Learning Interpretable Latent Dialogue Actions With Less Supervision

Vojtěch Hudeček and Ondřej Dušek
hudecek@ufal.mff.cuni.cz, odusek@ufal.mff.cuni.cz
Charles University, Faculty of Mathematics and Physics
Malostranské náměstí 25, 118 00 Prague, Czechia

Abstract

We present a novel architecture for explainable modeling of task-oriented dialogues with discrete latent variables to represent dialogue actions. Our model is based on variational recurrent neural networks (VRNN) and requires no explicit annotation of semantic information. Unlike previous works, our approach models the system and user turns separately and performs database query modeling, which makes the model applicable to task-oriented dialogues while producing easily interpretable action latent variables. We show that our model outperforms previous approaches with less supervision in terms of perplexity and BLEU on three datasets, and we propose a way to measure dialogue success without the need for expert annotation. Finally, we propose a novel way to explain semantics of the latent variables with respect to system actions.

1 Introduction

While supervised neural dialogue modeling is a very active research topic (Wen et al., 2017b; Lei et al., 2018; Peng et al., 2021), it requires a significant amount of work to obtain turn-level labels, usually with dialogue state annotation. We argue that in many real-world cases, it is very expensive to obtain the necessary labels or even to design an appropriate annotation schema. Consider a call center with various dialogues that has a lot of transcripts available, including the corresponding API queries, but has no capacity to label them. This motivates our research of approaches that minimize the need for expert annotation.

While most recent research focuses on pretrained language models (PLMs) and reaches state-of-the-art performance in standard supervised (Peng et al., 2021; Zhang et al., 2020b) or even few-shot (Peng et al., 2020; Wu et al., 2020) settings, these models still require full supervision. Furthermore, they lack the potential to interpret the model decisions. Some recent works try to address PLM interpretability with some success (Lin et al., 2019; Stevens and Su, 2021), but still face considerable difficulties due to PLMs' huge number of parameters and their structure. On the other hand, dialogue models using latent variables are able to infer interpretable attributes from unlabeled data (Wen et al., 2017a; Zhao et al., 2019). These models are mostly trained using variational autoencoders (VAE; Kingma and Welling, 2014; Serban et al., 2017). Improvements with discrete variables (Zhao et al., 2018; Shi et al., 2019) offer better interpretability, but the approaches are not directly applicable to task-oriented response generation as no distinction between the system and user roles is made, and database access or goal fulfillment are not considered; most research on unlabeled data only applies to a chit-chat setting.

Since interpretability and the ability to learn from unlabeled data are our primary goals, we choose working with RNN-based latent variable models over Transformer-based PLMs in our work.

Unlike previous latent-variable approaches, we shift the focus towards task-oriented systems and take tracked entity values and database access into account. Specifically, we base our approach on Shi et al. (2019)'s architecture. Shi et al. (2019) employ the VRNN model (Chung et al., 2015) and experiment with conditioning the prior distribution. However, their focus is on uncovering dialogue structure, and they model user and system utterances together. In contrast, we fully take advantage of the VRNN model's generative capabilities and apply it for response generation. Specifically, we train a specialized decoder for system response generation. Furthermore, we extend the VRNN model so that the system and the user utterances are modeled separately. This modification brings the following major advantages: (1) We can model different behaviors on the side of the system and the user, which is expected in a task-oriented setting; (2) We can focus on modeling latent system
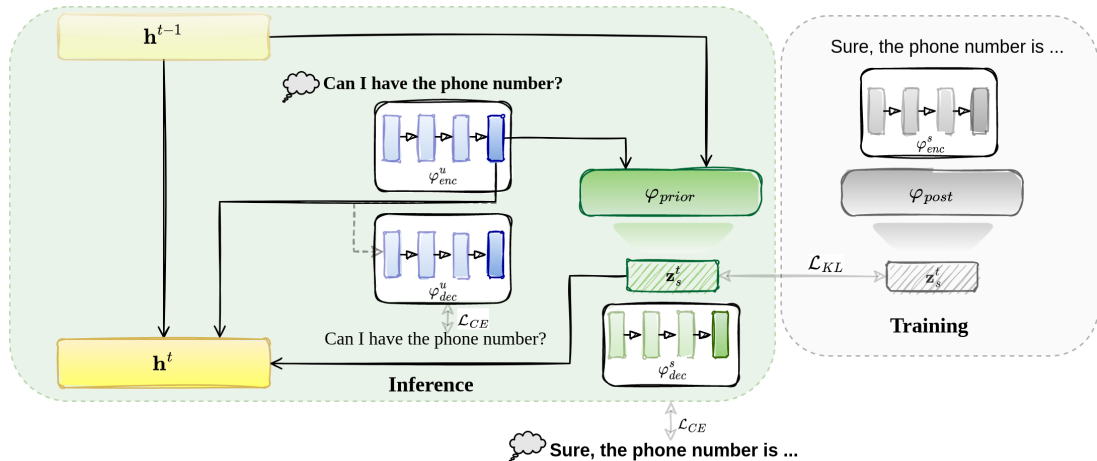
Figure 1: Visualization of our model architecture (one dialogue turn). Yellow boxes represent the turn-level VRNN's hidden state h^t. The user utterance is represented as the last hidden state of the encoder network φ_{enc}^u, which is trained as an autoencoder along with the decoder φ_{dec}^u. The system utterance, encoded by the network φ_{enc}^s, is an input to the posterior network φ_{post} that helps to train the prior network φ_{prior} to construct meaningful latent variables $\mathbf{z}_s$, which initialize the system utterance decoder φ_{dec}^s. Training uses the whole architecture, including the posterior network φ_{post}, while only uses the part shaded in green is used for inference. $\mathcal{L}_{CE}$ stands for cross-entropy loss, $\mathcal{L}_{KL}$ for KL-divergence loss.

actions in an explainable way; (3) We can predict the system response easily.

Task-oriented dialogue systems typically need to interact with an external database; otherwise, their responses cannot be grounded. Therefore, we assume that database queries and results are known, but no dialogue state annotation is available. This allows a direct application of our model for dialogue response generation in a task-oriented setting while still keeping the amount of needed supervision very low. This scenario reflects the intended use case, i.e. automating a call center based on recordings of previous human-human dialogues. At some point of the dialogue, a database query is performed by the human agent and we know exactly when and with which parameters.

Our contributions in this paper are as follows:

1. We propose a novel modification of the VRNN-based model for minimally supervised task-oriented dialogue generation, with interpretable latent variables to represent system actions.

2. We evaluate the system performance in a full task-oriented setting including the database interaction, going beyond previous works in this family of models. Our approach outperforms strong baselines in terms of BLEU and perplexity on three datasets and compares favorably to other baselines.

3. We present a straightforward way of interpreting the latent variables using a decision tree model.

We show that our model's latent variables explain most of our system's predicted responses and align well with gold-standard responses. Our experimental code is released on GitHub.[1]

2 Related Work

In the area of supervised dialogue systems, current leading research focuses on end-to-end sequence-to-sequence models (Lei et al., 2018). Recent works make use of large pre-trained language models (PLMs) based on the transformer architecture (Vaswani et al., 2017) such as GPT-2 (Radford et al., 2019) or BERT (Devlin et al., 2019). For example, Wu et al. (2020) propose finetuning BERT (Devlin et al., 2019) for task-oriented dialogue on multiple datasets; Zhang et al. (2020b) extended the GPT-2 PLM to model open-domain chit-chat.

However, we focus mainly on approaches that require less supervision. The hierarchical recurrent encoder-decoder (HRED) by Serban et al. (2016), where RNN hidden states represent the latent dialogue state, was among the first unsupervised neural dialogue models. However, the latent representations obtained from the vanilla autoencoder model trained with reconstruction loss suffer from poor generalization. For this purpose (Bowman et al., 2016), the usage of Variational Autoencoders (VAEs) (Kingma and Welling, 2014) was proposed. The VAE training maximizes the variational lower

---

[1] https://github.com/vojtsek/to-vrnn

298

bound of data log-likelihood. VAE distributions are invariant in time, therefore it are not suitable for modeling sequences. Chung et al. (2015) address this issue with the Variational Recurrent Neural Network model (VRNN). Serban et al. (2017) then used VRNN's latent variables to represent dialogue state. Recent works used modified Transformer architectures with specific training tasks to obtain in-context representations of dialogue utterances (Bao et al., 2020; Liu et al., 2021).

While both VAEs and Transformers improve generalization and consistency of the latent variables, they are not well interpretable. To obtain more interpretable latent states, generative models with discrete states such as hidden Markov models were applied (Zhai and Williams, 2014; Brychcín and Král, 2017). Wen et al. (2017a) used discrete latent variables to represent the state in a model trained using reinforcement learning. Another proposed approach was the usage of quantization techniques by Gunasekara et al. (2017), who perform clustering on utterances and model the dialogue as a sequence of clusters to predict future responses. Zhao et al. (2018) use VAEs in combination with Gumbel-Softmax to model discrete latent variables representing the dialogue utterances.

More recently, several works attempted to model latent system actions without any action-level annotation (Huang et al., 2020; Zhao et al., 2019; Lubis et al., 2020; Zhang et al., 2020a). However, they still rely on labeled data on different levels, such as turn-level dialogue state annotation. In a different line of research, Shi et al. (2019) aim to uncover the dialogue structure. They apply VRNNs to estimate dialogue state transition probabilities. The same goal of uncovering and understanding semantic structure of the dialogue is explored by Qiu et al. (2020), who propose a VRNN-based model with structured attention to achieve this goal, or Sun et al. (2021), who use an enhanced graph autoencoder. Our proposed model combines the latter two approaches, but it is distinct from both. It models system actions using latent variables, but it does not rely on any turn-level labels for dialogue state or language understanding. Moreover, our goal is not only to uncover the dialogue structure but rather to model system actions and generate responses.

3 Method

We assume that each dialogue turn t consists of a user utterance $\mathbf{x}_u^t$ and a system utterance $\mathbf{x}_s^t$. The context $\mathbf{c}^t$ in turn t is a sequence of user and system utterances up to the previous turn $t-1$. We expect that conditioning the generation of $\mathbf{x}_s^t$ on a latent variable $\mathbf{z}^t$ will allow the model to better incorporate context.

3.1 Background: VRNN

The VRNN model (Chung et al., 2015) can be seen intuitively as a recurrent network with a VAE in every timestep. It extends the VAE model to a sequence of observations generated from a series of hidden latent variables $\mathbf{z}$. Formally, we want to estimate the joint probability distribution of a sequence of observed and corresponding latent variables $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$. The conditional distribution $p(\mathbf{x}|\mathbf{z})$ is parameterized with a neural network. However, we still need to estimate the posterior $p(\mathbf{z}|\mathbf{x})$ in order to connect the latent variables with the observations. The VAE uses a variational approximation $q(\mathbf{z}|\mathbf{x})$ that allows to maximize the lower bound of log-likelihood of the data:

$$\begin{aligned}
\log p(\mathbf{x}) \geq &-\mathrm{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \\
&+\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})]
\end{aligned} \quad (1)$$

where KL is the Kullback-Leibler divergence. We consider a prior network φ_{prior} and a posterior network φ_{post}, which compute the parameters of $p(\mathbf{z})$ and $q(\mathbf{z}|\mathbf{x})$ respectively. In a VRNN, φ_{prior} and φ_{post} additionally depend on the RNN hidden state $\mathbf{h}^t$ to allow for a context-aware prior distribution. In each time step, we obtain the distribution parameters as follows:

$$\begin{aligned}
\theta_q &= \varphi_{post}(\mathbf{h}^t, \varphi_{enc}(\mathbf{x}^t)) \\
\theta_p &= \varphi_{prior}(\mathbf{h}^t)
\end{aligned} \quad (2)$$

where φ_{enc} is the encoder and θ_q, θ_p are parameters of the respective distributions (see Section 3.4). With distribution parameters available, we can sample the latent variable and predict the output:

$$\begin{aligned}
\mathbf{z}^t &\sim p(\mathbf{z}; \theta_p) \\
\mathbf{x}^t &= \varphi_{dec}(\mathbf{z}^t)
\end{aligned} \quad (3)$$

where φ_{dec} represents the decoder network. The update of the hidden state $\mathbf{h}^t$ is as follows:

$$\mathbf{h}^{t+1} = \mathrm{RNN}([\varphi_{enc}(\mathbf{x}^t), \varphi_z(\mathbf{z}^t)], \mathbf{h}^t) \quad (4)$$

where $[.,.]$ is concatenation, $\varphi_z(.)$ is a feature extractor and RNN() is a step transition function of a recurrent neural network, in our case an LSTM (Hochreiter and Schmidhuber, 1997).

Turn 1	**user:**	Is there a **moderately priced** restaurant serving **italian** food anywhere in town?
	system gold:	query italian moderate
	action:	QUERY()
	system hyp:	query **italian moderate**
Turn 2	**user/database:**	pizza express, Regent Street City Centre, 01223 324033, C.B 2, 1 D.B, centre
	system gold:	Pizza express serves italian food and is located in the town centre and is in the moderate price range .
	action:	OFFER()
	system hyp:	**Pizza hut Cherry Hinton** is a italian restaurant in the **centre** part of town
Turn 3	**user:**	what is the **address** and **phone number** ?
	system gold:	their address is Regent Street City Centre. their phone number is 01223 324033. can i help with anything else ?
	action:	GIVE_DETAILS()
	system hyp:	the phone number is **01223 324033**. There anything else i can help you with ?

Table 1: An example dialogue drawn from the CamRest676 validation set, illustrating the use of database information. We show the user input (or inserted database results), the gold-standard system response, system action annotation based on manual rules (cf. Section 5.2), and a prediction of our system (Ours-attn configuration using the database, cf. Table 3). In the first turn, a database query is constructed, the second turn illustrates how the result is retrieved and fed as input. Values inferred correctly by our system are depicted in green, wrong inference is in red.

3.2 Modeling task-oriented Dialogue

We use the VRNN model and extend it to fit the task-oriented setup. Our model's architecture is depicted in Figure 1. We employ a turn-level RNN that summarizes the context to its hidden state. In each dialogue turn, we model user and system utterances with separate autoencoders to account for different user and system behavior. The user utterance is modeled with a standard autoencoder; the last encoder hidden state $\varphi_{enc}^u(\mathbf{x}_u^t)$ provides the encoded representation. For the system part, we use a VAE with discrete latent variables $\mathbf{z}_s$ conditioned on the context RNN's hidden state $\mathbf{h}^{t-1}$ and the user utterance encoding $\varphi_{enc}^u(\mathbf{x}_u^t)$. Our model can thus be seen as a VRNN extended by an additional encoder-decoder module. The context RNN hidden state update looks as follows:

$$\mathbf{h}^{t+1} = \text{RNN}([\varphi_{enc}^u(\mathbf{x}_u^t), \varphi_z(\mathbf{z}_s^t)], \mathbf{h}^t) \quad (5)$$

For word-level encoding and decoding modules $(\varphi_{enc}^u, \varphi_{enc}^s, \varphi_{dec}^u, \varphi_{dec}^s)$, we use an RNN with LSTM cells. We further experiment with attention (Bahdanau et al., 2015) over user encoder hidden states in the system decoder. We train the model by minimizing a sum of the cross-entropy reconstruction loss on user utterances and the variational lower bound loss (Equation 1) on system responses.

When running in inference mode, only the prior distribution $p(\mathbf{z}_s)$ is considered, which does not require the system utterance on the input. Therefore, the model is able to generate the system response when provided with a user utterance on the input.

3.3 Database interaction

Task-oriented dialogue systems must provide accurate and complete information based on user re-

quests, which requires external database interaction. To support database access while avoiding costly turn-level annotation, we follow Bordes et al. (2017) and insert sparse database queries and results directly into the training data, forming special dialogue turns. Specifically, we identify turns that require database results, e.g. to inform about entity attributes or a number of matching entities, and insert a query-result pair in front of those turns (see Table 1).We argue that this is the minimal level of supervision required to successfully operate a task-oriented system with database access; it is significantly lower than the full dialogue-state supervision used by most systems. In addition, it is easily available in the wild (e.g., call center transaction logs). In practice, we observe that database queries are only inserted for 24% turns[2] on average. Note that this approach still covers the task of an explicit state tracker since the necessary entity values are provided when needed. To maintain consistency, database query results can be stored and used in follow-up questions.

Some experimental approaches, such as Raghu et al. (2021), learn database queries without annotation via reinforcement learning. Our framework could use this to handle database interaction more effectively. We leave this extension for future work.

3.4 Latent Variables

We use a set of n K-way ($K = 20; n = 1, 3, 5$) categorical variables to achieve good interpretability, following Zhao et al. (2018). This means that each variable is represented as a one-hot vector of

---

[2]This is the average over all datasets in our experiments (see Section 4.1). Per-dataset query counts are 36%, 23% and 11% for CamRest676, MultiWOZ and SMD respectively.

Data	Domains	Slots	Dialogues	T/D
MultiWOZ	7	29	10,437	13.71
SMD	3	15	3031	5.25
CamRest676	1	7	676	8.12

Table 2: Details of the used datasets giving number of domains, slots, dialogues and average number of turns per dialogue.

length K, and we use n such vectors. We use the Gumbel-Softmax distribution and the reparameterization trick (Jang et al., 2017). During inference, we apply argmax directly to the predicted distribution, instead of sampling from it.

4 Experiments

In this section, we focus on the quality of responses generated by our model as well as on model performance with respect to dialogue task success. We focus on theoretical modeling and feasibility at this stage, which we believe is sufficiently demonstrated by corpus-based evaluation complemented by manual checks. Detailed interpretation of the learned representations follows in Section 5.

4.1 Data

We evaluate the model performance on three datasets: CamRest676 (Wen et al., 2017b), Multi-WOZ 2.1 (Budzianowski et al., 2018; Eric et al., 2020) and Stanford Multidomain Dialogues (SMD; Eric et al., 2017)[3] All the datasets are task-oriented, i.e., they distinguish between user and system conversational roles. Furthermore, MultiWOZ and SMD include multiple conversation domains. The MultiWOZ dataset contains conversations between tourists and a system that provides information about the city they visit, e.g., restaurants, hotels or attractions and transit connections. SMD contains more concise dialogues between a driver and an in-car virtual assistant. CamRest676 contains only restaurant reservations. Detailed statistics are given in Table 2.

Database queries To include database information in the dialogues, we first identify all turns in the original datasets where database information is required, using handcrafted rules.[4] We then build

database query turns based on the respective state annotation (see example in Table 1). Note that database query parameters are the only annotation used to train our models apart from utterance texts; no other dialogue state annotation from the original datasets is used.

4.2 Experimental Setup

We evaluate two versions of our model: one that uses the attention mechanism (*attn*) and one without it (*noattn*).[5] Since our approach is the first to be evaluated in a task-oriented setting with this minimal level of supervision, comparing to prior works is difficult. Setups with full dialog state supervision are not comparable and dialog-state metrics are not applicable without the turn-level supervision. Therefore, we compare our models to standard architectures, such as vanilla LSTM or Transformer encoder-decoder, predicting in a sequence-to-sequence fashion using the same amount of supervision as our approach. We also compare to the HRED/VHRED models, which are perhaps the closest prior work to our approach. To put the results into perspective, we also include scores for fully supervised state of the art on our datasets. However, note that these scores are not directly comparable. Model parameters are selected by grid search (see Appendix A).[6]

4.3 Response quality

To evaluate the quality of individual responses, we compute BLEU score (Papineni et al., 2002) and perplexity on the test set (see Table 3).

Our architecture performs substantially better than (V)HRED, which commonly fails to pick up the necessary knowledge, especially on larger datasets. The attention-based versions perform better on BLEU, but lose slightly on perplexity. Comparing HRED and VHRED shows that using the variational approach generally improves the overall performance. While the GPT-2 PLM outperforms our approach on perplexity, it is worse on BLEU score, despite its huge capacity.

We compare to other relevant related works:

---

[3] We use standard splits for MultiWOZ 2.1 and SMD. We split CamRest676 in the 8:1:1 ratio, following previous work.

[4] These rules are very simple and require minimal effort: whenever database results are provided in the data (based on simple pattern matches over system actions), we prepend a database query based on ground-truth state. The assumption

is that in a real-world scenario, these queries would naturally be available – database queries induced by human operators can be logged along with client-operator conversations.

[5] The number and size of the variables are set based on a few cursory checks on the training data. Our models use 10 latent variables by default; we discuss the influence of the number of latent variables in Appendix B.

[6] The training is sensitive to some parameters, such-as the Gumbel-softmax temperature, but otherwise the model trains easily using conventional optimization methods.

model	db	CamRest676				SMD			MultiWOZ 2.1			
		BLEU	Ppl	MI	EMR	BLEU	Ppl	MI	BLEU	Ppl	MI	EMR
LSTM	✗	3.90	5.34	–	–	1.62	7.84	–	0.92	8.23	–	–
Transformer	✗	4.98	7.72	–	–	1.53	6.33	–	0.95	6.95	–	–
GPT-2	✗	15.40	1.18	–	–	9.26	2.46	–	9.40	2.77	–	–
GPT-2	✓	13.89	1.80	–	–	4.54	2.02	–	9.56	2.43	–	–
HRED	✗	2.70	13.92	–	0.02	1.25	12.50	–	2.98	29.61	–	0.01
VHRED	✗	4.34	11.76	0.21	0.02	3.75	11.94	0.20	4.65	32.74	0.15	0.01
VHRED	✓	8.50	10.23	0.17	0.36	3.94	11.86	0.19	3.82	16.61	0.07	0.04
Ours-noattn	✗	12.98	4.64	0.29	0.01	7.35	6.18	**0.53**	7.18	9.16	**0.42**	0.02
Ours-noattn	✓	15.10	4.45	**0.34**	0.24	9.24	**6.01**	0.47	11.3	5.17	0.27	0.05
Ours-attn	✗	**17.37**	5.07	0.16	0.09	12.30	6.36	0.04	**12.28**	10.19	0.06	0.04
Ours-attn	✓	17.10	**4.23**	0.22	**0.81**	**12.40**	6.11	0.11	11.86	**6.03**	0.05	**0.08**
*supervised SotA**	✓	25.50	–	–	–	14.40	–	–	19.40	2.50	–	–

Table 3: Model performance in terms of Entity Match Rate, BLEU for generated responses, Perplexity (Ppl), and Mutual Information (MI) between the generated response and the latent variables $\mathbf{z}_s$. We measure MI only for the models that use latent variables explicitly. The *db* column indicates systems which use database information. *Note that the supervised state-of-the-art scores are not directly comparable, as the systems use full turn-level supervision. Systems listed: CamRest676 (Peng et al., 2021); SMD (Qin et al., 2020); MultiWOZ (Lin et al., 2020a).

config	CamRest676	MultiWOZ 2.1.	
	gold	domain	action
random	0.167	0.143	0.093
majority	0.417	0.327	0.316
HRED	0.645	0.445	0.437
VHRED	0.521	0.357	0.323
GPT-2	0.650	0.601	0.552
Ours-attn	0.616	0.683	0.664
Ours-noattn	**0.753**	**0.704**	**0.691**
Ours-manual	0.587	–	–

Table 4: Accuracy of the domain and action decision-tree classifiers based on latent variables. For details about the manual annotation process, see Section 5.3.

model	success	query acc.
CamRest676		
VHRED	0.21	0.91
Ours-noattn	0.28	0.84
supervised SotA (Peng et al., 2021)	0.73	N/A
MultiWOZ		
Ours-noattn	0.10	0.98
supervised SotA (Peng et al., 2021)	0.85	N/A

Table 5: Dialogue success and query accuracy comparison for VHRED, *Ours-noattn* using the database and a state-of-the-art supervised system.

1. Shi et al. (2019) do not use their model for response generation, but they report a negative log likelihood of approximately $5.5 \cdot 10^4$ when reconstructing the CamRest676 test set. Our *Ours-noattn* model obtained $0.87 \cdot 10^4$, which suggests a better fit of the data.[7]

2. Wen et al. (2017a) measure response generation BLEU score on fully delexicalized Cam-Rest676 data. Their best reported result is 24.60, while our model gets 27.23 (30.10 with attention).

Based on manual checks, our models are able to generate relevant responses in most cases. As expected, only the models including database turns are able to predict correct entities (cf. Section 4.4). A relatively common error is informing about wrong slots, e.g. the model provides a phone num-

ber instead of an address or, even more frequently, provides wrong slot values (cf. Table 1).

4.4 Task-related performance

Without dialogue-state supervision, we cannot measure task-oriented metrics such as *inform* rate or *goal accuracy*. Therefore, we decided to measure dialogue success and entity match rate, which we adjust to the minimally supervised case (details follow). We also measure database query accuracy.

Dialogue success The dialogue success or *success rate* reflects the ratio of dialogues in which the system captures all the mentioned slots correctly and provides all the requested information. We follow previous works (Nekvinda and Dušek, 2021) and report corpus-based success score, as opposed to using a user simulator. However, measuring success rate without turn-level labels is not straightforward. We approximate tracking slot values turn-by-turn by checking for correct slot values upon database queries only, and we use this in-

---

[7]This comparison is only approximate since the exact data split is not described by Shi et al. (2019) – we are only able to use a test set of the same size, not the exact same instances.

formation to measure dialogue success. Note that this is not equivalent to having state tracking labels available at all turns, but we consider it a reasonable approximation given our limited supervision – database queries are crucial for presenting the correct entities to the user, which in turn decides the dialogue success. The generated query attributes directly show the captured slots.

Success rate results are shown in Table 5. Our system is not competitive with a fully supervised model, but outperforms the baselines (VHRED, GPT). Upon inspection, we see that the system is often able to recognize correct slots, however, it has difficulties capturing the correct values. However, the scores are promising considering the minimal supervision of our training.

Matching database entities   To evaluate the accuracy of the offered entities, we measure the Entity Match Rate (EMR), i.e. the ratio of generated responses with correct entities over all responses that mention some entity. Table 3 shows the results. We observe that the model performance without the database information is poor. However, including the database information improves the performance substantially, especially in the case of CamRest676 data. The MultiWOZ data is much more complex – it contains more slots and multiple domains that can also be combined in an individual dialogue. Nevertheless, we can still observe an improvement when we include the database queries. We also note that using attention improves EMR substantially – the latent variables alone cannot hold all information about particular values (cf. Section 5.2).

Database query accuracy   Further, we evaluate the accuracy of the database querying. This metric simply measures if the system queries the database at appropriate turns. The content of the query is not taken into account in this case, as it is already considered in the success rate. On MultiWOZ, we get a near-perfect accuracy, while our approach loses to VHRED on CamRest676 (see Table 5). We hypothesize that this discrepancy can be caused by different dialogue structures among theses two datasets. The dialogues in CamRest676 usually contain just zero or one query during a dialogue, so our model might generate more queries than necessary.

5   Latent Variable Interpretation

We believe that being able to explain and interpret the model behavior is crucial, especially in a setting without full supervision. Therefore, we design a set of experiments to evaluate the model behavior and investigate whether the model captures salient dialogue features in the latent variables obtained during training on CamRest676 and MultiWOZ. While it seems that the latent variables are mainly useful for interpretability or structure induction, they are likely also contributing to the performance as smaller latent spaces yield lower performance as we saw in preliminary experiments and show in Appendix B.

5.1   Clustering the actions

First, we want to assess whether similar variables represent similar actions. We follow Zhao et al. (2018) and define utterance clusters according to the latent variables that have been assigned to them by the model. We then use the homogeneity metric (Rosenberg and Hirschberg, 2007) to evaluate the clustering quality with respect to the reference classes determined by manually annotated system actions (which are used for evaluation only). Homogeneity reflects the amount of information provided by the clustering (and by extension, the latent vectors used) and is normalized to the interval [0, 1]. The reason of choosing this metric is that it is independent on the number of labels and their permutations. We provide the results in Table 6. The clusters formed on the CamRest676 data are more homogeneous than on MultiWOZ, likely because of the greater dataset complexity in the latter case. In all cases, our clusters are much more homogeneous than clustering formed by random assignment. We also compare favorably to stronger baseline that is based on clustering of the sentence representations. Specifically, in this approach we compute sentence representations using a BERT model tuned for sentence representations (Reimers and Gurevych, 2019) and then cluster the obtained sentence embeddings using K-means clustering.

5.2   Predictive power of the variables

To evaluate the predictive power of the obtained latent representations, we train a simple classifier that predicts the system action and current domain, using solely the obtained latent representations as input features. CamRest676 data does not include system action annotation, hence we manually de-
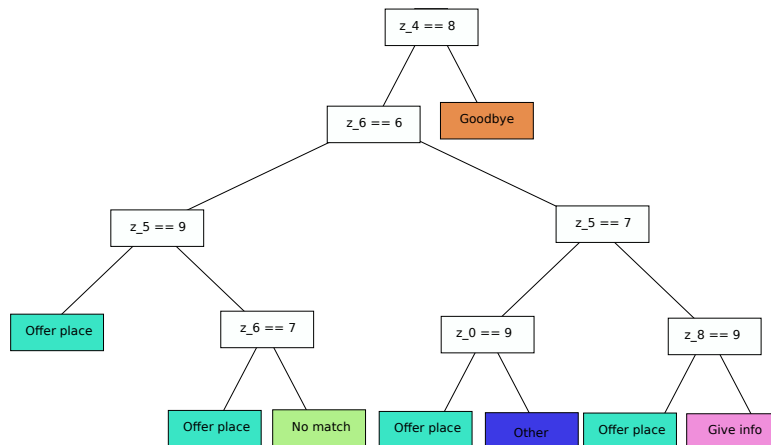
Figure 2: A visualization of a decision tree trained on the CamRest676 data to predict a system action from the contents of the latent variables. Each node represents a decision based on one latent variable value and the leaf node colors represent different system actions. When the condition in a given node is fulfilled, the algorithm proceeds into the right subtree, left otherwise. For clarity, we limit the maximum tree depth to 4. The limit lowers the accuracy slightly – the pictured tree achieves an accuracy of 73% on the CamRest676 data.

signed a set of rules to determine system actions. An example of this rule-based action annotation is shown in Table 1. For MultiWOZ, we predict both system action and the domain of the utterance.

To put our results into perspective, we include several baselines: trivial random and majority class baselines, and classifiers using representations obtained with other methods (HRED, VHRED, GPT). We use a decision tree (DT) classifier trained with the CART algorithm[8] and the *gini* split criterion, due to the its good interpretability. The results are shown in Table 4. Our classifier beats the random and majority baselines in all cases. More importantly, it also outperforms classification based on (V)HRED and GPT representations. This demonstrates that our approach produces high-quality interpretable representations. We also observe that using attention harms the performance of the action classifier as it makes it possible for the models to bypass the latent variables.

The information about domains and system actions is stored in categorical variables and can be extracted by a simple classification model such as the decision tree which allows us to interpret and explain the behavior of our model. For illustration, in Figure 2 we plot a DT with limited depth that achieves 73% accuracy when predicting the system action on the CamRest676 data.[9]

Target	Ours-noattn	sent-repr	random
CamRest676 action	0.65	0.45	0.20
MultiWOZ action	0.34	0.33	0.02
MultiWOZ domain	0.39	0.30	0.01

Table 6: Homogeneity for *Ours-noattn* configuration using the database vs. a clustering of sentence representations and random baseline.

5.3 Manual interpretation

To explore the interpretability of our representations even further, we manually annotate the latent variables to obtain a simple handcrafted classifier. Specifically, we draw a set of pairs of utterances and corresponding latent representations from the validation set. Then we present the representation (discrete) vectors to an expert annotator with a task of assigning an action that each vector represents, based on the sampled utterances. This way we obtain a mapping from the space of latent vectors to actions. We then apply this mapping to predict actions on the test set (the *-manual* entry in Table 4). Note that in this approach, we only allow assigning an action to a whole vector, unlike in the case of decision tree classifier that can take individual components into account. As the results show, this approach works well, despite the above limitation.

---

[8] https://scikit-learn.org/stable/modules/tree.html

[9] The aim is that latent variables hold high-level information, such as intents, actions or domains. This helps interpretability, but is not sufficient for generating appropriate and

factually correct responses – here we need to incorporate correct slot values. This detailed information is captured and carried over via the attention mechanism in *Ours-attn*. Potential alternatives are copy mechanisms (Lei et al., 2018) or delexicalization on the generated outputs (Henderson et al., 2014; Peng et al., 2021).

5.4 Mutual Information

Finally, we compute mutual information (MI) between the generated text and latent variables as well as among the latent variables themselves (see Table 3).[10] We see that using attention has a dramatic effect on the amount of MI between the latent variables and the generated text. It appears that since attention bypasses the latent vectors, the decoder does not need to use them to store information.

6 Conclusion and Future Work

We introduce a model for task-oriented dialogue with discrete latent variables that uses only minimal supervision and improves upon previous approaches (Chung et al., 2015; Serban et al., 2017). We also propose methods for task-based evaluation in this minimally supervised setting. Our system is not yet ready for interactive evaluation on full dialogues, considering the clear performance gap with respect to with fully supervised approaches. However, we demonstrate that it learns meaningful representations from minimal supervision (in a realistic setup corresponding to pre-existing call center call logs) and compares favorably to previous weakly supervised approaches. A detailed analysis reveals that the learned representations capture relevant dialogue features and can be used to identify system actions. Furthermore, the reason for choosing an action can be described in an explainable way. The results suggest that dialogue models with discrete latent variables can be successfully applied also in the task-oriented setting.

The main limitations of our current model are its problems with providing the correct slot values in responses. We plan address this issue in future work by incorporating explicit copy mechanisms (Lei et al., 2018), i.e. the model will learn to copy slot values from the context and from database results. We also plan to experiment with incorporating Transformer models into the variational autoencoder setup, following recent models such as the VAE-transformer (Lin et al., 2020b).

7 Acknowledgements

References

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 (ICLR2015)*, San Diego, CA, USA.

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. PLATO: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96, Online. Association for Computational Linguistics.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning End-to-End Goal-Oriented Dialog. In *5th International Conference on Learning Representations, ICLR 2017*, Toulon, France.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Tomáš Brychcín and Pavel Král. 2017. Unsupervised Dialogue Act Induction using Gaussian Mixtures. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 485–490, Valencia, Spain.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ – a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of EMNLP*.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186, Minneapolis, MN, USA.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj

---

[10]Since we measure MI between categorical variables, we quantize the continuous variables used in the VHRED model.

Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A Consolidated Multi-Domain Dialogue Dataset with State Corrections and State Tracking Baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428, Marseille, France.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49, Saarbrücken, Germany. Association for Computational Linguistics.

R Chulaka Gunasekara, David Nahamoo, Lazaros C Polymenakos, Jatin Ganhotra, and Kshitij P Fadnis. 2017. Quantized-dialog language model for goal-oriented conversational systems. In *DSTC6 – Dialog System Technology Challenges*, Long Beach, CA, USA. ArXiv:1812.10356.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 360–365.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Xinting Huang, Jianzhong Qi, Yu Sun, and Rui Zhang. 2020. Generalizable and explainable dialogue generation via explicit action learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3981–3991, Online. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR 2017)*.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, Banff, AB, Canada.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447, Melbourne, Australia.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020a. MinTL: Minimalist Transfer Learning for Task-Oriented Dialogue Systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 3391–3405, Online.

Zhaojiang Lin, Genta Indra Winata, Peng Xu, Zihan Liu, and Pascale Fung. 2020b. Variational transformers for diverse response generation. *CoRR*, abs/2003.12738.

Che Liu, Rui Wang, Jinghua Liu, Jian Sun, Fei Huang, and Luo Si. 2021. DialogueCSE: Dialogue-based contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2396–2406, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nurul Lubis, Christian Geishauser, Michael Heck, Hsien-chin Lin, Marco Moresi, Carel van Niekerk, and Milica Gasic. 2020. LAVA: Latent action spaces via variational auto-encoding for dialogue policy optimization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 465–479, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tomáš Nekvinda and Ondřej Dušek. 2021. Shades of BLEU, flavours of success: The case of MultiWOZ. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 34–46, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. SOLOIST: Building Task Bots at Scale with Transfer Learning and Machine Teaching. *Trans. Assoc. Comput. Linguistics*, 9:907–824.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot Natural Language Generation for Task-Oriented Dialog. In *Findings of EMNLP*, pages 172–182.

Libo Qin, Xiao Xu, Wanxiang Che, Yue Zhang, and Ting Liu. 2020. Dynamic Fusion Network for Multi-Domain End-to-end Task-Oriented Dialog. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 6344–6354, Online.

Liang Qiu, Yizhou Zhao, Weiyan Shi, Yuan Liang, Feng Shi, Tao Yuan, Zhou Yu, and Song-Chun Zhu. 2020. Structured attention for unsupervised dialogue structure induction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1889–1899, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI.

Dinesh Raghu, Nikhil Gupta, and Mausam. 2021. Unsupervised Learning of KB Queries in Task-Oriented Dialogs. *Trans. Assoc. Comput. Linguistics*, 9:374–390.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Proceedings of the 31th AAAI Conference on Artificial Intelligence*, page 3295–3301, San Francisco, CA, USA.

Weiyan Shi, Tiancheng Zhao, and Zhou Yu. 2019. Unsupervised Dialog Structure Learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1797–1807, Minneapolis, Minnesota.

Samuel Stevens and Yu Su. 2021. An investigation of language model interpretability via sentence editing. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 435–446, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yajing Sun, Yong Shan, Chengguang Tang, Yue Hu, Yinpei Dai, Jing Yu, Jian Sun, Fei Huang, and Luo Si. 2021. Unsupervised learning of deterministic dialogue structure with edge-enhanced graph autoencoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13869–13877, Virtual Event.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, Long Beach, CA, USA.

Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve Young. 2017a. Latent Intention Dialogue Models. In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, Sydney, Australia.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017b. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of EACL*, pages 438–449.

Chien-Sheng Wu, Steven Hoi, Richard Socher, and Caiming Xiong. 2020. ToD-BERT: Pre-trained natural language understanding for task-oriented dialogues. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 917–929, Online.

Ke Zhai and Jason D. Williams. 2014. Discovering latent structure in task-oriented dialogues. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–46, Baltimore, Maryland.

Yichi Zhang, Zhijian Ou, Min Hu, and Junlan Feng. 2020a. A Probabilistic End-To-End Task-Oriented Dialog Model with Latent Belief States towards Semi-Supervised Learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9207–9219, Online. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020b. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL): System Demonstrations*, pages 270–278, Online.

Tiancheng Zhao, Kyusong Lee, and Maxine Eskenazi. 2018. Unsupervised Discrete Sentence Representation Learning for Interpretable Neural Dialog Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1098–1107, Melbourne, Australia.

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1208–1218, Minneapolis, Minnesota. Association for Computational Linguistics.

A Training Parameters

The model is trained with gradient descent, using ADAM optimizer. We set the hyperparameters according to the BLEU and perplexity results of a grid search on the development set. Utterance encoder and decoder hidden sizes are 250, the context-LSTM hidden size is 100. The latent variables are 20-dimensional vectors, their number differs across experiments and is given in the main text. For the RNN components, we use a dropout probability of 0.3. The total model size is 7,047,529 parameters. The training time is 3-8 hours using one GPU, depending on dataset.

B Performance with Various Numbers of Latent Variables

	BLEU	Ppl	MI
Ours-noattn-1z	25.2	4.25	0.46
Ours-noattn-3z	26.8	4.24	0.26
Ours-noattn-5z	27.23	4.20	0.38
Ours-noattn-12z	29.83	4.12	0.35

Table 7: Evaluation of the model performance with respect to automatic measures of BLEU, Perplexity (Ppl) and Mutual Information (MI) on the CamRest676 data.

C Limitations and risks

We consider our work to be mostly fundamental research rather than a practical application. However, it has certain limitations. Firstly, the proposed way of including the database results is inflexible and it is hard to incorporate possible API changes. Also, although we show that the latent actions are possible to interpret and explain, with growing number of actions we likely worsen this possibility to interpret the variables. Another limitation of our current model is its inability to provide correct entities and slot values.

Another limitation and possible risk is that this system is very hard to control and deploying it in current form could produce undesired behavior.