_{m_n} = GRU_{enc}(h_{m_{n-1}}, w_{m_n}), n = 1, 2 \dots N_m \quad (4.2.1)$$

where GRU_{enc} is the token-level encoder GRU function, h_{m_n} is GRU's hidden state representation and $h_{m_0} = 0$, the null vector. So, the token encoder should give us a sequence of fixed size distributed order sensitive representations, $h_1, h_2 \dots h_M$ where M is the number of utterances in the conversation.

Now, we pass the sequence of utterance representations $h_1, h_2 \dots h_M$ through another GRU to compute a representation of the M utterances so far as in equation 4.2.2.

$$s_i = GRU_{uttr}(s_{i-1}, h_i), i = 1, 2 \dots M \quad (4.2.2)$$

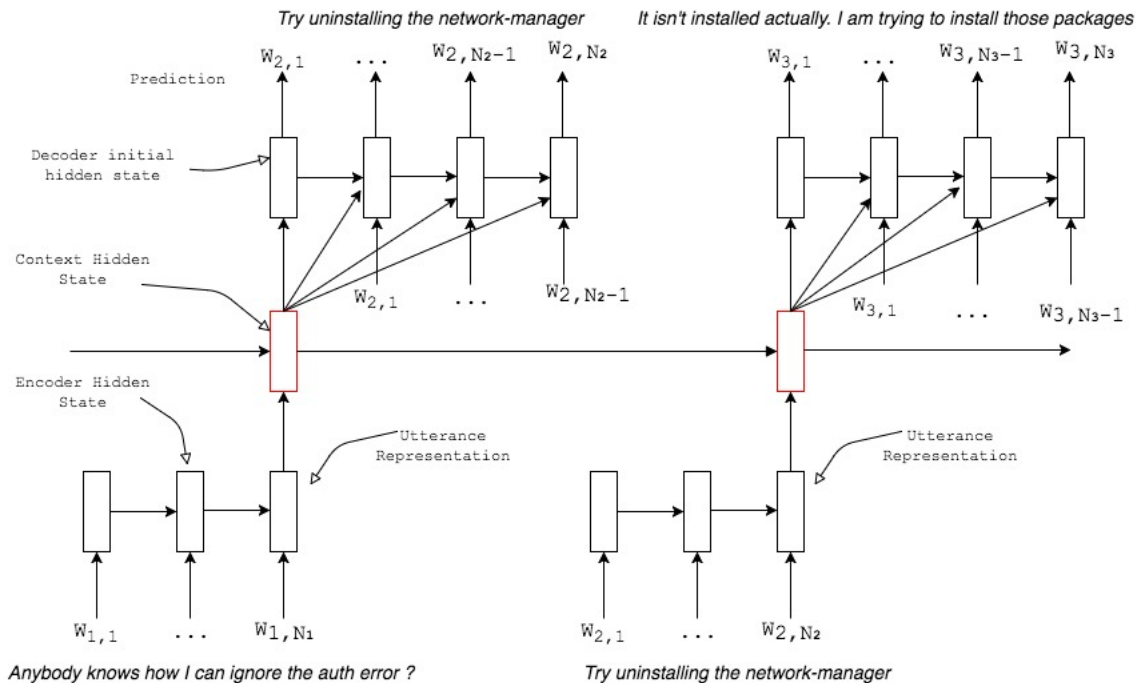


Fig. 4.1. The computational graph of the HRED architecture [141] for a sample dialog composed of three turns. The token encoder provides a representation for each utterance and the context encoder provides the condensed representation for the previous utterances. The decoder predicts tokens conditioned additionally on this context embedding.

where GRU_{uttr} is the utterance-level encoder GRU function, s_i is the GRU's hidden state representation and $s_0 = 0$, the null vector. The recurrent hidden state s_i summarizes the utterances that have been processed up to position i in addition to preserving the order information of both utterances and tokens spoken so far.

The RNN output token decoder is responsible for the prediction of the next utterance given the previous utterances. The probability of the next utterance prediction is given by

$$P(U_m/U_{1:m-1}) = \prod_{n=1}^{N_m} P(w_n|w_{1:n-1}, U_{1:m-1}) \quad (4.2.3)$$

where U_m is the m_{th} utterance, $U_{1:m-1}$ are the past utterances. The contribution of the previous utterances to the recurrent equation of the output token decoder is as in equation 4.2.4

$$h_{dec_{m,n}} = GRU_{dec}(h_{dec_{m,n-1}}, w_{m,n}, s_m), n = 1, \dots, N_m \quad (4.2.4)$$

where h_{dec_n} is the $n^{th}GRU_{dec}$'s recurrent hidden state and s_n is the recurrent hidden state of the utterance encoder from 4.2.2. It is a common practice to either set $h_{dec_{m,0}}$ to the null vector or initialize it s_m .

The model is trained by maximizing the log-likelihood of each conversation as in 4.2.5

$$LL = \sum_{m=1}^M \log P(U_m | U_{1:m-1}) = \sum_{m=1}^M \sum_{n=1}^{N_m} \log P(w_{m,n} | w_{m,1:n-1}, U_{1:m-1}) \quad (4.2.5)$$

While the adaptation of sequence to sequence based hierarchical generative model is a good first direction, such dialog systems have difficulties with training as there is a wide range of plausible responses, unlike the translation setting. Dialog datasets, especially open domain corpus like the Twitter dataset [127], are jammed with short generic responses like "yes", "thank you", "got it.thanks!" etc. Also, there are contrasting responses to the same types of utterances across different conversations.

During training, when the decoder sees the same output utterance for different utterance encoder hidden states, it manages to ignore the encoder hidden state over time regarding it as just noise and instead concentrates on being a better language model. Thus, the decoder doesn't generalize to new encoder states during inference and ends up predicting one of the many generic responses it has seen during training. Also, we cannot afford to remove the abundant generic responses from the dataset, like some sampling-based solutions, for class imbalance problems in classification.

Several approaches have been proposed in the literature to address the generic response generation issue. [80] propose to modify the loss function to increase the diversity in the generated responses. Multi-resolution RNN [138] addresses the above issue by additionally conditioning with entity information in the previous utterances. Alternatively, [148] uses

external knowledge from a retrieval model to condition the response generation. Latent variable models inspired by Conditional Variational Autoencoders (CVAEs) [147] are explored in [144, 189]. In Article 3, we propose to address this problem by jointly modeling the utterances with the dialog attributes of each utterance. Dialog attributes of an utterance refer to discrete features or aspects associated with an utterance like dialog-acts, sentiment, emotion, speaker identity, speaker personality, etc.

Chapter 5

Background : Efficient Locality Sensitive Hashing (LSH)-based Text Representation

This chapter covers the basics of Locality Sensitive Hashing (LSH) [24] based projection operations for efficient text representations discussed in Article 4.

5.1. Introduction

In the last decade, research in deep neural networks has lead to tremendous advances and state-of-the-art performance on wide range of Natural Language Processing (NLP) applications [8, 139]. The availability of high performance computing has enabled research in deep learning to focus largely on the development of deeper and more complex network architectures for improved accuracy. However, the increased complexity of the deep neural networks has become one of the biggest obstacles to deploy deep neural networks on-device such as mobile phones, smart watches and Internet of Things (IoT) devices [56]. On-device deployment of neural network models specifically require a tiny memory footprint and low inference latency.

Recently, [123, 124, 72, 62], introduced a class of on-device deep learning models learned via embedding-free lightweight binary LSH-based projections learned on-the-fly. The projection approach surmounts the need to store any embedding matrices, since the projections are dynamically computed. The computation of the representation is linear in the number of inputs in the sentence surmounting the need to maintain and lookup global vocabulary. Additionally, this further enables user *privacy* by performing inference directly on device without sending user data (e.g., personal information) to the server.

They propose a modified LSH-based projection operation, \mathbb{P} that dynamically generates a fixed binary projection representation $\mathbb{P}(x) \in [0, 1]^K$ for any text, x by extracting morphological input features like char (or token) n-gram & skip-gram features, parts of speech tags, etc. The first step of the projection operation involves a string-based hashing method to extract a fixed vector,

$$\vec{x} \in \mathbb{R}^d$$

from the morphological input features. Similar to the LSH via random projection methods, the projection vectors, P_k in the projection matrix P (contains K randomly generated vectors $\in \mathbb{R}^d$) transform the input \vec{x}_i to a binary hash representation denoted by $\mathbb{P}_k(\vec{x}_i) \in [0, 1]$ as in

$$\mathbb{P}_k(\vec{x}_i) := \text{sgn}[\langle \vec{x}_i, P_k \rangle]$$

where $\langle \cdot, \cdot \rangle$ denotes inner product and sgn denotes the sign of the inner product. This results in a K -bit vector representation $\mathbb{P}(x) \in [0, 1]^K$ one bit corresponding to each projection row.

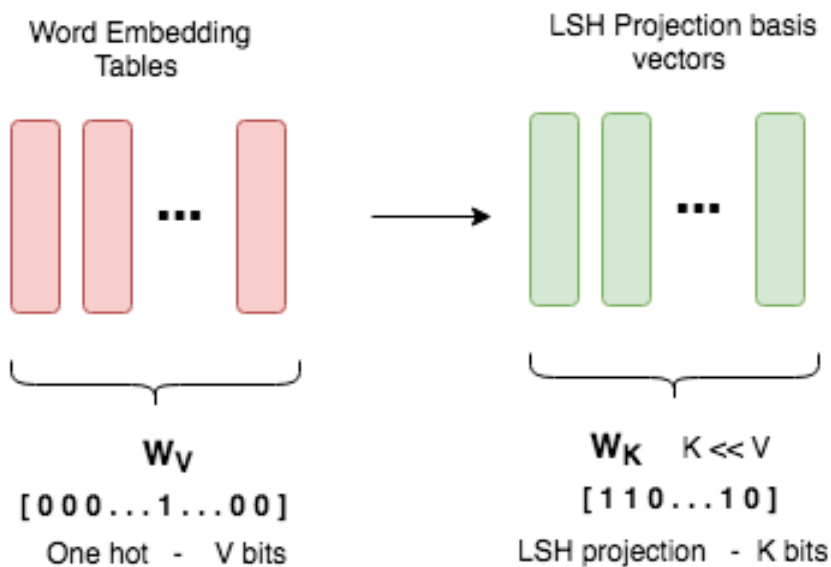


Fig. 5.1. Memory for V look-up vectors for each token vs storing $K(\ll V)$ vectors and linearly combining them for token representation. We consider $K = 1120$ following [124] in this paper.

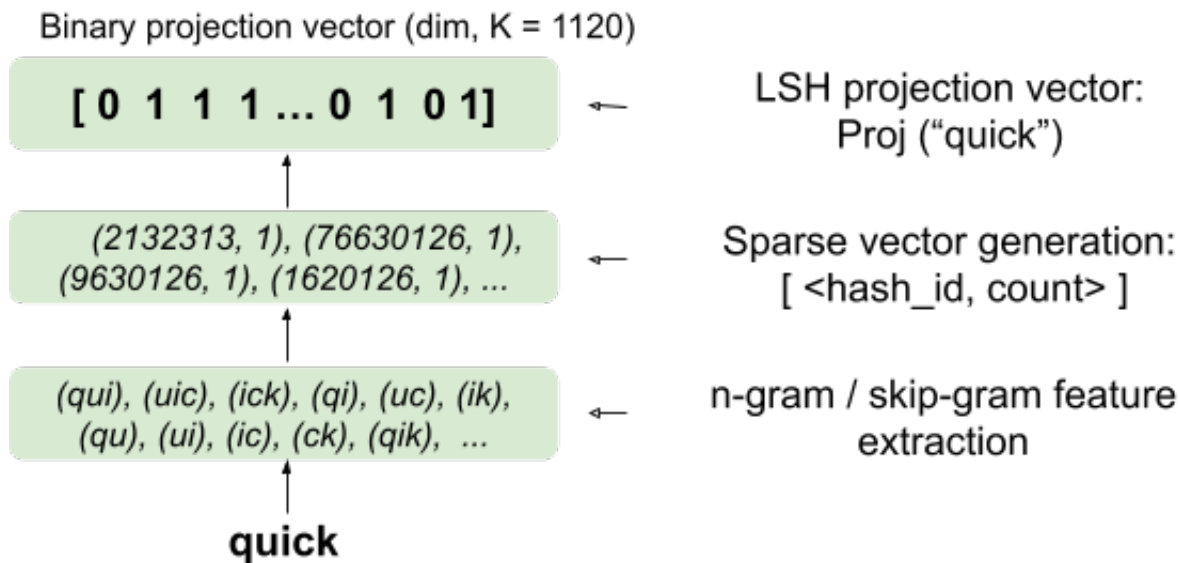


Fig. 5.2. Binary Locality-Sensitive Hashing (LSH) projection representation for text.

5.2. Motivation

The dependency on vocabulary size V , is one of the primary reasons for the huge memory footprint of embedding matrices. It is common to represent a token x by one-hot representation $\mathbb{Y}(x) \in [0, 1]^V$ and a distributed representation of the token is obtained by multiplying the one-hot representation with the embedding matrix, $W_V \in \mathbb{R}^{d \times V}$ as in

$$U_V(x) = W_V * \mathbb{Y}(x) \in \mathbb{R}^d$$

where $*$ refers to matrix-vector product.

One way to remove the dependency on the vocabulary size is to learn a smaller matrix $W_K \in \mathbb{R}^{d \times K}$ ($K \ll V$), as shown in Figure 5.1. For instance, 300-dimensional Glove embeddings, W_V [113] with 400k vocabulary size occupies > 1 GB while the W_K occupies only ≈ 1.2 MB for $K = 1000$ yielding a $1000\times$ reduction in size. Instead of learning a unique vector for each token in the vocabulary, we can think of the columns of this W_K matrix as a set of basis vectors and each token can be represented as a linear combination of basis vectors in W_K . We select the basis vectors from W_K for each token with a fixed K -bit binary vector instead of a V -bit one-hot vector.

The LSH *Projection function*, \mathbb{P} (Figure 5.2)[122] used in [124, 72, 62] does exactly this as it dynamically generates a fixed binary projection representation $\mathbb{P}(x) \in [0, 1]^K$ for any token x by extracting morphological input features like char (or token) n-gram & skip-gram

features, parts of speech tags etc. from x and a modified Locality-Sensitive Hashing (LSH) based transformation, \mathbb{L} as in

$$x \xrightarrow{\mathbb{F}} [f_1, \dots, f_n] \xrightarrow{\mathbb{L}} \mathbb{P}(x) \in [0, 1]^K$$

where \mathbb{F} extracts n-grams (or skip-grams), $[f_1, \dots, f_n]$ from the input text. Here, $[f_1, \dots, f_n]$ could refer to either character level or token level n-grams(or skip-grams) features. Given the LSH projection representation $\mathbb{P}(x)$, the distributed representation of the token x is represented as in

$$U_W(x) = W_K * \mathbb{P}(x) \in \mathbb{R}^d$$

It is worth noting that the projection operation \mathbb{P} can also be used to map an entire sentence directly to the $[0, 1]^K$ space.

Chapter 6

Prologue to First Article

6.1. Article Details

Do Neural Dialog Systems Use the Conversation History Effectively? An Empirical Study. Chinnadhurai Sankar, Sandeep Subramanian, Christopher Pal, Sarath Chandar, Yoshua Bengio, *57th Annual Meeting of the Association for Computational Linguistics, 2019*

Personal Contribution. There has been a surge of end-to-end neural dialog approaches [141, 138, 165] advocating different kinds of architectures, loss functions and training procedures addressing the utterance repetition problem. However, there is very little understanding of how or if these dialog models are leveraging dialog history effectively. With this motivation, I proposed the idea to analyze existing dialog models over multiple goal-oriented and chit-chat datasets within the ParlAI framework [106]. I set up the initial coding framework on top of ParlAI code-base. Sandeep Subramanian helped with extending the experimental coverage to other models and datasets. Additionally, he contributed to the initial draft of the paper while others helped improve the presentation.

6.2. Context

A common criticism of recent neural dialog systems is that they seldom understand or use the available dialog history effectively due to which they tend to generate repetitive utterances or meander away from the context of the conversation [79, 136]. [66] proposed a perturbation-based study on how language models attend to words near and far away in

the context. In this paper, we proposed to understand how neural dialog models use context via perturbation-based experiments along the lines of previous research efforts [66, 11].

6.3. Contributions

We separately train various types of neural dialog models on multi-turn datasets and study how the learned probability distribution over generated utterances behaves as we artificially perturb the conversation history. We measure behavior by looking at how much the per-token perplexity of the generated utterance increases under these perturbations. We experiment with 10 different types of perturbations on 4 multi-turn dialog datasets and find that commonly used neural dialog architectures like recurrent and transformer-based seq2seq models are rarely sensitive to most perturbations such as missing or reordering utterances, shuffling words, etc.

6.4. Recent Developments

This article was nominated for the best paper award at ACL 2019, Florence. There are multiple research efforts extending our perturbation-based experimental setting to other dialog domains such as Conversational Question Answering [26], Conversational Machine Comprehension [115], etc. [145] propose a similar experimental setting to study the robustness of representations learnt by BERT [36] for multiple-choice reading comprehension tasks. Research efforts like [132] propose reinforcement learning a method to effectively attend to longer dialog context.

Chapter 7

Do Neural Dialog Systems Use the Conversation History Effectively? An Empirical Study

7.1. Abstract

Neural generative models have become increasingly popular when building conversational agents. They offer flexibility, can be easily adapted to new domains, and require minimal domain engineering. A common criticism of these systems is that they seldom understand or use the available dialog history effectively. In this paper, we take an empirical approach to understanding how these models use the available dialog history by studying the sensitivity of the models to artificially introduced *unnatural* changes or perturbations to their context at test time. We experiment with 10 different types of perturbations on 4 multi-turn dialog datasets and find that commonly used neural dialog architectures like recurrent and transformer-based seq2seq models are rarely sensitive to most perturbations such as missing or reordering utterances, shuffling words, etc. Also, by open-sourcing our code, we believe that it will serve as a useful diagnostic tool for evaluating dialog systems in the future.

7.2. Introduction

With recent advancements in generative models of text [184, 166, 116], neural approaches to building chit-chat and goal-oriented conversational agents [150, 167, 141, 19, 137] have gained popularity with the hope that advancements in tasks like machine translation [8], abstractive summarization [134] should translate to dialog systems as well. While these models have demonstrated the ability to generate fluent responses, they still lack the

	No Perturbations	Token shuffling
1	Good afternoon ! Can I help you ?	I afternoon help you Good ? ! Can
2	Could you show me where the Chinesc-style clothing is located ? I want to buy a silk coat	the located Chinesc-style where is show a . buy you ? I clothing want coat silk me Could to
3	This way , please . Here they are . They're all handmade .	are handmade . way please This all Here they . , They're .
4	Model Response: How much is it ?	Model Response: How much is it ?

Tab. 7.1. An example of an LSTM seq2seq model with attention’s insensitivity to shuffling of words in the dialog history on the DailyDialog dataset.

ability to “understand” and process the dialog history to produce coherent and interesting responses. They often produce boring and repetitive responses like “Thank you.” [79, 136] or meander away from the topic of conversation. This has been often attributed to the manner and extent to which these models use the dialog history when generating responses. However, there has been little empirical investigation to validate these speculations.

In this work, we take a step in that direction and confirm some of these speculations, showing that models do not make use of a lot of the information available to it, by subjecting the dialog history to a variety of synthetic perturbations. We then empirically observe how recurrent [157] and transformer-based [166] sequence-to-sequence (seq2seq) models respond to these changes. The central premise of this work is that *models make minimal use of certain types of information if they are insensitive to perturbations that destroy them*. Worryingly, we find that 1) both recurrent and transformer-based seq2seq models are insensitive to most kinds of perturbations considered in this work 2) both are particularly insensitive even to extreme perturbations such as randomly shuffling or reversing words within every utterance in the conversation history (see Table 7.1) and 3) recurrent models are more sensitive to the ordering of utterances within the dialog history, suggesting that they could be modeling conversation dynamics better than transformers.

7.3. Related Work

Since this work aims at investigating and gaining an understanding of the kinds of information a generative neural response model learns to use, the most relevant pieces of work are where similar analyses have been carried out to understand the behavior of neural models in other settings. An investigation into how LSTM based *unconditional* language models use available context was carried out by [66]. They empirically demonstrate that models are sensitive to perturbations only in the nearby context and typically use only about 150 words of context. On the other hand, in conditional language modeling tasks like machine translation, models are adversely affected by both synthetic and natural noise introduced anywhere in the input [11]. Understanding what information is learned or contained in the representations of neural networks has also been studied by “probing” them with linear or deep models [2, 154, 33].

Several works have recently pointed out the presence of annotation artifacts in common text and multi-modal benchmarks. For example, [48] demonstrate that hypothesis-only baselines for natural language inference obtain results *significantly* better than random guessing. [64] report that reading comprehension systems can often ignore the entire question or use only the last sentence of a document to answer questions. [4] show that an agent that does not navigate or even see the world around it can answer questions about it as well as one that does. These pieces of work suggest that while neural methods have the potential to learn the task specified, its design could lead them to do so in a manner that doesn’t use all of the available information within the task.

Recent work has also investigated the inductive biases that different sequence models learn. For example, [161] find that recurrent models are better at modeling hierarchical structure while [158] find that feedforward architectures like the transformer and convolutional models are not better than RNNs at modeling long-distance agreement. Transformers however excel at word-sense disambiguation. We analyze whether the choice of architecture and the use of an attention mechanism affect the way in which dialog systems use information available to them.

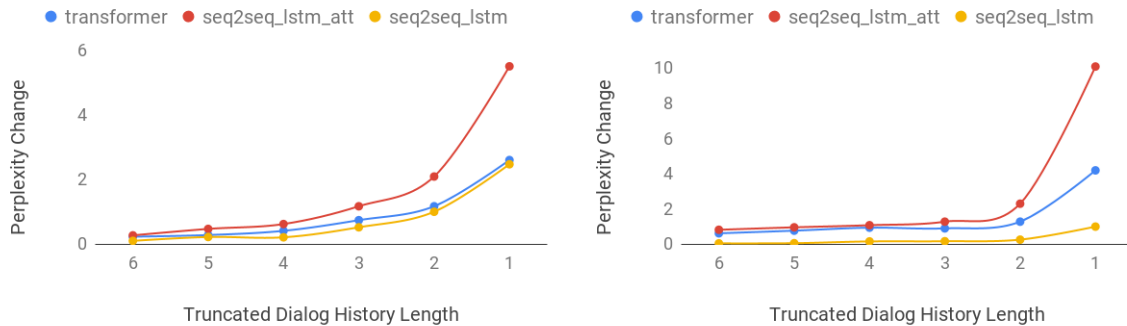


Fig. 7.1. The increase in perplexity for different models when only presented with the k most recent utterances from the dialog history for Dailydialog (left) and bAbI dialog (right) datasets. Recurrent models with attention fare better than transformers, since they use more of the conversation history.

7.4. Experimental Setup

Following the recent line of work on generative dialog systems, we treat the problem of generating an appropriate response given a conversation history as a conditional language modeling problem. Specifically we want to learn a conditional probability distribution $P_{\theta}(y|x)$ where y is a reasonable response given the conversation history x . The conversation history is typically represented as a sequence of utterances $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$, where each utterance \mathbf{x}_i itself is comprised of a sequence of words $x_{i_1}, x_{i_2} \dots x_{i_k}$. The response y is a single utterance also comprised of a sequence of words $y_1, y_2 \dots y_m$. The overall conditional probability is factorized autoregressively as

$$P_{\theta}(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^n P_{\theta}(y_i|y_{<i}, \mathbf{x}_1 \dots \mathbf{x}_n)$$

P_{θ} , in this work, is parameterized by a recurrent or transformer-based seq2seq model. The *crux of this work is to study how the learned probability distribution behaves as we artificially perturb the conversation history* $\mathbf{x}_1, \dots, \mathbf{x}_n$. We measure behavior by looking at how much the per-token perplexity (defined in Section 3.3.1) increases under these changes. For example, one could think of shuffling the order in which $\mathbf{x}_1 \dots \mathbf{x}_n$ is presented to the model and observe how much the perplexity of \mathbf{y} under the model increases. If the increase is only minimal, we can conclude that the ordering of $\mathbf{x}_1 \dots \mathbf{x}_n$ isn't informative to the model. For a complete

list of perturbations considered in this work, please refer to Section 7.4.2. All models are trained without any perturbations and sensitivity is studied *only at test time*.

7.4.1. Datasets

We experiment with four multi-turn dialog datasets.

bAbI dialog. is a synthetic goal-oriented multi-turn dataset [19] consisting of 5 different tasks for restaurant booking with increasing levels of complexity. We consider Task 5 in our experiments since it is the hardest and is a union of all four tasks. It contains $1k$ dialogs with an average of 13 user utterances per dialog.

Persona Chat. is an open domain dataset [188] with multi-turn chit-chat conversations between turkers who are each assigned a “persona” at random. It comprises of $10.9k$ dialogs with an average of 14.8 turns per dialog.

Dailydialog. is an open domain dataset [84] which consists of dialogs that resemble day-to-day conversations across multiple topics. It comprises of $13k$ dialogs with an average of 7.9 turns per dialog.

MutualFriends. is a multi-turn goal-oriented dataset [49] where two agents must discover which friend of theirs is mutual based on the friends’ attributes. It contains $11k$ dialogs with an average of 11.41 utterances per dialog.

7.4.2. Types of Perturbations

We experimented with several types of perturbation operations at the utterance and word (token) levels. All perturbations are applied in isolation.

Utterance-level perturbations. We consider the following operations 1) *Shuf* that shuffles the sequence of utterances in the dialog history, 2) *Rev* that reverses the order of utterances in the history (but maintains word order within each utterance) 3) *Drop* that completely drops certain utterances and 4) *Truncate* that truncates the dialog history to contain only the k most recent utterances where $k \leq n$, where n is the length of dialog history.

Word-level perturbations. We consider similar operations but at the word level within **every** utterance 1) *word-shuffle* that randomly shuffles the words within an utterance 2) *reverse* that reverses the ordering of words, 3) *word-drop* that drops 30% of the words uniformly 4) *noun-drop* that drops *all* nouns, 5) *verb-drop* that drops *all* verbs.

Models	Test PPL	Word Drop	Verb Drop	Noun Drop	Word Shuf	Word Rev
		Word level perturbations ($\Delta PPL_{[\sigma]}$)				
DailyDialog						
seq2seq_lstm	32.90 _[1.40]	1.58 _[0.15]	0.87 _[0.08]	1.06 _[0.28]	3.37 _[0.33]	3.10 _[0.45]
seq2seq_lstm_att	29.65 _[1.10]	2.03 _[0.25]	1.37 _[0.29]	2.22 _[0.22]	2.82 _[0.31]	3.29 _[0.25]
transformer	28.73 _[1.30]	1.20 _[0.69]	0.63 _[0.17]	2.60 _[0.98]	0.15 _[0.08]	0.26 _[0.18]
Persona Chat						
seq2seq_lstm	43.24 _[0.99]	1.81 _[0.25]	0.68 _[0.19]	0.75 _[0.15]	1.29 _[0.17]	1.95 _[0.20]
seq2seq_lstm_att	42.90 _[1.76]	2.47 _[0.67]	1.11 _[0.27]	1.20 _[0.23]	2.03 _[0.46]	2.39 _[0.31]
transformer	40.78 _[0.31]	0.54 _[0.08]	0.40 _[0.00]	0.32 _[0.18]	0.01 _[0.01]	0.00 _[0.06]
MutualFriends						
seq2seq_lstm	14.17 _[0.29]	0.28 _[0.11]	0.00 _[0.03]	0.61 _[0.39]	0.31 _[0.25]	0.56 _[0.39]
seq2seq_lstm_att	10.60 _[0.21]	1.56 _[0.20]	0.15 _[0.07]	3.28 _[0.38]	2.35 _[0.22]	4.59 _[0.46]
transformer	10.63 _[0.03]	0.75 _[0.05]	0.16 _[0.02]	1.50 _[0.12]	0.07 _[0.01]	0.13 _[0.04]
bAbi dialog: Task5						
seq2seq_lstm	1.28 _[0.02]	0.38 _[0.11]	0.01 _[0.00]	0.10 _[0.06]	0.09 _[0.02]	0.42 _[0.38]
seq2seq_lstm_att	1.06 _[0.02]	0.64 _[0.07]	0.03 _[0.03]	0.22 _[0.04]	0.25 _[0.01]	1.10 _[0.80]
transformer	1.07 _[0.00]	0.36 _[0.02]	0.25 _[0.06]	0.37 _[0.06]	0.00 _[0.00]	0.00 _[0.00]

Tab. 7.2. Model performance across multiple datasets and sensitivity to different perturbations. Columns 1 & 2 report the test set perplexity (without perturbations) of different models. Columns 3-7 report the **increase** in perplexity ($\Delta PPL_{[\sigma]}$) when models are subjected to different perturbations. The mean (μ) and standard deviation [σ] across 5 runs are reported. The model that exhibits the highest sensitivity (higher the better) to a particular perturbation on a dataset is in bold. *seq2seq_lstm_att* are the most sensitive models **24/40** times, while transformers are the least with **6/40** times.

7.4.3. Models

We experimented with two different classes of models - recurrent and transformer-based sequence-to-sequence generative models. All data loading, model implementations and evaluations were done using the ParlAI framework. We used the default hyper-parameters for all the models as specified in ParlAI.

Recurrent Models. We trained a seq2seq (*seq2seq_lstm*) model where the encoder and decoder are parameterized as LSTMs [53]. We also experiment with using decoders that use an attention mechanism (*seq2seq_lstm_att*) [8]. The encoder and decoder LSTMs have 2 layers with 128 dimensional hidden states with a dropout rate of 0.1.

Transformer. Our transformer [166] model uses 300 dimensional embeddings and hidden states, 2 layers and 2 attention heads with no dropout. This model is significantly smaller than the ones typically used in machine translation since we found that the model that resembled [166] significantly overfit on all our datasets.

While the models considered in this work might not be state-of-the-art on the datasets considered, we believe these models are still competitive and used commonly enough at least as baselines, that the community will benefit by understanding their behavior. In this paper, we use early stopping with a patience of 10 on the validation set to save our best model. All models achieve close to the perplexity numbers reported for generative seq2seq models in their respective papers.

7.5. Results & Discussion

Our results are presented in Table 7.3 and Figure 7.1. Table 7.3 reports the perplexities of different models on test set in the second column, followed by the **increase** in perplexity when the dialog history is perturbed using the method specified in the column header. Rows correspond to models trained on different datasets. Figure 7.1 presents the change in perplexity for models when presented only with the k most recent utterances from the dialog history.

We make the following observations:

- (1) Models tend to show only tiny changes in perplexity in most cases, even under extreme changes to the dialog history, suggesting that they use far from all the information that is available to them.
- (2) Transformers are insensitive to word-reordering, indicating that they could be learning bag-of-words like representations.
- (3) The use of an attention mechanism in *seq2seq_lstm_att* and transformers makes these models use more information from earlier parts of the conversation than vanilla seq2seq models as seen from increases in perplexity when using only the last utterance.

Models	Test PPL	Only Last	Shuf	Rev	Drop First	Drop Last
		Utterance level perturbations ($\Delta PPL_{[\sigma]}$)				
DailyDialog						
seq2seq_lstm	32.90 _[1.40]	1.70 _[0.41]	3.35 _[0.38]	4.04 _[0.28]	0.13 _[0.04]	5.08 _[0.79]
seq2seq_lstm_att	29.65 _[1.10]	4.76 _[0.39]	2.54 _[0.24]	3.31 _[0.49]	0.32 _[0.03]	4.84 _[0.42]
transformer	28.73 _[1.30]	3.28 _[1.37]	0.82 _[0.40]	1.25 _[0.62]	0.27 _[0.19]	2.43 _[0.83]
Persona Chat						
seq2seq_lstm	43.24 _[0.99]	3.27 _[0.13]	6.29 _[0.48]	13.11 _[1.22]	0.47 _[0.21]	6.10 _[0.46]
seq2seq_lstm_att	42.90 _[1.76]	4.44 _[0.81]	6.70 _[0.67]	11.61 _[0.75]	2.99 _[2.24]	5.58 _[0.45]
transformer	40.78 _[0.31]	1.90 _[0.08]	1.22 _[0.22]	1.41 _[0.54]	-0.1 _[0.07]	1.59 _[0.39]
MutualFriends						
seq2seq_lstm	14.17 _[0.29]	1.44 _[0.86]	1.42 _[0.25]	1.24 _[0.34]	0.00 _[0.00]	0.76 _[0.10]
seq2seq_lstm_att	10.60 _[0.21]	32.13 _[4.08]	1.24 _[0.19]	1.06 _[0.24]	0.08 _[0.03]	1.35 _[0.15]
transformer	10.63 _[0.03]	20.11 _[0.67]	1.06 _[0.16]	1.62 _[0.44]	0.12 _[0.03]	0.81 _[0.09]
bAbi dialog: Task5						
seq2seq_lstm	1.28 _[0.02]	1.31 _[0.50]	43.61 _[15.9]	40.99 _[9.38]	0.00 _[0.00]	4.28 _[1.90]
seq2seq_lstm_att	1.06 _[0.02]	9.14 _[1.28]	41.21 _[8.03]	34.32 _[10.7]	0.00 _[0.00]	6.75 _[1.86]
transformer	1.07 _[0.00]	4.06 _[0.33]	0.38 _[0.02]	0.62 _[0.02]	0.00 _[0.00]	0.21 _[0.02]

Tab. 7.3. Model performance across multiple datasets and sensitivity to different perturbations. Columns 1 & 2 report the test set perplexity (without perturbations) of different models. Columns 3-7 report the **increase** in perplexity when models are subjected to different perturbations. The mean (μ) and standard deviation [σ] across 5 runs are reported. The *Only Last* column presents models with **only** the last utterance from the dialog history. The model that exhibits the highest sensitivity (higher the better) to a particular perturbation on a dataset is in bold. *seq2seq_lstm_att* are the most sensitive models **24/40** times, while transformers are the least with **6/40** times.

- (4) While transformers converge faster and to lower test perplexities, they don't seem to capture the conversational dynamics across utterances in the dialog history and are less sensitive to perturbations that scramble this structure than recurrent models.

7.6. Conclusion

This work studies the behaviour of generative neural dialog systems in the presence of synthetically introduced perturbations to the dialog history, that it conditions on. We find that both recurrent and transformer-based seq2seq models are not significantly affected even by drastic and unnatural modifications to the dialog history. We also find subtle differences between the way in which recurrent and transformer-based models use available context. By open-sourcing our code, we believe this paradigm of studying model behavior by introducing perturbations that destroys different kinds of structure present within the dialog history can be a useful diagnostic tool. We also foresee this paradigm being useful when building new dialog datasets to understand the kinds of information models use to solve them.

Chapter 8

Prologue to Second Article

8.1. Article Details

Taskmaster-1: Toward a Realistic and Diverse Dialog Dataset. Bill Byrne*, Karthik Krishnamoorthi*, Chinnadhurai Sankar*, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, Kyu-Young Kim, *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*

Personal Contribution. (*) denotes co-first authorship. I helped Bill Byrne & team with feedback during the dialog collection process. Specifically, helped identify imbalance issues. I ran all the dataset analyses, set baselines with the current state-of-the-art neural conversational models, computed human baseline scores for better evaluation and collected multiple human ground truth responses for each dialog in the test set. In addition to other sections of the paper, I mainly contributed to the models/experiments section of the paper with help from Arvind Neelakantan. I also used the Pointer-Generator network coded by Semih Yavuz as one of the baselines in my experiments.

8.2. Context

A significant barrier to progress in data-driven approaches to building dialog systems is the lack of high quality, goal-oriented conversational data. Collection of goal oriented datasets usually is a costly process as it usually involves setting up a “Wizard of Oz” (WOz) system [65], where one human agent acts as the digital assistant and another human agent

acts as the user. While the two-person approach to data collection creates a realistic scenario for robust, spoken dialog data collection, this technique is time consuming, complex and expensive, requiring considerable technical implementation as well as administrative procedures to train and manage agents and crowdsourced workers. In this article, we propose to address this issue by an alternative self-dialog approach.

8.3. Contributions

In this article, we introduce the initial release of the Taskmaster-1 dataset which includes 13,215 task-based dialogs comprising six domains. The first involves a two-person, spoken "Wizard of Oz" (WOz) approach in which trained agents and crowdsourced workers interact to complete the task while the second is "self-dialog" in which crowdsourced workers write the entire dialog themselves. Taskmaster-1 has richer and more diverse language than the current popular benchmark in task-oriented dialog, MultiWOZ [20]. Beyond the corpus and the methodologies used to create it, we present several baseline models including state-of-the-art neural seq2seq architectures together with perplexity and BLEU scores. We also provide qualitative human performance evaluations for these models and find that automatic evaluation metrics correlate well with human judgments. We publicly released our corpus containing conversations, API call and argument annotations, and also the human judgments.

8.4. Recent Developments

This paper was selected as an oral presentation at EMNLP 2019. Post our dataset release, another relevant research effort [120] released the the largest public task-oriented dialogue corpus. It consists of over 16000 dialogues in the training set spanning 26 services belonging to 16 domains. While the number of dialogs collected by [120] exceeded our dataset, Taskmaster-1's self-dialog approach is still scalable and in-expensive to collect. Another distinct difference between the two approaches is that while [120] ground their dialogs to pre-defined API services and intents, Taskmaster-1 grounds on predefined slot types. Independence over predefined API services allows models trained with Taskmaster-1 dataset to generalize to new types of API services which uses the predefined slot (or argument) types. We believe that grounding the conversations with argument types could be easier to scale to new intents which share common argument types.

Chapter 9

Taskmaster-1: Toward a Realistic and Diverse Dialog Dataset

9.1. Abstract

A significant barrier to progress in data-driven approaches to building dialog systems is the lack of high quality, goal-oriented conversational data. To help satisfy this elementary requirement, we introduce the initial release of the Taskmaster-1 dataset which includes 13,215 task-based dialogs comprising six domains. Two procedures were used to create this collection, each with unique advantages. The first involves a two-person, spoken "Wizard of Oz" (WOz) approach in which trained agents and crowdsourced workers interact to complete the task while the second is "self-dialog" in which crowdsourced workers write the entire dialog themselves. We do not restrict the workers to detailed scripts or to a small knowledge base and hence we observe that our dataset contains more realistic and diverse conversations in comparison to existing datasets. We offer several baseline models including state of the art neural seq2seq architectures with benchmark performance as well as qualitative human evaluations. Dialogs are labeled with API calls and arguments, a simple and cost effective approach which avoids the requirement of complex annotation schema. The layer of abstraction between the dialog model and the service provider API allows for a given model to interact with multiple services that provide similar functionally. Finally, the dataset will evoke interest in written vs. spoken language, discourse patterns, error handling and other linguistic phenomena related to dialog system research, development and design.

9.2. Introduction

Voice-based “personal assistants” such as Apple’s SIRI, Microsoft’s Cortana, Amazon Alexa, and the Google Assistant have finally entered the mainstream. This development is generally attributed to major breakthroughs in speech recognition and text-to-speech (TTS) technologies aided by recent progress in deep learning [75], exponential gains in compute power [152, 58], and the ubiquity of powerful mobile devices. The accuracy of machine learned speech recognizers [51] and speech synthesizers [163] are good enough to be deployed in real-world products and this progress has been driven by publicly available labeled datasets. However, conspicuously absent from this list is equal progress in machine learned conversational natural language understanding (NLU) and generation (NLG). The NLU and NLG components of dialog systems starting from the early research work [173] to the present commercially available personal assistants largely rely on rule-based systems. The NLU and NLG systems are often carefully programmed for very narrow and specific cases [46, 135]. General understanding of natural spoken behaviors across multiple dialog turns, even in single task-oriented situations, is by most accounts still a long way off. In this way, most of these products are very much hand crafted, with inherent constraints on what users can say, how the system responds and the order in which the various subtasks can be completed. They are high precision but relatively low coverage. Not only are such systems unscalable, but they lack the flexibility to engage in truly natural conversation.

Yet none of this is surprising. Natural language is heavily context dependent and often ambiguous, especially in multi-turn conversations across multiple topics. It is full of subtle discourse cues and pragmatic signals whose patterns have yet to be thoroughly understood. Enabling an automated system to hold a coherent task-based conversation with a human remains one of computer science’s most complex and intriguing unsolved problems [173]. In contrast to more traditional NLP efforts, interest in statistical approaches to dialog understanding and generation aided by machine learning has grown considerably in the last couple of years [129, 18, 50]. However, the dearth of high quality, goal-oriented dialog data is considered a major hindrance to more significant progress in this area [18, 94].

To help solve the data problem we present Taskmaster-1, a dataset consisting of 13,215 dialogs, including 5,507 spoken and 7,708 written dialogs created with two distinct procedures. Each conversation falls into one of six domains: ordering pizza, creating auto repair

appointments, setting up ride service, ordering movie tickets, ordering coffee drinks and making restaurant reservations. For the spoken dialogs, we created a “Wizard of Oz” (WOz) system [65] to collect two-person, spoken conversations. Crowdsourced workers playing the “user” interacted with human operators playing the “digital assistant” using a web-based interface. In this way, users were led to believe they were interacting with an automated system while it was in fact a human, allowing them to express their turns in natural ways but in the context of an automated interface. We refer to this spoken dialog type as “two-person dialogs”. For the written dialogs, we engaged crowdsourced workers to write the full conversation themselves based on scenarios outlined for each task, thereby playing roles of both the user and assistant. We refer to this written dialog type as “self-dialogs”. In a departure from traditional annotation techniques [50, 129, 20], dialogs are labeled with simple API calls and arguments. This technique is much easier for annotators to learn and simpler to apply. As such it is more cost effective and, in addition, the same model can be used for multiple service providers.

Taskmaster-1 has richer and more diverse language than the current popular benchmark in task-oriented dialog, MultiWOZ [20]. Table 9.1 shows that Taskmaster-1 has more unique words and is more difficult for language models to fit. We also find that Taskmaster-1 is more realistic than MultiWOZ. Specifically, the two-person dialogs in Taskmaster-1 involve more real-word entities than seen in MultiWOZ since we do not restrict conversations to a small knowledge base. Beyond the corpus and the methodologies used to create it, we present several baseline models including state-of-the-art neural seq2seq architectures together with perplexity and BLEU scores. We also provide qualitative human performance evaluations for these models and find that automatic evaluation metrics correlate well with human judgments. We will publicly release our corpus containing conversations, API call and argument annotations, and also the human judgments.

9.3. Related work

9.3.1. Human-machine vs. human-human dialog

[140] discuss the major features and differences among the existing offerings in an exhaustive and detailed survey of available corpora for data driven learning of dialog systems. One important distinction covered is that of human-human vs. human-machine dialog data,

hline Statistic	Self-dialogs	MultiWOZ
# unique words	21,894	19,175
# unique named entities	8,218	1,338
# utterances	169,469	132,610
# dialogs	7,708	10,438
Avg. utterances per dialog	21.99	13.70
Avg. tokens per utterance	8.62	13.82
Perplexity	17.08	15.62
BLEU	6.53	11.02

Tab. 9.1. Statistics comparison: Self-dialogs vs MultiWOZ corpus both containing approximately 10k dialogues each.

each having its advantages and disadvantages. Many of the existing task-based datasets have been generated from deployed dialog systems such as the Let’s Go Bus Information System [121] and the various Dialog State Tracking Challenges (DSTCs) [178]. However, it is doubtful that new data-driven systems built with this type of corpus would show much improvement since they would be biased by the existing system and likely mimic its limitations [182]. Since the ultimate goal is to be able to handle complex human language behaviors, it would seem that human-human conversational data is the better choice for spoken dialog system development [20]. However, learning from purely human-human based corpora presents challenges of its own. In particular, human conversation has a different distribution of understanding errors and exhibits turn-taking idiosyncrasies which may not be well suited for interaction with a dialog system [182, 140].

9.3.2. The Wizard of Oz (WOz) Approach and MultiWOZ

The WOz framework, first introduced by [65] as a methodology for iterative design of natural language interfaces, presents a more effective approach to human-human dialog collection. In this setup, users are led to believe they are interacting with an automated

assistant but in fact it is a human behind the scenes that controls the system responses. Given the human-level natural language understanding, users quickly realize they can comfortably and naturally express their intent rather than having to modify behaviors as is normally the case with a fully automated assistant. At the same time, the machine-oriented context of the interaction, i.e. the use of TTS and slower turn taking cadence, prevents the conversation from becoming fully fledged, overly complex human discourse. This creates an idealized spoken environment, revealing how users would openly and candidly express themselves with an automated assistant that provided superior natural language understanding.

Perhaps the most relevant work to consider here is the recently released MultiWOZ dataset [20], since it is similar in size, content and collection methodologies. MultiWOZ has roughly 10,000 dialogs which feature several domains and topics. The dialogs are annotated with both dialog states and dialog acts. MultiWOZ is an entirely written corpus and uses crowdsourced workers for both assistant and user roles. In contrast, Taskmaster-1 has roughly 13,000 dialogs spanning six domains and annotated with API arguments. The two-person spoken dialogs in Taskmaster-1 use crowdsourcing for the user role but trained agents for the assistant role. The assistant’s speech is played to the user via TTS. The remaining 7,708 conversations in Taskmaster-1 are self-dialogs, in which crowdsourced workers write the entire conversation themselves. As [73, 108] show, self dialogs are surprisingly rich in content.

9.4. The Taskmaster Corpus

9.4.1. Overview

There are several key attributes that make Taskmaster-1 both unique and effective for data-driven approaches to building dialog systems and for other research.

Spoken and written dialogs: While the spoken sources more closely reflect conversational language [22], written dialogs are significantly cheaper and easier to gather. This allows for a significant increase in the size of the corpus and in speaker diversity.

Goal-oriented dialogs: All dialogs are based on one of six tasks: ordering pizza, creating auto repair appointments, setting up rides for hire, ordering movie tickets, ordering coffee drinks and making restaurant reservations.

ASSISTANT: How can I help you?

USER: Hi, could you help me with booking movie tickets for tonight?

ASSISTANT: What movie are you interested in?

USER: The Upside.

ASSISTANT: Did you have a theater in mind?

USER: Could you check if the Regal Neshaminy... No, AMC Neshaminy in Neshaminy, PA is playing it?

ASSISTANT: Could you spell that?

USER: Sure, n e s h a m i n y.

ASSISTANT: I have a showtime at 7:30 and at 10:30, is that okay?

USER: Yes, could you get two tickets for the 7:30?

ASSISTANT: One moment. Okay so that's 2 tickets for 7:30 at the AMC Neshaminy 24?

USER: Yes.

ASSISTANT: It'll be twenty-four ninety-nine for your tickets.

USER: That sounds great.

ASSISTANT: I've confirmed your tickets, they'll arrive via text shortly. Did you need any other information?

USER: No, that was it. Thank you so much for your help.

ASSISTANT: Great, no problem. I hope you have fun.

USER: I hope so, too. Thank you so much.

Fig. 9.1. Sample Taskmaster-1 two-person dialog

MAIN TASK: Users will pretend they are using a voice-powered personal digital assistant to book movie tickets for a film they ALREADY have in mind.

- (1) In several turns (not just one!), cover the following:
 - (a) Film name
 - (b) Number of people
 - (c) City
 - (d) Theater
 - (e) Time
 - (f) If applicable: 3D vs. IMAX vs. standard.
- (2) They may also want to know things like:
 - (a) Run time
 - (b) End time
 - (c) Director, actors, etc.
- (3) Make sure to CONFIRM all the relevant ticket details before the end of the dialogue INCLUDING:
 - (a) Total cost for two tickets
 - (b) Time, location, theater
- (4) You can assume you have the user’s account info with the ticket service—so no credit card information is necessary.
- (5) After confirming the details, end the conversation by confirming that the tickets are being sent to the user’s mobile device as a text message.

Fig. 9.2. Sample instructions for agents playing “assistant” role

Two collection methods: The two-person dialogs and self-dialogs each have pros and cons, revealing interesting contrasts.

Multiple turns: The average number of utterances per dialog is about 23 which ensures context-rich language behaviors.

API-based annotation: The dataset uses a simple annotation schema providing sufficient grounding for the data while making it easy for workers to apply labels consistently.

Size: The total of 13,215 dialogs in this corpus is on par with similar, recently released datasets such as MultiWOZ [20].

MAIN TASK: Pretend you are using your voice-powered digital assistant to book movie tickets.

- (1) Start by thinking of a particular movie PLAYING NOW in theaters that you'd like to see. (Use the internet to find one if necessary.)
- (2) Choose a DIFFERENT CITY from where you live, work, or happen to be at the moment.
- (3) Pretend you've decided to see this movie tonight and you're taking a friend.
- (4) The assistant will ask about all relevant details BUT you should make sure it covers all your needs.
- (5) You can assume you already have an account with the ticket service—so no credit card information is necessary.
- (6) The assistant will end the conversation by confirming that your tickets are being sent to your mobile device as a text message. (And you can respond thanks, goodbye, ok, etc. for a final closing turn, if you like).

Fig. 9.3. Sample instructions for crowdsourced workers playing “user” role

9.4.2. Two-person, spoken dataset

In order to replicate a two-participant, automated digital assistant experience, we built a WOz platform that pairs agents playing the digital assistant with crowdsourced workers playing the user in task-based conversational scenarios. An example dialog from this dataset is given in Figure 9.1.

9.4.2.1. WOz platform and data pipeline

While it is beyond the scope of this work to describe the entire system in detail, there are several platform features that help illustrate how the process works.

Modality: The agents playing the assistant type their input which is in turn played to the user via text-to-speech (TTS) while the crowdsourced workers playing the user speak aloud to the assistant using their laptop and microphone. We use WebRTC to establish the audio channel. This setup creates a digital assistant-like communication style.

Conversation and user quality control: Once the task is completed, the agents tag each conversation as either successful or problematic depending on whether the session had technical glitches or user behavioral issues. We are also then able to root out problematic users based on this logging.

Agent quality control: Agents are required to login to the system which allows us to monitor performance including the number and length of each session as well as their averages.

User queuing: When there are more users trying to connect to the system than available agents, a queuing mechanism indicates their place in line and connects them automatically once they move to the front of the queue.

Transcription: Once complete, the user’s audio-only portion of the dialog is transcribed by a second set of workers and then merged with the assistant’s typed input to create a full text version of the dialog. Finally, these conversations are checked for transcription errors and typos and then annotated, as described in Section 9.4.4.

9.4.2.2. *Agents, workers and training*

Both agents and crowdsourced workers are given written instructions prior to the session. Examples of each are given in Figure 9.2 and Figure 9.3. The instructions continue to be displayed on screen to the crowdsourced workers while they interact with the assistant. Instructions are modified at times (for either participant or both) to ensure broader coverage of dialog scenarios that are likely to occur in actual user-assistant interactions. For example, in one case users were asked to change their mind after ordering their first item and in another agents were instructed to tell users that a given item was not available. Finally, in their instructions, crowdsourced workers playing the user are told they will be engaging in conversation with “a digital assistant”. However, it is plausible that some suspect human intervention due to the advanced level of natural language understanding from the assistant side.

Agents playing the assistant role were hired from a pool of dialog analysts and given two hours of training on the system interface as well as on how to handle specific scenarios such as uncooperative users and technical glitches. Uncooperative users typically involve those who either ignored agent input or who rushed through the conversation with short phrases. Technical issues involved dropped sessions (e.g. WebRTC connections failed) or cases in

which the user could not hear the agent or vice-versa. In addition, weekly meetings were held with the agents to answer questions and gather feedback on their experiences. Agents typically work four hours per day with dialog types changing every hour. Crowdsourced workers playing the user are accessed using Amazon Mechanical Turk. Payment for a completed dialog session lasting roughly five to seven minutes was typically in the range of \$1.00 to \$1.30. Problematic users are detected either by the agent involved in the specific dialog or by post-session assessment and removed from future requests.

- (1) Think of a particular movie **PLAYING NOW** in theaters that you'd like to see. (Use the internet to find one if necessary.)
 - (2) Choose a **DIFFERENT CITY** from where you live, work, or happen to be at the moment.
 - (3) Pretend you've decided to see this movie tonight and you're taking a friend.
 - (4) Use the internet to look up the details of the city, the theater name, showtimes offered, ticket prices, and any additional options like 3D, etc.
 - (5) **MAIN TASK:** Pretend you call your personal assistant on the phone who will book the ticket for you. Write the conversation that would happen between you and your assistant in order to buy two tickets.
 - (6) **MAKE SURE** the assistant asks about all relevant details (see #4) **INCLUDING** the number of tickets needed. **BUT** you should choose the order that makes sense to you as far what details to ask (theater, times, etc)
 - (7) You can assume you already have an account with the ticket service—so no credit card information is necessary.
 - (8) The assistant should end the conversation by confirming that your tickets are being sent to your mobile device as a text message. (And you can respond thanks, goodbye, ok, etc. for a final closing turn, if you like).
- **YOUR TASK:** Write the conversation that results between you and your assistant. It must be at least 10 turns long (for both you and the assistant). Below we have provided 15 turns in case you need more. **KEEP IT NEW AND FRESH! DON'T REPEAT DIALOGUES FROM THE PAST!**

Fig. 9.4. Sample instructions for written “self-dialogs”

USER: Hi I would like to buy 2 tickets for Shazam!

ASSISTANT: What city would you like to see this movie?

USER: Ontario, California

ASSISTANT: Ok, I'll check that location for you.

USER: I would prefer the Edwards Ontario Mountain Village, since it's closest to me and my guest.

ASSISTANT: What time is best for you?

USER: Either 4 or 6 pm.

ASSISTANT: I'm sorry, but it looks like the 4:10 and the 6:10 pm showings are sold out.

USER: That's too bad. I really wanted to see that movie.

ASSISTANT: I'm sorry. Is there another movie you would like to see?

USER: How about Captain Marvel at the Edwards Ontario Mountain theater.

ASSISTANT: Show times are 3:45, 7:10 and 10:10 pm. Which would you like?

USER: I am interested in the 7:10 showing.

ASSISTANT: I'm sorry, it looks like the 7:10 showing is also sold out.

USER: Wow, that's too bad.

ASSISTANT: I'm sorry. Is there another movie you would like me to look up?

USER: No, I think I'll pass on the movies tonight since those were the two I really wanted to see.

ASSISTANT: If you want, I can check another theater.

USER: No, that's fine. Thank you for your help.

ASSISTANT: You're welcome.

Fig. 9.5. Sample one-person, written dialog

9.4.3. Self-dialogs (one-person written dataset)

While the two-person approach to data collection creates a realistic scenario for robust, spoken dialog data collection, this technique is time consuming, complex and expensive, requiring considerable technical implementation as well as administrative procedures to train and manage agents and crowdsourced workers. In order to extend the Taskmaster dataset at minimal cost, we use an alternative self-dialog approach in which crowdsourced workers write the full dialogs themselves (i.e. interpreting the roles of both user and assistant).

9.4.3.1. *Task scenarios and instructions*

Targeting the same six tasks used for the two-person dialogs, we again engaged the Amazon Mechanical Turk worker pool to create self-dialogs, this time as a written exercise. In this case, users are asked to pretend they have a personal assistant who can help them take care of various tasks in real time. They are told to imagine a scenario in which they are speaking to their assistant on the phone while the assistant accesses the services for one of the given tasks. They then write down the entire conversation. Figure 9.4 shows a sample set of instructions.

9.4.3.2. *Pros and cons of self-dialogs*

The self-dialog technique renders quality data and avoids some of the challenges seen with the two-person approach. To begin, since the same person is writing both sides of the conversation, we never see misunderstandings that lead to frustration as is sometimes experienced between interlocutors in the two-person approach. In addition, all the self-dialogs follow a reasonable path even when the user is constructing conversations that include understanding errors or other types of dialog glitches such as when a particular choice is not available. As it turns out, crowdsourced workers are quite effective at recreating various types of interactions, both error-free and those containing various forms of linguistic repair. The sample dialog in Figure 9.5 shows the result of a self-dialog exercise in which workers were told to write a conversation with various ticket availability issues that is ultimately unsuccessful.

Two more benefits of the self-dialog approach are its efficiency and cost effectiveness. We were able to gather thousands of dialogs in just days without transcription or trained agents,

USER: Finally, I need the table to be
for three people and 8pm.
ASSISTANT: One moment....OK, I
have your table for three
(**num.guests.accept**) at
8pm (**time.reservation.accept**)
reserved.

Fig. 9.6. Indicating transaction status with “accept” or “reject”

and spent roughly six times less per dialog. Despite these advantages, the self-dialog written technique cannot recreate the disfluencies and other more complex error patterns that occur in the two-person spoken dialogs which are important for model accuracy and coverage.

9.4.4. Annotation

We chose a highly simplified annotation approach for Taskmaster-1 as compared to traditional, detailed strategies which require robust agreement among workers and usually include dialog state and slot information, among other possible labels. Instead we focus solely on API arguments for each type of conversation, meaning just the variables required to execute the transaction. For example, in dialogs about setting up UBER rides, we label the “to” and “from” locations along with the car type (UberX, XL, Pool, etc). For movie tickets, we label the movie name, theater, time, number of tickets, and sometimes screening type (e.g. 3D vs. standard). A complete list of labels is included with the corpus release.

As discussed in Section 9.4.2.2, to encourage diversity, at times we explicitly ask users to change their mind in the middle of the conversation, and the agents to tell the user that the requested item is not available. This results in conversations having multiple instances of the same argument type. To handle this ambiguity, in addition to the labels mentioned above, the convention of either “accept” or “reject” was added to all labels used to execute the transaction, depending on whether or not that transaction was successful.

In Figure 9.6, both the number of people and the time variables in the assistant utterance would have the “.accept” label indicating the transaction was completed successfully. If the utterance describing a transaction does not include the variables by name, the whole sentence

Statistic	Self-dialogs	Two Person
# unique words	17,275	13,490
# utterances	110,074	132,407
# dialogs	5000	5000
Avg. utterances per dialog	22.01	24.04
Avg. tokens per utterance	8.62	7.54
Perplexity	16.28	6.44
BLEU	4.73	15.16
Joint-Perplexity	16.44	6.04
Joint-BLEU	5.80	13.09

Tab. 9.2. Statistics comparison: Self-dialogs vs two person corpus both containing 5k dialogs. Perplexity and BLEU are reported for Transformer baseline. Joint-Perplexity and Joint-BLEU are perplexity/BLEU scores from the joint training of self-dialogs and two-person but evaluated with their respective test sets.

is marked with the dialog type. For example, a statement such as *The table has been booked for you* would be labeled as **reservation.accept**.

9.5. Dataset Analysis

9.5.1. Self-dialogs vs MultiWOZ

We quantitatively compare our self-dialogs (Section 9.4.3) with the MultiWOZ dataset in Table 9.1. Compared to MultiWOZ, we do not ask the users and assistants to stick to detailed scripts and do not restrict them to have conversations surrounding a small knowledge base. Table 9.1 shows that our dataset has more unique words, and has almost twice the number of utterances per dialog than the MultiWOZ corpus. Finally, when trained with the Transformer [165] model, we observe significantly higher perplexities and lower BLEU scores for our dataset compared to MultiWOZ suggesting that our dataset conversations are difficult to model. Finally, Table 9.1 also shows that our dataset contains close to 10

times more real-world named entities than MultiWOZ and thus, could potentially serve as a realistic baseline when designing goal oriented dialog systems. MultiWOZ has only 1338 unique named entities and only 4510 unique values (including date, time etc.) in their dataset.

9.5.2. Self-dialogs vs Two-person

In this section, we quantitatively compare 5k conversations each of self-dialogs (Section 9.4.3) and two-person (Section 9.4.2). From Table 9.2, we find that self-dialogs exhibit higher perplexity (almost 3 times) compared to the two-person conversations suggesting that self-dialogs are more diverse and contains more non-conventional conversational flows which is inline with the observations in Section-9.4.3.2. While the number of unique words are higher in the case of self-dialogs, conversations are longer in the two-person conversations. We also report metrics by training a single model on both the datasets together.

9.5.3. Baseline Experiments: Response Generation

We evaluate various seq2seq architectures [156] on our self-dialog corpus using both automatic evaluation metrics and human judgments. Following the recent line of work on generative dialog systems [169], we treat the problem of response generation given the dialog history as a conditional language modeling problem. Specifically we want to learn a conditional probability distribution $P_\theta(U_t|U_{1:t-1})$ where U_t is the next response given dialog history $U_{1:t-1}$. Each utterance U_i itself is comprised of a sequence of words $w_{i_1}, w_{i_2} \dots w_{i_k}$. The overall conditional probability is factorized autoregressively as

$$P_\theta(U_t|U_{1:t-1}) = \prod_{i=1}^n P_\theta(w_{t_i}|w_{t_{1:i-1}}, U_{1:t-1})$$

P_θ , in this work, is parameterized by a recurrent, convolution or Transformer-based seq2seq model.

n-gram: We consider 3-gram and 4-gram conditional language model baseline with interpolation. We use random grid search for the best coefficients for the interpolated model.

Convolution: We use the *fconv* architecture [42] and default hyperparameters from the fairseq [110] framework. We train the network with ADAM optimizer [70] with learning rate of 0.25 and dropout probability set to 0.2.

Baseline Models	PPL	BLEU	Ratings (LIKERT)	Rank
GPT-2 (117M)	-	0.26	-	-
3-gram	38.12	0.20	-	-
4-gram	34.49	0.21	-	-
LSTM	25.73	4.45	-	-
Convolution	21.25	5.09	2.89	3
LSTM-attention	20.05	5.12	3.51	2
Transformer	18.19	6.11	3.22	1

Tab. 9.3. Evaluation of various seq2seq architectures [156] on our self-dialog corpus using both automatic evaluation metrics and human judgments. Human evaluation ratings in the 1-5 LIKERT scale (higher the better), and human ranking are averaged over 500 x 3 ratings (3 crowdsourced workers per rating).

LSTM: We consider LSTM models [54] with and without attention [9] and use the tensor2tensor [164] framework for the LSTM baselines. We use a two-layer LSTM network for both the encoder and the decoder with 128 dimensional hidden vectors.

Transformer: As with LSTMs, we use the tensor2tensor framework for the Transformer model. Our Transformer [165] model uses 256 dimensions for both input embedding and hidden state, 2 layers and 4 attention heads. For both LSTMs and Transformer, we train the model with ADAM optimizer ($\beta_1 = 0.85$, $\beta_2 = 0.997$) and dropout probability set to 0.2.

GPT-2: Apart from supervised seq2seq models, we also include results from pre-trained GPT-2 [117] containing 117M parameters.

We evaluate all the models with perplexity and BLEU scores (Table 9.3). Additionally, we perform two kinds of human evaluation - Ranking and Rating (LIKERT scale) for the top-3 performing models - Convolution, LSTM-attention and Transformer. For the ranking task, we randomly show 500 partial dialogs and generated responses of the top-3 models from the test set to three different crowdsourced workers and ask them to rank the responses based on their relevance to the dialog history. For the rating task, we show the model responses individually to three different crowdsourced workers and ask them to rate the responses on a 1-5 LIKERT scale based on their appropriateness to the dialog history. From Table-9.4, we

Evalation method	Inter-Annotator Reliability (Krippendorf’s Alpha)
Rating (1-5 LIKERT)	0.21
Ranking	0.29

Tab. 9.4. Inter-Annotator Reliability scores of seq2seq model responses computed for 500 self-dialogs from the test set, each annotated by 3 crowdsourced workers.

Model	Micro F1 (%)
Transformer	48.73
Transformer + copy	51.79

Tab. 9.5. API Argument prediction accuracy for Self-dialogs. API arguments are annotated as spans in the utterances.

see that inter-annotator reliability scores (Krippendorf’s Alpha) are higher for the ranking task compared to the rating task. From Table 9.3, we see that Transformer is the best performing model on automatic evaluation metrics. It is interesting to note that there is a strong correlation between BLEU score and human ranking judgments.

9.5.4. Baseline Experiments: Argument Prediction

Next, we discuss a set of baseline experiments for the task of argument prediction. API arguments are annotated as spans in the dialog (Section 9.4.4). We formulate this problem as mapping text conversation to a sequence of output arguments. Apart from the seq2seq Transformer baseline, we consider an additional model - an enhanced Transformer seq2seq model where the decoder can choose to copy from the input or generate from the vocabulary [101, 47]. Since all the API arguments are input spans, the copy model having the correct inductive bias achieves the best performance.

9.6. Conclusion

To address the lack of quality corpora for data-driven dialog system research and development, this paper introduces Taskmaster-1, a dataset that provides richer and more diverse language as compared to current benchmarks since it is based on unrestricted, task-oriented

conversations involving more real-word entities. In addition, we present two data collection methodologies, both spoken and written, that ensure both speaker diversity and conversational accuracy. Our straightforward, API-oriented annotation technique is much easier for annotators to learn and simpler to apply. We give several baseline models including state-of-the-art neural seq2seq architectures, provide qualitative human performance evaluations for these models, and find that automatic evaluation metrics correlate well with human judgments.

Chapter 10

Prologue to third Article

10.1. Article Details

Deep Reinforcement Learning For Modeling Chit-Chat Dialog With Discrete Attributes. Chinnadhurai Sankar, Sujith Ravi *20th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), 2019*

Personal Contribution. I came up with the idea of jointly modeling dialog with discrete dialog states during discussions with Sujith Ravi. I implemented the prototype of the model in Tensorflow and iterated on the prototype to significantly improve its performance with guidance from Sujith. The paper was largely written by me and Sujith helped improve its presentation.

10.2. Context

Open domain dialog systems face the challenge of being repetitive and producing generic responses [80, 142, 172]. Several approaches have been proposed in the literature to address the generic response generation issue [80, 148, 144, 189, 138]. This paper proposes to improve response generation quality in open domain dialog systems by jointly modeling the utterances with the dialog attributes of each utterance. Dialog attributes of an utterance refer to discrete features or aspects associated with an utterance like dialog-acts, sentiment, emotion, speaker identity, speaker personality, etc. This modeling framework also enables us to formulate the dialog attribute selection as a reinforcement learning (RL) problem and optimize the policy initialized by the supervised training using REINFORCE [183].

10.3. Contributions

In this paper we have two main contributions - 1) we propose a conditional utterance generation model in which the next utterance is conditioned on the dialog attributes corresponding to the next utterance. 2) we propose to formulate the dialog attribute selection as a reinforcement learning (RL) problem using REINFORCE [183]. Additionally, we annotate an existing open domain dialog dataset using dialog attribute classifiers trained with tagged datasets and demonstrate both quantitative (in terms of token perplexity/embedding metrics) and qualitative improvements (based on human evaluations) in generating interesting responses.

10.4. Recent Developments

This paper was awarded the best paper in SIGDIAL 2019. Following our paper, [132] extended our approach by replacing discrete dialog attributes with a continuous latent variable. They also follow a similar fine-tuning technique to frame the latent variable prediction as a reinforcement learning (RL) problem using REINFORCE [183].

Chapter 11

Deep Reinforcement Learning For Modeling Chit-Chat Dialog With Discrete Attributes

11.1. Abstract

Open domain dialog systems face the challenge of being repetitive and producing generic responses. In this paper, we demonstrate that by conditioning the response generation on *interpretable* discrete dialog attributes and *composed* attributes, it helps improve the model perplexity and results in diverse and interesting non-redundant responses. We propose to formulate the dialog attribute prediction as a reinforcement learning (RL) problem and use policy gradients methods to optimize utterance generation using long-term rewards. Unlike existing RL approaches which formulate the token prediction as a policy, our method reduces the complexity of the policy optimization by limiting the action space to dialog attributes, thereby making the policy optimization more practical and sample efficient. We demonstrate this with experimental and human evaluations.

11.2. Introduction

Following the success of neural machine translation systems [8, 156, 27], there has been a growing interest in adapting the encoder-decoder models to model open-domain conversations [150, 141, 142, 167]. This is done by framing the next utterance generation as a machine translation problem by treating the dialog history as the source sequence and the next utterance as the target sequence. Then the models are trained end-to-end with Maximum Likelihood (MLE) objective without any hand crafted structures like slot-value pairs, dialog manager, etc., used in conventional dialog modeling [74]. Such data driven

approaches are worth pursuing in the context of open-domain conversations since the next utterance distribution in open-domain conversations exhibit high entropy which makes it impractical to manually craft good features.

While the encoder-decoder approaches are promising, lack of specificity has been one of the many challenges [172] in modeling non-goal oriented dialogs. Recent encoder-decoder based models usually tend to generate generic or dull responses like “*I don’t know.*”. One of the main causes are the implicit imbalances present in the dialog datasets that tend to potentially handicap the models into generating uninteresting responses.

Imbalances in a dialog dataset can be broadly divided into two categories: *many-to-one* and *one-to-many*. *Many-to-one* imbalance occurs when the dataset contain very similar responses to several different dialog contexts. In such scenarios, decoder learns to ignore the context (considering it as noise) and behaves like a regular language model. Such a decoder would not generalize to new contexts and will end up predicting generic responses for all contexts. In the *one-to-many* case, the dataset may exhibit a different type of imbalance where a certain type of generic response may be present in abundance compared to other plausible interesting responses for the same dialog context [172]. When trained with a maximum-likelihood (MLE) objective, generative models usually tend to place more probability mass around the most commonly observed responses for a given context. So, we end up observing little variance in the generated responses in such cases. While these two imbalances are problematic for training a dialog model, they are also inherent characteristics of a dialog dataset which cannot be removed.

Several approaches have been proposed in the literature to address the generic response generation issue. [80] propose to modify the loss function to increase the diversity in the generated responses. Multi-resolution RNN [138] addresses the above issue by additionally conditioning with entity information in the previous utterances. Alternatively, [148] uses external knowledge from a retrieval model to condition the response generation. Latent variable models inspired by Conditional Variational Autoencoders (CVAEs) are explored in [144, 189]. While models with continuous latent variables tend to be uninterpretable, discrete latent variable models exhibit high variance during inference. [144] append discrete attributes such as sentiment to the latent representation to generate next utterance.

11.2.1. Contributions

New Conditional Dialog Generation Model. Drawing insights from [144, 190], we propose a *conditional utterance generation model* in which the next utterance is conditioned on the dialog attributes corresponding to the next utterance. To do this, we first predict the higher level dialog attributes corresponding to the next response. Then we generate the next utterance conditioned on the dialog context and predicted attributes. The dialog attribute of an utterance refers to discrete features or aspects associated with the utterance. Example attributes include dialog-acts, sentiment, emotion, speaker id, speaker personality or other user defined discrete features of an utterance. While previous research works lack the framework to learn to predict the attributes of the next utterance and mainly view the next utterance’s attribute as a control variable in their models, our method learns to predict the attributes in an end-to-end manner. This alleviates the need to have utterances annotated with attributes during inference.

RL for Dialog Attribute Selection. Further, it also enables us to formulate the dialog attribute selection as a reinforcement learning (RL) problem and optimize the policy initialized by the supervised training using REINFORCE [183]. While the Supervised pre-training helps the model to generate utterances coherent with the dialog history, the RL formulation encourages the model to generate utterances optimized for long term rewards like diversity, user-satisfaction scores etc. This way of optimizing the policy over the discrete dialog attribute space is more practical as the action space is low dimensional instead of the entire vocabulary (as common in policies which involve predicting the next token to generate).

By using REINFORCE [183] to further optimize the dialog attribute selection process, we then show improvements in specificity of the generated responses both qualitatively (based on human evaluations) and quantitatively (with respect to the *diversity* measures). The diversity scores, *distinct-1* and *distinct-2* are computed as the fraction of uni-grams and bi-grams in the generated responses as described in [80].

Improvements on Dialog datasets demonstrated through quantitative & qualitative Evaluations: Additionally, we annotate an existing open domain dialog dataset using dialog attribute classifiers trained with tagged datasets like Switchboard [43, 59], Frames [133] and demonstrate both quantitative (in terms of token perplexity/embedding metrics [131, 107])

and qualitative improvements (based on human evaluations) in generating interesting responses. In this work, we show results with two types of dialog attributes - sentiment and dialog-acts. It is worth investigating this approach as we need not invest much in training classifiers for very high accuracy and we show empirically that annotations from classifiers with low accuracy are able to boost token perplexity. We conjecture that the irregularities in the auto-annotated dialog attributes induce a regularization effect while training deep neural networks analogous to the dropout mechanism. Also, annotating utterances with many types of dialog attributes could increase the regularization effect and potentially tip the utterance generation in the favor of certain low frequency but interesting responses.

In this work, we are mainly interested in exploring the impact of the jointly modeling extra discrete dialog attributes along with dialog history for next utterance generation and their contribution to addressing the generic response problem. Although our approach is flexible enough to include latent variables additionally, we mainly focus on the contribution of dialog attributes to address the "generic" response issue in this work.

11.3. Attribute Conditional HRED

In this paper, we extend the *HRED* [141] model (elaborated in the Appendix section) by jointly modeling the utterances with the dialog attributes of each utterance. *HRED* is a encoder-decoder model consisting of a token-level RNN encoder and an utterance-level RNN encoder to summarize the dialog context followed by a token-level RNN decoder to generate the next utterance. Assuming that the next utterance and its dialog attributes are conditionally independent given the dialog context, the joint probability can be factorized into dialog attributes prediction, followed by next utterance generation conditioned on the predicted dialog attributes as shown in equation 11.3.1 .

$$P(U_m, DA_{1:K}|U_{1:m-1}) = \prod_{i=1}^K P(DA_i|U_{1:m-1}) * P(U_m|U_{1:m-1}, DA_{1:K}) \quad (11.3.1)$$

where $DA_{1:K}$ denote K different dialog attributes corresponding to the utterance U_m . U_m is the m_{th} utterance, $U_{1:m-1}$ are the past utterances. For instance, if we condition on three dialog attributes - *sentiment*, *dialog-acts* and *emotion*, we would have $K = 3$. Further, we assume that the dialog attributes are conditionally independent given the dialog context.

More simply, we predict the attributes of the next utterance and then, condition on the previous context & the predicted attributes to generate the next utterance.

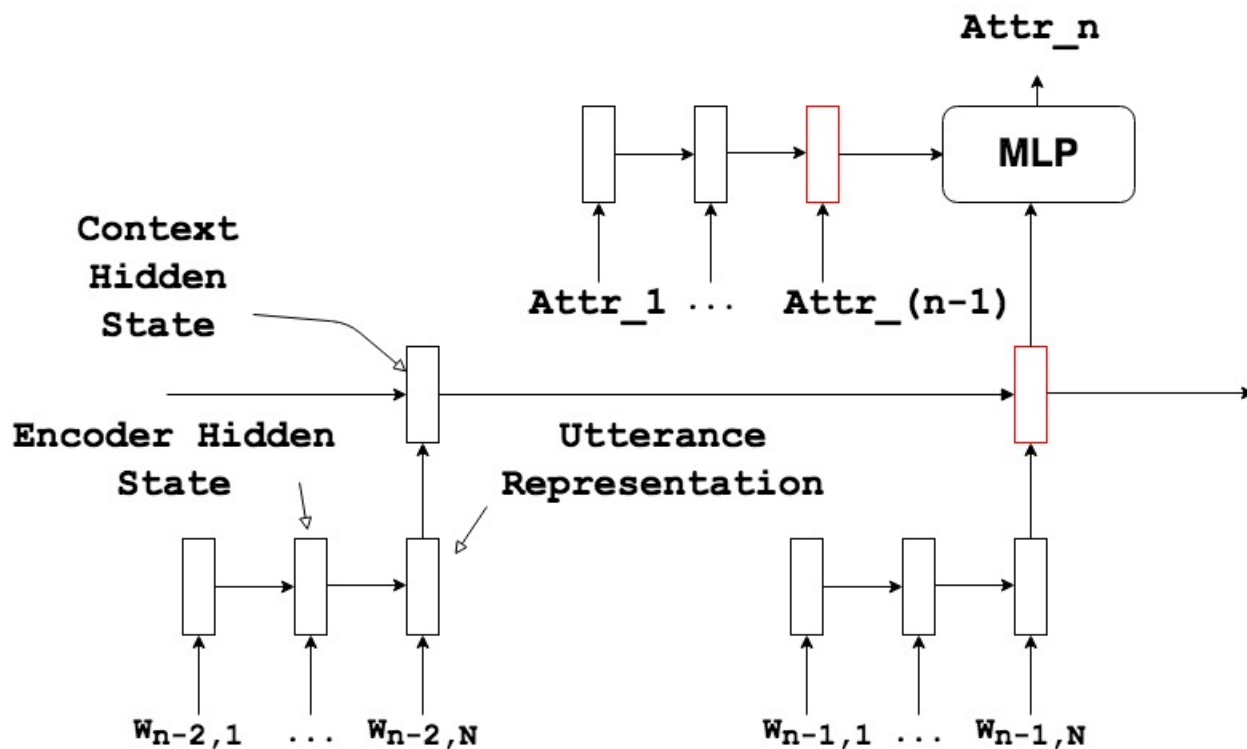


Fig. 11.1. Dialog attribute classification: We predict the dialog attribute of the next utterance based on the previous context and attributes corresponding to the previous utterances. Please note that we depict only a single attribute for convenience

11.3.1. Dialog Attribute Prediction

We predict the dialog attribute of the next utterance conditioned on the context vector i.e. summary of the previous utterances and the dialog attributes of the previous utterances. We first pass the attributes of all the previous utterances through an RNN. We combine only the last hidden state of this RNN with the context vector (represents the summary of all the previous utterances) to predict the dialog attribute of the next utterance as shown in Figure 11.1.

If the dialog dataset is not annotated with the dialog attributes, we build a classifier (with a manually tagged dataset) to annotate the dialog attributes. This classifier is a simple MLP. We empirically show that this classifier need not have high accuracy to improve the

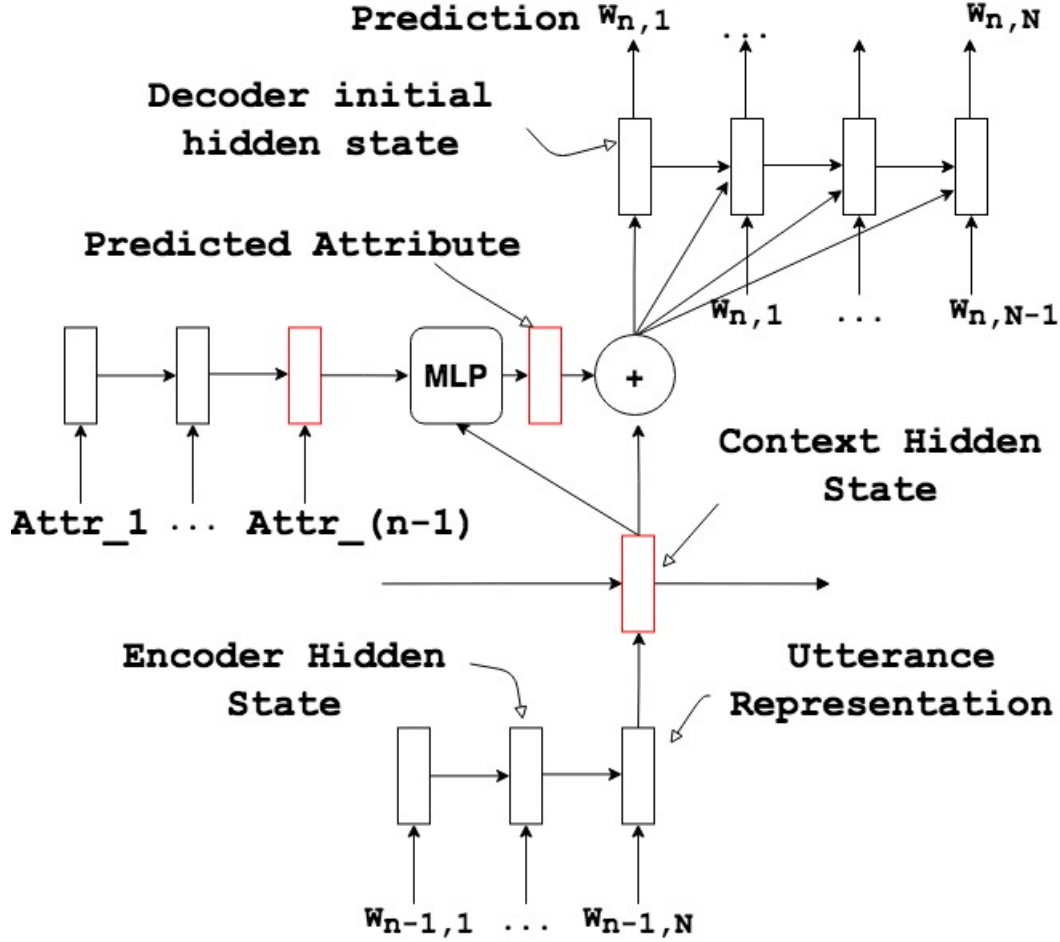


Fig. 11.2. Attribute Conditional HRED : Token generation is additionally conditioned on the predicted dialog attributes. The dialog attribute’s embedding is concatenated with the context vector.

dialog modeling. We hypothesize that few misclassified attributes could potentially provide a regularization effect similar to the dropout mechanism [151].

11.3.2. Conditional Response Generation

After the dialog attributes prediction, we generate the next utterance conditioned on the dialog context and the predicted attributes as shown in Figure 11.2. Token generation of the next utterance is modelled as in equation 11.3.2. The context and attributes are combined by concatenating their corresponding hidden states.

$$h_{dec_{m,n}} = f_{dec}(h_{dec_{m,n-1}}, w_{m,n-1}, \mathbf{c}_m) \quad (11.3.2)$$

where $h_{dec_{m,n}}$ is the recurrent hidden state of the decoder after seeing $n - 1$ words in the m -th utterance, f_{dec} is the token level response decoder, and

$$\mathbf{c}_m = [s_{m-1}; da_m^1; da_m^2; \dots; da_m^K] \quad (11.3.3)$$

where s_{m-1} is the summary of previous $m - 1$ utterances (recurrent hidden state of the utterance-level encoder), and $da_m^1, da_m^2, \dots, da_m^K$ are the K dialog attribute embeddings corresponding to the m -th utterance.

During inference, we first predict the dialog attributes of the dialog context. We then predict the dialog attribute of the next utterance conditioned on the predicted attribute and the hierarchical utterance representations. We combine the predicted attribute’s embedding vector with the context representation to generate the next utterance. Looking from another perspective, we could formulate the conditional utterance generation problem as a multi-task problem where we jointly learn to predict the dialog attributes and tokens of the next utterance.

11.3.3. RL for Dialog Attribute Prediction

Often the MLE objective does not capture the true goal of the conversation and lacks the framework which can take developer-defined rewards into account for modeling such goals. Also, the MLE-based seq2seq models fail to model long term influence of the utterances on the dialog flow causing coherency issues. This calls for a Reinforcement Learning (RL) based framework which has the ability to optimize policies for maximizing long term rewards. At the core, the MLE objective tries to increase the conditional utterance probabilities and influences the model to place higher probabilities over the commonly occurring utterances. On the other hand, RL based methods circumvent this issue by shifting the optimization problem to maximizing long term rewards which could promote diversity, coherency, etc.

Previous approaches [82, 71, 78] propose to model the token prediction of the next utterance as a reinforcement learning problem and optimize the models to maximize hand-crafted rewards for improving diversity, coherency, and ease of answering. Their approaches involves pre-training the encoder-decoder models with supervised training and then refining the utterance generation further with RL using the hand-engineered rewards. Their state space consists of the dialog context representation (encoder hidden states). Their action

space at a given time step includes all possible words that the decoder can generate (which is very large).

While this approach is appealing, policy gradient methods are known to suffer from high variance when using large action spaces. This makes training extremely unstable and requires significant engineering efforts to train successfully.

Another potential drawback with directly acting over the vocabulary space is that the RL optimization procedure tends to strip away the linguistic and natural language aspects learned during the supervised pre-training step, as observed in [71, 78]. Since the primary focus of the RL objective function is to improve the final reward (which may not emphasize on the linguistic aspects of the generated responses, for e.g., diversity scores), the optimization algorithm could lead the decoder into generating unnatural responses. We propose to avoid both the issues by reducing the action space to a higher level abstraction space i.e. the dialog attributes. Our action space comprises the discrete dialog attributes and the state space is the dialog context. Intuitively, this enables the RL policy to view the dialog attributes as control variables for improving dialog flow and modeling long term influence. For instance, if the input response was “*how old are you?*”, an RL policy optimized to maximize conversation length and engagement could choose to set one of the next utterance attributes as a question-type to generate a response like “*why do you ask?*” instead of a straightforward answer, to keep the conversation engaging. Thus, we believe that this approach enables the model to predict such rare but interesting utterances to which the MLE objective fails to give attention.

Our policy network comprises of the encoders and the attribute prediction network. Given the previous utterances $U_{1:m-1}$, the policy network first encodes them by using the encoders. Then this encoded representation is passed to the attribute prediction network. The output of the attribute prediction network is the action. While there are many ways to design the reward function, we adopt the *ease-of-answering* reward introduced by [82] - negative log-likelihood of a set of manually constructed dull utterances (usually the most commonly occurring phrases in the dataset) in response to the next generated utterance. Let \mathcal{S} be the set of dull utterances. With the sampled dialog-acts, $DA_{1:K}$ from the policy network, we generate the next utterance U_m using the decoder. Then we add this generated utterance to the context and predict the probability of seeing one of the dull utterances in

the $m + 1$ -th step. This is used to compute the reward as follows:

$$R = -\frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{1}{N_s} \log P(s | U_{1:m-1}, DA_{1:K}), \quad (11.3.4)$$

where N_s is the number of tokens in the dull utterance s . The normalization avoids the reward function attending to only the longer dull responses. We use REINFORCE [183] to optimize our policy, $P_{RL}(DA_{1:K} | U_{1:m-1})$. The expected reward is given by equation 11.3.5

$$J(\theta) = \mathbb{E}[R(U_{1:m-1}, DA_{1:K})] \quad (11.3.5)$$

The gradient is estimated as in equation 11.3.6

$$\nabla J(\theta_{RL}) = (R - b) \nabla \log P_{RL}(DA_{1:K} | U_{1:m-1}), \quad (11.3.6)$$

where b is the reward baseline (computed as the running average of the rewards during training). We initialize the policy with the supervised training and add an L2-loss to penalize the network weights from moving away from the supervised network weights.

11.4. Training Setup

Datasets: We first start with the Reddit-discourse dataset [187] for training dialog attribute classifiers and modeling utterance generation.

Reddit: The Reddit discourse dataset [187] is manually pre-annotated with dialog-acts via crowd sourcing. The dialog-acts comprise of *answer, question, humor, agreement, disagreement, appreciation, negative reaction, elaboration, announcement*. It comprises conversations from around 9000 randomly sampled Reddit threads with over 100000 comments and an average of 12 turns per thread.

Open-Subtitles: Additionally, we show results with the unannotated Open-Subtitles dataset [160] (we randomly sample up to 2 million dialogs for training and validation). We tag the dataset with dialog attributes using pre-trained classifiers.

We experiment with two types of dialog attributes in this paper - *sentiment and dialog-acts*. We annotate the utterances with sentiment tags - *positive, negative, neutral* using the Stanford Core-NLP tool [96]. We adopt the dialog-acts from two annotated dialog corpus - Switchboard [43] and Frames [133].

Switchboard: Switchboard corpus [43] is a collection of 1155 chit-chat style telephonic conversations based on 70 topics. [59] revised the original tags to 42 dialog-acts. In our

experiments, we restrict dialog-acts to the top-10 most frequently annotated tags in the corpus - *Statement-non-opinion*, *Acknowledge*, *Statement-opinion*, *Agree/Accept*, *Abandoned or Turn-Exit*, *Appreciation*, *Yes-No-Question*, *Non-verbal*, *Yes answers*, *Conventional-closing*. We consider the top-10 frequently annotated tags as a simple solution to avoid the class imbalance issue (the *Statement-non-opinion* act is tagged 72824 times, while *Thanking* is tagged only 67 times) for training the dialog attribute classifiers.

Frames: Frames [133] is a task oriented dialog corpus collected in the *Wizard-of-Oz* fashion. It comprises of 1369 human-human dialogues with an average of 15 turns per dialog. The wizards had access to a database of hotels and flights information and had to converse with users to help finalize vacation plans. The dataset has 20 different types of dialog-acts annotations. Like the Switchboard corpus, we adopt the top 10 frequently occurring acts in the dataset for our experiments - *inform*, *offer*, *request*, *suggest*, *switch-frame*, *no result*, *thank you*, *sorry*, *greeting*, *affirm*.

Model Details: We use two-layer GRUs [30] for both encoder and decoders with hidden sizes of 512. We restrict the vocabulary for both the datasets to top 25000 frequency occurring tokens. The dialog attribute classifier for dialog attributes is a simple 2-layer MLP with layer sizes of 256, and 10 respectively. We use the rectified linear unit (ReLU) as the non-linear activation function for the MLPs and use dropout rate of 0.3 for the token embeddings, hidden-hidden transition matrices of the encoder and decoder GRUs. For generation we use standard beam search (beam size 10).

Training Details: We ran our experiments on Nvidia Tesla-K80 GPUs and optimized using the ADAM optimizer with the default hyper-parameters used in [98, 99]. All models are trained with batch size 128 and a learning rate 0.0001.

11.5. Experimental Results

In this section, we present the experimental results along with qualitative analysis.

In Section 11.5.1, we discuss the dialog attribute classification results for different model architectures trained on the Reddit, Switchboard and Frames datasets.

In Section 11.5.2, we first demonstrate quantitative improvements (token perplexity/embedding based metrics) for the Attribute conditional HRED model with the manually annotated Reddit dataset. Further, we discuss the model perplexity improvements along

with sample conversations and human evaluation results on the Open-Subtitles dataset. We annotate it with sentiment and dialog-acts (from Switchboard/Frames datasets) using pre-trained classifiers described in Section 11.5.1.

Finally, in Section 11.5.3, we analyze the quality of the generated responses after RL fine-tuning using *diversity* scores (*distinct-1*, *distinct-2*), sample conversations and human evaluation results for diversity and relevance. The diversity scores, (*distinct-1*, *distinct-2*) are computed as the fraction of uni-grams and bi-grams in the generated responses following the previous work by [74].

11.5.1. Dialog Attribute Prediction

In this section, we present the experiments with the model architectures for the dialog attribute prediction - dialog-acts from Reddit, Switchboard and Frames datasets. First, we demonstrate the performance of the dialog-acts classifiers on the Reddit dataset as shown in Table 11.1.

Model	Acc(%)
$F(U_t)$	57
$F(DA_{t-1,t-2})$	54
$F(U_t, DA_{t-1,t-2})$	68

Tab. 11.1. Dialog-acts prediction accuracy in Reddit validation set.

The model $F(U_t)$ refers to the architecture which predicts the dialog-acts based on current utterance U_t alone. The tokens in the current utterance U_t are fed through a two-layer GRU and the final hidden state is used to predict the dialog-acts. The model $F(DA_{t-1,t-2})$ predicts the current utterance’s dialog-acts DA_t based on the dialog-acts corresponding to the previous two utterances. We consider the dialog-acts prediction problem as a sequence modeling problem where we feed the dialog-acts into a single-layer GRU and predict the current dialog-acts conditioned on the previous dialog-acts. We settled on conditioning on the dialog-acts corresponding to the previous two utterances alone as we didn’t observe any boost in the classifier performance from the older dialog-acts. As seen in Table 11.1, conditioning additionally on the dialog attributes helps improve classifier performance.

Next, we train classifiers to predict dialog-acts of utterances of the Switchboard and Frames corpus. In our experiments, the number of act types is 11 - the top 10 most frequently occurring acts in the corpus and "others" category covering the rest of the tags.

Corpus	Num Acts	Acc(%)
Reddit	9	68.1
Switchboard	11	67.9
Frames	11	71.1

Tab. 11.2. Dialog-acts prediction accuracy for classifiers trained on validation set of different datasets.

As seen from Table 11.2, classifier performance is not really high and yet, contribute to improvements in perplexity for the conditional Seq2Seq models (discussed in Section 11.5.2). While we aim for better classifier performance, it is important to note here that the primary objective of such dialog attribute classifiers is to tag unannotated open-domain dialog datasets. As future work, we will study how the classification errors influence response generation.

11.5.2. Utterance Evaluation

Following [141], we use token perplexity and embedding based metrics (average, greedy and extrema) [107, 131] for utterance evaluation.

Metric	LM	Seq2Seq	Seq2Seq+Attr
Perplexity	176	170	163
Greedy	-	0.47	0.54
Extrema	-	0.37	0.47
Average	-	0.67	0.62

Tab. 11.3. Perplexity and Embedding Metrics for the Reddit validation set.

Reddit: First, we evaluate Seq2Seq models trained on the manually annotated Reddit corpus as shown in Table 11.3. *Seq2Seq+Attr* refers to our model where we condition on the dialog-acts additionally. Please note that we use the notation "*Attr*" here to maintain

		Num Dialogs(in Millions)			
Model	Attributes	0.2 M	0.5 M	1 M	2 M
Seq2seq	-	101.63	80.05	74.78	67.28
Seq2seq	Sentiment	98.61	79.15	72.23	66.11
Seq2seq	Switchboard	97.03	77.81	71.51	64.21
Seq2seq	Frames	96.61	77.41	72.01	65.33
Seq2seq	Sentiment, Switchboard	96.67	78.01	72.17	66.01
Seq2seq	Sentiment, Frames	96.32	77.61	72.15	66.13
Seq2seq	Switchboard, Frames	94.80	77.40	71.18	65.01

Tab. 11.4. Validation Perplexity for the Open-Subtitles dataset.

generality as it may refer to other dialog attributes like sentiment later in this section. For both the baseline and conditional Seq2Seq models, we consider a dialog context involving the previous two turns as we did not observe significant performance improvement with three or more turns. We use a 2-layer GRU language model as a baseline for comparison. As seen from Table 11.3, *Seq2Seq+Attr* fares well both in terms of perplexity and embedding metrics. Higher perplexity observed in the Reddit corpus could be due to the presence of several topics in the dataset (exhibits high entropy) and fewer dialogs compared to other open domain dialog datasets.

Open-Subtitles: With promising results on the manually tagged Reddit corpus, we now evaluate our attribute conditional HRED model on the unannotated Open-Subtitles dataset. We tag the Open-Subtitles dataset with the sentiment tags using the Stanford Core-NLP tool [96] and dialog-acts from Frames & Switchboard corpus using the pre-trained classifiers described in Section 11.5.1.

In Table 11.4, we compare the model perplexity when trained on varying dialog corpus size. In most of the cases, we observe that the conditioning with acts from both the frames and switchboard yields the lowest perplexity. We observe that the perplexity improvement is substantial for smaller datasets which is also corroborated from the experiments with the Reddit dataset.

Human Evaluation: Following the human evaluation setting in [82], we randomly sample 200 input message and the generated outputs from the *Seq2Seq+Attr* & *Seq2Seq* models. We present each of them to 3 judges and ask them to decide which of the two outputs is 1) most relevant and 2) diverse or interesting. Ties are permitted. Results for human evaluation are shown in Table 11.8. We observe that *Seq2Seq+Attr* performs better than the *Seq2Seq* model both in terms of *diversity* and *relevance*.

Seq2Seq+Attr vs Seq2Seq			
Metric	Wins(%)	Losses(%)	Ties(%)
<i>Diversity</i>	42	24.16	33.84
<i>Relevance</i>	40.16	36.83	23.01

Tab. 11.5. Human Evaluation results: *Seq2Seq+Attr* vs *Seq2Seq*

Please note that the *Seq2Seq+Attr* model performs better in terms of diversity compared to the relevancy. This is in line with our expectations, as the purpose of dialog attribute annotations is to help the model focus better on less-frequent responses.

Additionally, we present a few sample conversations in Table 11.6, where we observe that the *Seq2Seq+Attr* model generates more interesting responses.

11.5.3. RL For Dialog Attribute Prediction

For the RL fine-tuning, we report the diversity scores of the generated responses with the models trained on the Open-Subtitles dataset in Table 11.7. The diversity scores, *distinct-1* and *distinct-2* are computed as the fraction of uni-grams and bi-grams in the generated responses following the previous work by [79].

We use the model conditioned on acts from both Switchboard and Frames for the *Seq2Seq+Attr* and *RL* cases. The action space for the policy in this case, covers the 10 acts from Switchboard and Frames each. We choose a collection of commonly occurring phrases in the Open-Subtitles dataset as the set of dull responses, \mathbb{S} for the reward computation in equation 11.3.4. We observe that the RL fine-tuning improves over the conditional seq2seq in terms of the diversity scores.

Input:	i wish i was home watching tv.
Seq2Seq:	i dont know what i was thinking about
Seq2Seq+Attr:	i cant wait to see it.
Input:	He used from his charity to settle legal problems.
Seq2Seq:	i have no idea what youre talking about
Seq2Seq+Attr:	i dont think he is going to be a president.
Input:	tell us how you really feel
Seq2Seq:	i dont understand why
Seq2Seq+Attr:	lmao i could hella picture your reaction

Tab. 11.6. Sample conversations

Model	<i>distinct-1</i>	<i>distinct-2</i>
Seq2Seq	0.004	0.013
Seq2Seq+Attr	0.005	0.018
RL	0.011	0.033

Tab. 11.7. Diversity scores on the Open-Subtitles validation set after RL fine-tuning .

Human Evaluation: As described in Section 11.5.2, we present each of the 200 randomly sampled input-response pairs of the *Seq2Seq + Attr* and *RL* models to 3 judges and ask to them rate each sample for *diversity* and *relevance*. From Table 11.8, we can see that the *RL* model significantly performs better both in terms of *diversity* and *relevance*.

Qualitative Analysis: In Table 11.9, we present the percentage of the commonly occurring generic responses from the Open-Subtitles dataset in the validation set samples corresponding to the *RL* and *Seq2Seq + Attr* models. We observe very low percentages of such generic responses in the samples after RL fine-tuning. It is interesting to note that RL

RL vs Seq2Seq+Attr			
Metric	Wins(%)	Losses(%)	Ties(%)
<i>Diversity</i>	54.66	28.50	16.84
<i>Relevance</i>	43.33	26.62	30.05

Tab. 11.8. Human Evaluation results:*RL vs Seq2Seq+Attr*

model has successfully learned to minimize the generation of other dull responses like *I would love to be* , *I would love to see*, *I dont want to* apart from expected the dull responses, \mathbb{S} (used in the reward computation). At the same time, RL model has scored higher in terms of the *Relevancy* metric, as seen in Table 11.8 which indicates that the RL fine-tuning actually explores interesting responses whilst avoiding the generic responses.

Generic Responses	RL(%)	Seq2Seq + Attr(%)
thank you so much	7.56	7.32
i dont understand why	0.0	15.64
i would love to see	0.66	5.65
i dont know how	0.0	13.97
i dont want to	1.66	3.99
i dont know why	0.0	3.66
i would love to be	0.99	2.21
i have no idea	4.31	3.33

Tab. 11.9. Percentage of generic responses after RL fine-tuning.

Additionally, we present a few sample conversations in Table 11.10, where we observe that the *RL* model generates more diverse and relevant responses.

11.6. Related Work

There are several works focusing on dialog-acts classification and clustering based analysis [125, 90, 67, 5, 34, 153, 39]. [144] additionally add sentiment feature to the latent variables in the VAE setting for utterance generation. In our work, we use dialog attributes from different sources - Switchboard and Frames corpus to model utterance generation in a

Input:	i'm honestly a bit confused why no one has brought me or my books any cake
Seq2Seq+Attr:	i dont think i dont think anything
RL:	i cant wait to see you in the city.
Input:	ive been playing spaceship with my year old niece for the past few days
Seq2Seq+Attr:	i dont even know what i was talking about.
RL:	this is the best thing ive ever seen.
Input:	it makes me so happy that you like them
Seq2Seq+Attr:	i dont know what i was thinking about it
RL:	i was just thinking about the same thing

Tab. 11.10. Sample conversations

more realistic setting. As for the RL setting, existing research efforts include [82, 37, 57] which formulate the token prediction as a RL policy in Seq2Seq models. However, searching over a huge vocabulary space typically involves training with huge number of samples and careful fine-tuning of the policy optimization algorithms. Additionally, as discussed in Section 11.3.3, it requires precautionary measures to prevent the RL algorithm from removing the linguistic aspects of the generated utterances. In another related research work, [135] use dialog-acts as one among their hand crafted features to select responses from an ensemble of dialog systems. They use dialog-acts in their RL policy, however their action space comprises of responses from an ensemble of dialog models. They include dialog-acts in their features for their distributed state representation.

11.7. Conclusion

In this work, we address the dialog utterance generation problem by jointly modeling previous dialog context and discrete dialog attributes. We analyze both quantitatively (model perplexity and other embedding based metrics) and qualitatively (human evaluation, sample conversations) to validate that *composed* dialog attributes help generate interesting responses. Further, we formulate the dialog attribute prediction problem as a reinforcement learning problem. We fine tune the attribute selection policy network trained with supervised learning using REINFORCE and demonstrate improvements in diversity scores compared to the Seq2Seq model. In the future, we plan to extend the model for additional dialog attributes like emotion, speaker persona etc. and evaluate the controllability aspect of the responses based on the dialog attributes.

Chapter 12

Prologue to Fourth Article

12.1. Article Details

Transferable Neural Projection Representations. Chinnadhurai Sankar, Sujith Ravi, Zornitsa Kozareva *2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019)*

Personal Contribution. After discussions with Sujith Ravi and Zornitsa Kozareva, I came up with the idea for training on-device neural text representations along the lines of the skip-gram model [104]. I implemented the prototype of the model and iterated on the prototype to significantly improve its performance. Zornitsa and I wrote up the paper, while Sujith helped improve its presentation.

12.2. Context

The tremendous success of deep learning models and the explosion of mobile, IoT devices coupled together with the growing user privacy concerns have led to the need for deploying deep learning models on-device for inference. This has led to new research in compressing large and complex deep learning models for low power on-device deployment. Recently, [124] developed an on-device neural text classification model. They proposed to reduce the memory footprint of large neural networks by replacing the input word embeddings with Locality Sensitive Hashing (LSH) [24] projection-based representations. However, the projections in [124] are static and currently do not leverage pre-training on large unsupervised corpora, which is an important property to make the projections transferable to new tasks.

In this paper, we propose to combine the best of both worlds by learning transferable neural projection representations over randomized LSH projections.

12.3. Contributions

We introduce a new neural architecture inspired by the skip gram model of [104] and combined with a deep MLP plugged on top of LSH projections. In order to make this model train better, we introduce a novel regularizing loss function critical for generalization. We conduct a qualitative analysis of the nearest neighbours in the learned representation spaces and a quantitative evaluation via similarity, language modeling and NLP tasks.

12.4. Recent Developments

Efficient neural representations for on-device models have been explored extensively since the publication of the paper. For instance, [62] train tiny neural networks just 200 Kilobytes in size that improve over prior CNN and LSTM models and achieve near state of the art performance on multiple long document classification tasks. They also explore transfer learning capabilities to further improve the performance in limited data scenarios. In another work [72], authors introduce novel on-device sequence model for text classification using recurrent LSH-based projection representations.

Chapter 13

Transferable Neural Projection Representations

13.1. Abstract

Neural word representations are at the core of many state-of-the-art natural language processing models. A widely used approach is to pre-train, store and look up word or character embedding matrices. While useful, such representations occupy huge memory making it hard to deploy on-device and often do not generalize to unknown words due to vocabulary pruning.

In this paper, we propose a skip-gram based architecture coupled with Locality-Sensitive Hashing (LSH) projections to learn efficient dynamically computable representations. Our model does not need to store lookup tables as representations are computed on-the-fly and require low memory footprint. The representations can be trained in an unsupervised fashion and can be easily transferred to other NLP tasks. For qualitative evaluation, we analyze the nearest neighbors of the word representations and discover semantically similar words even with misspellings. For quantitative evaluation, we plug our transferable projections into a simple LSTM and run it on multiple NLP tasks and show how our transferable projections achieve better performance compared to prior work.

13.2. Introduction

Pre-trained word representations are at the core of many neural language understanding models. Among the most popular and widely used word embeddings are word2vec [104], GloVe [112] and ELMO [114]. The biggest challenge with word embedding is that they

require lookup and a large memory footprint, as we have to store one entry (d -dim vector) per word and it blows up.

In parallel, the tremendous success of deep learning models and the explosion of mobile, IoT devices coupled together with the growing user privacy concerns have led to the need for deploying deep learning models on-device for inference. This has led to new research in compressing large and complex deep learning models for low power on-device deployment. Recently, [124] developed an on-device neural text classification model. They proposed to reduce the memory footprint of large neural networks by replacing the input word embeddings with projection-based representations. [124] used n -gram features to generate binary LSH [24] randomized projections on the fly surpassing the need to store word embedding tables and reducing the memory size. The projection models reduce the memory occupied by the model from $O(|V|)$ to $O(n_{\mathbb{P}})$, where $|V|$ refers to the vocabulary size and $n_{\mathbb{P}}$ refers to number of projection operations [122]. Two key advantages of the projection-based representations over word embeddings are: (1) they are fixed and have low memory size; (2) they can handle out of vocabulary words. However, the projections in [124] are static and currently do not leverage pre-training on large unsupervised corpora, which is an important property to make the projections transferable to new tasks.

In this paper, we propose to combine the best of both worlds by learning transferable neural projection representations over randomized LSH projections. We do this by introducing new neural architecture inspired by the skip gram model of [104] and combined with a deep MLP plugged on top of LSH projections. In order to make this model train better, we introduce new regularizing loss function, which minimizes the cosine similarities of the words within a mini-batch. The loss function is critical for generalization.

In summary, our model (1) requires a fixed and low memory footprint, (2) can handle out of vocabulary words and misspellings, (3) captures semantic and syntactic properties of words; (4) can be easily plugged to other NLP models and (5) can support training with data augmentation by perturbing characters of input words. To validate the performance of our approach, we conduct a qualitative analysis of the nearest neighbours in the learned representation spaces and a quantitative evaluation via similarity, language modeling and NLP tasks.

13.3. Neural Projection Model

We propose a novel model (NP-SG) to learn compact neural representations that combines the benefit of representation learning approaches like skip-gram model with efficient LSH projections that can be computed on-the-fly.

13.3.1. Vanilla Skip-Gram Model

In the skip-gram model [104], we learn continuous distributed representations for words in a large fixed vocabulary, \mathbb{V} to predict the context words surrounding them in documents. We maintain an embedding look up table, $v(w) \in \mathbb{R}^d$ for every word, $w \in \mathbb{V}$.

For each word, w_t in the training corpus of size T , the set of context words $\mathbb{C}_t = \{w_{t-W_t}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+W_t}\}$ includes W_t words to the left and right of w_t respectively. W_t is the window size randomly sampled from the set $\{1, 2, \dots, N\}$, where N is the maximum window size. Given a pair of words, $\{w_c, w_t\}$, the probability of w_c being within the context window of w_t is given by equation 13.3.1.

$$\begin{aligned} P(w_c|w_t) &= \sigma(v'(w_c)^\top v(w_t)) \\ &= \frac{1}{1 + \exp(-v'(w_c)^\top v(w_t))} \end{aligned} \tag{13.3.1}$$

where v, v' are input and context embedding look up tables.

13.3.2. Neural Projection Skip-Gram (NP-SG)

In the neural projection approach, we replace the input embedding look up table, $v(w)$ in equation 13.3.1 with a deep n -layer MLP over the binary projection, $\mathbb{P}(w)$ as shown equation 13.3.2.

$$v_{\mathbb{P}}(w) = \mathbb{N}(f_n(\mathbb{P}(w))) \tag{13.3.2}$$

where $v_{\mathbb{P}}(w) \in \mathbb{R}^d$, f_n is a n -layer deep neural network encoder with *ReLU* non-linear activations after each layer except for the last layer as shown in Figure 13.1. \mathbb{N} refers to a normalization applied to the final layer of f_n . We experimented with Batch-normalization, L2-normalization and layer normalization; batch-normalization works the best.

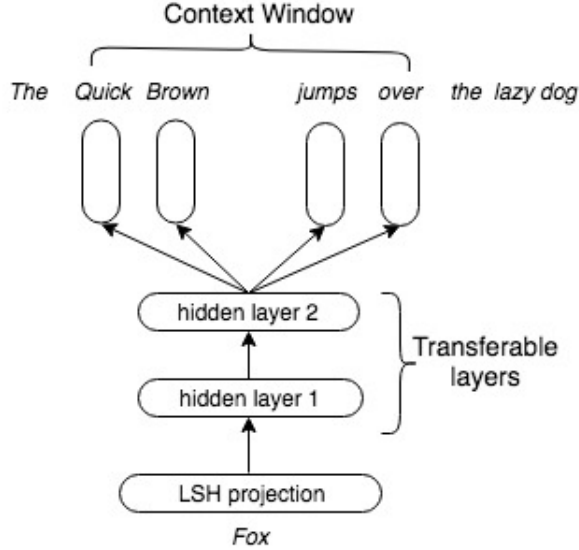


Fig. 13.1. Neural Projection Skip-gram (NP-SG) model

The binary projection $\mathbb{P}(w)$ is computed using locality-sensitive projection operations [122] which can be performed on-the-fly (i.e., without any embedding look up) to yield a fixed, low-memory footprint binary vector. Unlike [124] which uses *static* projections to encode the entire input text and learn a classifier, NP-SG creates a *trainable* deep projection representation for words using LSH projections over character-level features combined with contextual information learned via the skip-gram architecture.

13.3.3. Training NP-SG Model

We follow a similar approach as [104] and others for training our neural projection skip-gram model (NP-SG). We define the training objective to maximize the probability of predicting the context words given the current word. Formally, the model tries to learn the word embeddings by maximizing the objective, $J(\theta)$ known as negative sampling (NEG), given by equation 13.3.3.

$$J(\theta) = \sum_{t=1}^T \sum_{w_c \in C_t} J_{w_t, w_c}(\theta) \quad (13.3.3)$$

$$J_{w_t, w_c}(\theta) = \log(\mathbb{P}(w_c | w_t)) + \sum_{i=1, w_i \sim P_n(w)}^k \log(1 - \mathbb{P}(w_i | w_t)) \quad (13.3.4)$$

where k is the number of randomly sampled words from the training corpus. Following [105], we sample words according to the noise distribution $P_n(w) \propto U(w)^{3/4}$, where $U(w)$ is the unigram distribution of the training corpus.

Model improvements: Training an NP-SG model as is, though efficient, may not lead to highly discriminative representations. During training, we noticed that the word representations, $v_{\mathbb{P}}(w)$ were getting projected in a narrow sub-space where the cosine similarities of all the words in the dataset were too close to 1.0. This made the convergence slower and led to poor generalization.

13.3.4. Discriminative NP-SG Models

To encourage the word representations to be more spaced out in terms of the cosine similarities, we introduce an additional explicit regularizing L2-loss function. With the assumption that the words in each mini-batch are randomly sampled, we add a L2-loss over the cosine similarities between all the words within a mini-batch, as shown in equation 13.3.6.

$$Loss = -J(\theta) + L_2^{cs}(\mathbf{w}_{mb}) \quad (13.3.5)$$

$$L_2^{cs}(\mathbf{w}_{mb}) = \lambda \cdot \text{sqrt}\left(\sum_{0 < i < j \leq mb} CS(w_i, w_j)^2\right) \quad (13.3.6)$$

where $CS(w_i, w_j)$ refers to the cosine similarity between w_i and w_j , mb refers to the mini-batch size and \mathbf{w}_{mb} refers to the words in the mini-batch. We enforce this using a simple outerproduct trick. We extract the cosine-similarities between all the words within a mini-batch in a single shot by computing the outer-product of the L_2 row normalized word representations corresponding to each minibatch $\hat{v}_{\mathbb{P}}(\mathbf{w}_{mb})$, as shown in equation 13.3.7.

$$L_2^{cs}(\mathbf{w}_{mb}) = \frac{\lambda}{2} \cdot \|\text{Flatten}(\hat{v}_{\mathbb{P}}(\mathbf{w}_{mb}) \cdot \hat{v}_{\mathbb{P}}(\mathbf{w}_{mb})^{\top})\|_2^2 \quad (13.3.7)$$

13.3.5. Improved NP-SG Training

Since the NP-SG model does not have a fixed vocabulary size, we can be flexible and leverage a lot more information during training compared to standard skip-gram models which require vocabulary pruning for feasibility.

To improve training for NP-SG model, we augment the dataset with input words after applying character level perturbations to them. The perturbations are such that they are commonly occurring misspellings in documents. We mainly experiment with three types of perturbation operation APIs [41].

- *insert(word, n)* : We randomly choose n chars from the character vocabulary and insert them randomly into the input *word*. We ignore the locations of first and last character in the word for the *insert* operation. Example transformation: *sample* \rightarrow *samnple*.
- *swap(word, n)* : We randomly swap the location of two characters in the word n times. As with the *insert* operation, we ignore the first and last character in the word for the *swap* operation. Example transformation: *sample* \rightarrow *sapmle*.
- *duplicate(word, n)* : We randomly duplicate a character in the word by n times. Example transformation: *sample* \rightarrow *saample*.

We would like to note that the perturbation operations listed above are not exhaustive and we plan to experiment with more operations in the future.

13.4. Training Setup

13.4.1. Dataset

We train our skipgram models on the wikipedia data XML dump, *enwik9*. We extract the normalized English text from the XML dump using the Matt Mahoney’s pre-processing perl script.

We fix the vocabulary to the top $100k$ frequently occurring words. We sub-sample words in the training corpus, dropping them with probability, $P(w) = 1 - \sqrt{t/freq(w)}$, where $freq(w)$ is the frequency of occurrence of w in the corpus and we set the threshold, t to 10^{-5} . We perturb the input words with a probability of 0.4 using a randomly chosen perturbation described in Section 13.3.5.

13.4.2. Implementation Details

We fix the number of random projections to 80 and the projection dimension to 14. We use a 2-layer MLP (sizes: [2048, 100]) regularized with dropout (with probability of

<i>Dataset</i>	<i>SG (10M)</i>	<i>NP-SG (w/oOP)</i>	<i>NP-SG (1M)</i>	<i>NP-SG (2M)</i>	<i>NP-SG (4M)</i>
EN-MTurk-287	0.5409	0.0107	0.5629	0.5517	0.5494
EN-WS-353-ALL	0.5930	0.0710	0.4891	0.5215	0.5370
EN-WS-353-REL	0.5359	0.0203	0.4956	0.5746	0.5671
EN-WS-353-SIM	0.6242	0.1043	0.4994	0.5116	0.5111
EN-RW-STANFORD	0.1505	0.0401	0.0184	0.0375	0.0835
EN-VERB-143	0.2452	0.0730	0.1333	0.1500	0.2108

Tab. 13.1. Similarity Tasks: # of params, 100k vocabulary size for skipgram baseline, 100 embedding size.

0.65) and weight decay (regularization parameter of 0.0005) to transform the binary random projections to continuous word representation. For the vanilla skipgram model, we fix the embedding size to 100. For both models, we use 25 negative samples for the NEG loss. We learn the parameters using the Adam optimizer [69] with a default learning rate of 0.001, clipping the gradients which have a norm larger than 5.0. We initialize the weights of the MLP using Xavier initialization, and output embeddings uniformly random in the range $[-1.0, 1.0]$. We use a batch size of 1024 in all our experiments. We found that $\lambda = 0.01$ for the outerproduct loss to be working better after experimenting with other values. Training time for our model was around 0.85 times that of the skipgram model. Both the models were trained for 10 epochs.

13.5. Experiments

We show both qualitative and quantitative evaluation on multiple tasks for the NP-SG model.

13.5.1. Qualitative Evaluation and Results

Table 13.2 shows the nearest neighbors produced by NP-SG for select words. Independent of whether it is an original or misspelled word, our NP-SG model accurately retrieves relevant and semantically similar words.

<i>Word</i>	<i>Nearest neighbours</i>
king	reign, throne, kings, knights, vii, regent
kingg	vii, younger, peerage, iv, tiberius, frederick
woman	man, young, girl, child, girls, women
wwoamn	man, herself, men, couple, herself, alive
city	town, village, borough, township, county
ciity	town, village, borough, county, unorganized
time	few, times, once, entire, prominence, since
tinme	times, once, takes, taken, another, only
zero	two, three, seven, one, eight, four
zzero	two, three, five, six, seven, four

Tab. 13.2. Sampled nearest neighbors for NP-SG.

13.5.2. Quantitative Evaluation and Results

We evaluate our NP-SG model on similarity, language modeling and text classification tasks. Similarity tests the ability to capture words, while language modeling and classification warrant the ability to transfer the neural projections.

13.5.2.1. Similarity Task

We evaluate our NP-SG word representations on 4 different widely used benchmark datasets for measuring similarities.

Dataset:

MTurk-287 [119] has 287 pairs of words and was constructed by crowdsourcing the human similarity ratings using Amazon Mechanical Turk. *WS353* [40] has 353 pairs of similar English words rated by humans and is further split into *WS353-SIM*. *WS353-REL* [3] captures different types of similarities and relatedness. *RW-STANFORD* [95] has 2034 rare word pairs sampled from different frequency bins.

Evaluation:

For all the datasets, we compute the Spearman’s rank correlation coefficient between the rankings computed by skip-gram models (baseline SG and NP-SG) and the human rankings. We use the cosine similarity metric to measure word similarity.

Results:

Table 13.1 shows that NP-SG, with a significantly smaller number of parameters comes close to the skip-gram model (SG) and even outperforms it with 2.5x-10x compression. NP-SG gets better representations even with misspellings which cannot be handled by vanilla SG.

It is interesting to note that the vanilla skip-gram model does well on *WS353-SIM* compared to *WS353-REL*. This behavior is reversed in our NP-SG model, which indicates that it captures meronym-holonym relationships better than the vanilla skip-gram model. Although NP-SG handles out of vocabulary words in the form of misspellings, it needs further improvement for the rare word similarity task. We plan to improve it by including context word n-gram features in the LSH projection function, allowing NP-SG to also leverage information from the context words in the case of rare words and provide word sense disambiguation.

13.5.2.2. Language Modeling

We applied NP-SG to language modeling task on the Penn Treebank (PTB)[159] dataset. We consider a single layer LSTM with hidden size of 2048 for the language model task. With the input embedding size of 200, we observed a perplexity of ≈ 120 on the test set after training for 5 epochs. We replace the input embeddings in the LSTM with transferable encoder layer of the NP-SG model. We train the LSTMs with and without pretrained initializations. Since we observed convergence issues with the single layer NP-SG representation, we considered 2-layer MLP with layer sizes (1024, 256) for the NP-SG representations. We found that while the model without pretrained NP-SG layer got stuck at a perplexity of around 300, the pretrained version converged to a perplexity of 140, comparable to the embedding based network. We leave the analysis of the impact of the deeper NP-SG layers to the future work.

13.5.2.3. *Text Classification*

For the text classification evaluations, we used two different tasks and datasets. For the dialog act classification task, we used the MRDA dataset from the ICSI Meeting Recorder Dialog Act Corpus [1]. MRDA is a multiparty dialog annotated with 5 dialog act tags. For the question classification task, we used the TREC dataset [85]. The task is given a question to predict the most relevant category.

We trained a single layer LSTM (hidden size: 256) with and without the pretrained NP-SG layers. Overall, we observed accuracy improvements of +5.7% and +3.75% compared to baseline models without pretrained NP-SG initializations on TREC and MRDA respectively.

13.6. Conclusion

In this paper, we introduced a new neural architecture (NP-SG), which learns transferable word representations that can be efficiently and dynamically computed on-device without any embedding look up. We proposed an unsupervised method to train the new architecture and learn more discriminative word representations. We compared the new model with a skip-gram approach and showed qualitative and quantitative comparisons on multiple language tasks. The evaluations show that our NP-SG model learns better representations even with misspellings and reaches competitive results with skip-gram on similarity tasks, even outperforming with 2.5x-10x fewer parameters.

Chapter 14

Conclusion

This thesis has touched various topics around neural approaches for modeling dialog.

The work on perturbation-based analysis evaluates how sensitive different neural generative dialog models are to perturbations in dialog history, where sensitivity is measured as a change in perplexity. It considers a range of utterance-level and sentence-level perturbations, three model variants, and 4 different datasets. The overall finding is that all models are surprisingly insensitive to dialog history. By open-sourcing our code, we believe this paradigm of studying model behavior by introducing perturbations that destroys different kinds of structure present within the dialog history can be a useful diagnostic tool. We also foresee this paradigm being useful when building new dialog datasets to understand the kinds of information models use to solve them. This work calls for more analysis and in-depth studies of how dialog context is modeled in current neural architectures. We believe that this work could potentially inspire more analysis-based research efforts to understand why neural conversational models generate generic responses and possibly lead to quantitative techniques to study the shortcomings of current state of the art neural conversational models. Follow up works could focus on the impact of some of the following factors over the sensitivity of neural dialog models

- Neural architectures for dialog context encoding,
- Dialog training loss functions & metrics,
- Nature of imbalances present in the dialog dataset,

To address the lack of quality corpora for data-driven dialog system research and development, the second work introduces Taskmaster-1, a dataset that provides richer and

more diverse language as compared to current benchmarks since it is on unrestricted, task-oriented conversations involving more real-world entities. In addition, we present two data collection methodologies, both spoken and written, that ensure both speaker diversity and conversational accuracy. Our straightforward, API-oriented annotation technique is much easier for annotators to learn and simpler to apply. We give several baseline models including state-of-the-art neural seq2seq architectures, provide qualitative human performance evaluations for these models, and find that automatic evaluation metrics correlate well with human judgments. While there have been other larger dataset releases following our work, they have largely been grounded in dialog intent types where as our framework involves grounding conversations in argument types. We believe that grounding the conversations in argument types over intents is a scalable solution to approaching modeling task oriented dialogs as the number of possible intents could be exponentially large compared to the number of argument types. As a future work, we wish to collect datasets that consist of a training set with conversations involving only certain combinations of argument types and benchmark generalization capabilities of neural models to unseen combinations of argument types.

The third work describes a learning procedure for training chitchat seq2seq-based dialog agents that are conditioned on dialogue context and discrete dialogue attributes. We analyze both quantitatively (model perplexity and other embedding-based metrics) and qualitatively (human evaluation, sample conversations) to validate that *composed* dialog attributes help generate interesting responses. Further, we formulate the dialog attribute prediction problem as a reinforcement learning problem. We fine-tune the attribute selection policy network trained with supervised learning using REINFORCE and demonstrate improvements in diversity scores compared to the Seq2Seq model. In the future, we plan to extend the model for additional dialog attributes like emotion, speaker persona etc. and evaluate the controllability aspect of the responses on the dialog attributes.

The final work aims at producing learned word representations on-the-fly, without having to use gigabytes of memory for a lookup table. Instead, a locality-sensitive hashing (LSH) projection on the characters level features of a word is transformed using a MLP to create

a trainable representation of that word. The main aim of this approach is to reduce the large memory overhead required for storing conventional embedding look-up matrices. We compared the new model with a skip-gram approach and showed qualitative and quantitative comparisons on multiple language tasks. The evaluations show that our model learns better representations even with misspellings and reaches competitive results with skip-gram on similarity tasks, even outperforming with 2.5x-10x fewer parameters. In the immediate future, we wish to extend this work by including the the morphological features of the context words additionally to generate a context dependent on-device representation of words. As for the future research directions, LSH-based text representation could potentially be used to succinctly represent dialog utterances which are usually short (5 to 9 words on an average). We believe that LSH-based projection representations could lead to deploying conversational models directly on edge and IoT devices.

Bibliography

- [1] Janin Adam, Don Baron, Jane Edwards, Dan Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, and Chuck Wooters. The icsi meeting corpus. In *Proceedings of the 5TH SIGdial Workshop on Discourse and Dialogue*, pages 364–367, 2003.
- [2] Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*, 2016.
- [3] Eneko Agirre, Enrique Alfonseca, Keith B. Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, May 31 - June 5, 2009, Boulder, Colorado, USA*, pages 19–27, 2009.
- [4] Ankesh Anand, Eugene Belilovsky, Kyle Kastner, Hugo Larochelle, and Aaron Courville. Blindfold baselines for embodied qa. *arXiv preprint arXiv:1811.05013*, 2018.
- [5] Jeremy Ang, Yang Liu, and Elizabeth Shriberg. Automatic dialog act segmentation and classification in multiparty meetings. In *ICASSP (1)*, pages 1061–1064, 2005.
- [6] Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019, 2019.
- [7] H. Aust, M. Oerder, F. Seide, and V. Steinbiss. The philips automatic train timetable information system. *Speech Communication*, 17(3):249–262, 1995.
- [8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings Of The International Conference on Representation Learning (ICLR 2015)*, 2015.
- [9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.
- [10] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. End-to-end attention-based large vocabulary speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, pages 4945–4949, 2016.

- [11] Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*, 2017.
- [12] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.
- [13] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003.
- [14] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166, 1994.
- [15] Stuart Bird, Sue Browning, Roger Moore, and Martin Russell. Dialogue move recognition using topic spotting techniques. In *Spoken Dialogue Systems-Theories and Applications*, 1995.
- [16] Dan Bohus, Antoine Raux, Thomas K Harris, Maxine Eskenazi, and Alexander I Rudnicky. Olympus: an open-source framework for conversational spoken language interface research. In *Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies*, pages 32–39. Association for Computational Linguistics, 2007.
- [17] Dan Bohus and Alexander I. Rudnicky. Ravenclaw: dialog management using hierarchical task decomposition and an expectation agenda. In *8th European Conference on Speech Communication and Technology, EUROSPEECH 2003 - INTERSPEECH 2003, Geneva, Switzerland, September 1-4, 2003*, 2003.
- [18] Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. *ICLR*, 2017.
- [19] Antoine Bordes and Jason Weston. Learning end-to-end goal-oriented dialog. *CoRR*, abs/1605.07683, 2016.
- [20] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *EMNLP*, 2018.
- [21] Ronald Carter and Michael McCarthy. *Cambridge grammar of English: a comprehensive guide; spoken and written English grammar and usage*. Ernst Klett Sprachen, 2006.
- [22] Wallace Chafe and Deborah Tannen. The relation between written and spoken language. *Annual Review of Anthropology*, 1987.
- [23] Sarath Chandar, Chinnadhurai Sankar, Eugene Vorontsov, Samira Ebrahimi Kahou, and Yoshua Bengio. Towards non-saturating recurrent units for modelling long-term dependencies. *CoRR*, abs/1902.06704, 2019.
- [24] Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thirty-fourth Annual ACM Symposium on Theory of Computing, STOC '02*, pages 380–388, New York, NY, USA, 2002. ACM.

- [25] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. A survey on dialogue systems. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35, Nov 2017.
- [26] Ting-Rui Chiang, Hao-Tong Ye, and Yun-Nung Chen. An empirical study of content understanding in conversational question answering, 2019.
- [27] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *ArXiv e-prints*, June 2014.
- [28] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.
- [29] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [30] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *ArXiv e-prints*, December 2014.
- [31] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [32] Kenneth Mark Colby. Modeling a paranoid mind. *Behavioral and Brain Sciences*, 4(4):515–534, 1981.
- [33] Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*, 2018.
- [34] Nigel Crook, Ramón Granell, and Stephen G. Pulman. Unsupervised classification of dialogue acts using a dirichlet process mixture model. In *Proceedings of the SIGDIAL 2009 Conference, The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 11-12 September 2009, London, UK*, pages 341–348, 2009.
- [35] Kevin Crowston. Amazon mechanical turk: A research tool for organizations and information systems scholars. In Anol Bhattacharjee and Brian Fitzgerald, editors, *Shaping the Future of ICT Research. Methods and Approaches*, pages 210–221, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- [36] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.
- [37] B. Dhingra, L. Li, X. Li, J. Gao, Y.-N. Chen, F. Ahmed, and L. Deng. Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. *ArXiv e-prints*, September 2016.

- [38] Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. The second conversational intelligence challenge (convai2). In Sergio Escalera and Ralf Herbrich, editors, *The NeurIPS '18 Competition*, pages 187–208, Cham, 2020. Springer International Publishing.
- [39] Aysu Ezen-Can and Kristy Elizabeth Boyer. Unsupervised classification of student dialogue acts with query-likelihood clustering. In *Proceedings of the 6th International Conference on Educational Data Mining, Memphis, Tennessee, USA, July 6-9, 2013*, pages 20–27, 2013.
- [40] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: the concept revisited. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, pages 406–414, 2001.
- [41] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 50–56, 2018.
- [42] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional Sequence to Sequence Learning. *Arxiv*, 2017.
- [43] John J. Godfrey, Edward C. Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the 1992 IEEE International Conference on Acoustics, Speech and Signal Processing - Volume 1, ICASSP'92*, pages 517–520, Washington, DC, USA, 1992. IEEE Computer Society.
- [44] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [45] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [46] Google. Actions on google. 2019.
- [47] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. *ACL*, 2016.
- [48] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*, 2018.
- [49] H. He, A. Balakrishnan, M. Eric, and P. Liang. Learning Symmetric Collaborative Dialogue Agents with Dynamic Knowledge Graph Embeddings. *arXiv e-prints*, April 2017.
- [50] Matthew Henderson, Blaise Thomson, and Steve Young. Deep neural network approach for the dialog state tracking challenge. *SIGDIAL*, 2013.
- [51] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*, 2012.

- [52] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, page 91, 1991.
- [53] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [54] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997.
- [55] Jason L. Hutchens and Michael D. Alder. Introducing megahal. In *Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning, NeMLaP3/CoNLL '98*, pages 271–274, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.
- [56] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016.
- [57] N. Jaques, S. Gu, D. Bahdanau, J. M. Hernández-Lobato, R. E. Turner, and D. Eck. Sequence Tutor: Conservative Fine-Tuning of Sequence Generation Models with KL-control. *ArXiv e-prints*, November 2016.
- [58] Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, and Gaurav Agrawal et al. In-datacenter performance analysis of a tensor processing unit. *ISCA*, 2017.
- [59] D. Jurafsky, R. Bates, N. Coccaro, R. Martin, M. Meteer, K. Ries, E. Shriberg, A. Stolcke, P. Taylor, and C. Van Ess-Dykema. Automatic detection of discourse structure for speech recognition and understanding. In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 88–95, Dec 1997.
- [60] Filip Jurčiček, Ondřej Dušek, Ondřej Plátek, and Lukáš Žilka. Alex: A statistical dialogue systems framework. In *International Conference on Text, Speech, and Dialogue*, pages 587–594. Springer, 2014.
- [61] Filip Jurčiček, Blaise Thomson, and Steve J. Young. Natural actor and belief critic: Reinforcement algorithm for learning parameters of dialogue systems modelled as pomdps. *TSLP*, 7(3):6:1–6:26, 2011.
- [62] Prabhu Kaliamoorthi, Sujith Ravi, and Zornitsa Kozareva. PRADO: Projection attention networks for document classification on-device. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5012–5021, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [63] Anjali Kannan and Oriol Vinyals. Adversarial evaluation of dialogue models. *CoRR*, abs/1701.08198, 2017.
- [64] Divyansh Kaushik and Zachary C Lipton. How much reading does reading comprehension require? a critical investigation of popular benchmarks. *arXiv preprint arXiv:1808.04926*, 2018.
- [65] John F Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 1984.

- [66] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. Sharp nearby, fuzzy far away: How neural language models use context. *arXiv preprint arXiv:1805.04623*, 2018.
- [67] Hamed Khanpour, Nishitha Guntakandla, and Rodney D. Nielsen. Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2012–2021, 2016.
- [68] Seokhwan Kim, Luis Fernando D’Haro, Rafael E Banchs, Jason Williams, and Matthew Henderson. Dialog state tracking challenge 4. 2015.
- [69] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [70] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
- [71] S. Kottur, J. M. F. Moura, S. Lee, and D. Batra. Natural Language Does Not Emerge ‘Naturally’ in Multi-Agent Dialog. *ArXiv e-prints*, June 2017.
- [72] Zornitsa Kozareva and Sujith Ravi. ProSeqo: Projection sequence networks for on-device text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3894–3903, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [73] Ben Krause, Marco Damonte, Mihai Dobre, Daniel Duma, Joachim Fainberg, Federico Fancellu, Emmanuel Kahembwe, Jianpeng Cheng, and Bonnie Webber. Edina: Building an open domain socialbot with self-dialogues. *Arxiv*, 2017.
- [74] Krista Lagus and Jukka Kuusisto. Topic identification in natural language dialogues using neural networks. In *Proceedings of the SIGDIAL 2002 Workshop, The 3rd Annual Meeting of the Special Interest Group on Discourse and Dialogue, Thursday, July 11, 2002 to Friday, July 12, 2002, Philadelphia, PA, USA*, pages 95–102, 2002.
- [75] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015.
- [76] Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Xiang Li, Yaoqin Zhang, Zheng Zhang, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. Convlab: Multi-domain end-to-end dialog system platform. *CoRR*, abs/1904.08637, 2019.
- [77] Esther Levin and Roberto Pieraccini. A stochastic model of computer-human interaction for learning dialogue strategies. In *Fifth European Conference on Speech Communication and Technology, EUROSPEECH 1997, Rhodes, Greece, September 22-25, 1997*, 1997.
- [78] M. Lewis, D. Yarats, Y. N. Dauphin, D. Parikh, and D. Batra. Deal or No Deal? End-to-End Learning for Negotiation Dialogues. *ArXiv e-prints*, June 2017.
- [79] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A Diversity-Promoting Objective Function for Neural Conversation Models. *ArXiv e-prints*, October 2015.

- [80] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. In *The North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 110–119, 2016.
- [81] J. Li, M. Galley, C. Brockett, G. P. Spithourakis, J. Gao, and B. Dolan. A Persona-Based Neural Conversation Model. *ArXiv e-prints*, March 2016.
- [82] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, and D. Jurafsky. Deep Reinforcement Learning for Dialogue Generation. *ArXiv e-prints*, June 2016.
- [83] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. *CoRR*, abs/1603.06155, 2016.
- [84] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*, 2017.
- [85] Jimmy J. Lin and Boris Katz. Building a reusable test collection for question answering. *JASIST*, 57(7):851–861, 2006.
- [86] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.
- [87] Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *CoRR*, abs/1603.08023, 2016.
- [88] Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *CoRR*, abs/1603.08023, 2016.
- [89] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. *CoRR*, abs/1901.11504, 2019.
- [90] Yang Liu. Using SVM and error-correcting codes for multiclass dialog act classification in meeting corpus. In *INTERSPEECH 2006 - ICSLP, Ninth International Conference on Spoken Language Processing, Pittsburgh, PA, USA, September 17-21, 2006*, 2006.
- [91] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [92] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.
- [93] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *CoRR*, abs/1506.08909, 2015.
- [94] Ryan Lowe, Nissan Pow, Iulian V. Serban, and Joelle Pineau. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *SIGDIAL*, 2015.

- [95] Thang Luong, Richard Socher, and Christopher D. Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 104–113, 2013.
- [96] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014.
- [97] S. McGlashan, N. Fraser, N. Gilbert, E. Bilange, P. Heisterkamp, and N. Youd. Dialogue management for telephone information systems. In *Proceedings of the third conference on Applied natural language processing*, pages 245–246. Association for Computational Linguistics, 1992.
- [98] S. Merity, N. Shirish Keskar, and R. Socher. Regularizing and Optimizing LSTM Language Models. *ArXiv e-prints*, August 2017.
- [99] S. Merity, N. Shirish Keskar, and R. Socher. An Analysis of Neural Language Modeling at Multiple Scales. *ArXiv e-prints*, March 2018.
- [100] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing LSTM language models. *CoRR*, abs/1708.02182, 2017.
- [101] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *ICLR*, 2017.
- [102] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tür, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, and Geoffrey Zweig. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23:530–539, 2015.
- [103] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048, 2010.
- [104] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey 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.*, pages 3111–3119, 2013.
- [105] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey 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.*, pages 3111–3119, 2013.
- [106] Alexander H Miller, Will Feng, Adam Fisch, Jiasen Lu, Dhruv Batra, Antoine Bordes, Devi Parikh, and Jason Weston. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*, 2017.

- [107] Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *ACL*, pages 236–244, 2008.
- [108] Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M. Khapra. Towards exploiting background knowledge for building conversation systems. *EMNLP*, 2018.
- [109] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814. Omnipress, 2010.
- [110] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *NAACL Demonstrations*, 2019.
- [111] Alexandros Papangelis, Mahdi Namazifar, Chandra Khatri, Yi-Chia Wang, Piero Molino, and Gokhan Tur. Plato dialogue system: A flexible conversational ai research platform. *ArXiv*, abs/2001.06463, 2020.
- [112] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics, 2014.
- [113] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.
- [114] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke 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)*, pages 2227–2237. Association for Computational Linguistics, 2018.
- [115] Chen Qu, Liu Yang, Minghui Qiu, W. Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. Bert with history answer embedding for conversational question answering. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR’19*, 2019.
- [116] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/language_understanding_paper.pdf, 2018.
- [117] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *Arxiv*, 2019.
- [118] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [119] Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 337–346, 2011.

- [120] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset, 2019.
- [121] Antoine Raux, Brian Langner, Alan W Black, and Maxine Eskenazi. Let’s go: Improving spoken dialog systems for the elderly and non-natives. *Eurospeech*, 2003.
- [122] Sujith Ravi. Projectionnet: Learning efficient on-device deep networks using neural projections. *CoRR*, abs/1708.00630, 2017.
- [123] Sujith Ravi. Efficient on-device models using neural projections. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5370–5379, 2019.
- [124] Sujith Ravi and Zornitsa Kozareva. Self-governing neural networks for on-device short text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 804–810, 2018.
- [125] Norbert Reithinger and Martin Klesen. Dialogue act classification using language models. In *EuroSpeech*, 1997.
- [126] Hang Ren, Weiqun Xu, Yan Zhang, and Yonghong Yan. Dialog state tracking using conditional random fields. In *Proceedings of the SIGDIAL 2013 Conference, Metz, France, August, 2013*.
- [127] A. Ritter, C. Cherry, and W. B. Dolan. Data-driven response generation in social media. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 583–593, 2011.
- [128] Lina Maria Rojas-Barahona, Milica Gasic, Nikola Mrksic, Pei-Hao Su, Stefan Ultes, Tsung-Hsien Wen, Steve J. Young, and David 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*, pages 438–449, 2017.
- [129] Lina Maria Rojas-Barahona, Milica Gasic, Nikola Mrksic, Pei-Hao Su, Stefan Ultes, Tsung-Hsien Wen, Steve J. Young, and David Vandyke. A network-based end-to-end trainable task-oriented dialogue system. *EACL*, 2017.
- [130] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1.
- [131] Vasile Rus and Mihai Lintean. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 157–162. Association for Computational Linguistics, 2012.
- [132] Abdelrhman Saleh, Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, and Rosalind Picard. Hierarchical reinforcement learning for open-domain dialog, 2019.
- [133] Hannes Schulz, Jeremie Zumer, Layla El Asri, and Shikhar Sharma. A frame tracking model for memory-enhanced dialogue systems. *CoRR*, abs/1706.01690, 2017.

- [134] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.
- [135] I. V. Serban, C. Sankar, M. Germain, S. Zhang, Z. Lin, S. Subramanian, T. Kim, M. Pieper, S. Chandar, N. R. Ke, S. Rajeshwar, A. de Brebisson, J. M. R. Sotelo, D. Suhubdy, V. Michalski, A. Nguyen, J. Pineau, and Y. Bengio. A Deep Reinforcement Learning Chatbot. *ArXiv e-prints*, September 2017.
- [136] I. V. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, and Y. Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference (AAAI)*, 2017.
- [137] Iulian V Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, Sarath Chandar, Nan Rosemary Ke, et al. A deep reinforcement learning chatbot. *arXiv preprint arXiv:1709.02349*, 2017.
- [138] Iulian Vlad Serban, Tim Klinger, Gerald Tesauro, Kartik Talamadupula, Bowen Zhou, Yoshua Bengio, and Aaron 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.*, pages 3288–3294, 2017.
- [139] Iulian Vlad Serban, Ryan Lowe, Laurent Charlin, and Joelle Pineau. Generative deep neural networks for dialogue: A short review. In *NIPS, Let’s Discuss: Learning Methods for Dialogue Workshop*, 2016.
- [140] Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. A survey of available corpora for building data-driven dialogue systems. *nnn*, 2017.
- [141] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of AAAI*, 2016.
- [142] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. *CoRR*, abs/1605.06069, 2016.
- [143] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *CoRR*, abs/1803.02155, 2018.
- [144] X. Shen, H. Su, Y. Li, W. Li, S. Niu, Y. Zhao, A. Aizawa, and G. Long. A Conditional Variational Framework for Dialog Generation. *ArXiv e-prints*, April 2017.
- [145] Chenglei Si, Shuohang Wang, Min-Yen Kan, and Jing Jiang. What does bert learn from multiple-choice reading comprehension datasets?, 2019.
- [146] A. Simpson and N. M Eraser. Black box and glass box evaluation of the sundial system. In *Third European Conference on Speech Communication and Technology*, 1993.
- [147] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28: Annual*

- Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3483–3491, 2015.
- [148] Y. Song, R. Yan, X. Li, D. Zhao, and M. Zhang. Two are Better than One: An Ensemble of Retrieval- and Generation-Based Dialog Systems. *ArXiv e-prints*, October 2016.
- [149] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM 2015)*, 2015. In press.
- [150] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A neural network approach to context-sensitive generation of conversational responses. *arXiv preprint arXiv:1506.06714*, 2015.
- [151] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.
- [152] Dave Steinkrau, Patrice Y. Simard, and Ian Buck. Using gpus for machine learning algorithms. *ICDAR*, 2005.
- [153] Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373, 2000.
- [154] Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. *arXiv preprint arXiv:1804.00079*, 2018.
- [155] David Suendermann-Oeft, Vikram Ramanarayanan, Moritz Teckenbrock, Felix Neutatz, and Dennis Schmidt. Halef: An open-source standard-compliant telephony-based modular spoken dialog system: A review and an outlook. In *Natural language dialog systems and intelligent assistants*, pages 53–61. Springer, 2015.
- [156] Ilya Sutskever, Oriol Vinyals, and Quoc 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*, pages 3104–3112, 2014.
- [157] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.
- [158] Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. Why self-attention? a targeted evaluation of neural machine translation architectures. *arXiv preprint arXiv:1808.08946*, 2018.
- [159] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. The penn treebank: An overview, 2003.
- [160] Jörg Tiedemann. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in*

- Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, 2009.
- [161] Ke Tran, Arianna Bisazza, and Christof Monz. The importance of being recurrent for modeling hierarchical structure. *arXiv preprint arXiv:1803.03585*, 2018.
- [162] Gokhan Tur. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. John Wiley and Sons, January 2011.
- [163] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alexander Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *Arxiv*, 2016.
- [164] Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. Tensor2tensor for neural machine translation. *CoRR*, 2018.
- [165] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.
- [166] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.
- [167] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.
- [168] Oriol Vinyals and Quoc V. Le. A neural conversational model. *CoRR*, abs/1506.05869, 2015.
- [169] Oriol Vinyals and Quoc V. Le. A neural conversational model. *Arxiv*, 2015.
- [170] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461, 2018.
- [171] Zhuoran Wang and Oliver Lemon. A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information. In *Proceedings of the SIGDIAL 2013 Conference*, pages 423–432, 2013.
- [172] B. Wei, S. Lu, L. Mou, H. Zhou, P. Poupart, G. Li, and Z. Jin. Why Do Neural Dialog Systems Generate Short and Meaningless Replies? A Comparison between Dialog and Translation. *ArXiv e-prints*, December 2017.
- [173] Joseph Weizenbaum. Eliza a computer program for the study of natural language communication between man and machine. *Computational Linguistics*, 1966.
- [174] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.
- [175] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina Maria Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve J. Young. A network-based end-to-end trainable task-oriented dialogue system. *CoRR*, abs/1604.04562, 2016.

- [176] Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve J. Young. Latent intention dialogue models. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 3732–3741, 2017.
- [177] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *Proceedings Of The International Conference on Representation Learning (ICLR 2015)*, 2015. In Press.
- [178] Jason Williams, Antoine Raux, and Matthew and Henderson. The dialog state tracking challenge series: A review. *Dialog and Discourse*, 2016.
- [179] Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 404–413, 2013.
- [180] Jason D Williams. An empirical evaluation of a statistical dialog system in public use. In *Proceedings of the SIGDIAL 2011 Conference*, pages 130–141. Association for Computational Linguistics, 2011.
- [181] Jason D Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.
- [182] Jason D. Williams and Steve J. Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 2007.
- [183] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.
- [184] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.
- [185] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019.
- [186] Steve Young, Milica Gasic, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.
- [187] Amy X. Zhang, Bryan Culbertson, and Praveen Paritosh. Characterizing online discussion using coarse discourse sequences. In *Proceedings of the 11th International AAAI Conference on Weblogs and Social Media, ICWSM ’17*, 2017.
- [188] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*, 2018.
- [189] T. Zhao, R. Zhao, and M. Eskenazi. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. *ArXiv e-prints*, March 2017.
- [190] H. Zhou, M. Huang, T. Zhang, X. Zhu, and B. Liu. Emotional Chatting Machine: Emotional Conversation Generation with Internal and External Memory. *ArXiv e-prints*, April 2017.

