MIXTURE OF DECODERS FOR DIVERSE DIALOG RESPONSE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture modeling is a long established machine learning technique for learning large sets of multi-modal data. While it is known that sequence-to-sequence models for dialog response generation suffer from the problem of low diversity, we hypothesize that it is because sequence-to-sequence models tend to learn a degenerate uni-modal distribution of responses. We then propose to incorporate a mixture of decoders into sequence-to-sequence models and try to make each decoder learn specialized topics in order to improve the diversity of generated responses. Our model is developed under the framework of conditional variational autoencoder (CVAE). We evaluate our approach on an open domain chat corpus and show improvement over strong baselines in quantitative measures and human evaluation.

1 INTRODUCTION

Data-driven approaches to dialog response generation have become increasingly popular in the past few years Ritter et al. (2011), and with the success of deep learning in language generation, especially machine translation Sutskever et al. (2014), sequence-to-sequence models Shang et al. (2015); Vinyals & Le (2015) have started to dominate the field of dialog systems. While providing a high degree of automation and flexibility, deep learning models notoriously tend to generate dull and generic responses all the time Li et al. (2016). Unlike machine translation, the complexity of response generation problem is very high in the way that each context (or source) may have multiple plausible responses, and those responses for the same context can be very different from each other in both lexical choices and syntactic structures. In other words, the output distribution for response generation is multi-modal, and its automation requires learning a one-to-many mapping between utterances Liu et al. (2018). Previous sequence-to-sequence learning approaches usually end up learning a degenerate output distribution whose mode is at the dull responses. This is widely known as "modal collapse" in machine learning community Tolstikhin et al. (2017).

In order to learn such multi-modal distributions, we propose a general method combined with sequence-to-sequence learning, termed "mixture of decoders (MoD)". As suggested by the name, the model introduces a discrete latent variable, and each category is associated with a dedicated decoder for that mixture. Incorporating latent variables into encoder-decoder models to capture discourse-level information has been previously studied Serban et al. (2017); Zhao et al. (2017). We argue that mixture modelling has the following advantages. First, discrete latent variables are easier than continuous variables for interpretation, since mixtures can be directly mapped to proposed semantic or pragmatic categorizations, such as topics and dialog acts. Second, since the latent variables are discrete, it is made possible to incorporate prior knowledge of topics or dialog acts by means of semi-supervised learning, which is developed and discussed in this paper. Third, during the decoding process, a mixture of decoders can potentially provide a diverse pool of hypotheses, if the decoders are sufficiently specialized, and such diversity in the candidate pool leads to a higher chance in producing more diverse responses.

For learning and inference, we adopt the framework of conditional variational autoencoder Sohn et al. (2015). Although we only evaluate decoders based on recurrent neural networks (RNN), we conjecture that the mixture modelling approach can be applied to other types of decoders, such as convolutional neural networks Zhang et al. (2017) and Transformer Vaswani et al. (2017). To our knowledge, we are the first to apply mixture modelling to dialog response generation problem with

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

deep learning. From the perspective of machine learning, our model is an extension to M2 model of semi-supervised VAE Kingma et al. (2014) by making the prior distribution conditioned on dialog contexts.

2 Related Work

Improving the diversity of dialog response generation and retrieval Deb et al. (2019) has been an active research problem. Li et al. (2016) employed a decoding objective based on mutual information between contexts and responses; Li et al. (2017a) used reinforcement learning techniques for training the decoder to generate responses that maximize pre-defined rewards instead of perplexities; Li et al. (2017b); Xu et al. (2017) adopted adversarial learning, in which a generator is trained to deceive a discriminator that tries to differentiate between generated responses and human responses. Beside changing training and decoding objectives, Liu et al. (2018); Lison & Bibauw (2017) considered reweighting data points by penalizing those with overly frequent responses or by emphasizing high-quality responses. Du & Black (2019) proposed a boosting-based approach for increasing the diversity by iteratively training the model with the data weighted differently at each iteration. Our approach aims to augment existing models and increase their capacity, and can be used together with aforementioned methods. Next, we survey in detail the related work on language model and generation from the following two aspects:

2.1 Latent Variable Model

For dialog response generation, Serban et al. Serban et al. (2017) proposed to incorporate a continuous latent variable into their sequence-to-sequence model. The latent variable is conditioned on the dialog context, and the generation of responses depends on the same latent variable. Learning and inference of the latent variable involves a prior network conditioned on the context, and a posterior (or recognition) network conditioned on both the context and the response. Zhao et al. Zhao et al. (2017) experimented with almost the same idea as Serban et al., in the hope that the continuous vector representations can be informative of discourse and pragmatics. They additionally developed a way to leverage linguistic knowledge by training the prior network and the recognition network with hand-crafted features. Du et al. Du et al. (2018) advanced one step further in this line of ideas by learning separate latent variables for each word in the responses. Unlike these models with continuous latent variables, prior knowledge can be imparted onto a mixture model by training it discriminatively with labels. Discrete latent variables also allows better interpretability than continuous variables.

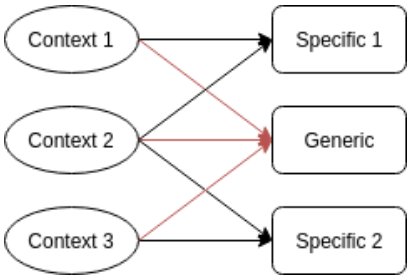


Figure 1: Sequence-to-sequence model learns a degenerate unimodal distribution of responses, as illustrated by the red lines.

2.2 Mixture Model for Language

There has been some recent work on combining mixture modelling and sequence-to-sequence learning for machine translation. He et al. He et al. (2018) imposes a uniform prior on the mixtures. Shen et al. Shen et al. (2019) uses a multinomial distribution over mixtures that is agnostic to contexts, and update the parameters via expectation-maximization (EM). They also explored several variants on the learning procedure of their mixture model. In our work, the prior distribution of mixtures is conditioned on contexts, enabling the model to make sharper decisions of latent actions based on dialog history.

Originally proposed for language modelling, mixture-of-softmax (MoS) Yang et al. (2018) aims to enhance the capacity of RNN decoders by learning a mixture of output word distributions on top of the bottleneck layer in the decoder. This constitutes a mixture model in which the weight of each mixture component is conditioned on the RNN hidden state at each time step. No posterior inference

about the mixtures is made, and the parameters are learned through maximum likelihood estimation. We compare MoS with our approach in detail in the next section.

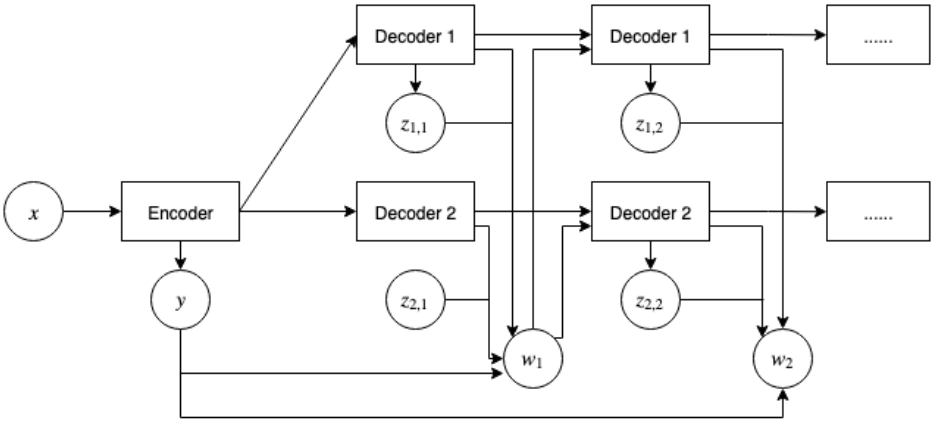


Figure 2: The graphical representation of our mixture of decoders model, where each decoder is a VAD. The figure shows the first two recurrent steps for two decoders. Circles are variables. Attention mechanism is left out for simplicity.

3 METHODS

3.1 FORMULATION

Given dialog context x, the standard sequence-to-sequence model learns to estimate the probability of response $\mathbf{w}$ through

$$p(\mathbf{w} \mid x) = \prod_i p_\theta(w_i \mid w_1 \dots w_{i-1}, x)$$

Previous work Zhao et al. (2017) based on conditional variational autoencoder adds a continuous variable z to connect x and $\mathbf{w}$:

$$\begin{aligned} p(\mathbf{w} \mid x) &= \int p_\theta(\mathbf{w}, z \mid x) dz \\ &= \int p_\theta(\mathbf{w} \mid x, z) p_\theta(z \mid x) dz \end{aligned}$$

Variational Autoregressive Decoder (VAD) Du et al. (2018) augment the idea by adding latent variables to each word in the response, and each latent variable depends not only on the context but also all previous words in the response:

$$\begin{aligned} p(\mathbf{w} \mid x) &= \int p_\theta(\mathbf{w}, \mathbf{z} \mid x) dz \\ &= \int \big(\prod_i p_\theta(w_i \mid z_i, w_1 \dots w_{i-1}, x) \cdot \\ &\quad p_\theta(z_i \mid w_1 \dots w_{i-1}, x) \big) dz \end{aligned}$$

To add mixtures into the model, we introduce mixture variable y and estimate the probability of responses through

$$\begin{aligned} p(\mathbf{w} \mid x) &= \sum_y \int p_\theta(\mathbf{w}, \mathbf{z}, y \mid x) dz \\ &= \sum_y p_\theta(y \mid x) \int p_\theta(\mathbf{w}, \mathbf{z} \mid x, y) dz \end{aligned} \tag{1}$$

Maximizing the objective above is intractable in general. We adopt mean-field variational inference and propose the variational posterior

$$q_\phi(\mathbf{z}, y \mid \mathbf{w}, x) = q_\phi(y \mid \mathbf{w}, x) q_\phi(\mathbf{z} \mid \mathbf{w}, x, y)$$

and the parameters is learned through maximizing the evidence variational lower bound (ELBO):

$$\begin{aligned} &\log p_\theta(\mathbf{w} \mid x) \geq \\ &\quad \mathbb{E}_q(y, \mathbf{z})[\log p_\theta(\mathbf{w} \mid x, y, \mathbf{z}) + \log p_\theta(y \mid x) + \\ &\quad \log p_\theta(\mathbf{z} \mid x, y, \mathbf{w}) - \log q_\phi(y, \mathbf{z} \mid x, \mathbf{w})] \end{aligned} \tag{2}$$

The prior distribution of $\mathbf{w}$ and $\mathbf{z}$ are parameterized as:

$$p_\theta(y \mid x) \sim Cat(\pi_\theta(x))$$

$$p_\theta(\mathbf{z} \mid x, y, \mathbf{w}) \sim \mathcal{N}(\mu_\theta(x, y, \mathbf{w}), diag(\sigma^2_\theta(x, y, \mathbf{w}))$$

and the posterior distributions as:

$$q_\phi(y \mid x, \mathbf{w}) \sim Cat(\pi_\phi(x, \mathbf{w}))$$

$$q_\phi(\mathbf{z} \mid x, y, \mathbf{w}) \sim \mathcal{N}(\mu_\phi(x, y, \mathbf{w}), diag(\sigma^2_\phi(x, y, \mathbf{w}))$$

3.2 DECODING

To search for best hypotheses, we score each candidate with the marginal likelihood of its words. However, marginalizing the continuous variable $\mathbf{z}$ can be tricky when decoding. To keep the algorithm deterministic (rather than randomized) and fast, we choose to be greedy at each time step, using $\mathbf{z}$ with maximum likelihood, which equals the mean of the prior Gaussian distribution. To marginalize out y, we break down the search process into two stages:

- **Nominate**. We perform normal beam search process for each mixture. The hypotheses are aggregated into one candidate pool.
- **Vote**. Each mixture scores all the candidates. Their scores are weighted according to the conditional prior of mixtures and added up for ranking candidates.

Algorithm 1 Pseudocode for decoding procedures.

```
procedure DECODE(src)
    c ← []
    for each decoder d do
        c ← c + BeamSearch(d, src)
    for h in c do
        h.score ← Σ_d P(h | src, d)P(d | src)
    return h with highest score
```

3.3 ARCHITECTURES

We use bi-directional long short-term memory network Schuster & Paliwal (1997); Hochreiter et al. (1997) to encode dialog history. Each mixture has its own LSTM for decoding. The last hidden states of the encoder, $\mathbf{h}^e_T$, are fed to a multi-layered perceptron (MLP) to obtain the initial hidden states of the decoder. Attention mechanism is used, and the previous hidden state of the decoder, $\mathbf{h}^d_{y,t-1}$, is used as the key to query encoder hidden states, and attention context, $\mathbf{c}_{y,t}$ is computed as the weighted sum of projected encoder hidden states.

$$d_{y,t} = \langle \mathbf{W}_e \mathbf{h}^e_t, \mathbf{W}^d_y \mathbf{h}^d_{y,T} \rangle \qquad e_{y,t} = \text{softmax}(\mathbf{d}_y) \qquad \mathbf{c}_{y,t} = \sum_i e_{y,i} \mathbf{h}^e_t$$

The prior mixture probability is calculated with the last hidden states of the encoder:

$$\pi_\theta(y) = MLP_{\pi,\theta}(\mathbf{h}^e_T)$$

and the prior distribution of $\mathbf{z}$ at each time step t are parameterized with the previous decoder hidden state and the current attention context:

$$\mu_{\theta,y}(z_t) = MLP_{\mu,\theta,y}([\mathbf{h}^d_{t-1}; \mathbf{c}_{y,t}])$$
$$\log \sigma^2_{\theta,y}(z_t) = MLP_{\sigma,\theta,y}([\mathbf{h}^d_{y,t-1}; \mathbf{c}_{y,t}])$$

We use an additional bi-directional LSTM encoder to encode responses into states $\mathbf{h}^p_t$ for posterior inference. The posterior distribution of mixtures is computed with the last hidden states of response encoder:

$$\pi_\phi(y) = MLP_{\pi,\phi}(\mathbf{h}^p_T)$$

and the posterior of $\mathbf{z}$ at each time step t is computed with the next hidden states of response encoder:

$$\mu_{\phi,y}(z_t) = MLP_{\mu,\phi,y}(\mathbf{h}^p_{t+1})$$
$$\log \sigma^2_{\phi,y}(z_t) = MLP_{\sigma,\phi,y}(\mathbf{h}^p_{t+1})$$

3.4 Semi-Supervised Learning

During experiments, we found degeneracy of latent mixtures to be an issue. This is similar to what is reported from Shen et al. (2019). They reported two types of degeneracy:

1. Only one decoder gets trained because "rich get richer". This means the decoder that learns fast will keep getting more weight. The other decoders end up being insufficiently trained.
2. The discrete latent variable is ignored. The prior and posterior of mixtures often end up with uniform distribution, or a multinomial distribution that varies very little across different dialog contexts.

In order to force the mixture model to make sharper decisions and be context-sensitive, we bootstrap labels to train the posterior network discriminatively. We do this by simply adding the log likelihood of labels to the training objective in Equation (2):

$$\mathcal{L}(\mathbf{w}, x, l) = \log q_\phi(y = l \mid \mathbf{w}, x)$$

The rest of the objective in (2) remains intact. Note that for labeled data, we still maximize the expected likelihood of words over all mixtures, instead of the likelihood from the labeled mixture. This is different from Kingma et al. (2014). We still allow each mixture to learn the data from other mixtures, but with lower weights. This is because we found through experiments that if trained with only a small subset of the data, the decoders of each mixture may generate responses with more specialized words, but in a highly monotonous way.

To bootstrap labels, we group utterances based on their lexical choices, in the hope that the resulting mixtures would reflect different topics. More specifically, we embed each utterance using the weighted sum of the Glove embeddings Pennington et al. (2014) of words in the utterance, normalized by subtracting the largest singular vector of all embeddings, as described in Arora et al. (2016). Then we perform k-means clustering on these embeddings, with the same number of clusters as mixtures in the response generation model. In addition, we assign "dull" responses to one specific mixture. Responses are considered "dull" if it contains n-grams ($n \geq 4$) that are highly frequent in those responses generated by sequence-to-sequence baseline. The mixture is specially handled in training. It still has access to responses from other mixtures, but the other mixtures do not learn resposes that are labeled as "dull".

3.5 Model Comparison

We now discuss the distinction between mixture of decoders and mixture of softmax in detail. The focus is on the efficient use of parameters. In mixture of decoders, the last hidden state of decoder,

	MoD	MoS
Output		✓
Bottleneck	✓	
Attention	✓	
RNN Decoder	✓	

Table 1: Parameter sharing in different layers in mixture of decoders (MoD) and mixture of softmax (MoS). Check mark indicates each mixture uses its own parameters.

	MoD	MoS
Order	$O(KEH)$	$O(KEV)$
K = 3	$< 9M$	$> 18M$
K = 5	$< 11M$	$> 30M$

Table 2: Comparison of the order and numeric values of the number of parameters used by MoD and MoS .

	Human	Seq2Seq	VAD	MoS	MoD	MoD (without z)	MoD (Supervised)
Length	10.35	8.93	7.88	**10.53**	12.34	9.24	14.01
BLEU	N/A	0.135	0.126	0.159	**0.169**	0.152	0.157
ROUGE-L	N/A	0.155	0.153	0.164	**0.170**	0.158	0.160
Cos. Sim. to targets	1.0000	0.0481	0.0603	0.0611	0.0699	**0.0820**	0.0715
Cos. Sim. to persona	0.1246	0.0614	0.0480	0.0491	0.0668	0.0515	**0.0810**
Distinct uni-gram	6327	353	566	424	617	590	**661**
Distinct bi-gram	44376	931	1559	1264	2106	1866	**2413**
Distinct tri-gram	77884	1499	2437	2159	3744	3224	**4415**

Table 3: Evaluation results on Persona dataset.

the attention context, and the continuous latent variable are used for word prediction. The concatenations of aforementioned representations are first passed through a bottleneck layer (for the sake of reducing the number of parameters):

$$\mathbf{b}_{y,t} = tanh(\mathbf{W}_b[\mathbf{h}^t_{d,y}; \mathbf{c}_{y,t}; \mathbf{z}_{y,t}])$$

The size of bottleneck is usually chosen to be the size of word embeddings. Then the output is used for calculating the distribution over words

$$p(w_t) = softmax(\mathbf{E}\mathbf{b}_{y,t})$$

Note that the weight matrix in the output layer is shared across mixtures, for reducing the number of parameters to learn. On the other hand, mixture of softmax first predicts mixture probabilities from the output of the bottleneck layer

$$\pi_{y,t} = softmax(\mathbf{W}_\pi \mathbf{b}_t)$$

Then the probability of words are marginalized over mixtures

$$p(w_t) = \sum_y \pi_{y,t} softmax(\mathbf{E}_y \mathbf{b}_t)$$

with each mixture using its own weights in the final output layer. The difference of how mixture of decoders and mixture of softmax share weights is summarized in Table 1.

Next, we compare the number of parameters used by mixture of decoders and mixture of softmax. Let V be the size of vocabulary, E be the size of word embeddings, K be the number of mixtures, and H be the size of any other hidden variables including attention contexts and decoder states. We calculate the order of number of parameters used by both models, and the actual values when the number of mixtures is at 3 and 5. The results are summarized in Table 2. It shows the number of parameters used by mixture of decoders is much smaller; the number of parameters used by mixture of softmax grows very fast with the number of mixtures.

Now we can see how mixture of decoders is advantageous over mixture of softmax. By utilizing distinct set of parameters for each mixture across multiple layers, mixture of decoders learns a rich set of latent representations while keeping the number of parameters small.

Context	Sorry to hear that. Making crafts takes my stress away. I enjoy it.
Persona	I love animals and have two dogs.
Baseline	What do you do for a living?
MoD	Do you like dogs? I love them.
Context	I grew up with Morgans. I love horses.
Persona	I also enjoy cooking meals with food from our garden.
Baseline	I do not have any pets. Do you have any pets?
MoD	Do you like cooking? I love to bake.
Context	Great! How are you?
Persona	I am married and have two kids of my own.
Baseline	I am good, just got back from the gym.
MoD	I am great. Just hanging out with my wife and two kids.

Table 4: Positive examples of generated responses from the MoD model with better reference to persona.

3.6 Implementation and Training Details

All models are implemented using PyTorch. The hidden size of all LSTM encoders and decoders are 512. The size of embeddings of words is 300. We trained our models using Adam optimization algorithm Kingma & Ba (2014) with learning rate 0.001. Dropout Srivastava et al. (2014) of rate 0.25 is used wherever applicable. Batch size is adjusted according to the number of mixtures. We trained our model on a single GPU (GTX 1080 Ti). We observed that the training speeds of mixture of decoders and mixture of softmax are usually 3 to 5 times slower than baseline models without mixtures.

4 Experiments and Analysis

4.1 Data

We evaluate our model for dialog response generation on Persona dataset Zhang et al. (2018), an open-domain dialog corpus. Each person is give a list of persona descriptions in simple sentences, and they are required to converse according to the given persona. We use the last utterance for each response as source, with persona descriptions prepended. During pre-processing, we truncate all trailing punctuations. The corpus has a vocabulary around 20,000 words.

4.2 Quantitative Analysis

We use the following sets of metrics for automatic evaluation of the quality of generated responses.

4.2.1 Count Statistics

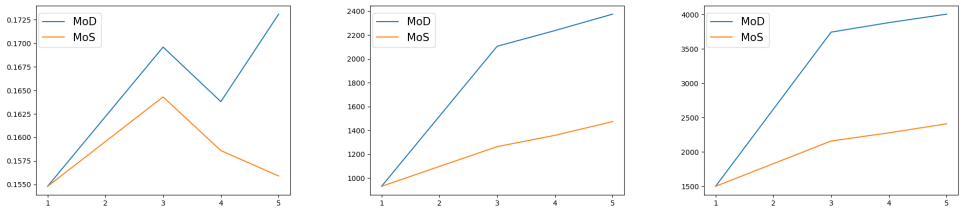
We show the average length of generated responses. To measure lexical diversity, we count the number of distinct n-grams in all the generated responses.

4.2.2 Word Overlap Based Metrics

We use BLEU Papineni et al. (2002) and ROUGE-L Lin (2004) scores as automatic evaluation metrics. While the reliability of such metrics has been criticized Liu et al. (2016), there is also evidence supporting the correlation of these metrics with human judgment for task-oriented domains Sharma et al. (2017).

4.2.3 Word Embedding Based Metrics

We evaluate the semantic similarity between generated responses and human responses and associated persona descriptions by the cosine distance of their sentence embeddings. We use the word averaging approach by Arora et al. (2016) to embed the responses, which has been demonstrated to be very good at capturing lexical level semantic similarity.



(a) ROUGE (b) Number of bi-grams (c) Number of tri-grams

Figure 3: Quantitative results by varying the number of mixtures. X-axis is for number of mixtures and y-axis for metrics. The number of mixtures are experimented with K = 1, 3, 4, 5.

The main results are listed in Table 3. The number of mixtures for MoS and MoD is 3. We also consider the baseline in which we remove the continuous latent variable from MoD. BLEU scores are correlated with lengths due to its brevity penalty. We found that MoS generate responses that have closest lengths to human responses on average. VAD tend to generate short responses. MoDs tend to generate longer responses. On manual inspection, we found that a fraction of those long responses generated by MoDs are self-repeating. As for metrics based on word overlapping, MoD beat baseline models. Adding continuous latent variables to MoD helps improve these metrics.

When it comes to semantic similarity based on embeddings, unsupervised MoD with discrete latent variables only performed the best on matching ground truth responses, while supervised MoD achieved the best result on matching persona descriptions. Full MoD also performed well comparing against baseline models.

As for lexical diversity. we can see that by adding either discrete or continuous latent variables into encoder-decoder models, better diversity is achieved, since VAD, MoS, and MoD all outperformed sequence-to-sequence baseline. Adding continuous latent variables helps improve lexical diversity, while adding supervision into MoD can make further improvement.

We also experimented with the effects of number of mixtures on MoS and MoD. The maximum number of mixtures is limited to 5 due to GPU memory constraint. Results are summarized in Figure 3. The general trend is that when the number of mixtures is small, increasing the number may improve lexical diversity, but not always help matching words in references, as shown in ROUGE-L. Note the trend that ROUGE-L starts to decline for MoS at $K = 3$.

4.3 Qualitative Analysis

Model	**Win**	**Loss**	**Tie**
Ours vs VAD	$47.0 \pm 12.1\%$	$25.0 \pm 10.3\%$	$28.0 \pm 19.8\%$
Ours vs MoS	$42.0 \pm 12.2\%$	$30.0 \pm 6.6\%$	$28.0 \pm 11.0\%$

Table 5: Human evaluation results.

	Example 1	**Example 2**	**Example 3**
Mix. 1	What kind of music do you like	Do you like movies ? I love horror movies	Do you like to watch TV
Mix. 2	What city are you from	Yes I do . I am a full time student	What are you majoring in
Mix. 3	Hi how are you today	I am great . How are you	That sounds like a lot of fun
Mix. 4	I am a huge fan of red hot chili peppers . They are my favorite	I am a huge fan of French fries . You	Do you like to cook ? I love it
Mix. 5	What do you do for a living	What do you like to do in your spare time	I do not have a lot of free time . What about you

Table 6: Examples of generated responses from each mixture in supervised MoD.

While the mixture model can improve word diversity, we need to verify the diversified responses are still as relevant. We ask 4 annotators to evaluate a randomly sampled subset of 100 examples. We compare our model to two baseline models: VAD and MoS with same number of mixtures. Each context is paired with two responses – one from the baseline model and one from our model. The annotators are asked to choose the best response, based on (1) appropriateness, and (2) relevance to given persona descriptions. Ties are allowed. The results are shown in Table 5. On average, about 30 percent of the time the annotators showed no preferences, and our mixture of decoders model are preferred over VAD and MoS for more than 40 percent of the time, and lost less than 30 percent of the trials. Note that all individual tests show annotators preferred MoD over baseline models, except for one case, where the annotator chose MoS over MoD slightly more often. We also show some of the examples on how MoD did well in Table 4.

To better understand what each mixture is learning about semantic topics, in Table 4, we select some examples from the most frequently generated responses for each mixtures. It can be observed that Mixture 1 tend to focus on hobbies, especially entertainment. Mixture 2 covers education. Mixture 3 contains conversation starters. Mixture 4 specifically concentrates on food. Mixture 5 tend to give general inquiries of occupation and leisure.

To evaluate how diverse are the responses generated by different mixtures, we follow Shen et al. (2019) and use pairwise BLEU to score the hypotheses set. The lower the pairwise BLEU scores, the more diverse the hypotheses. Additionally, we count the average number of distinct n-grams of hypotheses. We compare supervised MoD of 5 mixtures against the other baseline models. The top 5 hypotheses from the baseline are used, while the top hypothesis from each mixture in MoD are selected. The results are listed in Table 7. The sequence-to-sequence baseline has the highest pairwise BLEU. VAD, MoS, unsupervised MoD have slightly lower pairwise BLEU and slightly better lexical diversity in hypotheses sets. With supervision, pairwise BLEU score significantly reduced for MoD, indicating each mixture decoder became more specialized.

	Seq2Seq	VAD	MoS	MoD	MoD (Supervised)
BLEU	0.58	0.49	0.47	0.46	**0.22**
Distinct uni-gram	22.8	22.3	23.4	23.8	**24.5**
Distinct bi-gram	59.7	58.7	70.1	75.4	**86.9**
Distinct tri-gram	76.4	74.0	98.0	106.7	**122.9**

Table 7: Diversity evaluation of hypotheses sets.

5 Conclusion

In this paper we investigate the idea of combining mixture modeling with sequence-to-sequence model to improve the diversity issue in automated dialog response generation. Our model is developed under the framework of autoencoding variational Bayes. A discrete latent variable is introduced to capture latent dialog actions or topics. Such latent variable is easy to interpret. As illustrated in our qualitative analysis, each mixture specializes in certain topics to some extent. We also demonstrated the possibility of leveraging prior knowledge for learning latent topics through semi-supervised learning. By examining the hypotheses set, we confirm that the source of the superior diversity of mixture of decoders model is indeed the specialization of decoders. Our model performed reasonably well, comparing against other latent variable models and mixture models, in terms of automatic metrics. Moreover, our claim is backed by human evaluation through A/B test, and the responses generated by our model is often preferred over baseline models. As a future direction, we believe that a more sophisticated way (with more fine-grained linguistic features) to supervised the mixture of decoders model can further improve its performance.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. 2016.

Budhaditya Deb, Peter Bailey, and Milad Shokouhi. Diversifying reply suggestions using a matching-conditional variational autoencoder. In *Proceedings of the 2019 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers), pp. 40–47, 2019.

Jiachen Du, Wenjie Li, Yulan He, Ruifeng Xu, Lidong Bing, and Xuan Wang. Variational autoregressive decoder for neural response generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3154–3163, 2018.

Wenchao Du and Alan W Black. Boosting dialog response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 38–43, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1005.

Xuanli He, Gholamreza Haffari, and Mohammad Norouzi. Sequence to sequence mixture model for diverse machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pp. 583–592, 2018.

Sepp Hochreiter, J urgen Schmidhuber, and Corso Elvezia. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pp. 3581–3589, 2014.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, pp. 110–119, 2016.

Jiwei Li, Will Monroe, and Dan Jurafsky. Learning to decode for future success. *arXiv preprint arXiv:1701.06549*, 2017a.

Jiwei Li, Will Monroe, Tianlin Shi, Sėbastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2157–2169, 2017b.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

Pierre Lison and Serge Bibauw. Not all dialogues are created equal: Instance weighting for neural conversational models. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 384–394. Association for Computational Linguistics; Stroudsburg, PA, 2017.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2122–2132, 2016.

Yahui Liu, Wei Bi, Jun Gao, Xiaojiang Liu, Jian Yao, and Shuming Shi. Towards less generic responses in neural conversation models: A statistical re-weighting method. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2769–2774, 2018.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Alan Ritter, Colin Cherry, and William B Dolan. Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 583–593, 2011.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pp. 1577–1586, 2015.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *arXiv preprint arXiv:1706.09799*, 2017.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. Mixture models for diverse machine translation: Tricks of the trade. *arXiv preprint arXiv:1902.07816*, 2019.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pp. 3483–3491, 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Ilya O Tolstikhin, Sylvain Gelly, Olivier Bousquet, Carl-Johann Simon-Gabriel, and Bernhard Schölkopf. Adagan: Boosting generative models. In *Advances in Neural Information Processing Systems*, pp. 5424–5433, 2017.

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*, pp. 5998–6008, 2017.

Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

Zhen Xu, Bingquan Liu, Baoxun Wang, SUN Chengjie, Xiaolong Wang, Zhuoran Wang, and Chao Qi. Neural response generation via gan with an approximate embedding layer. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 617–626, 2017.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. Breaking the softmax bottleneck: A high-rank RNN language model. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HkwZSG-CZ.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2204–2213, 2018.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. Deconvolutional paragraph representation learning. In *Advances in Neural Information Processing Systems*, pp. 4169–4179, 2017.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 654–664, 2017.