

Dialogue Transformers

Vladimir Vlasov,^{*} Johannes E. M. Mosig,[†] and Alan Nichol[‡]

Rasa

(Dated: November 19, 2019)

We introduce a dialogue policy based on a transformer architecture [1], where the self-attention mechanism operates over the sequence of dialogue turns. Recent work has used hierarchical recurrent neural networks to encode multiple utterances in a dialogue context, but we argue that a pure self-attention mechanism is more suitable. By default, an RNN assumes that every item in a sequence is relevant for producing an encoding of the full sequence, but a single conversation can consist of multiple overlapping discourse segments as speakers interleave multiple topics. A transformer picks which turns to include in its encoding of the current dialogue state, and is naturally suited to selectively ignoring or attending to dialogue history. We compare the performance of the Transformer Embedding Dialogue (TED) policy to an LSTM and to the REDP [2], which was specifically designed to overcome this limitation of RNNs.

I. INTRODUCTION

Conversational AI assistants promise to help users achieve a task through natural language. Interpreting simple instructions like *please turn on the lights* is relatively straightforward, but to handle more complex tasks these systems must be able to engage in multi-turn conversations.

The goal of this paper is to show that the transformer architecture [1] is more suitable for modeling multi-turn conversations than the commonly used recurrent models.

We intentionally choose simple architectures in order to compare the basic mechanisms that are in the heart of the sequence encoding. The proposed TED architecture should be thought of as a candidate building block for use in developing state-of-the-art architectures in various dialogue tasks.

Each utterance in a conversation does not necessarily have to be a response to the most recent utterance by the other party. Groz and Sidner [3] consider conversations as an interleaved set of *discourse segments*, where a discourse segment (or topic) is a set of utterances which directly respond to each other. These sequences of turns may not directly follow one another in the conversation. An intuitive example of this is the need for sub-dialogues in task-oriented dialogue systems. Consider this conversation:

```
BOT: Your total is $15.50 - shall I charge
      the card you used last time?
USER: Do I still have credit on my account
      from that refund I got?
BOT: Yes, your account is $10 in credit.
USER: Ok, great.
BOT: Shall I place the order?
USER: Yes.
BOT: Done. You should have your items tomorrow.
```

a. Dialogue Stacks The assistant's question *Shall I place the order?* prompts the return to the task at hand: completing a purchase. One model is to think of these sub-dialogues as existing on a stack, where new topics are pushed on to the stack when they are introduced and popped off the stack once concluded.

In the 1980s, Groz and Sidner [3] argued for representing dialogue history as a stack of topics, and later the RavenClaw [4] dialogue system implemented a dialogue stack for the specific purpose of handling sub-dialogues. While a stack naturally allows for sub-dialogues to be handled and concluded, the strict structure of a stack is also limiting. The authors of RavenClaw argue for explicitly tracking topics to enable the contextual interpretation of the user intents. However, once a topic has been popped from the dialogue stack, it is no longer available to provide this context. In the example above, the user might follow up with a further question like *so that used up my credit, right?*. If the topic of refund credits has been popped from the stack, this can no longer help clarify what the user wants to know. Since there is in principle no restriction to how humans revisit and interleave topics in a conversation, we are interested in a more flexible structure than a stack.

b. Recurrent Neural Networks A common choice in recent years has been to use a recurrent neural network (RNN) to process the sequence of previous dialogue turns, both for open domain [5, 6] and task-oriented systems [7]. Given enough training data, an RNN should be able to learn any desired behaviour. However, in a typical low-resource setting where no large corpus for training a particular task is available, there is no guarantee that an RNN will in fact learn to generalize these behaviours. Previous work on modifying the basic RNN structure to include inductive biases for this behaviour into a dialogue policy was conducted by Vlasov et al. [2] and Sahay et al. [8]. These works aim to overcome a feature of RNNs which is undesirable for dialogue modeling. RNNs by default consume the *entire* sequence of input elements to produce an encoding, unless a more complex structure like a Long Short-Term Memory (LSTM) cell is trained on sufficient data to explicitly learn that it should 'forget'

^{*} vlasov@rasa.com

[†] j.mosig@rasa.com

[‡] alan@rasa.com

parts of a sequence.

c. Transformers The transformer architecture has in recent years replaced recurrent neural networks as the standard for training language models, with models such as Transformer-XL [9] and GPT-2 [10] achieving much lower perplexities across a variety of corpora and producing representations which are useful for a variety of downstream tasks [11, 12]. In addition, transformers have recently shown to be more robust to unexpected inputs (such as adversarial examples) [13]. Intuitively, because the self-attention mechanism preselects which tokens will contribute to the current state of the encoder, a transformer can ignore uninformative (or adversarial) tokens in a sequence. To make a prediction at each time step, an LSTM needs to update its internal memory cell, propagating this update to further time steps. If an input at the current time step is unexpected, the internal state gets perturbed and at the next time step the neural network encounters a memory state unlike anything encountered during training. **A transformer accounts for time history via a self-attention mechanism, making the predictions at each time step independent of each other.** If a transformer receives an irrelevant input, it can ignore it and use only the relevant previous inputs to make a prediction.

Since a transformer chooses which elements in a sequence to use to produce an encoder state at every step, we hypothesise that it could be a useful architecture for processing dialogue histories. The sequence of utterances in a conversation may represent multiple interleaved topics, and the transformer’s self-attention mechanism can simultaneously learn to disentangle these discourse segments and also to respond appropriately.

II. RELATED WORK

a. Transformers for open-domain dialogue Multiple authors have recently used transformer architectures in dialogue modeling. Henderson et al. [14] train response selection models on a large dataset from Reddit where both the dialogue context and responses are encoded with a transformer. They show that these architectures can be pre-trained on a large, diverse dataset and later fine-tuned for task-oriented dialogue in specific domains. Dinan et al. [15] used a similar approach, using transformers to encode the dialogue context as well as background knowledge for studying grounded open-domain conversations. Their proposed architecture comes in two forms: a retrieval model where another transformer is used to encode candidate responses which are selected by ranking, and a generative model where a transformer is used as a decoder to produce responses token-by-token. **The key difference with these approaches is that we apply self-attention at the discourse level, attending over the sequence of dialogue turns** rather than the sequence of tokens in a single turn.

b. Topic disentanglement in task-oriented dialogue Recent work has attempted to produce neural architectures for dialogue policies which can handle interleaved discourse segments in a single conversation. Vlasov et al. [2] introduced the Recurrent Embedding Dialogue Policy (REDP) architecture. The ablation study in this work highlighted that the improved performance of REDP is due to an attention mechanism over the dialogue history and a copy mechanism to recover from unexpected user input. This modification to the standard RNN structure enables the dialogue policy to ‘skip’ specific turns in the dialogue history and produce an encoder state which is identical before and after the unexpected input. Sahay et al. [8] develop this line of investigation further by studying the effectiveness of different attention mechanisms for learning this masking behaviour.

In this work we do not augment the basic RNN architecture but rather replace it with a transformer. By default, an RNN processes every item in a sequence to calculate an encoding. REDP’s modifications were motivated by the fact that not all dialogue history is relevant. Taking this line of reasoning further, we can use self-attention in place of an RNN, so there is no *a priori* assumption that the whole sequence is relevant, but rather that the dialogue policy should select which historical turns are relevant for choosing a response.

III. TRANSFORMER AS A DIALOGUE POLICY

We propose the Transformer Embedding Dialogue (TED) policy, which greatly simplifies the architecture of the REDP. Similar to the REDP, we do not use a classifier to select a system action. Instead, **we jointly train embeddings for the dialogue state and each of the system actions by maximizing a similarity function between them.** At inference time, the current state of the dialogue is compared to all possible system actions, and the one with the highest similarity is selected. A similar approach is taken by [14, 16, 17] in training retrieval models for task-oriented dialogue.

Two time steps (i.e. dialogue turns) of the TED policy are illustrated in Figure 1. A step consists of several key parts.

a. Featurization Firstly, the policy featurizes the user input, system actions and slots.

The TED policy can be used in an end-to-end or in a modular fashion. The modular approach is similar to that taken in POMDP-based dialogue policies [18] or Hybrid Code Networks [7, 19]. An external natural language understanding system is used and the user input is featurized as a binary vector indicating the recognized intent and the detected entities. The dialogue policy predicts an action from a fixed list of system actions. System actions are featurized as binary vectors representing the action name, following the REDP approach explained in detail in [2].

By end-to-end we mean that there is no supervision

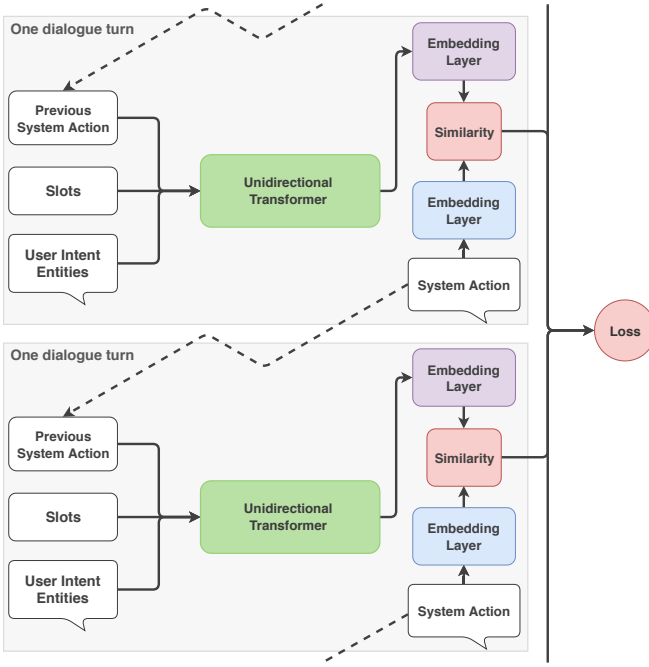


FIG. 1. A schematic representation of two time steps of the transformer embedding dialogue policy.

beyond the sequence of utterances. That is, there are no gold labels for the NLU output or the system action names. The end-to-end TED policy is still a retrieval model and does not generate new responses. In the end-to-end setup, user and system utterances are encoded as bag-of-words vectors.

Slots are always featurized as binary vectors, indicating their presence, absence, or that the value is not important to the user, at each step of the dialogue. We use a simple slot tracking method, overwriting each slot with the most recently specified value.

b. Transformer The input to the transformer is the sequence of user inputs and system actions. Therefore, we leverage the self-attention mechanism present in the transformer to access different parts of dialogue history dynamically at each dialogue turn. The relevance of previous dialogue turns is learned from data and calculated anew at each turn in the dialogue. Crucially, this allows the dialogue policy to take a user utterance into account at one turn but ignore it completely at another.

c. Similarity All dialogue contexts and system actions are embedded into a single semantic vector space and we use the dot-product loss [2, 14, 20] to maximize the similarity with the target label and minimize similarities with sampled incorrect ones. Since softmax cannot assign zeroes to its output we use 1.5-entmax [21] in the loss function, where correct examples are labeled as 1 and incorrect ones as 0. The global loss is an average of all loss functions from all time steps. At inference time, the dot-product similarity serves as a ranker for the next utterance retrieval problem.

In modular training, we use a balanced batching

strategy [22] to mitigate class imbalance as some system actions are far more frequent than others.

IV. EXPERIMENTS

The aim of our experiments is to analyze the performance of the transformer and an LSTM on multi-turn conversations. Specifically, we want to test the TED policy on the task of picking out relevant turns in the dialogue history for next action prediction. Therefore, we need a conversational dataset for which system actions depend on the dialogue history across several turns. This requirement precludes question-answering datasets such as WikiQA [23] as candidates for evaluation.

In addition, system actions need to be labeled to evaluate next action retrieval accuracy. Note, that metrics such as Recall@k [24] could be used on unlabeled data, but since typical dialogues contain many generic responses, such as “yes”, that are correct in a large number of situations, the meaningfulness of Recall@k is questionable. We therefore exclude unlabeled dialogue corpora such as the Ubuntu Dialogue Corpus [24] or MetalWOZ [25] from our experiments.

To our knowledge, the only publicly available dialogue datasets that might satisfy both our criteria are the REDP dataset [2], MultiWOZ [26, 27] and Google Taskmaster-1 [28].

All experiments are performed using the Rasa framework [19] and all code, hyperparameters, and data required to reproduce these experiments are available online at <https://github.com/RasaHQ/TED-paper>.

A. Comparing the end-to-end and modular approaches on MultiWOZ

As we described in Section III, the TED policy can either take the original user utterance, or an externally classified user intent as input. In the present section we establish that both approaches are valid, based on the recently released MultiWOZ 2.1 dataset [26, 27]. Unfortunately, we cannot use MultiWOZ to evaluate how well the TED policy handles dialogue complexity, since MultiWOZ is almost history independent, as we shall demonstrate here.

MultiWOZ 2.1 is a dataset of 10438 human-human dialogues for a Wizard-of-Oz task in seven different domains: hotel, restaurant, train, taxi, attraction, hospital, and police. In particular, the dialogues are between a user and a clerk (wizard). The user asks for information and the wizard, who has access to a knowledge base about all the possible things that the user may ask for, provides that information or executes a booking. The dialogues are annotated with labels for the wizard’s actions, as well as the wizard’s knowledge about the user’s goal after each user turn.

For our experiments, we split the MultiWOZ 2.1 dataset into a training and a test set of 7249 and 1812 dialogues, respectively. Unfortunately, we had to neglect 1377 dialogues altogether, since their annotations are incomplete.

1. End-to-end training

We begin our series of experiments with an end-to-end retrieval setup, where the user utterance is used directly as input to the TED policy, which then has to retrieve the correct response from a predefined list (extracted from MultiWOZ).

The wizard’s behaviour depends on the result of queries to the knowledge base. For example, if only a single venue is returned, the wizard will probably refer to it. We marginalize this knowledge base dependence by (i) delexicalizing all user and wizard utterances [29], and (ii) introducing status slots that indicate whether a venue is available, not available, already booked, or unique (i.e. the wizard is going to recommend or book a particular venue in the next turn). These slots are featurized as a 1-of-K binary vector.

We run experiments with and without the addition of status slots.

To compute the accuracy and F1 scores of the TED policy’s predictions, we assign the action labels (e.g. `request_restaurant`) that are provided by the MultiWOZ dataset to the output utterances, and compare them to the correct labels. If multiple labels are present, we concatenate them in alphabetic order to a single label.

Table I shows the resulting F1 scores and accuracies on the hold-out test set. The discrepancy between the F1 score and the accuracy stems from the fact that some labels, s.a. `bye_general`, occur very frequently (4759 times) compared to most other labels, s.a. `recommend_restaurant_select_restaurant` which only occurs 11 times.

The fact that accuracy and F1 scores are generally low compared to 1.0 stems from a deeper issue with the MultiWOZ dialog dataset. Specifically, because more than one particular behaviour of the wizard would be considered ‘correct’ in most situations, the MultiWOZ dataset is unsuitable for supervised learning of dialogue policies. Put differently, some of the wizards’ actions in MultiWOZ are not deterministic, but probabilistic. For example, it cannot be learned when the wizard should ask if the user needs anything else, since this is the personal preference of the people who take the wizard’s role. We elaborate on this and several other issues of the MultiWOZ dataset elsewhere.

Comparing row ‘end-to-end’ to row ‘end-to-end (with status slots)’ with `max_history = 10` in Table I, we see that introducing status slots raises the accuracy by 0.035, and the F1 score by 0.068, but scores remain low compared to 1.0. We shall explore the reason for these low scores in conjunction with analysing the results of the

model	max_history	accuracy	F1 score
end-to-end	10	0.61	0.21
	2	0.61	0.21
end-to-end (with status slots)	10	0.64	0.28
	2	0.62	0.24
modular	10	0.66	0.49
	2	0.65	0.45
modular (with status slots)	10	0.73	0.63
	2	0.69	0.55

TABLE I. Accuracy and F1 scores of the TED policy in end-to-end and modular mode, evaluated on the MultiWOZ 2.1 dataset. All scores concern prediction at the action level. The ‘with status slots’ tag means that status slots have been added, which is necessary to marginalize the dependence on the wizard’s knowledge base. While the modular architecture generally achieves higher scores, taking into account more than 2 steps in the dialogue history barely increases the performance in either case.

modular architecture predictions in the next section.

2. Modular training

We now repeat the above experiment, using the same subset of MultiWOZ dialogues, but now adopting the modular approach. We simulate an external natural language understanding pipeline and provide gold user intents and entities to the TED policy instead of the original user utterances. We extract the intents from the changes in the Wizard’s belief state. This belief state is provided by the MultiWOZ dataset in the form of a set of slots (e.g. `restaurant_area`, `hotel_name`, etc.) that get updated after each user turn. A typical user intent is thus `inform{"restaurant_area": "south"}`. The user does not always provide new information, however, so the intent might be simply `inform` (without any entities). If the last user intent of the dialog was uninformative in this way, we assume it is a farewell and thus annotate it as `bye`.

Using the modular approach instead of end-to-end learning roughly doubles the F1 score and also increases accuracy slightly, as can be seen in Table I. This is not surprising since the modular approach receives additional supervision.

Although the scores suggest that the modular TED policy performs better than the end-to-end TED policy, the *kinds* of mistakes made are similar. We demonstrate this with one example dialog from our test set, named SNG0253, that is displayed in Figure 2.

The second column of Figure 2 shows the end-to-end predictions. The two predicted responses are both sensible, i.e. the replies could have come from a human. Nevertheless, both results are marked as wrong, because according to the goal dialogue (first column), the first response should only have included its second sentence (`request_train`, but not

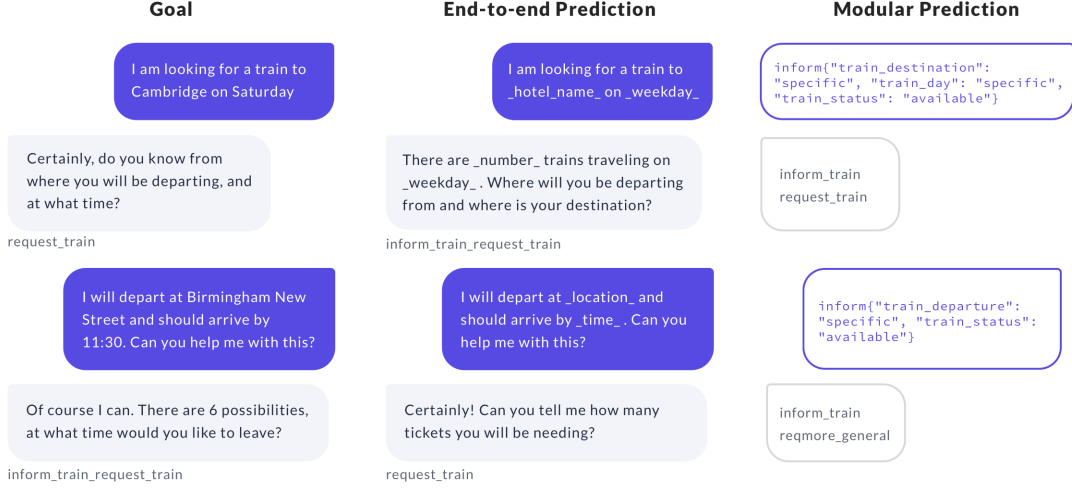


FIG. 2. MultiWOZ 2.1 dialogue SNG0253, as-is (first column), as predicted by the end-to-end TED policy (second column), and as sequence of user intents and system actions predicted by modular TED policy (third column). The two predictions are sensible, yet incorrect according to the labels. Furthermore, the end-to-end and modular TED policies make similar kinds of mistakes.

`inform_train`). For the fourth turn, however, it is the other way around: According to the target dialogue the response should have included additional information about the train (`inform_train_request_train`), whereas the predicted dialogue only asked for more information (`request_train`).

The third column shows that the modular TED policy makes the same kinds of mistakes: instead of predicting only `request_train`, it predicts to take both actions, `inform_train` and `request_train` in the second turn. In the final turn, instead of `request_train`, the modular TED policy predicts `reqmore_general`, which means that the wizard asks if the user requires anything else. This reply is perfectly sensible and does in fact occur in similar dialogues of the training set (see, e.g., Dialogue PMUL1883). Thus, *the* correct behaviour doesn't exist and it is impossible to achieve high scores, as reflected by the test scores of Table I.

3. History independence

As Table I shows, taking into account only the last two turns (i.e. the current user utterance or intent, and one system action before that), instead of the last 10 turns, the accuracy and F1 scores remain unchanged for end-to-end and decrease by 0.04 or less for the modular architecture when no status slots are present. If status slots are included, the accuracy and F1 scores decrease by no more than 0.04 for end-to-end and no more than 0.08 for the modular architecture. Thus, MultiWOZ appears to be nearly history-independent, and therefore we cannot evaluate how well the TED policy handles dialogue complexity and thus proceed with a different dataset.

B. Conversations containing sub-dialogues

We repeat experiments on the dataset from [2]. This dataset was specifically designed to test the ability of a dialogue policy to handle non-cooperative or unexpected user input. It consists of task-oriented dialogues in hotel and restaurant reservation domains containing cooperative (user provides necessary information related to the task) and non-cooperative (user asks a question unrelated to the task or makes chit-chat) dialogue turns. One of the properties of this dataset is that the system repeats the previously asked question after any non-cooperative user behavior. This dataset is also used in [8] to compare the performance of different attention mechanisms.

Figure 3 shows the performance of different dialogue policies on the held-out test dialogues as a function of the amount of conversations used to train the model. The TED policy performs on par with REDP without any specifically designed architecture to solve the task and significantly outperforms simple LSTM-based policy. In the extreme low-data regime, the TED policy is outperformed by REDP. It should be noted that REDP relies heavily on its copy mechanism to predict the previously asked question after a non-cooperative digression. However, the TED policy, being both simpler and more general, achieves similar performance without relying on dialogue properties like repeating a question. Moreover, due to the transformer architecture, the TED policy trains faster than REDP and requires fewer training epochs to achieve the same accuracy.

Figure 4 visualizes the attention weights of the TED policy on an example dialogue. This example dialogue contains several chit-chat utterances in a row in the middle of the conversation. The Figure shows that the series of chit-chat interactions is completely ignored by the self-

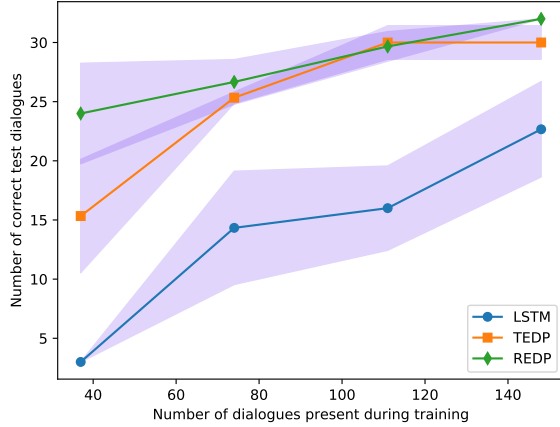


FIG. 3. Performance of REDP, TED policy, and an LSTM baseline policy on the dataset from [2]. Lines indicate mean and shaded area the standard deviation of 3 runs. The horizontal axis indicates the amount of training conversations used to train the model and the vertical axis is the number of conversations in the test set in which every action is predicted correctly.

attention mechanism when task completion is attempted (i.e. further required questions are asked). Note, that the learned weights are sparse, even though the TED policy does not use a sparse attention architecture. Importantly, the TED policy chooses key dialogue steps from the history that are relevant for the current prediction and ignores uninformative history. Here, we visualize only one conversation, but the result is the same for an arbitrary number of chit-chat dialogue turns.

V. CONCLUSIONS

We introduce the transformer embedding dialogue (TED) policy in which a **transformer’s self-attention mechanism operates over the sequence of dialogue turns**. We argue that this is a more appropriate architecture than an RNN due to the presence of interleaved topics in real-life conversations. We show that the TED policy can be applied to the MultiWOZ dataset in both a modular and end-to-end fashion, although we also identified that this dataset is not ideal for supervised learning of dialogue policies, due to a lack of history depen-

dence and a dependence on individual crowd-worker preferences. We also perform experiments on a task-oriented dataset specifically created to test the ability to recover from non-cooperative user behaviour. The TED policy outperforms the baseline LSTM approach and performs on par with REDP, despite TED being faster, simpler, and more general. We demonstrate that learned attention weights are easily interpretable and reflect dialogue logic. At every dialogue turn, a transformer picks which previous turns to take into account for current predic-

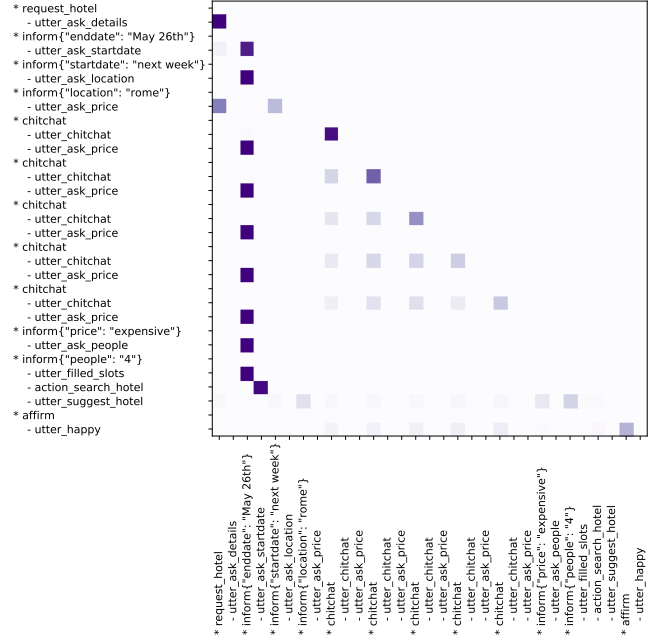


FIG. 4. Attention weights of the TED policy on an example dialogue. On the vertical axis are predicted dialogue turns, and on the horizontal axis is the dialogue history over which the TED policy attends. We use a unidirectional transformer, so the upper triangle is masked with zeros to avoid attending to future dialogue turns.

tion, selectively ignoring or attending to different turns of the dialogue history.

Acknowledgments

We would like to thank the Rasa team and Rasa community for feedback and support. Special thanks to Elise Boyd for supporting us with the illustrations.

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[2] Vladimir Vlasov, Akela Drissner-Schmid, and Alan Nichol. Few-shot generalization across dialogue tasks. *arXiv preprint arXiv:1811.11707*, 2018.

[3] Barbara J Grosz and Candace L Sidner. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3):175–204, 1986.

- [4] Dan Bohus and Alexander I Rudnicky. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3):332–361, 2009.
- [5] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 553–562. ACM, 2015.
- [6] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
- [7] Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv preprint arXiv:1702.03274*, 2017.
- [8] Saurav Sahay, Shachi H. Kumar, Eda Okur, Haroon Syed, and Lama Nachman. Modeling intent, dialog policies and response adaptation for goal-oriented interactions. In *Proceedings of the 23rd Workshop on the Semantics and Pragmatics of Dialogue - Full Papers*, London, United Kingdom, September 2019. SEMDIAL. URL http://semdial.org/anthology/Z19-Sahay_semdial_0019.pdf.
- [9] Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.
- [10] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.
- [11] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.
- [12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [13] Yu-Lun Hsieh, Minhao Cheng, Da-Cheng Juan, Wei Wei, Wen-Lian Hsu, and Cho-Jui Hsieh. On the robustness of self-attentive models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1520–1529, 2019.
- [14] Matthew Henderson, Ivan Vulić, Daniela Gerz, Iñigo Casanueva, Paweł Budzianowski, Sam Coope, Georgios Spithourakis, Tsung-Hsien Wen, Nikola Mrkšić, and Pei-Hao Su. Training neural response selection for task-oriented dialogue systems. *arXiv preprint arXiv:1906.01543*, 2019.
- [15] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*, 2018.
- [16] Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*, 2016.
- [17] Shikib Mehri, Evgeniia Razumovskaia, Tiancheng Zhao, and Maxine Eskenazi. Pretraining methods for dialog context representation learning. *arXiv preprint arXiv:1906.00414*, 2019.
- [18] Jason D Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.
- [19] Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*, 2017.
- [20] Ledell Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. Starspace: Embed all the things! *arXiv preprint arXiv:1709.03856*, 2017.
- [21] Ben Peters, Vlad Niculae, and André FT Martins. Sparse sequence-to-sequence models. *arXiv preprint arXiv:1905.05702*, 2019.
- [22] Note1. Balanced batching was introduced in Rasa 1.3.
- [23] Yi Yang, Wen-tau Yih, and Christopher Meek. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018. Association for Computational Linguistics, 2015. doi:10.18653/v1/D15-1237.
- [24] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. 2016. URL <http://arxiv.org/abs/1506.08909>.
- [25] MetaLWOz, 2019. URL <https://www.microsoft.com/en-us/research/project/metalwoz/>.
- [26] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasić. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*, 2018.
- [27] Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tur. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*, 2019.
- [28] Bill Byrne, Karthik Krishnamoorthi, Chinnadurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. Taskmaster-1: Toward a realistic and diverse dialog dataset. In *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*, 2019. URL https://storage.googleapis.com/dialog-data-corpus/TASKMASTER-1-2019/landing_page.html.
- [29] Nikola Mrkšić, Diarmuid O Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. Neural belief tracker: Data-driven dialogue state tracking. *arXiv preprint arXiv:1606.03777*, 2016.