Deep Embedding of Conversation Segments

Abir Chakraborty
Microsoft India
abir.chakraborty@gmail.com

Anirban Majumder
Flipkart Internet Private Ltd.
majumder.a@flipkart.com

Abstract

We introduce a novel conversation embedding by extending Bidirectional Encoder Representations from Transformers (BERT) framework. Specifically, information related to "turn" and "role" that are unique to conversations are augmented to the word tokens and the next sentence prediction task predicts a segment of a conversation possibly spanning across multiple roles and turns. It is observed that the addition of role and turn substantially increases the next sentence prediction accuracy. Conversation embeddings obtained in this fashion are applied to (a) conversation clustering, (b) conversation classification and (c) as a context for automated conversation generation on new datasets (unseen by the pre-training model).

We found that clustering accuracy is greatly improved if embeddings are used as features as opposed to conventional tf-idf based features that do not take role or turn information into account. On classification task, a fine-tuned model on conversation embedding achieves accuracy comparable to an optimized linear SVM model on tf-idf based features. Finally, we present a way of capturing variable length context in sequence-to-sequence models by utilizing this conversation embedding and show that BLEU score improves over a vanilla sequence to sequence model without context.

1 Introduction

Embedding of natural language units (word, sentence or paragraph) deals with the problem of finding a vector space representation of these units that can be used in downstream applications of classification, summarization or token identification. For example word embeddings (Mikolov et al., 2013a,b,c; Pennington et al., 2014) have found application in information retrieval (Manning et al., 2008), document classification (Sebastiani, 2002; Kim), question answering (Tellex et al., 2003; Minaee and Liu, 2017), named entity recognition (Turian et al., 2010) and parsing (Socher et al., 2013). Extending the same concept to sentences and documents one can also find the corresponding vector representations independently (Le and Mikolov, 2014) or by suitably averaging the word vectors (Kusner et al., 2015).

While aforementioned embeddings are created without optimizing for (or even considering) downstream applications there are recent approaches that seek optimal representations based on pre-training (Radford et al., 2018; Howard and Ruder, 2018; Peters et al., 2018). These applications can be at sentence-level such as natural language inference (Bowman et al., 2015; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005) where the semantic relationship between sentences are captured or at word-level tasks (Rajpurkar et al., 2016; Wang et al., 2018). There are two different approaches for applying pre-trained embeddings, (a) feature-based ((Peters et al., 2018), where the model architecture is task-specific and pre-training is a feature of the architecture) and (b) fine-tuning (Radford et al., 2018; Devlin et al., 2019) (where the pre-training architecture is quite generic to handle a variety of downstream tasks and model parameters are later fine-tuned for specific tasks). While most of the pre-training architectures use unidirectional language models (Radford et al., 2018; Peters et al., 2018), Bidirectional Encoder Representations from Transformers (BERT, (Devlin et al., 2019)) uses a different strategy to learn sentence/paragraph representations and achieves best scores on a variety of tasks.

Even though there are different strategies for creating word, sentence and document level embeddings, there is no study available in literature that deals with $conversation\ embedding$. While a piece of conversation may look very similar to a

Proceedings of the 18th International Conference on Natural Language Processing, pages 555–563
Silchar, India. December 16 - 19, 2021.

paragraph (and one can probably start with paragraph embedding to embed conversations) it has two important additional pieces of information, namely, turns and roles. A turn can consist of a single word, sentence or multiple sentences (all belong to a single role) and a conversation can have many participants (roles) where it is crucial to distinguish who is saying what. An efficient representation of a complete conversation (or part of it) should take into account the role and turn information (and their congruence) for downstream applications.

The most important application of conversation embedding is in the area of automated dialogue generation. Starting with the vanilla sequence-to-sequence model (Sutskever et al., 2014; Vinyals and Le, 2015) there are different approaches to capture the "context" so that meaningful responses can be generated (Sordoni et al., 2015b; Mei et al., 2017). The "context" continuously grows as the conversation progresses and can be defined in terms of everything that has happened in the conversation so far or key words from earlier turns extracted by some attention based algorithms (Bahdanau et al., 2015). There could be different approaches to capture a context, e.g., (a) separate RNNs for previous turns and roles, (b) attention over previous turns or (c) a global vector representing counts of tokens from previous turns etc. However, all of them have limitations either in capturing all the required information or in their ability to deal with a continuously increasing context length. An embedding that can map a variable length context (i.e., a conversation segment) into a numeric vector while including key pieces of information required for generating the next response would be immensely helpful in automatic dialogue generation. This is what has been attempted in this work where we create conversation embedding using BERT and apply to various downstream tasks. Our contributions are

1. Extension of BERT based *sentence representation* to *conversation representation* by adding the notion of roles and turns and thus creating an embedding of conversation segments hitherto unavailable in literature.

2. We show that with the inclusion of roles and turns during pre-training the next sentence prediction accuracy increases.

3. Application of these pre-trained models on conversation clustering shows better accuracy over tf-idf based features.

4. We demonstrate how conversation embedding can be used to capture context in sequence-to-sequence models and thereby improving the BLEU score.

2 Related Work

Very little work is available in the literature on conversation embedding, especially that treats conversations with all its associated complexities. Most of the work has been on word embedding (non-neural, (Brown et al., 1992; Ando and Zhang, 2005; John Blitzer and Pereira, 2006; Pennington et al., 2014) and neural (Mikolov et al., 2013a,b,c; Liu et al., 2017)), sentence embedding (Le and Mikolov, 2014) and embedding of paragraphs (Dai et al., 2015). Recent approaches involving pre-training and fine-tuning also deal with sentences and sentence pairs (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019) and the downstream tasks are mostly classifications (and not sequence generation). On conversation embedding, the closest that we see is from Mehri et al. (2019) where multiple pretraining objectives are explored. Conversations are encoded using recurrent neural network (RNN) and no information from roles or turns are included.

The importance of capturing "context" for relevant response generation is well understood. (Sordoni et al., 2015b) tried capturing the context initially using bag-of-words representation (Sordoni et al., 2015a) and later by a hierarchical recurrent encoder-decoder (HRED) approach (Serban et al., 2016a) applied to the movie dataset (Banchs, 2012) with only one previous utterance appearing as the context. Here, a dialogue D consisting of a series of utterances $\{U_1, U_2, \ldots, U_M\}$ was decomposed as

$$p_\theta(U_1, U_2, \ldots, U_M) = \prod_{m=1}^{M} \prod_{n=1}^{N_m} p_\theta(w_{m,n} | w_{m,<n}, U_{<m}) \quad (1)$$

and two encoders were used to encode context and current input. A second approach based on stochastic latent variables (called VHRED model) to capture dependency amongst multiple time steps is proposed by (Serban et al., 2016b) and it was shown that VHRED generated responses preferred over HRED. HRED architecture is further modified by (Li et al., 2018) to take bidirectional GRU as

input to the encoder, for application to multi-modal input scenario (Agarwal et al., 2018) and in a GAN set up for generating a sequence response (Su et al., 2018).

In the next section we describe our approach of creating conversation embedding from BERT. Subsequently, we demonstrate three applications of this embedding, namely, clustering, classifications and dialogue generation. Finally, conclusions are drawn.

3 Conversation Embedding Using BERT

BERT represents a sentence or a pair of sentences by their Wordpiece tokens (Sennrich et al., 2016), segment names (A or B) and token-wise positions. While this representation is enough for contiguous sentences from a paragraph it does not take into account (1) role or speaker and (2) turn or depth of the conversation. In this work we add these two additional pieces of information along with the rest of the embedding. To give an example, Table 1 is a snippet of a typical conversation with the corresponding roles (Customer and Agent) and turns $(0, 1, 2, \ldots)$:

The corresponding text, role and turn tokens for input to a BERT model will be as shown in Fig. 1. While the role in the present case takes only two values, $i.e.$, agent and customer (although there is no restriction) the turn can be as high as 200 in our conversation data. Thus, the turn embedding can be similar to the position embedding used in Transformer, $i.e.$, sines and cosines. However, in the present work we have projected both the turn and role values to a fixed-dimensional vector and learnt the corresponding embeddings (Gehring et al., 2017). These embeddings are added together along with the position and segmentation embeddings defined in the original BERT paper.

The pre-training steps of BERT are partly based on tasks defined earlier, namely, masked language model (MLM) and next sentence prediction (NSP). However, in case of MLM the masked word can be from different roles and turns. Similarly, the sentence pairs in NSP can span across multiple roles and turns. Thus, both MLM and NSP will drive better understanding of the conversation structure.

Task #3: Middle Sentence Prediction

In addition to NSP and MLM we have also introduced a new task, namely, middle sentence prediction (MSP). As the name suggests, we choose any two alternate turns (both coming from the same role) and try to predict the middle turn (different role). Similar to the NSP task, MSP is also converted into a binary classification problem where 50% of the time the actual middle turn is chosen and for the rest of the time another turn from a different conversation is picked randomly (while maintaining the role). In this way, the model should be able to understand the conversation structure better that will also help in applications like automated response generation.

4 Experiments

The data for all the experiments presented here are from conversations between customer service agents and existing/new customers who contact the customer service for their order related issues. The conversations between customer and agents are divided into sessions which we merged together to generate a single conversation. For BERT model pre-training we have randomly selected 100,000 chats of various different topics. These conversations are of varying number of turns and tokens (as can be seen in Table 2). For text normalization following steps are carried out (a) lower casing, (b) replacing entities like number, customer, city and state names, url, date, ticket number etc. by their corresponding tokens and finally (c) removing empty spaces, multiple punctuation and special characters. We have restricted the number of words in the vocabulary to 30,000 (same as what was considered in the original BERT paper (Devlin et al., 2019)).

4.1 Pre-training

We use the base configuration (Devlin et al., 2019) for pre-training, $i.e.$, number of layers, $L = 12$, hidden dimension, $H = 768$ and number of self-attention heads, $A = 12$. Both turn and role are projected into the same hidden dimension H as used for segments and positions. There are only two words in the role vocabulary ("A" and "C") whereas we have taken 128 as the turn vocabulary dimension. The maximum sequence length used in all the examples is 128.

We have created two different pre-training datasets from the 100,000 chats. The first one does not contain any MSP task and has 1.7 million examples for NSP and MLM tasks created by having a duplication factor of 5 (for random masking and

Turn	Role	Text
0	**Agent**	Please proceed with your query.
1	**Customer**	1want my order delay for one day. On the date of 26 nov
2	**Agent**	I certainly understand your concern. Let me check that.
3	**Customer**	Okk
4	**Agent**	Thanks for waiting. On checking details ...

Table 1: Sample conversation and the corresponding text, roles and turns

Statistics	Turn	Token
Minimum	1	1
Maximum	195	8409
Mean	13	23
50^{th} percentile	11	14
75^{th} percentile	17	26
90^{th} percentile	25	52

Table 2: Statistics of Turns and Tokens in the pre-training dataset

binary label selection). The second dataset has all possible (∼1M) MSP examples (no duplication) along with NSP and MLM examples created with a duplication factor of 2 (678k) resulting in 1.77 million data points. Thus, the two datasets are similar in size but having a totally different distribution of task types. We use a batch size of 32 for all examples and train for 3 epochs (∼170k steps), which takes around 80 hours in a 12 GB Tesla GPU machine.

The corresponding MLM and NSP task performance for these datasets are shown in Table 3. The effect of adding the role and turn is clear in the next sentence prediction with a much better accuracy. It can also be seen that adding MSP task increases (next or masked) sentence prediction accuracy. Addition of role and turn on the other hand has little effect on the MLM accuracy, mostly because these masked words are more dependent on surrounding words (rather than the roles or turns of the surrounding words).

Once we have pre-trained BERT models (with roles and turns) we can use the representation of [CLS] (from the top layer or from multiple layers, (Devlin et al., 2019)) as a representation (embedding) of the entire conversation. This representation (a vector of length H, 768 in the present case) then can be used for many potential downstream predictions as it has captured the entire conversation in a fixed length vector. Here we apply our pre-trained BERT models for clustering and automated response generation. For all the applications, including MSP, we use the representation of the [CLS] token at the top layer as an embedding. In addition, we also fine tune the BERT model for intent prediction.

4.2 Conversation Clustering

The data for conversation clustering consist of a different set of 50,000 conversations (again taken from conversations between customers and agents). Each conversation has a labeled intent (based on agents' tagging of the corresponding issue) and the distribution of these intents in the dataset is shown in Table 4. Each conversation is converted into a fixed length feature vector (dimension 768) using the pre-trained models described in the previous section. A t-SNE (van der Maaten and Hinton, 2008) plot of this dataset is shown in Fig. 2 where interactions amongst different intents and existence of multiple intents in a conversation are captured to a certain extent. For example, most of the intents have some overlap with "others" intent and "status" (order) is closely related to "delivery". Also, conversations with "others" tag and falling in the range of component-1 > 0 and component-2 > 10 seem to have no overlap with any of the other existing intents. A sample of 10 conversations from this region shown below confirms that:

- check my last order i want to know about which battery inbuilt
- can u update the name from [[name]] [[name]] to [[name]] [[name]] in the invoice ?

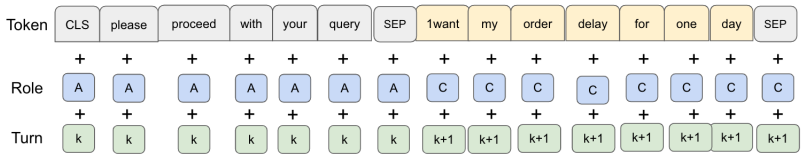

Figure 1: Conversation segments and corresponding BERT input representation using tokens, roles (A - agents, C - customer) and turns (k is the turn number varying between 1 and #turns).

Type	MLM Accuracy	NSP Accuracy
Set-1, no role and turn	82.4%	89.0%
Set-1, with role and turn	83.5%	96.4%
Set-2 (MSP), with role and turn	81.6%	97.5%

Table 3: Pre-training test results on the two different datasets

Intent	Percentage
Cancel	11.2
Delivery	16.5
Return & Refund	18.3
Status Inquiry	22.1
Others	31.9

Table 4: Distribution of pre-defined intents in the clustering dataset

Figure 2: Clustering of conversations by t-SNE (van der Maaten and Hinton, 2008) applied on conversation embeddings

Feature	Accuracy
tf-idf based feature	31.07%
word2vec based feature	33.28%
BERT embedding (no MSP)	44.08%
BERT embedding (with MSP)	42.98%

Table 5: Accuracy of k-means clustering of conversations

- [[order-id]] item missing product
- i do not want to change the address i want to change the payment method
- what is mean by no cost emi ?
- hi . i want to remove my default mobile num [[phone]] and change to [[phone]] i have the option to change email but not mobile
- i want to buy screen guard with phone but u charge extra delivery charges for screen guard yes
- call me back
- what is the offer for this phone i am not able to understand hello can you type you there
- ji mujhe emi pe phone lena h

where the last sentence has both English and Hindi words.

Although tags are not always very accurate and there can be multiple intents in a single conversation, we apply k-means algorithm (with 5 clusters) on the BERT embedding and calculate the accuracy. We also extract tf-idf based features (uni-grams and bi-grams) from the conversations (without taking role and turn information into account) and apply k-means algorithm with 5 clusters. Once cluster numbers are obtained for individual data points we apply Hungarian algorithm (Papadimitriou and Steiglitz, 1998) to map cluster number to intents (class names) and compute the accuracy. Instead of clustering all 50000 data points we randomly sample 10000 data points 10 times and apply k-means algorithm 5 times (for both tf-idf and BERT based feature representation) and compute the average accuracy. The results are shown in Table 5 where it is clear that BERT based feature (on an unseen dataset) has resulted in a much better accuracy underscoring the efficiency of this approach of embedding a conversation.

4.3 Conversation Classification

For classification we consider another set of 40,000 conversations (different from what is used in pre-training or clustering) with intents (labels) provided by the customer service agents. The distribution of these intents in this data is similar to what is shown in Table 4. We fine-tune the pre-trained BERT model for 32,000 data points and apply on the rest 8,000 examples. We use the same vocabulary of pre-training (with 30,000 words) for tokenization with maximum sequence length of 128. We fine tune the BERT model for 5 epochs with a batch size of 32 and learning rate of 2×10^{-5}. For comparison, we consider a linear SVM model with tf-idf based features. Hyper-parameters and ranges shown in Table 6 are considered for grid search. The results are presented in Table 7. It can be seen that BERT fine-tuned model achieves comparable accuracy (slightly higher) on completely unseen data.

hyperparameter	range
maximum document frequency	0.5, 0.75, 1.0
n-gram range	1, 2, 3
idf usage	True, False
tf-idf norm	L1, L2
α	10^{-4}, 10^{-5}, 10^{-6}
Regularization	L2, elasticnet

Table 6: Hyperparameter set for SVM classifier

Model	Accuracy
Linear SVM	72.04%
BERT fine-tuned	72.13%
BERT fine-tuned (with MSP)	72.24%

Table 7: Accuracy of different classifiers

4.4 Conversation Generation

The final example shows another application of BERT based features for automatic dialogue generation. As discussed previously, sequence-to-sequence (or $seq2seq$) models are not naturally amenable to accommodate conversation contexts and various approaches have been tried in the past. We try to generate response tokens r_t^k for turn k by maximizing the probability of response $\mathbf{r}^k = \{r_1^k, r_2^k, \ldots, r_T^k\}$

$$p(\mathbf{r}^k|\mathbf{i}^k) = \prod_{t=1}^{T} p(r_t^k|r_1^k, \ldots, r_{T-1}^k, \mathbf{i}^k, \mathbf{c}^{1:k-1}) \quad (2)$$

where $\mathbf{i}^k$ and $\mathbf{c}^{1:k-1}$ are the input and context for the kth turn, respectively. Since the context contains everything that has happened so far in the conversation, $i.e.$, $\mathbf{c}^{1:k-1} = \{\mathbf{i}^1, \mathbf{r}^1, \mathbf{i}^2, \mathbf{r}^2, \ldots, \mathbf{i}^{k-1}, \mathbf{r}^{k-1}\}$ it is an ever growing list and difficult to encode in a fixed length vector. In this work, we convert a variable length context into a fixed length feature vector using a pre-trained BERT model, i.e.,

$$\mathbf{c}^{1:k-1} = BERT(\{\mathbf{i}^1, \mathbf{r}^1, \mathbf{i}^2, \mathbf{r}^2, \ldots, \mathbf{i}^{k-1}, \mathbf{r}^{k-1}\}) \in \mathbb{R}^H \quad (3)$$

where H is the embedding dimension in BERT model.

Figure 3 shows the schema of applying context embedding. The model is based on an encoder-decoder pair modified for context embedding. At k-th turn, the context embedding represents turns 1 to $k-1$ that is used as an initial hidden state to the encoder (after a linear transformation). Next, tokens of the k-th turn are fed into the encoder one-by-one and the corresponding encoder outputs are recorded. The final encoder output (h_5 in Fig. 3) is concatenated with the context embedding (again with another linear transformation) and used as the initial state of the decoder. Following Bahdanau style attention (Bahdanau et al., 2015) the decoder state is compared with encoder outputs to compute attentions weights that are applied on the encoder outputs to get 'context vector'. This context vector is concatenated subsequently with the $k+1$-th turn tokens before given as input to the decoder to generate decoder outputs. Although not used in this work, context embedding can also be included in the attention weight calculation.

The data for conversation modeling is also taken from prior customer interaction with agents. However, we have considered only conversations for a specific issue, $i.e.$ 'status check'. We have manually extracted 3,872 conversations with 14,978 turns (data points) that have only this intent and no other additional intents displayed in the same conversation. The median number of turns in these conversations is 6 (75^{th} percentile is 8 and 90^{th} percentile is 15). We fixed the encoder sequence length to 32 and decoder sequence length to 128 (90^{th} percentile is 128). The context (which is a cumulative of previous turns) that is passed to the BERT model ideally should be less than 128 (the maximum sequence length considered in the BERT model). However, in our dataset the maximum number of tokens in a context is 363 while 99^{th}

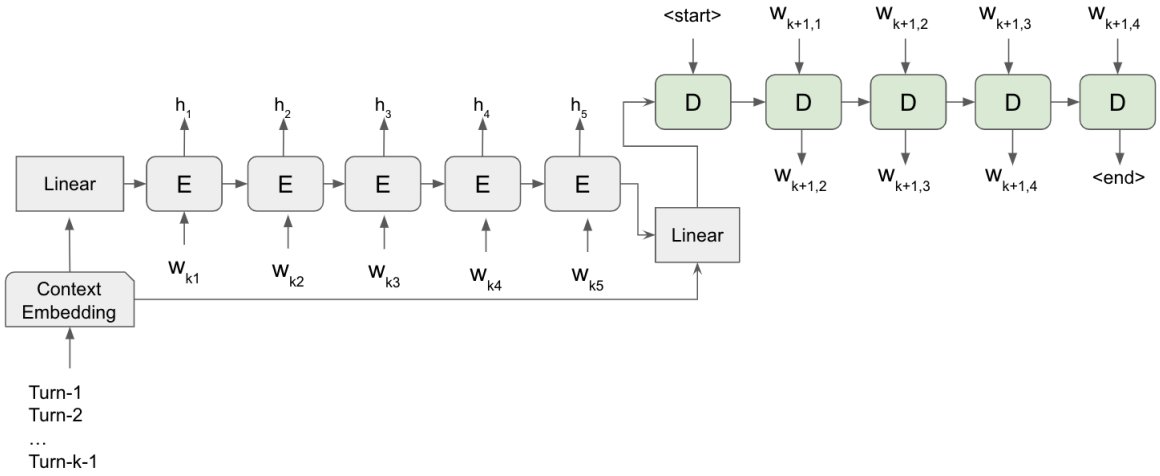

Figure 3: Encoder-decoder model with context embedding. Context embedding is used to set the initial hidden state of both the encoder (E) and decoder (D).

percentile is 109. Thus, for less than 1% of the cases the context will be truncated before being evaluated by BERT model.

We have used GRU for encoder and decoder (single layer) with a hidden dimension of 256 (same for the token embedding dimension). The inputs are reversed before given as input to the encoder. Before passing the context embedding (vector of dimension 768) to the encoder and decoder linear layers with ReLU activation (projecting 768 to 256) are used. In case of decoder there is an additional linear layer (with ReLU activation) that acts on the concatenated vector of last encoder hidden state and context vector (i.e., of dimension $768 + 256 = 1024$).

We have used 80% of the data for training and 20% for validation. All the model components are trained by Adam optimizer (Kingma and Ba, 2014) with default values and batch size of 16. Model performance is evaluated by BLEU score (Papineni et al., 2002) where the validation data is evaluated at the end of every epoch and tested for improving BLEU score. If the score does not improve for 4 consecutive epochs training is terminated. Table 8 shows the BLEU scores (BLEU-2 and BLEU-3 indicate BLEU scores for bi-grams and tri-grams) with and without conversation embeddings. The best rating is obtained when MSP task was not included in BERT pre-training which is not intuitive. However, both conversation embedding based models result in a better BLEU score than vanilla seq2seq model.

Model	BLEU-2	BLEU-3
no context embedding	0.2284	0.2037
with MSP	0.2354	0.2116
without MSP	**0.2403**	**0.2177**

Table 8: BLEU-2 and BLEU-3 for conversation response generated by different seq2seq models

5 Conclusion

We have introduced an embedding (representation) of a conversation (or conversation segment) by augmenting role and turn information to word tokens and utilizing BERT for pre-training. This pre-trained model can be used either to generate features from new conversations or can be fine-tuned further on specific tasks. In this work we have explored both the options. Pre-trained model based conversation features are used for (a) conversation clustering and (b) for representing contexts in a conversation for predicting the next response. In case of clustering we show that embedding based features result in higher accuracy when compared to tf-idf based features. Similarly, for conversation modeling embedding feature based context representation drove higher BLEU score when compared to a vanilla seq2seq model without any contextual information. We also fine-tune a pre-trained model for conversation classification on new dataset and obtain accuracy similar to what is given my a linear SVM model trained on tf-idf based features. With these examples we show the general applicability

of the current approach on modeling various tasks involving conversation data.

References

Shubham Agarwal, Ondrej Dusek, Ioannis Konstas, and Verena Rieser. 2018. Improving context modelling in multimodal dialogue generation. In *INLG*.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

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

Rafael E. Banchs. 2012. Movie-dic: a movie dialogue corpus for research and development. In *ACL*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. *EMNLP*.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Andrew M. Dai, Christopher Olah, and Quoc V. Le. 2015. Document embedding with paragraph vectors. In *NIPS Deep Learning Workshop*.

Jacob Devlin, Ming-Wei, Chang Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. *Proceedings of the Third International Workshop on Paraphrasing*.

J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. 2017. Convolutional sequence to sequence learning.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *ACL*.

Ryan McDonald John Blitzer and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. *Proceedings of the 2006 conference on empirical methods in natural language processing*, page 120–128.

Yoon Kim. Convolutional Neural Networks for Sentence Classification.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization.

Matt J. Kusner, Yu Sun, Nicholas I. Kolki, and Kilian Q. Weinberger. 2015. From Word Embeddings To Document Distances. *ICML*.

Quoc V. Le and Tomas Mikolov. 2014. "Distributed Representations of Sentences and Documents. *ICML*, pages 1188–1196.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Christopher Joseph Pal. 2018. Towards deep conversational recommendations. *ArXiv*, abs/1812.07617.

Li-Ping Liu, Francisco J. R. Ruiz, Susan Athey, and David M. Blei. 2017. Context selection for embedding models. In *NIPS*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Sch. 2008. *Introduction to Information Retrieval*. Cambridge.

Shikib Mehri, Evgeniia Razumovskaia, Tiancheng Zhao, and Maxine Eskenazi. 2019. Pretraining methods for dialog context representation learning.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2017. Coherent dialogue with attention-based language models. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *ICLR Workshop*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *NIPS*, pages 3111–3119.

Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. *HLT-NAACL*.

Shervin Minaee and Zhu Liu. 2017. Automatic Question-Answering Using A Deep Similarity Neural Network.

Christos H Papadimitriou and Kenneth Steiglitz. 1998. Combinatorial optimization: algorithms and complexity. *Courier Corporation.*

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. *EMNLP*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *xxx*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *EMNLP*.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *ArXiv*, abs/1508.07909.

Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016a. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*.

Iulian Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2016b. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing With Compositional Vector Grammars. *ACL*, pages 1–47.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jing Nie. 2015a. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. *ArXiv*, abs/1507.02221.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jing Nie, Jianfeng Gao, and William B. Dolan. 2015b. A neural network approach to context-sensitive generation of conversational responses. In *HLT-NAACL*.

Hui Xu Su, Xiaoyu Shen, Pengwei Hu, Wenjie Li, and Yun Chen. 2018. Dialogue generation with gan. In *AAAI*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, pages 3104–3112.

Stefanie Tellex, Boris Katz, Jimmy Lin, Aaron Fernandes, and Gregory Marton. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the SIGIR Conference on Research and Development in Informaion Retrieval*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the ACL*, pages 384–394.

Oriol Vinyals and Quoc V Le. 2015. A neural conversational model. *Proceedings of the 31st International Conference on Machine Learning*, pages 3104–3112.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. *NAACL*.