Assessing Word embedding Performance in Multi-Domain Dialogue State Tracking

Sanchit Nevgi
College of Information and Computer Sciences
University of Massachusetts, Amherst

Abstract

Recent advancements have made Task-oriented dialogue systems easily accessible, by reducing reliance on hand-crafted features. Dialogue State Tracking is a core component of these systems. Traditional approaches fall short in tracking unknown slot values and cannot easily adapt to new, unseen domains. Recent efforts in dialogue state tracking have progressed towards open vocabulary and generation-based approaches, where the models can generate slot values from the dialogue history itself. In this paper, we assess the quality of various word embeddings when used in the utterance encoder of a task-oriented system. Our results show that pre-trained word vectors perform better in the zero-shot approach.

1 Introduction

Advancements in neural architectures [**??**] have led to improvements in dialogue systems A dialogue system is an entity you can interface with via text/voice that has certain capabilities. These capabilities can be broadly classified into three groups [**?**].

Question Answering: QA agents allow users to query large-scale Knowlege Bases (KB-QA) or document collection in natural lanaguage (text-QA). In the real-world, text-QA agents are most commonly used in search engines such as Google, Bing, while KB-QA are widely used in voice assistants.

Task-oriented systems: Task-oriented systems work towads a well-specified goal, such as movie ticket booking, usually in a multi-turn fashion. The dialogue system keeps track of the dialogue state, uses information supplied by user to constrain search, prompts user for required information necessary task completion.

Social chatbots: The agent needs to converse seamlessly with users, provide useful recommendations. A good dialogue agent should demonstrate emotional connect with the user. For example, showing excitement when user shares good news, empathy when user is feeling sad, etc.

A dialogue system consists for 4 components —

1. Natural Language Understanding
2. Dialogue State Tracking
3. Dialogue Policy
4. Natural Language Generation

Dialogue state tracking (DST) is a core component in task-oriented dialogue systems. The goal of DST module is to extract user goals/intentions expressed during conversation and to encode them as a compact set of dialogue states, i.e., a set of slots and their corresponding values [**?**]. Traditionally DST modules used a fixed ontology, where the slot-values were known prior. The task was framed as as a classification problem. However, these models do not scale well and are unable to handle unseen domains. Recent approaches follow a generation approach to handle Out Of Vocabulary terms.

2 Definitions

Ontology: An ontology defines the domain of a task-oriented system. It is a structured representation of an external database. The ontology defines all entity attributes called *slots* and all the possible *values* for each slot. In practice, a full ontology is difficult to obtain in advance. Some slots such as dates, names are unbounded. Slots are classified as *Informable* and *Requestable* slots. An *informable* slot (area, price-range) allows the user to constrain the search query, while *requestable* slots (phone-number, operational-time) represent additional information that a user can request from the dialogue system.

Dialogue Act: The semantic representation of each turn of a dialogue is known as a *dialogue act*. A dialogue act has an implicit *intent*, *slot-value* pairs or both.

Slot-filling dialogues: The user and the system converse with the goal of *slot-filling*. The system must collect all the necessary information from the user to formulate an appropriate query. In the movie ticket booking domain, some examples of slots are `movie-name`, `time`, `price`, `location`. Slots are *informable* if the user can provide the *slot-value*, used to constrain the search; or slots are *requestable*, where the user can ask the value of the slot from the system (eg `phone_number`)

Knowledge Base: A Knowledge Base (KB) is a structured database conisiting of facts in the form of subject-predicate-object triples (s, r, t), where $s, t \in \mathcal{E}$ are entities and $r \in \mathcal{R}$ are predicates or relations.

3 Related Work

Learning task-oriented dialogs end-to-end requires defining a user simulator and a dialogue agent. **?** define baseline tasks to measure neural methods's performances. Traditional dialog systems in task-oriented dialogs require lot of domain-specific handcrafting, which makes it difficult to scale. These limitations are mitigated in end-to-end trained systems. [**?**] shows that, compared to handcrafted slot-filling baseline, end-to-end dialog

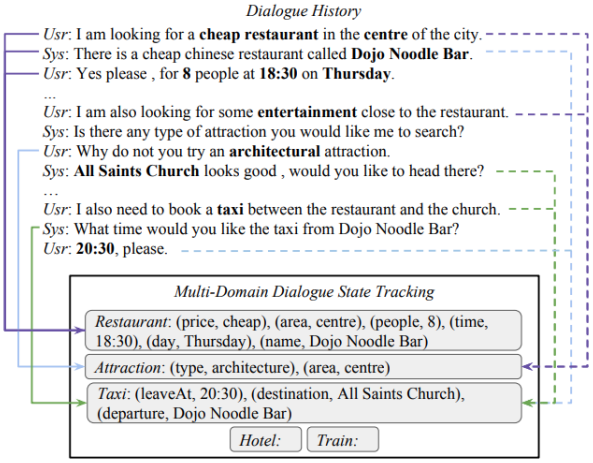

Figure 1: Sample dialogues and corresponding Dialogue State Tracking

system based on *Memory networks* [**?**] can reach promising, but imperfect performance and can even perform non-trivial operations.

Successful goal-oriented dialog systems model coversation as partially observable Markov deicsion processes. Require handcrafted features for state and action space representations, restricting to narrow domains. To measure the performance of task-oriented dialog systems, 5 common tasks have been proposed. They are,

Task 1 - Issuing API calls

Task 2 - Updating API calls

Task 3 - Displaying options

Task 4 - Providing extra information

Task 5 - Full dialogs

In [**?**], the authors use a restaurant KB, which has `name`, `cuisine`, `location`, `price range`, `address`, `phone number`. Each example in the dataset comprises of dialog utterances from user and bot, as well as API calls and the resulting facts. They split the KB into two, disjoint sets of *cuisines*, and *locations*, to test the models capability to handle Out-of-Vocabulary scenario. The dialog systems are evaluated in a ranking, not a generation setting. At each turn of dialog, they test whether they can predict bot utterances by selecting candidates.

Further, the model is evaluated on real concierge service where the dialogs are shorter, and the vocabulary more diverse. Some of the findings are that the set of user requests is much wider compared to dataset, from managing restaurant reservation to asking for recommendations. Also, the users do not stay focused on the request. The facts about restaurants are not structured like KB and finally the users and operators make typos, spelling and grammatical erros. Some approaches to modeling task-oriented systems are outlined below,

1. Rule-based systems - Use hand-crafted rules such as word matches, positions in dialog, entity detections, dialog state.

2. Information Retrieval models: Use a TF-IDF matching for each possible candidate response, compute the matching score between input and response. The score is computed as a TF-IDF weighted cosine similarity between the bag-of-words of input and response.

3. Nearest neighbour - Finds the most similar conversation in the training set and use that response. Word overlap as scoring method. Sorted by decreasing co-occurence frequency.

4. Supervised embedding models - Predict next response given the previous conversation. Scored for input x as $f(x, y) = (Ax)^T By$. Trained with margin ranking loss

5. Memory Networks - We know that ssing word embeddings fails when the dialogue has open-ended entities, since embeddings use approximate word match. They cannot handle OOV words. The solution is to match type features and augment vocabulary with 7 words (cuisine, location, etc).

Neural Belief Tracker: **?** proposes an end-to-end trained DST module known at the *Neural Belief Tracker*. A *belief tracker*, estimates the user's goal at every step of the dialog. They mitigate two issues with traditional approaches to building DST, namely that NLU models require large amount of training data and hand-crafting lexicons for capturing liguistic variation in users's language is cumbersome.

Using recent advances in *representation learning*, Neural Belief Tracking (NBT) framework reasons over pre-trained word vectors, learning to compose them into distributed representations of user utterances and dialogue context. The *dialogue state tracking* component serves to interpret user input and update the *belief state*, which is the system's internal representation of teh state of the conversation. The dialogue system is supported by *domain ontology*, which describes the range of user intents the system can process. The ontology defines a collection of slots and the values each slot can take.

The task is non-trivial due to lexical variation, dynamics of context and noisy Automated Speech Recognition (ASR) output. Traditional approaches use separate modules to handle lexical variability in single dialogue turn. However the turn-level SLU and cross-turn DST can be coalesced into a single model, but they rely on manually constructed semantic dictionaries.

Some systems use template-based matching systems while others train independent binary models that decide if slot-value pair was expressed in user utterance. SLU has been treated as sequence labeling problem.

The proposed Neural Belief Tracking model uses pre-trained vectors. The input consists of the system dialogue acts preceding the user input, the user utterance, a single slot-value pair it needs to make a decision about. To perform belief tracking, NBT model iterates over all candidate slot-value pairs and decides which ones have just been expressed by the user. *"I'm looking for good pizza"* entails FOOD=ITALIAN.

Representation Learning module: The vector embeddings for unigram, bigram and trigram are concatenated and projected to a fixed-

size representation. The canditate slot-value pair and user utterance interact through the *semantic decoding* module. The slot and value representation are concatenated, projected to the same size at user utterance vector and finally a similarity score is computed.

To conclude, the NBT couples Spoken Language Understanding and Dialogue State Tracking without relying on hand-crafted semantic lexicons. Further, the model performance improves with the semantic quality of underlying word vectors.

Trade DST: The Trade model [**?**] comprises of three components — an utterance encoder, a slot gate, and a slot generator. Instead of predicting the probability of every predfined ontology term, the model directly generates the slot values. The utterance encoder encodes the dialogue utterances into a sequence of fixed-length vectors.

The *State Generator* generates slot values using text from the input source using a copy mechanism. **?** use a soft-gated pointer generator copy over index-based copying as this helps to extract slot values that are synonyms of values present in the ontology (eg *inexpensive* $\rightarrow$ *cheap*). The decoder uses a GRU to predict the value for each *(domain, slot)* independently. The input to the decoder is a summed vector representation of the domain and slot.

The *Slot gate* is a three-way classifier that maps the dialogue history context vector, the current turn vector representation and for each *(domain, slot)* pair, it independently classifies as one of *(not-mentioned, dont-care, ptr)*, where if the *slot* is not mentioned or no preference is specified we ignore it, otherwise we pass the repsentation to the state generator to extract its value.

Non-Autoregressive DST Generation based approaches to dialogue state tracking fall short in two ways; (1) they do not allow models to explicitly learn signals across domains and slots to detect potential dependencies among *(domain, slot)* pairs and (2) existing models follow auto-regressive approaches which incur high time cost when the dialogue evolves over multiple domains and multiple turns. **?** propose a non-autoregressive framework which factors in potential dependencies among the domain, slot pairs.

The NADST model consists of 3 components, an encoder, a fertility decoder and a state decoder. Like earlier models, the *encoders* encode sequences of dialogue history, delexicalized dialogue history, and domain and slot tokens into continuous representations. The *fertility decoder* learns potential dependencies between the domain, slots pairs using an attention mechanism. The embedding weights are shared across all three components. The encoders use token-level embedding and positional encoding to encode the input dialogue history and (domain, slot) pairs into continuous representations. The encoded domains and slots are then input to stacked self-attention and feed-forward network to obtain relevant signals across dialogue history and generate a fertility for each domain, slot pair. The predicted fertilities are used to form an input sequence to the state decoder for nonautoregressive decoding. The output from the state decoder is used as a query to attend on this memory and copy tokens from the dialogue history to generate a dialogue state.

4 Dataset

MultiWOZ is a large fully-labeled collection of human-human written conversations spanning over multiple domains and topics [**?**]. It has 3,406 single-domain dialogues and 7,032 multi-domain dialogues consisting of 2-5 domains. There are on average 13.5 *turns per dialogue*. The average sentence lengths are 11.75 and 15.12 for users and wizards respectively. Each dialogue consists of a pre-defined user goal, and multiple user and system utterances as well as an implicit belief state. The data spans 7 domains — `Restaurant`, `Hotel`, `Taxi`, `Train`, `Attraction`, `Hospital`, `Police`.

MultiWOZ 2.1 is a recent effort to address the shortcomings of `v2.0`. The original utterances were re-annotated completely to account for the noisy annotations. Additionally, the

Figure 2: MultiWOZ dataset statistics

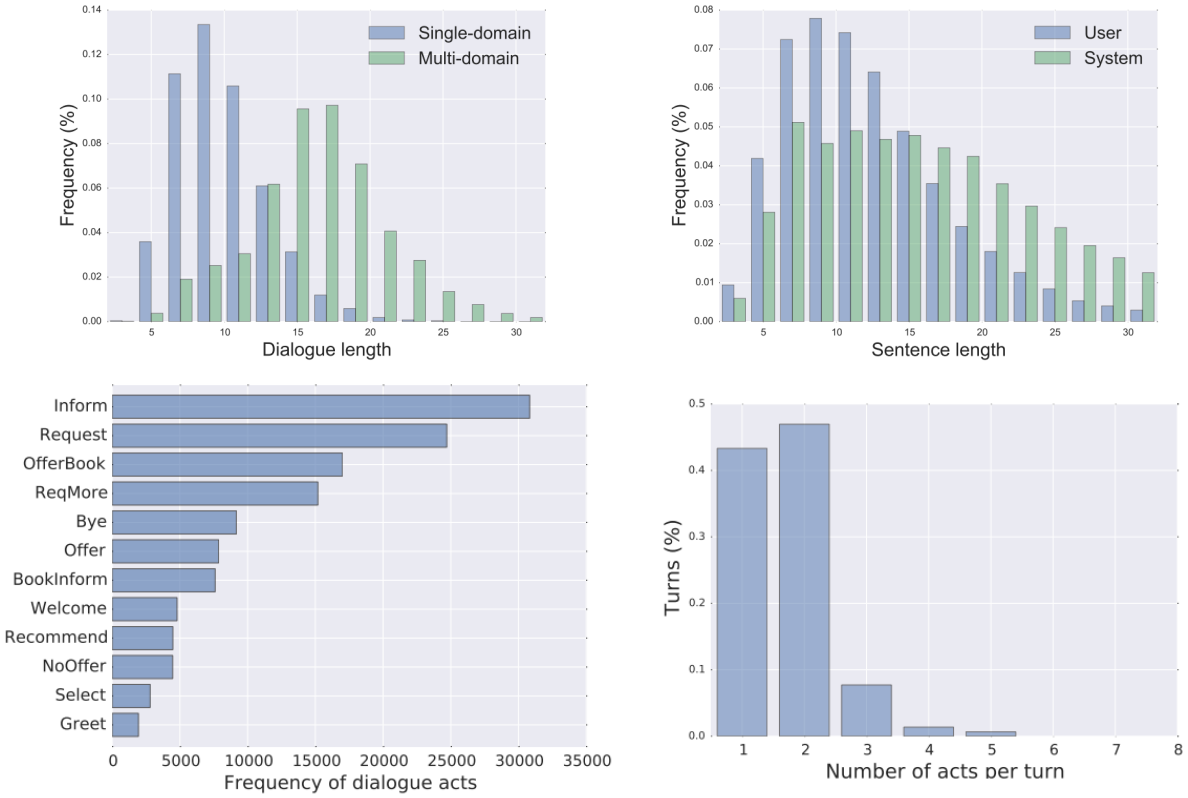

Table 1: Summary of the MultiWOZ dataset

Domain	Slots	Split		
		Train	Dev	Test
Attraction	area, name, type	3,381	416	394
Hotel	area, bookday, num_people, internet, name, parking, pricerange, stars, type	3,103	484	494
Restaurant	area, bookday, num_people, book_time, pricerange	2,717	401	395
Taxi	arriveby, departure, destination, leaveat	3,813	438	437
Train	arriveby, bookpeople, day, departure, destination, leaveat	1,654	207	195

user dialog acts were annotated, which were missing from the original dataset. The correction impacted 32% of state annotations across 40% of dialogue turns.

4.1 Data Collection Process

Data is collected by simulating tourist desk information agents and users by Mechanical Turkers. To facilitate data collection, [**?**] provide an easy-to-operate interface. An intial set of trials were used to identify a set of workers that perform annotation well. These workers were then asked to annotate the real dialogues. A large number of workers are used to facilitate diversity.

User side: A goal is conveyed to a user and the various constraints are communicated gradually to avoid information overload and mimic a natural conversation.

System side: The system agent or *wizard* is asked to perform a role of a clerk and provide information to the user. They have access to a fixed ontology for reference. The wizard either

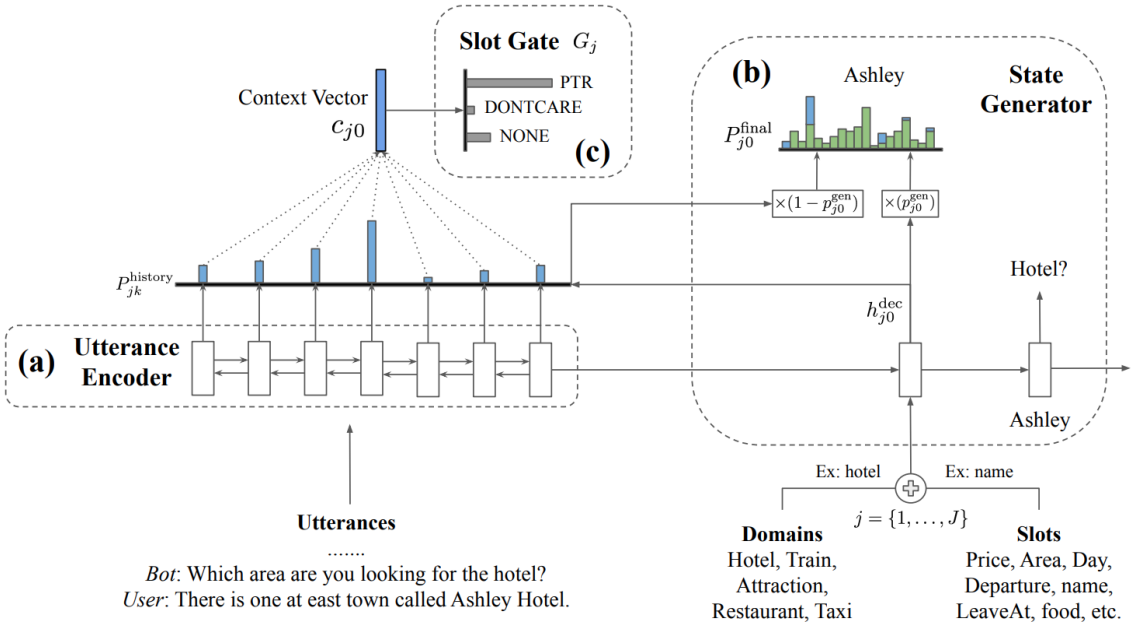

Figure 3: TRADE model architecture. We modify the utterance encoder component

gives the result of the user supplied query or requests for more information necessary for the search query. The wizards communicate via a web form, which is also used to implicitly track the belief state of the dialogue.

The dataset comes with a pre-defined train/test/dev split of `8k/1k/1k` dialogues, for easier reproducibility, and we continue to use the same split in all our experiments. Since, some of the dialogues ended up with the user goal unmet; the validation and test sets only contain dialogues that were successful.

4.2 Data Pre-processing

We perform the following pre-processing steps on the data. We *delexicalize* the data using the scripts provided in the MultiWOZ repository. The entities mentioned in the dialogue acts, such as *restaurant-name*, are encoded as one-hot vectors. We generate a vocabulary from the input and a mapping of index to words is generated for easier embedding. Further, the dialogues are divided into train/val/test split and the text is lowercased. Further, the *hospital* and *police* domain are removed from the dataset as they are only present in the train, and have few dialogues.

5 Approach

Let $X = \{(U_t, R_t)\}$ be the set of user utterances and system utterances up to turn t. We encode the dialog history using a bi-directional gated recurrent units (GRU) to get the dialogue history. Encoding the dialogue history enables us to handle multi-turn dialogue.

ConceptNet is a Knowledge Base consisting of 8m facts and relations [**?**]. ConceptNet Numberbatch are vectors built using an ensemble that combines data from ConceptNet, word2vec, GloVe, and OpenSubtitles 2016, using a variation of a method known as *retrofitting* [**?**]. The embedding of a node are created to minimize the vectors's euclidean distance to its graph neighbours as well as to the distance of equivalent pre-trained embeddings such as GloVE [**?**], fastText. The resulting embeddings capture the graph knowledge.

5.1 Metrics

For our experiments, we use the following metrics,

1. **Joint Accuracy**: The joint accuracy is the ratio of dialogue turns where **all** slots are classified correctly. This metric gives

Table 2: Experimental Results

Model	Metrics		
	Joint Accuracy	Slot Accuracy	F1
Trade DST*	0.480	0.969	0.87
NADST	0.469	0.971	0.880
NADST + ConceptNet	0.460	0.966	0.835
NADST + GloVE	0.477	0.992	0.890

us a good estimate of the robustness of the overall system.

2. **Slot Accuracy**: This is the ratio of correctly classified slots over all dialogue turns.

3. **F1 score**: The F1 score computed over all the slots.

6 Experiments

The models are built in PyTorch 1.5. We use the *Adam* optimizer with $betas = (0.9, 0.98)$ and a learning rate $\alpha = 0.001$. The dropout is set to $dropout = 0.2$. Further we use a batch size of 32 in all our experiments. The model is trained on a TitanX 12GB gpu for 30 epochs or until convergence. We use a a train/dev/test split of `8k/1k/1k` examples, divided according to class proportions. Additionally, we leave out the *hospital* and *police* domain, since they constitute $\leq 10\%$ of the train data and are absent from the test and dev set.

For our baseline, we use the TradeDST model architecture, consisting of an utterance decoder, slot gate and state generator; described previously. The utterance encoder uses an embedding layer of $d = 256$, initialized uniformaly at random, which is also jointly trained. The TradeDST model is trained on MultiWOZ 2.0 dataset, an earlier version compared to our dataset. The annotations in MultiWOZ 2 was dominated by `none` class. *Models trained on this dataset were biased towards predicting `none` class, resulting in a higher score (as seen from the baseline TradeDST model). This imbalance was corrected in MultiWOZ 2.1, which makes it more challenging. The results are summarized in Table 2.

Next, we initalize the embedding layer with GLoVe CommonCrawl embeddings 300-d. These embeddings are trained on 42B tokens with approximately 1.9M vocabulary size. We filter the vocabulary based on our target dataset. The model is trained with the same hyperparamters as our baseline model. We see that there is an improvement in the model performance across all the metrics.

Finally, we induce external knowledge into the utterance encoder through the ConceptNet numberbatch embeddings. However, there is no significant improvement over the baseline, which we hypothesize could be due to 2 reasons. (1) ConceptNet is a huge knowledge graph with over 8m entities and relations. The majority of these relations are unhelpful for our dialogue system. (2) ConceptNet is still vastly incomplete for it to be useful.

7 Conclusion

Task-oriented dialogue systems continue to be widely used. A dialogue state tracker keeps track of the dialogue state at each turn. In this paper, we evalutate the performance of various word embeddings in the utterance encoder module, in the scope of multi-domain dialogue state tracking. We see from our experiments that using pre-trained embeddings such as GloVE improve the performance of the model in zero-shot learning in terms of the slot extraction. However, we also notice incorporating Knowledge Base embeddings in the model doesn't help and an alternative is needed to induce external knowledge.

References

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

Jianfeng Gao, Michel Galley, and Lihong Li. Neural approaches to conversational ai. *Found. Trends Inf. Retr.*, 13:127–298, 2019.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2019.

Antoine Bordes and Jason Weston. Learning end-to-end goal-oriented dialog. *ArXiv*, abs/1605.07683, 2016.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NIPS*, 2015.

Nikola Mrksic, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve J. Young. Neural belief tracker: Data-driven dialogue state tracking. *ArXiv*, abs/1606.03777, 2016.

Hung T. Le, Richard Socher, and Steven C. H. Hoi. Non-autoregressive dialog state tracking. *ArXiv*, abs/2002.08024, 2020.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *EMNLP*, 2018a.

Hugo Liu and Push Singh. Conceptnet — a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22:211–226, 2004.

Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, 2016.

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

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling, 2018b.

Seokhwan Kim, Michel Galley, Chulaka Gunasekara, Sungjin Lee, Adam Atkinson, Baolin Peng, Hannes Schulz, Jianfeng Gao, Jinchao Li, Mahmoud Adada, Minlie Huang, Luis Lastras, Jonathan K. Kummerfeld, Walter S. Lasecki, Chiori Hori, Anoop Cherian, Tim K. Marks, Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, and Raghav Gupta. The eighth dialog system technology challenge, 2019.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Xiang Li, Yaoqin Zhang, Zheng Zhang, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. Convlab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019a.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines, 2019.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Xiang Li, Yaoqin Zhang, Zheng Zhang, Jin chao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. Convlab: Multi-domain end-to-end dialog system platform. In *ACL*, 2019b.

Jinchao Li, Baolin Peng, Sungjin Lee, Jianfeng Gao, Ryuichi Takanobu, Qi Zhu, Minlie Huang, Hannes Schulz, Adam Atkinson, and Mahmoud Adada. Results of the multi-domain task-completion dialog challenge. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, Eighth Dialog System Technology Challenge Workshop*, February 2020.

Young-Bum Kim, Sungjin Lee, and Karl Stratos. Onenet: Joint domain, intent, slot prediction for spoken language understanding. *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 547–553, 2017.