INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

BERT-DST: Scalable End-to-End Dialogue State Tracking with Bidirectional Encoder Representations from Transformer

Guan-Lin Chao[1], Ian Lane[1,2]

[1] Electrical and Computer Engineering, Carnegie Mellon University, USA
[2] Language Technologies Institute, Carnegie Mellon University, USA
`{guanlinchao, lane}@cmu.edu`

Abstract

An important yet rarely tackled problem in dialogue state tracking (DST) is scalability for dynamic ontology (*e.g., movie, restaurant*) and unseen slot values. We focus on a specific condition, where the ontology is unknown to the state tracker, but the target slot value (except for *none* and *dontcare*), possibly unseen during training, can be found as word segment in the dialogue context. Prior approaches often rely on candidate generation from n-gram enumeration or slot tagger outputs, which can be inefficient or suffer from error propagation. We propose BERT-DST, an end-to-end dialogue state tracker which directly extracts slot values from the dialogue context. We use BERT as dialogue context encoder whose contextualized language representations are suitable for scalable DST to identify slot values from their semantic context. Furthermore, we employ encoder parameter sharing across all slots with two advantages: (1) Number of parameters does not grow linearly with the ontology. (2) Language representation knowledge can be transferred among slots. Empirical evaluation shows BERT-DST with cross-slot parameter sharing outperforms prior work on the benchmark scalable DST datasets Sim-M and Sim-R, and achieves competitive performance on the standard DSTC2 and WOZ 2.0 datasets.

Index Terms: dialogue state tracking, belief tracking, task-oriented dialogue systems, BERT

1. Introduction

Dialogue state tracking (DST), a core component in today's task-oriented dialogue systems, maintains user's intentional states through the course of a dialogue. The dialogue states predicted by DST are used by the downstream dialogue management component to produce API calls to a backend database and generate responses to the user [1]. A dialogue state is often expressed as a collection of slot-value pairs. The set of slots and their possible values are often domain-specific, defined in a *domain ontology*. Many state-of-the-art approaches operate on a fixed ontology, by performing classification over a predefined set of slot values or iteratively scoring slot-value pairs from the ontology [2, 3, 4]. However, such models can be inefficient or infeasible when the ontology is dynamic (*e.g.*, *movie, restaurant*), innumerable (*e.g.*, *time*), or simply not exposed by an external database [5, 6].

In this paper, we study this practical problem in DST – scalability with unknown ontology and unseen slot values, with a specific condition: the target slot value (except for *none* and *dontcare*) always appears as word segment in the dialogue context. Previous approaches often require a *candidate list*, which can be an exhaustive list of n-grams in the dialogue context or slot tagging outputs from a separate language understanding (LU) module [5, 7, 8]. Using n-gram candidate generation might be inefficient because the number of candidates the DST scorer needs to iterate through is proportional to the length of the dialogue context. Although the LU-generated candidate list can be shorter, the DST scorer cannot recover from missing target candidates incurred by LU errors [5, 6].

We introduce BERT-DST[1], a scalable end-to-end dialogue state tracker, based on the BERT model [9], that directly predicts slot values from the dialogue context with no dependency on candidate generation. In our framework, BERT is adopted to produce contextualized representations of dialogue context (Section 3.1), which are used to by the classification and span prediction modules to predict the slot value as *none*, *dontcare* or a text span in the dialogue context (Section 3.2, 3.3). The advantages of using BERT as dialogue context encoder include: (1) The contextualized word representations are suitable for extracting slot values from semantic context. (2) Pre-trained on large-scale language modeling datasets, BERT's word representations are good initialization to be fine-tuned to our DST problem (Section 2). Moreover, we employ parameter sharing in the BERT dialogue encoder across all slots, which reduces the number of model parameters. Contextualized language representation can also benefit from more training examples of other slots (Section 3.5). To prevent overfitting, we apply the slot value dropout technique, originally introduced in [10, 5, 6]. This step is critical for extracting unseen slot values from their contextual patterns (Section 3.6). Through empirical evaluation, we show the effectiveness of BERT-DST with parameter sharing and slot value dropout. BERT-DST achieves state-of-the-art performance of 80.1% and 89.6% joint goal accuracy on the benchmark scalable DST datasets Sim-M and Sim-R [11], and competitive results on the standard DSTC2 [12] and WOZ 2.0 [13] datasets (Section 5).

2. BERT

In this section, we briefly describe BERT [9] and how its architecture can be applied to scalable DST in our framework.

BERT is a multi-layer bidirectional Transformer encoder [14], which is a stack of multiple identical layers each containing a multi-head self-attention and a fully-connected sub-layer with residual connections [15]. The input to BERT is a sequence of tokens, which can be concatenation of a pair of sentences. The input sequence is prepended by a special `[CLS]` token whose final hidden state is used as the aggregate sequence representation. The final hidden states of the other tokens are used as token-level representations. Besides word embedding and positional embedding used in the original Transformer model, BERT's input layer adds an additional segment embedding to differentiate tokens from the pair of sentences.

[1]Code is available at https://github.com/guanlinchao/bert-dst

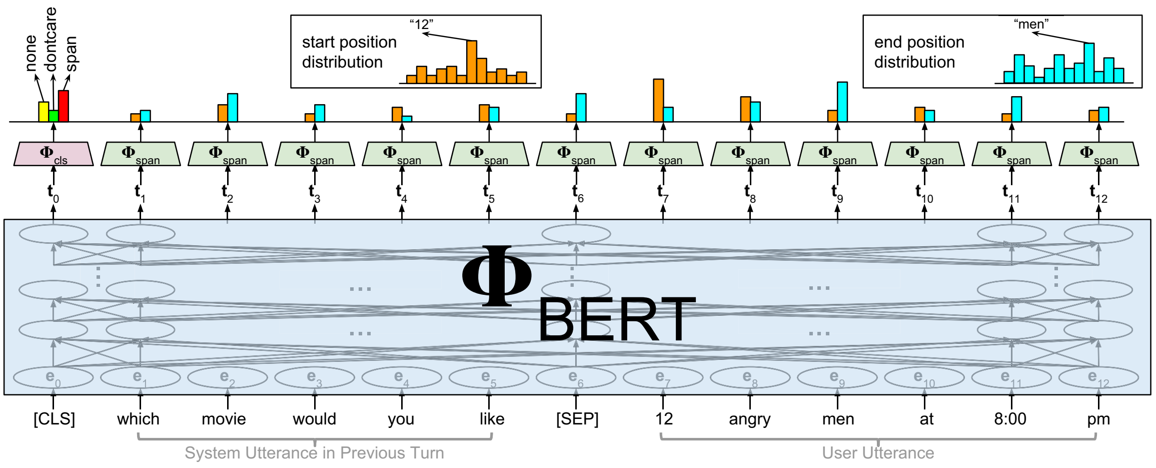


Figure 1: *Architecture of the proposed BERT-DST framework. The diagram is color-coded such that modules with the same color share the same parameters. For each user turn, BERT-DST takes as input the recent dialogue context (system utterance in previous turn and the user utterance), and outputs turn-level dialogue state. BERT dialogue context encoding module Φ_{BERT} (blue) produces contextualized sentence-level and token-level representations of the dialogue context. The per-slot classification module Φ_{cls} (red) uses the sentence-level representation to generate a categorical distribution over three types of slot values {none, dontcare and span}. The per-slot span prediction module Φ_{span} (green) gathers the token-level representations and output the start and end positions (span) of the slot value. Note that the dialogue context encoding module Φ_{BERT} allows parameter sharing across all slots.*

To learn bidirectional contextualized representations and inter-sentence relationship, BERT model is pre-trained on two unsupervised language modeling tasks: masked language modeling [16] and next sentence prediction, using the BooksCorpus [17] and the English Wikipedia corpora. The procedures of language model pre-training are detailed in [9]. With extra projection layers and fine-tuning the deep structure, BERT has been successfully applied to various tasks such as reading comprehension, named entity recognition, sentiment analysis, *etc*.

Our proposed application of BERT to scalable DST is in spirit similar to the Stanford Question Answering Dataset (SQuAD) task [18]. In SQuAD, the input is a question and a reading passage. If the reading paragraph contains the answer to the question, the output is a segment of text from the paragraph, represented by its span (start and end positions). Otherwise, the model should output *unanswerable*. Similarly, in our targeted case of scalable DST, a slot's value can be *none*, *dontcare*, or a word segment from the dialogue context. Our proposed framework uses BERT's contextualized sentence-level and token-level representations to determine the type of slot value (*none*, *dontcare*, or *span*), and the span of the specified slot value from the dialogue context. Using BERT as dialogue context encoder provides the following advantages. The contextualized word representations are suitable for extracting slot values from contextual patterns. With large-scale language model pre-training, BERT's word representations are good initialization to be fine-tuned to our DST problem.

3. BERT-DST

In this section, we describe in detail the proposed BERT-DST framework, as shown in Figure 1. For each user turn, BERT-DST takes the recent dialogue context as input and outputs the turn-level dialogue state. First, the dialogue context input is encoded by the BERT-based encoding module to produce contextualized sentence-level and token-level representations. The sentence-level representation is then used by the classification module to generate a categorical distribution over three types of slot values: *none*, *dontcare* or a *span* from the input. The span prediction module gathers the token-level representations and outputs the slot value's start and end positions. Finally, an update mechanism is used to track dialogue states across turns.

3.1. Dialogue Context Encoding Module

The dialogue context encoding module is based on BERT. We use the system utterance from the previous turn and the current turn user utterance as dialogue context input, represented as a token sequence in BERT's input format. The first token is `[CLS]`, followed by the tokenized system utterance, `[SEP]`, and tokenized user utterance. Let $[x_0, x_1, \cdots, x_n]$ denote the input token sequence. BERT's input layer embeds each token x_i into an embedding $\mathbf{e}_i$, which is the sum of three embeddings:

$$\begin{aligned}\text{BERTinput}(x_i) &= E_{\text{tok}}(x_i) + E_{\text{seg}}(i) + E_{\text{pos}}(i) \\ &= \mathbf{e}_i \in \mathbb{R}^d, \;\; \forall\, 0 \le i \le n \qquad (1)\end{aligned}$$

where $E_{\text{tok}}(x_i)$ is WordPiece embedding [19] for token x_i, $E_{\text{seg}}(i) \in \{\mathbf{e}_{\text{first}}, \mathbf{e}_{\text{second}}\}$ is segment embedding whose value is determined by whether the token belongs to the first or second sentence, and $E_{\text{pos}}(i)$ is positional embedding [20] for the i-th token.

The embedded input sequence $[\mathbf{e}_0, \cdots, \mathbf{e}_n]$ is then passed to BERT's bidirectional Transformer encoder, whose final hidden states are denoted by $[\mathbf{t}_0, \cdots, \mathbf{t}_n]$.

$$\begin{aligned}&[\mathbf{t}_0, \cdots, \mathbf{t}_n] = \text{BiTransformer}([\mathbf{e}_0, \cdots, \mathbf{e}_n]) \\ &\mathbf{t}_i \in \mathbb{R}^d, \;\; \forall\, 0 \le i \le n \qquad (2)\end{aligned}$$

The contextualized sentence-level representation $\mathbf{t}_0$, *i.e.*, the final state corresponding to the `[CLS]` token, is passed to the classification module. The contextualized token-level representations $[\mathbf{t}_1, \cdots, \mathbf{t}_n]$ are used by the span prediction module.

The parameters in the dialogue context encoding module, denoted by Φ_{BERT}, are initialized from a pre-trained BERT checkpoint and then fine-tuned on our DST dataset.

3.2. Classification Module

The classification module's input is the sentence-level representation $\mathbf{t}_0$ from the dialogue context encoding module. For each slot $s \in S$ in the collection of all informable slots S, the classification module predicts the value of s to be one of the three classes $\{$*none*, *dontcare*, *span*$\}$ using linear projection and softmax.

$$\mathbf{a}^s = \mathbf{W}^s_{cls}\mathbf{t}_0 + \mathbf{b}^s_{\text{cls}} = [a^s_{\text{none}}, a^s_{\text{dontcare}}, a^s_{\text{span}}] \in \mathbb{R}^3 \quad (3)$$

$$\mathbf{p}^s = \text{softmax}(\mathbf{a}^s) = [p^s_{\text{none}}, p^s_{\text{dontcare}}, p^s_{\text{span}}] \quad (4)$$

$$\text{slot\_value}^s = \text{argmax}_{c \in \{\text{none, dontcare, span}\}}(p^s_c) \quad (5)$$

The per-slot classification parameters, denoted by $\Phi^s_{\text{cls}} = \{\mathbf{W}^s_{\text{cls}}, \mathbf{b}_{\text{cls}}\}$, are trained from scratch on our DST dataset.

3.3. Span Prediction Module

For each slot $s \in S$, the span prediction module takes as input the token-level representations $[\mathbf{t}_1, \cdots, \mathbf{t}_n]$ from the dialogue context encoding module. Each token representation $\mathbf{t}_i$ is linearly projected through a common layer whose output values α^s_i and β^s_i correspond to start and end positions respectively. Softmax is then applied to the position values to produce a probability distribution over all tokens, by which the slot value span (start and end positions) of s can be determined.

$$[\alpha^s_i, \beta^s_i] = \mathbf{W}^s_{\text{span}}\mathbf{t}_i + \mathbf{b}^s_{\text{span}} \in \mathbb{R}^2, \forall\, 1 \leq i \leq n \quad (6)$$

$$\mathbf{p}^s_\alpha = \text{softmax}(\alpha^s) \quad (7)$$

$$\mathbf{p}^s_\beta = \text{softmax}(\beta^s) \quad (8)$$

$$\text{start\_pos}^s = \text{argmax}_i(p^s_{\alpha,i}) \quad (9)$$

$$\text{end\_pos}^s = \text{argmax}_i(p^s_{\beta,i}) \quad (10)$$

The per-slot span prediction parameters, denoted by $\Phi^s_{\text{span}} = \{\mathbf{W}^s_{\text{span}}, \mathbf{b}^s_{\text{span}}\}$, are trained from scratch on our DST dataset.

3.4. Dialogue State Update Mechanism

To track dialogue states across turns, we employ a rule-based update mechanism. In each turn, if the model's turn prediction for a slot is *dontcare* or a specified value (*i.e.*, any value other than *none*), it will be used to update the dialogue state. Otherwise, the dialogue state of the slot remains the same as the previous turn.

3.5. Parameter Sharing

Although our classification and span prediction modules are slot-specific, we notice that the contextualized representations generated by the dialogue context encoding module can be shared among slots; *i.e.*, we can apply parameter sharing in the dialogue context encoding module across all slots. Sharing dialogue context encoder parameters Φ_{BERT} across all slots not only drastically reduces the number of model parameters. It also allows knowledge transfer among slots, which may potentially benefit contextual relation understanding. In the following sections, we call the joint architecture of slot-specific BERT-DST models as **BERT-DST_SS** and the BERT-DST model with encoding module parameter sharing as **BERT-DST_PS**.

3.6. Slot Value Dropout

Slot value dropout, or targeted feature dropout, was originally proposed to address the under-training problem of contextual features in slot-filling [10]. The problem happens when models tend to overfit to frequent slot values in training data instead of learning contextual patterns, which adversely harms the performance on OOV slot values. To improve the robustness for unseen slot values, in the training phase, we replace each of the target slot value tokens by a special `[UNK]` token at a certain probability.

4. Experiments

We evaluate our models using *joint goal accuracy* [12], a standard metric for DST. The model's prediction has to jointly match all the informable slot labels to be considered correct.

4.1. Datasets

We evaluate our models on four benchmark datasets: Sim-M, Sim-R [11], DSTC2 [12] and WOZ 2.0 [13].

Sim-M and Sim-R are specialized for scalable DST, which contain human-paraphrased simulated dialogues in the movie and restaurant domains. They have span annotations for all specified slot values (*i.e.* values other than *none* and *dontcare*) in the system and user utterances. In the event that the target slot value has multiple spans in the dialogue context, we use the span of the last occurrence as reference. The prevalence of out-of-vocabulary (OOV) values in Sim-M's *movie* and Sim-R's *restaurant_name* slots makes them particularly challenging and suitable for scalable DST evaluation.

DSTC2 and WOZ 2.0 are standard benchmarks for task-oriented dialogue systems, which are both in the restaurant domain and share the same ontology. In DSTC2, automatic speech recognition (ASR) hypotheses of user utterances are provided to assess DST models' robustness against ASR errors, so we use the top ASR hypothesis for validation and testing. In WOZ 2.0, the user interface is typing and collection of user utterances exhibit higher degree of lexical variation. ASR errors and flexible language use can cause problems in defining slot value spans, which is basis for our targeted scalable DST condition. The problem arises when erroneous ASR hypotheses do not contain a user's intended specified value or the user uses a *creative* expression in which a clear boundary of a value's span can be hard to define. For example, "My wife thinks she likes international but I don't want to take out a loan." is annotated with `price_range=cheap`. Such challenging instances in DSTC2 and WOZ 2.0 set the performance upper bound for our proposed scalable DST framework.

Note that in the evaluation, we do not apply an output canonicalization step to handle other possible valid variations of span. Therefore our model's predicted slot value span has to exactly match the label span to be considered correct.

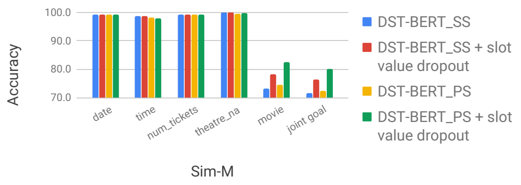
4.2. Implementation Details

We use the pre-trained *[BERT-Base, Uncased]* model which has 12 hidden layers of 768 units and 12 self-attention heads for lower-cased input text[2]. The span prediction loss for $\{$*none, dontcar*$\}$ slots is set to zero. The total loss is defined as $(0.8\mathcal{L}^{\text{xent}}_{\text{cls}} + 0.1\mathcal{L}^{\text{xent}}_{\text{span\_start}} + 0.1\mathcal{L}^{\text{xent}}_{\text{span\_end}})$, where $\mathcal{L}^{\text{xent}}$ denotes the cross entropy loss for the corresponding prediction target. We update all layers in the model using ADAM optimization [21] with an initial learning rate $2e^{-5}$ and early stopping on the validation set. During training, we use 30% dropout rate [22] on the dialogue context encoder outputs. Various rates of slot value dropout are experimented and reported in Section 5.

[2]https://github.com/google-research/bert

Table 1: *Comparison with prior approaches on Sim-M and Sim-R datasets (joint goal accuracy). * indicates statistically significant improvement over BERT-DST model (paired sample t-test; $p < 0.01$). † indicates the corresponding model should be considered as a kind of oracle because the candidates are ground truth slot-tagging labels, i.e. the targeted slot value is guaranteed to be in the candidate list and considered by DST.*

DST Models	Sim-M	Sim-R
DST + LU Candidates [7]	50.4%	87.1%
DST + Oracle Candidates† [5]	96.8%	94.4%
BERT-DST_SS	71.6%	87.4%
+ slot value dropout	76.3%*	87.6%
BERT-DST_PS	72.3%	88.6%*
+ slot value dropout	**80.1%***	**89.6%***



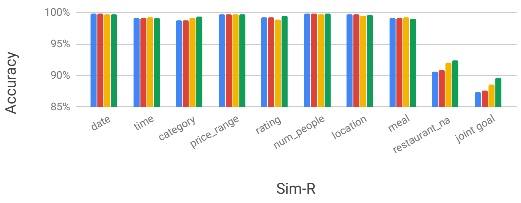


Figure 2: *Per-slot and joint goal accuracy of the proposed models on Sim-M and Sim-R datasets.*

5. Results and Discussion

Table 1 presents the performance of the proposed BERT-DST models compared to prior work on the scalable DST datasets Sim-M and Sim-R. In [7, 5], the DST component scores slot values from a candidate list, which is slot tagging predictions of a jointly-trained language understanding component (*DST + LU Candidates*), or the ground truth slot tagging labels (*DST + Oracle Candidates*). We compare our proposed model with the (*DST + LU Candidates*) baseline because in practice an oracle candidate list that always contains the target slot label is rarely available. On both Sim-M and Sim-R, BERT-DST_SS outperforms the baseline model. We attribute the performance gain to the effective contextualized representations obtained from the BERT dialogue encoding module. BERT-DST_PS with slot value dropout achieves further statistically significant improvement over BERT-DST_SS. The comparison of per-slot and joint goal accuracy of the different BERT-DST models is shown in Figure 2. We observe that it is mainly the slots with OOV values (*movie* for Sim-M and *restaurant_name* for Sim-R) that benefit from the encoder parameter sharing and slot value dropout techniques. The accuracy improvement on these bottleneck slots eventually leads to gain in the joint goal accuracy.

To investigate the effect of slot value dropout, we compare the performance with different slot value dropout probabilities of BERT-DST_PS on Sim-M and Sim-R datasets, as shown in Figure 3. While a proper selection of slot value dropout rate can result in slight improvement on Sim-R, the effect of slot value dropout is more pronounced on Sim-M. Because of the high OOV value rate of the *movie* slot (100% OOV in test set), higher slot value dropout rate can be helpful for extracting unseen slot values from contextual patterns.

Table 2 presents the performance of BERT-DST with prior approaches on the standard DSTC2 and WOZ 2.0 datasets. Our work is more comparable with the top group frameworks, which are also designed for scalable DST to handle unknown ontology. The middle group of models require a predefined ontology to perform classification or scoring over a predefined set of possible slot values. On DSTC2, BERT-DST_PS shows comparable performance with prior scalable DST models, although not as high as the state-of-the-art models. On WOZ 2.0, BERT-DST_PS achieves competitive results with state-of-the-art models, which demonstrates BERT-DST's capability in understanding sophisticated language. Note that it is not our goal to achieve state-of-the-art performance on the standard datasets. Instead, BERT-DST is tasked to handle unknown ontology and unseen slot values and does not require a separate candidate generation module.

Table 2: *Comparison with prior approaches on DSTC2 and WOZ 2.0 datasets (joint goal accuracy). We report the average and standard deviation of test set accuracy of 5 model runs with random training data shuffling and normal initialization on classification and span prediction weights.*

DST Models	DSTC2	WOZ 2.0
DST + LU Candidates [7]	67.0%	-
DST + n-gram Candidates [8]	68.2±1.8%	-
DST + Oracle Candidates [5]	70.3%	-
Pointer Network [6]	72.1%	-
Delex.-Based Model [2]	69.1%	70.8%
Delex. + Semantic Dict. [2]	72.9%	83.7%
Neural Belief Tracker [2]	73.4%	84.2%
GLAD [3]	74.5±0.2%	88.1±0.4%
StateNet [4]	**75.5%**	**88.9%**
BERT-DST_PS	69.3±0.4%	87.7±1.1%

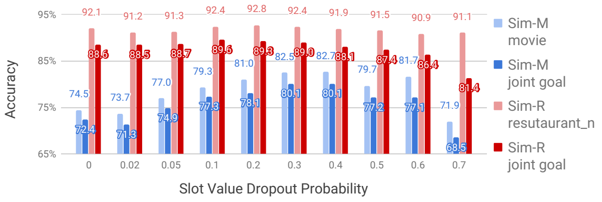


Figure 3: *Comparison of different slot value dropout probabilities of the BERT-DST_PS model on Sim-M and Sim-R datasets.*

6. Conclusions

We introduce BERT-DST, a scalable end-to-end dialogue state tracker to handle unknown ontology and unseen slot values. Not requiring candidate value generation, BERT-DST directly predicts slot values from the dialogue context. The key component is the BERT dialogue context encoding module which produces contextualized representations effective for extracting slot values from the contextual patterns. Empirical evaluation on Sim-M and Sim-R datasets shows the efficacy of the proposed BERT-DST model with the slot value dropout technique and encoder parameter sharing across all slots.

7. References

[1] S. Young, M. Gašić, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, "The hidden information state model: A practical framework for pomdp-based spoken dialogue management," *Computer Speech & Language*, 2010.

[2] N. Mrkšić, D. Ó. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[3] V. Zhong, C. Xiong, and R. Socher, "Global-locally self-attentive encoder for dialogue state tracking," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[4] L. Ren, K. Xie, L. Chen, and K. Yu, "Towards universal dialogue state tracking," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[5] A. Rastogi, D. Hakkani-Tür, and L. Heck, "Scalable multi-domain dialogue state tracking," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

[6] P. Xu and Q. Hu, "An end-to-end approach for handling unknown slot values in dialogue state tracking," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[7] A. Rastogi, R. Gupta, and D. Hakkani-Tur, "Multi-task learning for joint language understanding and dialogue state tracking," in *Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2018.

[8] R. Goel, S. Paul, T. Chung, J. Lecomte, A. Mandal, D. Hakkani-Tur, and A. A. AI, "Flexible and scalable state tracking framework for goal-oriented dialogue systems," in *NeurIPS Conversational AI workshop*, 2018.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Computing Research Repository*, vol. arXiv:1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[10] P. Xu and R. Sarikaya, "Targeted feature dropout for robust slot filling in natural language understanding," in *Annual Conference of the International Speech Communication Association*, 2014.

[11] P. Shah, D. Hakkani-Tür, G. Tür, A. Rastogi, A. Bapna, N. Nayak, and L. Heck, "Building a conversational agent overnight with dialogue self-play," *Computing Research Repository*, vol. arXiv:1801.04871, 2018. [Online]. Available: http://arxiv.org/abs/1801.04871

[12] M. Henderson, B. Thomson, and J. D. Williams, "The second dialog state tracking challenge," in *Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2014.

[13] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gasic, L. M. R. Barahona, P.-H. Su, S. Ultes, and S. Young, "A network-based end-to-end trainable task-oriented dialogue system," in *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] W. L. Taylor, "cloze procedure: A new tool for measuring readability," *Journalism Bulletin*, 1953.

[17] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *International Conference on Computer Vision (ICCV)*, 2015.

[18] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[19] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *Computing Research Repository*, vol. arXiv:1609.08144, 2016. [Online]. Available: http://arxiv.org/abs/1609.08144

[20] A. Vaswani, S. Bengio, E. Brevdo, F. Chollet, A. Gomez, S. Gouws, L. Jones, Ł. Kaiser, N. Kalchbrenner, N. Parmar *et al.*, "Tensor2Tensor for neural machine translation," in *Conference of the Association for Machine Translation in the Americas*, 2018.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2014.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 2014.