Interspeech 2018
2-6 September 2018, Hyderabad

Semi-supervised learning for information extraction from dialogue

Anjuli Kannan, Kai Chen, Diana Jaunzeikare, Alvin Rajkomar

Google Brain, Google, Inc., USA
{anjuli, kaichen, dianajzk, alvinrajkomar}@google.com

Abstract

In this work we present a method for semi-supervised learning from transcripts of dialogue between humans. We consider the scenario in which a large amount of transcripts are available, and we would like to extract some semantic information from them; however, only a small number of transcripts have been labeled with this information. We present a method for leveraging the unlabeled data to learn a better model than could be learned from the labeled data alone. First, a recurrent neural network (RNN) encoder-decoder is trained on the task of predicting nearby turns on the full dialogue corpus; next, the RNN encoder is reused as a feature representation for the supervised learning problem. While previous work has explored the use of pre-training for non-dialogue corpora, our method is specifically geared toward the dialogue use case. We demonstrate an improvement on a clinical documentation task, particularly in the regime of small amounts of labeled data. We compare several types of encoders, both in the context of a classification task and in a human-evaluation of their learned representations. We show that our method significantly improves the classification task in the case where only a small amount of labeled data is available.

1. Introduction

A key problem in spoken language understanding is automatically extracting local information from a dialogue between two people. For example, a company with a customer support line may wish to extract structured information about each caller's problem (e.g., the product, details of the issue). Similarly, a healthcare provider may be interested in automatic documentation of issues discussed in a patient-doctor visit (e.g., current symptoms, medical history) [1]. We describe this information as *local* because it can be inferred from a small segment of the dialogue (as opposed to the overall topic). With continuing improvement of automatic speech recognition, the problem is now a matter of extracting information from text transcripts [2].

While the dialogue corpus itself may be large, it is common for it to come with few or no labels. Labeling must be done by humans and may require expert knowledge, which can be very costly. In these settings, semi-supervised learning, in which we leverage a large amount of unlabeled data to improve a model trained on a small amount of labeled data, is essential.

This paper presents a method for using semi-supervised learning for extraction of local structured information from dialogue transcripts. It is geared toward the regime in which we have access to only a small set of labeled transcripts, but also to a large amount of unlabeled transcripts.

Our approach has two stages. In the first, an RNN encoder-decoder is trained over the full, unlabeled corpus to perform the following task: given one turn, it must predict the next or previous turn. This results in an RNN *encoder* which learns compactly how to represent a dialogue turn, and an RNN *decoder* which learns how to generate a dialogue turn.

In the second stage, a separate RNN encoder-decoder (or encoder-classifier) model is trained to predict the provided labels, given each turn of dialogue. This second model is trained only on the small amount of supervised data, but its encoder is initialized with the weights of the first model's encoder, enabling transfer of knowledge from the prior task. The second model's weights are then fine-tuned to the task of interest. Applying this method to a clinical documentation task (described further in 1.2), we observe an improvement in several key metrics, especially when the amount of labeled data is small.

This method also generalizes to cases where multiple consecutive dialogue turns may be required to extract a piece of structured information: we simply train an encoder to represent a segment of k consecutive turns rather than a single turn.

1.1. Related work

The use of RNN-based methods for semi-supervised learning of text classification has previously been explored on non-dialogue corpora. Most similar to our work, [3], [4] and [5] also employ a two-stage training process in which the first stage (referred to as *pre-training*) trains an RNN encoder in an unsupervised manner, and the second stage leverages this encoder on a supervised task.

The first stage of our training process differs in several ways. First, inspired by [3], we investigate the behavior of a sequence autoencoder; [3] showed that an RNN pre-trained with an autoencoder would surpass previous text classification baselines. Next, to leverage the turn structure unique to dialogue, we propose a variation that predicts nearby turns and observe that it outperforms the autoencoder. This is closely related to the "skip-thought" method [6], which encodes a sentence by training an encoder to predict the surrounding sentences. In a broader sense, pre-training is very related to multi-task and transfer learning [7, 8], since we use a loss function in one task (pre-training) to improve the performance on another task. Our method is also related to recent works in zero-resource machine translation [9, 10, 11, 12].

In the realm of dialogue systems, RNNs have been used extensively to understand spoken language, typically for slot-filling [13, 14, 15, 16], but none of these use pre-training, beyond that of word embeddings. [17] has used nearby-turn prediction to train "turn embeddings" as part of an end-to-end dialogue system, which is similar to the first stage of our training process. Our work differs in the second stage: 1) we demonstrates the effectiveness of pre-training on an information extraction task; 2) we analyze our method under various conditions and compare it directly to autoencoders.

1.2. Task

To demonstrate the effectiveness of our method, we present results on a clinical documentation task. A physician's note includes a section listing the affirmation, denial, and description of symptoms necessary to evaluate a patient. However, patients

10.21437/Interspeech.2018-1318

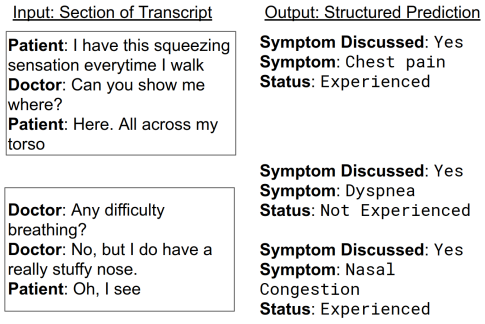

Figure 1: *Examples of symptom extraction task. Given a snippet of a patient-doctor transcript, the task is to predict (a) whether or not a symptom is present in the snippet, (b) the name of the symptom, (c) the status of the symptom.*

may express the experience of symptoms in numerous ways: "I have a headache", "My head hurts", "There's a throbbing behind my eye". This information can even spread across multiple consecutive turns: "I feel a throbbing." "Where?" "My head.". For each turn of each transcript, our goal is to extract and classify any symptoms mentioned, as well as their status (whether the patient has experienced it). See Figure 1 for an example.

We choose to conduct our experiments on this task because it requires more complex language understanding than simple keyword spotting. A simple baseline that uses a medical named entity recognition model to annotate symptoms recalls only about 20% of the symptoms in our data.

The remainder of this paper is organized as follows. First, we explain our method of semi-supervised learning in more detail. Next we describe details of our experiments on the clinical documentation task, and show the results together with human-evaluation of the turn representations learned from pre-training.

2. Semi-supervised learning for information extraction

This work presents and analyzes a semi-supervised learning method for extracting structured information from dialogue transcripts. The method consists of two stages. For consistency with prior work, we refer to these as *unsupervised pre-training* and *supervised classification*.

2.1. Unsupervised pre-training

The first stage pre-trains an RNN encoder as part of an encoder-decoder (also known as sequence-to-sequence) model [18, 19]. Broadly, this class of models solve the following problem: given a pair of token-sequences $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, ..., x_m)$ is called the input sequence and $\mathbf{y} = (y_1, ..., y_n)$ is called the output sequence, we would like to maximize $P(\mathbf{y}|\mathbf{x})$. These models contain two key components.

First, an RNN encoder consumes the input sequence $\mathbf{x}$ one token at a time to produce a hidden state, $\mathbf{h}(\mathbf{x})$. Conceptually, this is a dense vector representation of the sequence $\mathbf{x}$. Next, an RNN decoder estimates an output distribution over sequences $\mathbf{y}^*$ conditional on $\mathbf{h}(\mathbf{x})$. Maximizing $P(\mathbf{y}|\mathbf{x})$ over a training corpus forces the encoder to learn to compactly represent the most important information from the input sequence.

RNN encoder-decoder models are now ubiquitous for language understanding tasks. The key insight of pre-training is that we can train such a model on one task for which there is a large amount of unlabeled data, and then re-apply what the model has learned to another task with few labels [3, 4, 5].

In our model, the input sequence $\mathbf{x}$ is the tokens comprising a turn of dialogue, so the encoder will learn a dialogue turn representation. We compare three flavors of pre-training that differ in how the output sequence $\mathbf{y}$ is defined (see Figure 2). Since the encoder is trained to be maximally effective in predicting $\mathbf{y}$, the way in which $\mathbf{y}$ is defined will directly impact the representation of $\mathbf{x}$, a dialogue turn, that is learned. Our objective is to understand how the definition of $\mathbf{y}$ in the unsupervised pre-training stage will impact the performance of the supervised classification model.

Autoencoder In the first method, we train a sequence autoencoder, since [3] showed promising results with this method. This means that the output sequence $\mathbf{y}$ is identical to the input sequence $\mathbf{x}$. In other words, the model must learn to reconstruct a dialogue turn itself. While this seems simplistic, [3] has shown that it is highly effective because $\mathbf{h}(\mathbf{x})$ acts as a sort of bottleneck: in order for the decoder to reconstruct sequence given only $\mathbf{h}(\mathbf{x})$, the encoder must learn a compact representation. On the other hand, one drawback of this method is that correct reconstruction is more about the words themselves rather than the overall meaning.

Next-turn prediction To force the encoder to encapsulate more of the meaning of a turn, we also consider next-turn prediction as a second method for unsupervised pre-training. Here, given the input sequence $\mathbf{x_t}$, the output sequence $\mathbf{y}$ is $\mathbf{x_{t+1}}$. Essentially this is the same model as a neural conversation model [20, 21, 22, 23], but we are only interested in the encoder, and will throw away the decoder. We hypothesize that this objective forces the encoder to learn something about the semantic intent of $\mathbf{x_t}$, rather than simply its lexical features, in order to predict the reply $\mathbf{x_{t+1}}$. For example, if $\mathbf{x_t}$ contains a question, the encoder should encapsulate this to predict the next speaker's answer. We argue that this results in a stronger turn representation than from an autoencoder.

Skip-turn prediction Finally, we consider a generalization of next-turn prediction, which is to predict not just the next turn but other nearby turns as well. In other words, given input sequence $\mathbf{x_t}$, we create $2k$ different input examples: $(\mathbf{x_t}, \mathbf{x_{t-k}})$, $(\mathbf{x_t}, \mathbf{x_{t-k+1}})$, ... $(\mathbf{x_t}, \mathbf{x_{t-1}})$, $(\mathbf{x_t}, \mathbf{x_{t+1}})$, ... $(\mathbf{x_t}, \mathbf{x_{t+k-1}})$, $(\mathbf{x_t}, \mathbf{x_{t+k}})$ for some value of k. This is inspired by the Skip-Thought [6] and Skip-Gram [24] models. The premise of these models is that a word or sentence is defined by its context words or sentences; here we apply the same concept to dialogue turns.

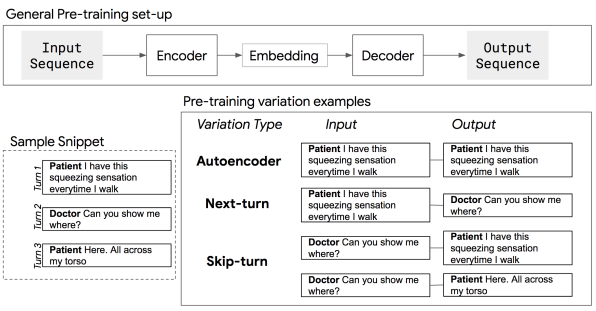

Figure 2: *Depiction of general pre-training set-up and how the input and output sequences are defined for each type of encoder.*

2.2. Supervised classification

In the second phase of our method, we perform a traditional supervised text classification task. Since our goal is to investigate the effectiveness of the unsupervised pre-training, we keep the supervised phase relatively simple. We assume that we have a small set of labeled dialogue turns $(\mathbf{x_t}, \mathbf{z_t})$, where $\mathbf{x_t}$ is a dialogue turn, and $\mathbf{z_t}$ is a sequence of labels that humans have given to that dialogue turn. For our particular task, $\mathbf{x_t}$ is from a patient-doctor conversation, and $\mathbf{z_t}$ consists of three tokens: a token indicating whether or not a symptom is discussed in $\mathbf{x_t}$, the name of the symptom, and the status of that symptom.

The objective of the model is to maximize $P(\mathbf{z_t}|\mathbf{x_t})$. Since both $\mathbf{x_t}$ and $\mathbf{z_t}$ are sequences, we again apply an RNN encoder-decoder architecture for the classification task. We posit that in the case where $\mathbf{z_t}$ is a single token (label), the decoder can be replaced with a feed-forward neural net classifier. Since we are studying the impact of pre-training, a more extensive exploration of supervised classification architectures, as well as more sophisticated handling of cases in which multiple symptoms are mentioned in a turn, are considered to be beyond the scope of this work. These questions will be explored in later work.

The key link with the first phase is that we initialize the RNN encoder with the weights from the RNN encoder that was trained in the first phase, then allow them to continue training. The first phase should teach the RNN how to effectively and compactly encode the meaning of a turn, while the second phase should learn how to apply this representation to a classification task.

This method can also be generalized to cases where multiple turns of context may be required for classification. For this scenario, we choose a context window size k and train an encoder to encode not a single turn, but a segment of k consecutive turns. In other words, during pre-training, the input sequence consists of turns $\mathbf{x}_{\mathbf{t}-\frac{\mathbf{k}-\mathbf{1}}{\mathbf{2}}}, \cdots, \mathbf{x_t}, \cdots, \mathbf{x}_{\mathbf{t}+\frac{\mathbf{k}-\mathbf{1}}{\mathbf{2}}}$ concatenated together. The output sequence will also consist of k turns: the same k turns for the autoencoder, the next k turns for next-turn prediction, and either the previous or next k turns for skip-turn prediction. In the second stage we continue to use the same context of k turns and initialize the encoder from the first stage.

3. Experimental details

Our experiments are conducted on a set of about 90K human-transcribed and de-identified medical encounters (details see [2]). The transcripts consist of a series of turns that each begin with a special token indicating the type of speaker (e.g. clinician, patient or caregiver).

In the first stage, unsupervised pre-training is performed on the full set of conversations, comprising 10 million speaker turns. In the second stage, supervised classification is performed on labels created by human experts on 566 conversations. Each turn in the transcripts was labeled in three ways: whether a symptom was discussed in the turn, the category of symptom (out of 170 pre-specified common symptoms), and the status of the symtpom (experienced, not experienced, or not applicable). See examples in Figure 1. These labels comprise a basic building block for the "review of systems" in a clinical note.

In total, humans labeled about 7K symptom occurrences across the 566 conversations (80/20 train/test split, 79 as test and 487 as training). The most frequent symptoms are: pain (7.0%), cough (5.6%) and shortness-of-breath (5.5%). In the next section we show results on two versions of the training data: *Symptom600* is only ten percent of the training set (about 600 symptom occurrences, randomly chosen), while *Symptom6K* is the full training set. Note that 73% of dialogue turns do not have a symptom associated with them. We keep these "negative" examples in both training and test sets because part of the task is to determine if there is a symptom present.

We conduct the experiments with a 3-turn context window because more than one turn may be required to determine a symptom's status. Consider the simplest example: "Any pain?" "Yes, in my back." In order to conclude that back pain is experienced, both turns must be considered together.

4. Results

The main result of this paper is that our symptom extraction model achieves the best results when the encoder has been pre-trained with the skip-turn or next-turn methods, especially when the training data size is small. We use three metrics for our model. Symptom Present F1 (SPF1) is the F1 score for identifying which turns mention a symptom. This is important to measure because most dialogue turns do not mention any symptom. Among those that do mention a symptom, Symptom Accuracy (SympAcc) is the percentage of examples for which the model correctly classifies the symptom, and Status Accuracy (StatusAcc) is the percentage of examples the model correctly classifies the status.

4.1. Baseline model

Our baseline model is a long short-term memory (LSTM) [25] encoder-decoder model without pre-training. The decoder outputs a sequence of 1-3 tokens (symptom-presence, name, status), which is chosen through beam search as described in [18].

We tuned the hyper-parameters of our baseline model on a dev set (subset of training), and report the results on the test set. Our final model is a single layer bidirecional LSTM with 512 hidden nodes, with small dropout (0.1) on the Symptom600 set and no dropout on Symptom6K. We used the Adam optimizer and tuned the learning rates on each training set.

For our pre-training experiments we use the same model architecture as the baseline, but with the encoder component pre-trained with the various objectives previously described (autoencoder, next-turn, skip-turn). We do not show a comparison with other text classification models (e.g. bag-of-words, support vector machines), as [3] demonstrates that an RNN model pre-trained with a sequence autoencoder meets or surpasses all those models on a range of text classification tasks.

As a side note, a "chance baseline" using the training set's label distribution will get SPF1 of 27% and SympAcc of 7%, which is much worse than our models.

4.2. Comparing pre-training methods on small data

Table 1 compares the various pre-training methods on the symptom extraction task for Symptom600. Without any pre-training (first row), the model barely learns anything: it classifies just 9% of symptoms correctly, while all three pre-training methods have much higher accuracies. This shows that pre-training is essential to making the symptom extraction model work with a small amount of labeled training data.

Among the pre-training methods, next-turn and skip-turn encoders perform better than autoencoder. We hypothesize that this is because the next-turn and skip-turn tasks are harder, so that pre-training leads to a stronger encoder. Those methods also require better understanding of intent, rather than simply copying lexical features. In Section 4.4, we will analyze the encoders in more detail.

Pre-training	SPF1	SympAcc	StatusAcc
None (baseline)	67%	9%	44%
Autoencoder	78%	36%	57%
Next-turn	83%	32%	63%
Skip-turn	87%	39%	66%

Table 1: *A comparison of various pre-training methods on Symptom600. All three pre-training methods (rows 2-4) outperform the baseline model that uses no pre-training (row 1) on all three metrics, with the skip-turn encoder performing the best.*

4.3. Impact of more data

Next we investigate the impact of increasing the amount of labeled training data for the second stage. The first block of Table 2 shows our model's results when we train the model on Symptom6K, which has 10x more labeled data than Symptom600. All the models with pre-training still maintain a gain over the baseline model without pre-training, but it is narrower. The difference between different encoder types also narrowed. This shows that our method is most effective when the amount of labeled training data is small.

In the second block of Table 2, we freeze the encoder's parameters after pre-training, and only allow the rest of the model to train. Here we focus on the *intrinsic* quality of different encoders after pre-training. Again, we see a larger gap between different encoders: both next-turn and skip-turn encoders have much better performance than autoencoder.

Pre-training	SPF1	SympAcc	StatusAcc
None (baseline)	81%	40%	55%
Autoencoder	86%	47%	61%
Next-turn	87%	50%	67%
Skip-turn	88%	52%	65%
Autoencoder (frozen)	80%	35%	54%
Next-turn (frozen)	87%	46%	63%
Skip-turn (frozen)	86%	45%	63%

Table 2: *A comparison of various pre-training methods on Symptom6K (10x more labeled data than Table 1). The gap between pre-training (rows 2-4) and no pre-training (row 1) has narrowed. Freezing the encoders (rows 5-7) demonstrates a larger gap between autoencoder and the other two encoders.*

4.4. Encoder analysis

We have shown that skip-turn and next-turn encoders are more effective at improving the model's performance than the autoencoder. To investigate further why this is the case, we compare the representations learned by the autoencoder and next-turn encoder. First, we run each of the 6.7 million unique turns in our corpus through each encoder and extract the final hidden state, a 1024-dimensional dense vector that we call a *turn embedding*. Then, given a turn, we can compute its nearest neighbors in the turn embedding space based on cosine distance. We use a context of 1-turn to train the encoders.

See Table 3 for an example. Using autoencoder, the nearest neighbors of a short sentence "Yeah I was a little more anxious." are similar in syntax but not semantics, while the neighbors for the next-turn encoder are semantically similar. Even for much longer turns, next-turn encoder can still provide neighbors with similar meaning, as shown in Table 4.

Next, to quantify the semantic similarity, we manually selected 230 turns related to symptoms, and asked medical documentation experts to judge the similarity between each turn and its nearest 5 neighbors. Raters assigned scores between 1 to 5 based on their semantic similarity, i.e. whether they are substitute-able in a doctor-patient conversation, with 5 being most similar. See Table 5 for the results. Note that these symptom-related turns are much longer than the typical turn as much of the corpus contains social chatting ("how are you") and short answers ("okay"). Overall, the next-turn encoder has a much higher similarity score than autoencoder. For both methods, their neighbors are less similar for longer turns; this is not surprising since encoding a longer turn is a more difficult task. This result helps explain why the next-turn encoder achieved better performance than the autoencoder in symptom extraction.

Autoencoder	Next-turn encoder
Yeah I was taking the ultram.	Well I was a little anxious.
Yeah I was using them more.	I was just really depressed.
Yeah I was laying there.	Yeah I was like really anxiety dressed, depressed.
Yeah I do have the back pain.	Yeah I'm just a little bit nervous.
Yeah I was curious about that.	Yeah and I got real nervous.

Table 3: *Neighbors of "Yeah I was a little more anxious."*

Turn	Nearest neighbor
Yeah but the only problem is that I have to urinate sometimes every two hours.	Like every two hours to go to the bathroom.
Yeah. and I lost another 2 pounds, and she was happy with that.	I had lost all that weight. I was so proud of myself.
Yeah, my stomach didn't like it, and then you put me on fosamax and that didn't -	With the actonel because it made me sick

Table 4: *Examples of neighbors for long turns based on next-turn encoder. The neighbors are still similar in meaning.*

Turn Length	Autoencoder	Next-turn
Short (1-10 tokens)	1.632	3.521
Medium (11-15 tokens)	1.120	2.485
Long (15+ tokens)	1.111	2.322
Overall	1.286	2.771

Table 5: *Average semantic similarity score assigned by humans to turns' neighbors in the "turn embedding" space. On average raters assign higher (better) scores to the next-turn encoder.*

5. Conclusions

In this work we presented a method for semi-supervised learning of local information extraction from dialogue transcripts. We tested our method on a clinical documentation task that required locating and classifying symptoms in a doctor-patient conversation. We observed that this method significantly improved the performance of an RNN encoder-decoder model, especially when only a small amount of labeled training data was available. Future work will investigate how this method may be impacted by errors introduced by ASR and how to mitigate this impact.

6. Acknowledgements

The authors would like to thank Izhak Shafran, Patrick Nguyen, and Yonghui Wu for helpful discussions and feedback; Chris Co, Nina Gonzalez, Michael Pearson and Jack Po for contributions to data and labeling; and Kat Chou, Greg Corrado, Claire Cui, and Jeff Dean for supporting this work.

7. References

[1] A. Verghese, N. Shah, and R. Harrington, "What this computer needs is a physician: Humanism and artificial intelligence," *JAMA*, Dec. 2017.

[2] C. C. Chiu, A. Tripathi, K. Chou, C. Co, N. Jaitly, D. Jaunzeikare, A. Kannan, P. Nguyen, H. Sak, A. Sankar, J. Tansuwan, and N. Wan, "Speech recognition for medical conversations," *CoRR*, vol. abs/1711.07274, 2017.

[3] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning.," *NIPS*, 2015.

[4] P. Ramachandran, P. J. Liu, and Q. V. Le, "Unsupervised pretraining for sequence to sequence learning," *CoRR*, vol. abs/1611.02683, 2016.

[5] A. Radford, R. Jozefowicz, and I. Sutskever, "Learning to generate reviews and discovering sentiment," *CoRR*, vol. abs/1704.01444, 2017.

[6] R. Kiros, Y. Zhu, R. Salakhutdinov, R. Zemel, A. Torralba, R. Urtasun, and S. Fidler, "Skip-thought vectors," in *Proceedings of NIPS*, 2015.

[7] M. T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning.," in *Proceedings of ICLR*, 2015.

[8] B. Zoph, D. Yuret, J. May, and K. Knight, "Transfer learning for low-resource neural machine translation.," in *Proceedings of EMNLP*, 2016.

[9] O. Firat, B. Sankaran, Y. Al-Onaizan, F. Yarman-Vural, and K. Cho, "Zero-resource translation with multilingual neural machine translation.," in *Proceedings of EMNLP*, 2016.

[10] D. He, Y. Zia, T. Qin, L. Wang, N. Yu, T. Liu, and W. Ma, "Dual learning for machine translation.," in *Proceedings of NIPS*, 2016.

[11] Y. Cheng, W. Xu, Z. He, W. He, H. Wu, M. Sun, and Y. Liu, "Semi-supervised learning for neural machine translation.," in *Proceedings of ICLR*, 2018.

[12] M. Artetxe, G. Labaka, E. Agirre, and K. Cho, "Unsupervised neural machine translation," in *Proceedings of ICLR*, 2018.

[13] G. Mesnil, , X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," *Interspeech*, 2013.

[14] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, Hakkani-Tur D, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, 2015.

[15] K. Yao, G. Zweig, M-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding in interspeech," *Interspeech*, 2013.

[16] K. Yao, B. Peng, Y. Zhang, D. Yu, G. Zweig, and Y. Shi, "Spoken language understanding using long short-term memory neural networks," *IEEE SLT*, 2014.

[17] A. Bayer, E. Stepanov, and G. Riccardi, "Towards end-to-end spoken dialogue systems with turn embeddings," in *Proc. Interspeech 2017*, 2017, pp. 2516–2520.

[18] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[19] K. Cho, B. van Merriënboer, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," in *Proc. EMNLP*, 2014.

[20] O. Vinyals and Q. Le, "A neural conversation model," in *ICML Deep Learning Workshop*, 2015.

[21] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J. Nie, J. Gao, and B. Dolan, "A neural network approach to context-sensitive generation of conversation responses," in *Proceedings of NAACL-HLT*, 2015.

[22] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," in *Proceedings of ACL-IJCNLP*, 2015.

[23] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Hierarchical neural network generative models for movie dialogues," in *arXiv preprint arXiv:1507.04808*, 2015.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of ICLR Workshop*, 2013.

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