

Diversifying Reply Suggestions using a Matching-Conditional Variational Autoencoder

Budhaditya Deb

Peter Bailey

Milad Shokouhi

Microsoft Search, Assistance and Intelligence
{budeb, pbailey, milads}@microsoft.com

Abstract

We consider the problem of diversifying automated reply suggestions for a commercial instant-messaging (IM) system (Skype). Our conversation model is a standard matching based information retrieval architecture, which consists of two parallel encoders to project messages and replies into a common feature representation. During inference, we select replies from a fixed response set using nearest neighbors in the feature space. To diversify responses, we formulate the model as a generative latent variable model with Conditional Variational Auto-Encoder (M-CVAE). We propose a constrained-sampling approach to make the variational inference in M-CVAE efficient for our production system. In offline experiments, M-CVAE consistently increased diversity by $\sim 30 - 40\%$ without significant impact on relevance. This translated to a 5% gain in click-rate in our online production system.

1 Introduction

Automated reply suggestions or smart-replies (SR) are increasingly becoming common in many popular applications such as Gmail (2016), Skype (2017), Outlook (2018), LinkedIn (2017), and Facebook Messenger.

Given a message, the problem that SR solves is to suggest short and relevant responses that a person may select with a click to avoid any typing. For example, for a message such as `Want to meet up for lunch?` an SR system may suggest the following three responses `{Sure; No problem!; Ok}`. While these are all relevant suggestions, they are semantically equivalent. In this paper, we consider how we can diversify the suggestions such as with `{Sure; Sorry I can't; What time?}` without losing any relevance. Our hypothesis is that encompassing greater semantic variability and intrinsic diversity will lead to higher click-rates for suggestions.

Smart-reply has been modeled as an sequence-to-sequence (S2S) process (Li et al., 2016; Kannan et al., 2016; Vinyals and Le, 2015) inspired by their success in machine translation. It has also been modeled as an Information Retrieval (IR) task (Henderson et al., 2017). Here, replies are selected from a fixed list of responses, using two parallel *Matching* networks to encode messages and replies in a common representation. Our production system uses such a Matching architecture.

There are several practical factors in favor of the *Matching-IR* approach. Production systems typically maintain a curated response-set (to have better control on the feature and to prevent inappropriate responses) due to which they rarely require a generative model. Moreover, inference is efficient in the matching architecture as vectors for the fixed response set can be pre-computed and hashed for fast lookup. Qualitatively, S2S also tends to generate generic, and sometimes incorrect responses due to label and exposure bias. Solutions for S2S during training (Wiseman and Rush, 2016) and inference (Li et al., 2016) have high overhead. Matching architectures on the other hand, can incorporate a global normalization factor during training to mitigate this issue (Sountsov and Sarawagi, 2016).

In practice we found that the Matching model retrieves responses which are semantically very similar in lexical content and underlying intent as shown in (Table 1). This behavior is not surprising and even expected since we optimize the model as a point estimation on golden message-reply (m-r) pairs. In fact, it illustrates the effectiveness of encoding similar intents in the common feature space. While this leads to individual responses being highly relevant, the model needs to diversify the responses to improve the overall relevance of the set by covering a wider variety of intents. We hypothesize that diversity would improve the click rates in our production system. This is the

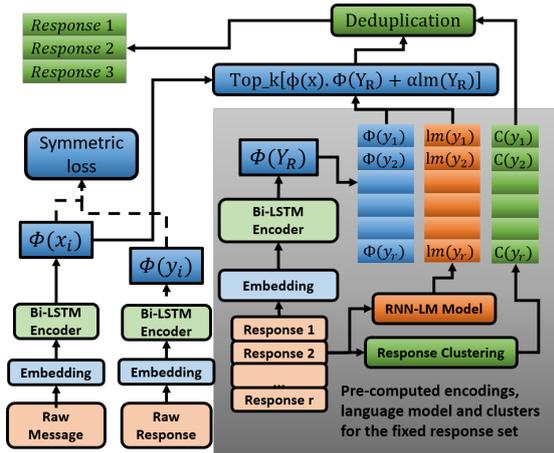


Figure 1: Training and inference graph for *Matching*. During inference, the response side stack is pre-computed (shaded grey).

main focus of this paper. We provide two baselines approaches using lexical clustering and maximal marginal relevance (MMR) for diversification in the Matching model.

Since we typically do not have multiple responses in one-on-one conversational data (and thus cannot train for multiple-intents), we consider a generative Latent Variable Model (LVM) to learn the hidden intents from individual m-r pairs. Our key hypothesis is that intents can be encoded through a latent variable, which can be then be utilized to generate diverse responses.

To this end, we propose the Matching-CVAE (M-CVAE) architecture, which introduces a generative LVM on the Matching-IR model using the neural variational autoencoder (VAE) framework (Kingma and Welling, 2014). M-CVAE is trained to generate the vector representation of the response conditioned on the input message and a stochastic latent variable. During inference we sample responses for a message and use voting to rank candidates. To reduce latency, we propose a constrained sampling strategy for M-CVAE which makes variational inference feasible for production systems. We show that the Matching architecture maintains the relevance advantages and inference-efficiency required for a production system while CVAE allows diversification of responses.

We first describe our current production model and diversification approaches. Next, we present our key contribution: Matching-CVAE. Finally we report on our results from offline and online experiments, including production system performance.

| Message | Matching | M-CVAE |
|--------------------------------|--|--|
| I'm betting on snow tomorrow. | I hope so I hope so too I hope so! I hope so too. I hope so. | I hope so too . As am I. I bet you are. You will not I might too |
| It's time for me to go to bed. | Go to sleep Go to sleep! Wake up Goodnight Wake up! | Goodnight Night! Same lol Sweet dreams Goodnight! |

Table 1: The top responses (without de-duplication) for Matching and M-CVAE.

2 Matching Model

Our training data consists of message reply (m-r) pairs $[x_i, y_i]$ from one-on-one IM conversations¹. A parallel stack of embedding and bi-directional LSTM layers encodes the raw text of m-r by concatenating the last hidden state of the backward and forward recurrences as $\Phi_X(x_i)$ and $\Phi_Y(y_i)$ (Figure 1). The encodings are trained to map to a common feature representation using the *symmetric-loss*: a probabilistic measure of the similarity as a normalized dot product $\Theta_{x_i y_i} = \Phi_X(x_i) \cdot \Phi_Y(y_i)$ in equation 1. We maximize the $-\ln p(\Theta)$ during training.

Note the denominator in the *symmetric-loss* is different from a softmax (where the marginalization is usually over the y terms) to approximate $p(y_i|x_i)$. Instead, it the sums over each message w.r.t. all responses and vice-versa. This normalization (analogous to a Jaccard index) in both directions enforces stronger constraints for a dialog pair². Thus, it is more appropriate for a conversational model where the goal is *conversation compatibility* rather than *content similarity*. Symmetric loss improved the relevance in our model. We omit the results here, to focus on diversity.

$$p(\Theta_{x_i y_i}) = \frac{e^{\Theta_{x_i y_i}}}{\sum_{y_j} e^{\Theta_{x_i y_j}} + \sum_{x_j} e^{\Theta_{x_j y_i}} - e^{\Theta_{x_i y_i}}} \quad (1)$$

$$S_k(x) = \text{softmax}_k[\text{top}_k[\Phi_X(x) \cdot \Phi_Y(Y_R) + \alpha \text{lm}(Y_R)]] \quad (2)$$

During inference, we pre-compute the response vectors $\Phi_Y(Y_R)$ for a fixed response set Y_R . We encode an input x as $\Phi_X(x)$, and find the K nearest responses Y_{R_k} , using a score composed of the dot product of $\Phi_X(x)$ and $\Phi_Y(Y_R)$ and a

¹Multi-user conversations were difficult to align reliably, given highly restricted access to preserve our users privacy.

²Li et al. (2016) made a similar argument with Mutual Information penalty during inference.

language-model penalty $lm(Y_R)^3$ in equation 2. The $lm(Y_R)$ is intended to suppress very specific responses similar to (Henderson et al., 2017). The α parameter is tuned separately on an evaluation set. We de-duplicate the Y_{R_k} candidates and select top three as suggested replies. The training and inference graph is shown in Figure 1.

2.1 Response Diversification

The matching model by itself, retrieves very similar responses as shown in Table 1. Clearly, the responses need to be de-duplicated to improve the quality of suggestions. We present two baseline approaches to increase diversity.

Lexical Clustering (LC): Table 1, motivates the use of simple lexical rules for de-duplication. We cluster responses which only differ in punctuations (Thanks!, Thanks.), contractions (cannot:can't, okay:ok), synonyms (yeah, yes, ya) etc. We further refine the clusters by joining responses with one-word edit distance between them (Thank you so much. Thank you very much) except for negations. During inference, we de-duplicate candidates belonging to the same clusters.

Maximal Marginal Relevance (MMR): As a way to increase the diversity in IR, (Carbonell and Goldstein, 1998) introduced the MMR criterion to penalize the query-document similarity with inter-document similarity to rank candidates using *marginal* relevance.

In the context of the SR, we apply the MMR principle as follows. First, we select the K candidates, (with scores $S_k(x)$ and response vectors $\Phi_Y(Y_{R_k})$) using equation 2. Next, we compute the novelty N_k (or marginal relevance) of the k^{th} response with respect to the other $K-1$ candidates using equation 3. Finally, we re-rank the candidates using the MMR score computed from equation 4. Our MMR implementation is an approximation of the original (which is iterative). Nevertheless, it allows the ranking in one single forward pass and thus is very efficient in terms of latency.

$$N_k = \frac{1}{K-1} \sum_{j \neq k}^K \text{CosSim}(\Phi_Y(Y_{R_k}), \Phi_Y(Y_{R_j})) \quad (3)$$

$$\text{MMR}_k(x) = \beta S_k(x) - (1 - \beta) N_k \quad (4)$$

Table 3 shows that LC and MMR are quite effective at reducing duplicates. We have also ex-

³We train an LSTM language model on the training data.

plored other clustering approaches using embeddings from unsupervised models, but they were not as effective as LC or MMR.

3 Matching-CVAE (M-CVAE)

Neither MMR nor LC solves the core issue with diversification i.e., learning to suggest diverse responses from individual m-r pairs. Privacy restrictions prevent any access to the underlying training data for explicit annotation and modeling for intents. Instead, we model the hidden intents in individual m-r pairs using a latent variable model (LVM) in M-CVAE.

In M-CVAE we generate a response vector conditioned on the message vector and a stochastic latent vector. The generated response vector is then used to select the corresponding raw response text.

M-CVAE relies on two hypotheses. First, the encoded vectors are accurate distributional indexes for raw text. Second, the latent variable encodes intents (i.e. a manifold assumption that similar intents have the same latent structure). Thus, samples from different latent vectors can be used to generate and select diverse responses within the Matching-IR framework.

We start with a base Matching model which encodes an m-r pair as $\Phi_X(x_i)$ and $\Phi_Y(y_i)$. We assume a stochastic vector z which encodes a latent intent, such that $\Phi_Y(y_i)$ is generated conditioned on $\Phi_X(x_i)$ and z . The purpose of learning the LVM is to maximize the probability of observations Φ_X, Φ_Y by marginalizing over z . This is typically infeasible in a high dimensional space.

Instead, the variational framework seeks to learn a posterior $Q_\phi(z|\Phi_X, \Phi_Y)$ and a generating $p_\varphi(\Phi_Y|\Phi_X, z)$ function to directly approximate the marginals. In the neural variational framework (Kingma and Welling, 2014) and the conditional variant CVAE (Sohn et al., 2015), the functionals Q_ϕ and p_φ are approximated using non-linear neural layers⁴, and trained using Stochastic Gradient Variational Bayes (SGVB).

We use two feed forward layers for Q_ϕ and p_φ as shown in equations 5 and 7. Here, \leftrightarrow denotes the concatenation of two vectors. To sample from Q_ϕ , we use the re-parameterization trick of Kingma (2014). First, we encode the input vectors interpreted as mean and variance $[\mu, \sigma^2]$. Next, we transform to a multivariate Gaussian form by

⁴Also referred as inference/recognition and reconstruction networks, they appear like an auto-encoder network.

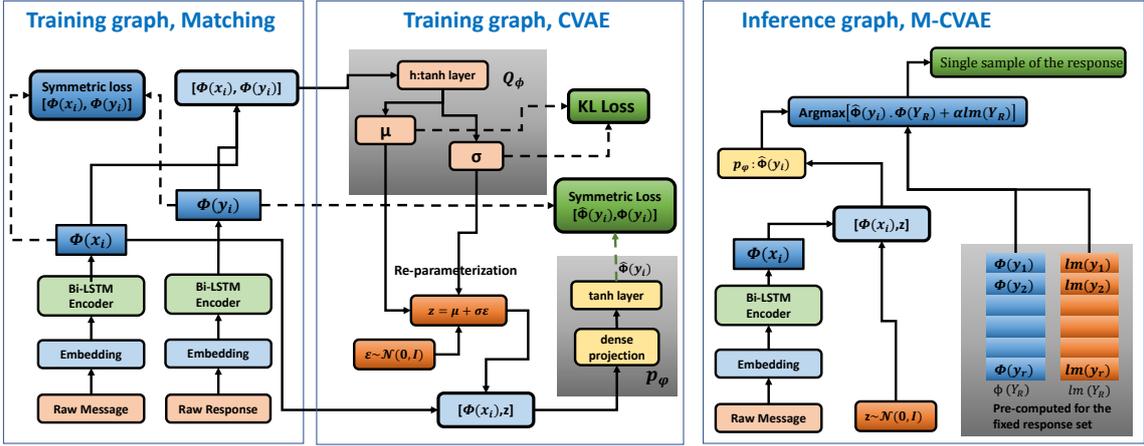


Figure 2: Matching (left) and CVAE (center) training models. Dotted arrows show the inputs to the loss functions. Right side shows the M-CVAE inference network where the shaded region shows the pre-computed values of the fixed response set..

sampling $\varepsilon \sim \mathcal{N}(0, I)$, and apply the linear transformation in equation 6. We reconstruct the response vector as $\hat{\Phi}_Y$ with p_ϕ (equation 7). Figure 2 shows the complete M-CVAE architecture.

The network is trained with the evidence lower bound objective (ELBO) by conditioning the standard VAE loss with message vector Φ_X in equation 8. The first term can be computed in closed form as it is the KL Divergence between two Normal distributions. The second term denotes the reconstruction loss for the response vector. We compute the reconstruction error using the symmetric loss, $p(\hat{\Phi}_Y(y_i), \Phi_Y(y_i))$ from equation 1 in the training minibatch. As is standard in SGVB, we use only one sample per item during training.

$$\begin{aligned} h &= \tanh(w_{\mu_1}^\phi \cdot \overleftarrow{\Phi_X} \Phi_Y^\rightarrow + b_{\mu_1}^\phi) \\ \mu &= w_{\mu_2}^\phi \cdot h + b_{\mu_2}^\phi \\ \sigma &= \exp((w_{\sigma_2}^\phi \cdot h + b_{\sigma_2}^\phi) / 2) \end{aligned} \quad (5)$$

$$z \sim Q_\phi = \mu + \sigma \cdot \varepsilon, \text{ where } \varepsilon \sim \mathcal{N}(0, I) \quad (6)$$

$$\hat{\Phi}_Y : p_\phi = w_2^\phi \cdot \tanh(w_1^\phi \cdot z \overleftarrow{\Phi_X}^\rightarrow + b_1^\phi) + b_2^\phi \quad (7)$$

$$\begin{aligned} ELBO &= -D_{KL}[Q_\phi(z|\Phi_X, \Phi_Y) \| p(z|\Phi_X, \Phi_X)] \\ &\quad + E[\ln p_\phi(\Phi_Y|z, \Phi_X)] \end{aligned} \quad (8)$$

$$Predict : \text{Argmax}_{Y_R}[\hat{\Phi}_Y(y_i) \cdot \Phi_Y(Y_R) + lm(Y_R)] \quad (9)$$

3.1 Inference in CVAE

During inference, We pre-compute the response vectors $\Phi_Y(Y_R)$ and $lm(Y_R)$ scores as before. However, instead of matching the message vector with the response vectors, we find the nearest neighbors of the *generated* response vector, $\hat{\Phi}_Y$

with $\Phi_Y(Y_R)$. We use a sampling and voting strategy to rank the response candidates.

Sampling Responses: To generate $\hat{\Phi}_Y$, we first sample $z \sim \mathcal{N}(0, I)$, concatenate with $\Phi(x)$ and generate $\hat{\Phi}(y)$ with the decoder p_ϕ from equation 7. The sampling process is shown Figure 2 (right).

Voting Responses: The predicted response sample for a given input and a z sample is given by equation 9. In each sample, a candidate response (argmax) gets the winning vote. We generate a large number of such samples and use the total votes accumulated by responses as a proxy to estimate the likelihood $p(y|x)$. Finally, we use the voting-score to rank the candidates in M-CVAE.

3.2 Constrained sampling in CVAE

To deploy M-CVAE in production we needed to solve two issues. First, generating a large number of samples significantly increased the latency compared to Matching. Reducing the number of samples leads to higher variance where M-CVAE can sometimes select diverse but irrelevant responses (compared to Matching which selects relevant but duplicate responses). We propose a *constrained sampling* strategy which solves both these problems by allowing better trade off between diversity and relevance at a reduced cost.

We note that the latency bottleneck is essentially in the large dot product with pre-computed response vectors (our response set size is $\sim 30k$) in equation 9. Here, the number of matrix multiplications for N samples is $600 * 30000 * N$ (with encoding dimension size of 600). However, during the sampling process, only a few relevant can-

didates actually get a vote. Thus, we can reduce this cost by pre-selecting top K candidates using the Matching score (eq. 2) and then pruning the response vector to the selected K candidates. This *constrains* the dot-product in each sampling step to only K vectors, and reduces the number of matrix multiplications for N samples to $600 * K * N$, where $K \ll 30000$.

By pruning the response set, we are able to fit all the sampling vectors within a single matrix, and apply the entire sampling and voting step as matrix operations in one forward pass through the network. This leads to an extremely efficient graph and allows us to deploy the model in production.

Sampling with MMR: As seen in Table 1, the candidates selected using Matching score can have very low diversity to begin with and can reduce the effectiveness of M-CVAE. To diversify the initial candidates, we can use our MMR ranking approach as follows. We first select top $2K$ responses using Matching and compute the MMR scores from equation 4. Next, we use the MMR scores to select the top K diverse responses for use in constrained sampling in M-CVAE.

All the inference components (Matching, MMR, and constrained sampling), when applied together requires just one forward pass through the network. Thus, we can not only trade-off diversity and relevance, but also control the latency at the same time. Constrained sampling was critical for deploying to production systems.

4 Experiments and Results

Our current production model in Skype is a parallel Matching stack (Figure 1) with embedding size of 320 and 2 Bi-LSTM layers with hidden size of 300 for both messages and replies. The token vocabulary is $\sim 100k$ (tokens with a minimum frequency of 50 in the training set), and the response set size is $\sim 30k$. It selects top 15 candidates and de-duplicates using lexical clustering to suggest three responses. The entire system is implemented on the Cognitive Toolkit (CNTK) which provides efficient training and run-time libraries, particularly suited to RNN based architectures.

We analyze the M-CVAE model in comparison to this production model ⁵. The production model is also used as the control for online A/B

⁵Since the production model has gone through numerous parameter tuning and flights, we assume it to be a strong baseline to compare with.

testing, so it is natural to use the same model for offline analysis. To train the M-CVAE, we use the base Matching model, freeze its parameters, and then train the CVAE layers on top. We apply a dropout rate of 0.2 after the initial embedding layer (for both Matching and M-CVAE) and use the Adadelta learner for training. We use the loss on a held out validation set for model selection.

Training data: We sample ~ 100 million pairs of m-r pairs from one-on-one IM conversations. We filter out multi-user and multi-turn conversations since they were difficult to align reliably. We set aside 10% of the data to compute validation losses for model selection. The data is completely eyes-off i.e., neither the training nor the validation set is accessible for eyes-on analysis.

Response set: To generate the response set, we filter replies from the m-r pairs with spam, offensive, and English vocabulary filters and clean them of personally identifiable information. Next, we select top 100k responses based on frequency and then top 30k based on lm-scores. We pre-compute the lm-scores, lexical-clusters and encodings for the response set and embed them inside the inference graphs as shown in Figure 1 and 2.

Evaluation metrics and set: The model predicts three responses per message for which we compute two metrics: *Defects* (a response is deemed incorrect) and *Duplicates* (at least 2 out of 3 responses are semantically similar). We use crowd sourced human judgments with at least 5 judges per sample. Judges are asked to provide a binary *Yes/No* answer on defects and duplicates. Judge consensus (inter annotator agreement) of 4 and above is considered for metrics, with 3 deemed as *no-consensus* (around 5%). Since training/validation sets are not accessible for analysis, we created an *evaluation* set of 2000 messages using crowd sourcing for reporting our metrics.

M-CVAE parameters: We consider three parameters for ablation studies in M-CVAE: size of latent vector z , number of samples s and the response pruning size k for constrained sampling. The results are shown in Table 2. The M-CVAE numbers (row 2 onwards) are relative to the base Matching model in row 1. First, row 2 shows that latent vector size of 256 provides a suitable balance between defects and duplicates, but in general, the size of the latent variable is not a significant factor in performance. Next, in row 3, we see that the response-pruning size k , is an effective

| | Fixed params | Ablation | Defects | Duplicates |
|-----------------------------------|--|---------------|--------------|---------------|
| 1. Matching 2-layer BiLSTM | | | | |
| | | - | 7.76 | 31.66 |
| 2. M-CVAE | k=15, s=300 different latent vector size | z = 64 | +2.46 | -13.39 |
| | | z = 128 | +0.71 | -14.48 |
| | | z = 256 | +0.24 | -11.72 |
| | | z = 512 | +1.74 | -12.62 |
| | | k = 15 | +0.24 | -11.72 |
| 3. M-CVAE | z=256, s=300 different top k constraint size | k = 30 | +2.82 | -14.55 |
| | | k = 50 | +2.97 | -14.93 |
| | | k = 100 | +3.2 | -15.32 |
| | | Un-const | +3.26 | -15.71 |
| | | s = 100 | +1.64 | -15.77 |
| 4. M-CVAE | k=15, z=256 different # samples | s = 200 | +1.15 | -14.42 |
| | | s = 300 | +0.24 | -11.72 |
| | | s = 1000 | +0.15 | -11.62 |

Table 2: Relative change in *Defect* and *Duplicate* metrics for different hyper-parameters of M-CVAE w.r.t. to the baseline Matching (row 1). In all cases, M-CVAE significantly reduces duplicates with minor increase in defects w.r.t. to the baseline model. Row highlighted in green is the configuration chosen for online A/B test.

| | Matching | | M-CVAE | |
|---------------------------|-------------|--------------|-------------|--------------|
| | Defects | Duplicates | Defects | Duplicates |
| w/o LC | 7.18 | 48.57 | 7.46 | 24.32 |
| + LC | 7.76 | 31.66 | 8.00 | 19.94 |
| + LC, MMR, β : 0.01 | 8.33 | 29.72 | 9.80 | 16.98 |
| + LC, MMR, β : 0.05 | 9.97 | 23.29 | 9.65 | 15.76 |
| + LC, MMR, β : 0.1 | 12.03 | 19.43 | 10.18 | 15.05 |
| + LC, MMR, β : 0.2 | 13.25 | 16.08 | 11.13 | 13.64 |
| + LC, MMR, β : 0.5 | 13.57 | 13.25 | 11.68 | 12.12 |

Table 3: Contribution of baseline diversification techniques, LC and MMR on duplicate reduction.

control to trade-off defects and duplicates. Thus, constrained sampling not only reduces the latency but also provides quality control required in a production system. In row 4, we see that more samples lead to better metrics but the improvements are marginal beyond 300 samples. In all cases, M-CVAE significantly reduces duplicates (by as much as 40%) without any major increase in defects. We select the model with hyper-parameters $[k = 15, z = 256, s = 300]$ for further analysis.

Diversification with LC: The first two rows of Table 3 analyzes the impact of LC based de-duplication. LC can significantly reduce the duplicates in the base matching model. However, M-CVAE (even without LC) reduces the rates by almost 50% as shown in column 4 in row 1. Using LC as a post processing step after M-CVAE, can give further boosts in diversity (row 2).

Diversification with MMR: Table 3 also reports the impact of MMR re-ranking. For Matching+MMR, duplicates can reduce significantly as we increase the β parameter, but at the cost of

| | Common params | Defects | Duplicates |
|----------|--------------------|-------------|--------------|
| S2S | 2 layers LSTM | 13.27 | 22.1 |
| Matching | 3 layers, FF | 8.70 | 36.77 |
| Matching | 2 layers, BiLSTM | 7.76 | 31.66 |
| M-CVAE | S=300, k=15, z=256 | 8.00 | 19.94 |

Table 4: Summary metrics comparing architectures.

| Architecture | Samples | Top K | Latency (ms) |
|--------------------|---------|-----------|--------------|
| S2S beam search | - | 15 (beam) | 207.5 |
| Matching (FF) | - | - | 7.12 |
| Matching (BiLSTM) | - | - | 9.08 |
| Matching + MMR | - | 15 | 9.6 |
| M-CVAE (un-const) | 300 | - | 99.12 |
| M-CVAE (const) | 300 | 15 | 10.03 |
| M-CVAE(const)+ MMR | 300 | 30/15 | 10.67 |

Table 5: Latency (in ms) in production servers.

increased defects. With MMR+M-CVAE, further diversification can be achieved, and typically at a lower defect rate. This shows the advantage of using M-CVAE which conditions the responses on the message and hence has stronger controls on the relevance than MMR.

Comparison with other architectures: We have considered two other architectures for our SR system. First is a standard S2S with attention (Bahdanau et al., 2014) with equivalent parameters for embedding and LSTMs as our base model, and inference using beam search decoding with width 15. Second, is a feed-forward (instead of an LSTM) based Matching encoder architecture which is equivalent to the one in (Henderson et al., 2017). All models use LC for de-duplication after 15 candidate responses are selected. Table 4 validates our architectural preference towards Matching/Bi-LSTM which has a superior performance in terms of defects.

Inference latency: Architecture choices were also driven by latency requirements in our production system. The results are summarized in Table 5 for different architectures. S2S and unconstrained sampling in M-CVAE were unsuitable for production due to their high latencies. With constrained sampling (including MMR), the latency increases marginally compared to the base model, and allows us to put the model in production.

Online experiments: Offline metrics were used principally for selecting the best candidate models for online A/B experiments. We selected M-CVAE model with parameters $[z=256, k=15, s=300]$ from Table 2. Using our existing production model as the control, and a treatment group consisting of 10% of our IM client users (with

| | % Change | P-Value |
|--|---------------|---------------|
| Overall Click Rate (user level) | +4.71% | 0.0003 |
| Position 1 Click Rate | +1.01% | 0.5558 |
| Position 2 Click Rate | +10.32% | 0.0001 |
| Position 3 Click Rate | +6.71% | 0.019 |

Table 6: Click rates for the M-CVAE flighted model. The control is the Matching model in production.

the same population properties as the control), we conducted an online A/B test for two weeks. Table 6 shows that the click-rate for M-CVAE compared to the Matching model increased by ~5% overall.

Gains were driven by the increase in the 2nd (10.3%) and 3rd (6.7%) suggested reply positions with virtually no impact in the 1st position. This correlates with our offline analysis since M-CVAE typically differs from the base model at these two positions. Intuitively, the three positions point to the *head*, *torso* and *tail* intents of responses⁶. Gains at these positions show that M-CVAE extracts diverse responses without sacrificing the relevance of these tail intents.

Driven by these gains, we have switched our production system in Skype to use M-CVAE for 100% of users.

5 Related work

Several researchers have used CVAEs (Sohn et al., 2015) for generating text (Miao et al., 2016; Guu et al., 2018; Bowman et al., 2016), modeling conversations (Park et al., 2018), diversifying responses in dialogues (Zhao et al., 2017; Shen et al., 2017) and improving translations (Schulz et al., 2018). These papers use S2S architectures which we found impractical for production. We demonstrate similar objectives without having to rely on any sequential generative process, in an IR setting.

VAE has been also used in IR (Chaidaroon and Fang, 2017) to generate hash maps for semantically similar documents and top-n recommendation systems (Chen and de Rijke, 2018). In contrast, we demonstrate semantic-diversity in intents in a conversational IR model with M-CVAE.

Novelty and diversity are well-studied problems in IR (Yue and Joachims, 2008; Clarke et al., 2008) where it is assumed that document topics are available (and not latent) during training. Diversification effect as shown in (Chen and Karger, 2006) relies on relevance (click) data, and thus is not

⁶which was validated by the absolute click rates for each of these positions but not shown in the table

directly applicable in our system. MMR (Carbonell and Goldstein, 1998) is a relevant prior work which we use as a baseline.

6 Conclusions

We formulate the IR-based conversational model as a generative LVM, optimized with the CVAE framework. M-CVAE learns to diversify responses from single m-r pairs without any supervision. Online results show that diversity increases the click rates in our system. Using efficient constrained sampling approach, we have successfully shipped the M-CVAE model to production.

Increase in click rates over millions of users is incredibly hard. We have also experimented with the M-CVAE model trained for suggesting replies to emails in Outlook Web App (significantly different characteristics than IM) and seen similar gains. The results across domains suggests strong generalization properties of the M-CVAE model and validates our hypothesis that increased diversity leads to higher click-rates by encompassing greater semantic variability of intents.

Perhaps the most important quality of the M-CVAE is that response vector can be flexibly conditioned on the input and thus a transduction process. In contrast, in the Matching IR model, response vectors are pre-computed and independent of the input. M-CVAE thus opens up new avenues to improve the quality of responses further through personalization and stylization. This is the subject of future work.

Acknowledgments

We gratefully acknowledge the contributions of Lei Cui, Shashank Jain, Pankaj Gulhane and Naman Mody in different parts of our production system on which this work builds upon. We also thank Chris Quirk and Kieran McDonald for their insightful feedback during the initial development of this work. Finally, we thank our partner teams (Skype, infrastructure, and online experimentation) for their support.

References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. [Neural Machine Translation by Jointly Learning to Align and Translate](#). *CoRR*, abs/1409.0473.
- Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Ben-

- gio. 2016. Generating sentences from a continuous space. In *CoNLL*.
- Jaime Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *SIGIR*.
- Suthee Chaidaroon and Yi Fang. 2017. Neural Variational Inference for Text Processing. In *SIGIR*.
- Harry Chen and David R. Karger. 2006. Less is more. Probabilistic models for retrieving fewer relevant documents. In *SIGIR*.
- Yifan Chen and Maarten de Rijke. 2018. A Collective Variational Autoencoder for Top-N Recommendation with Side Information. *CoRR*, abs/1807.05730.
- Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and Diversity in Information Retrieval Evaluation. In *SIGIR*.
- CNTK. [The Microsoft Cognitive Toolkit. https://www.microsoft.com/en-us/cognitive-toolkit/](https://www.microsoft.com/en-us/cognitive-toolkit/).
- Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating Sentences by Editing Prototypes. *Transactions of the Association of Computational Linguistics*, 6:437–450.
- Matthew Henderson, Rami Al-Rfou’, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient Natural Language Response Suggestion for Smart Reply. *CoRR*, abs/1705.00652.
- Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Gregory S. Corrado, László Lukács, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. Smart Reply: Automated Response Suggestion for Email. In *KDD*.
- Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. *ICLR*.
- Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B. Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *HLT-NAACL*.
- Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural Variational Inference for Text Processing. In *ICML*.
- Microsoft. 2018. [Designed to be fast - The Outlook on the web user experience gets simpler and more powerful.](#)
- Yookoon Park, Jaemin Cho, and Gunhee Kim. 2018. A Hierarchical Latent Structure for Variational Conversation Modeling. In *NAACL*.
- Jeff Pasternack and Nimesh Chakravarthi. 2017. [Building Smart Replies for Member Messages.](#)
- Philip Schulz, Wilker Aziz, and Trevor Cohn. 2018. A Stochastic Decoder for Neural Machine Translation. In *ACL*.
- Xiaoyu Shen, Hui Su, Yanran Li, Wenjie Li, Shuzi Niu, Yang Zhao, Akiko Aizawa, and Guoping Long. 2017. A Conditional Variational Framework for Dialog Generation. In *ACL*.
- Skype-Team. 2017. [Introducing Cortana in Skype.](#)
- Kihyuk Sohn, Honglak Lee, and Xinchun Yan. 2015. Learning Structured Output Representation using Deep Conditional Generative Models. In *NIPS*.
- Pavel Soutsov and Sunita Sarawagi. 2016. Length bias in Encoder Decoder Models and a Case for Global Conditioning. In *EMNLP*.
- Oriol Vinyals and Quoc V. Le. 2015. A Neural Conversational Model. In *ICML Deep Learning Workshop*.
- Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-Sequence Learning as Beam-Search Optimization. In *EMNLP*.
- Yisong Yue and Thorsten Joachims. 2008. Predicting Diverse Subsets Using Structural SVMs. In *ICML*.
- Tiancheng Zhao, Ran Zhao, and Maxine Eskénazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *ACL*.