Automating Template Creation for Ranking-Based Dialogue Models

Jingxiang Chen **Heba Elfardy** **Simi Wang** **Andrea Kahn** **Jared Kramer**
{jxchen,helfardy,simiwang,kahna,jaredkra}@amazon.com
Amazon
Seattle, WA, USA

Abstract

Dialogue response generation models that use template ranking rather than direct sequence generation allow model developers to limit generated responses to pre-approved messages. However, manually creating templates is time-consuming and requires domain expertise. To alleviate this problem, we explore automating the process of creating dialogue templates by using unsupervised methods to cluster historical utterances and selecting representative utterances from each cluster. Specifically, we propose an end-to-end model called Deep Sentence Encoder Clustering (DSEC) that uses an auto-encoder structure to jointly learn the utterance representation and construct template clusters. We compare this method to a random baseline that randomly assigns templates to clusters as well as a strong baseline that performs the sentence encoding and the utterance clustering sequentially.

To evaluate the performance of the proposed method, we perform an automatic evaluation with two annotated customer service datasets to assess clustering effectiveness, and a human-in-the-loop experiment using a live customer service application to measure the acceptance rate of the generated templates. DSEC performs best in the automatic evaluation, beats both the sequential and random baselines on most metrics in the human-in-the-loop experiment, and shows promising results when compared to gold/manually created templates.

1 Introduction

Dialogue response generation has been an active area of research in recent years. Response generation can be used in human-to-bot conversational systems (Qiu et al., 2017) or to generate quick replies in human-to-human conversational systems (Kannan et al., 2016; Pasternack et al., 2017).

Response generation approaches fall under two broad categories: (1) direct sequence generation using an encoder-decoder architecture (Vinyals and Le, 2015; Serban et al., 2016) or (2) response ranking, in which the model developer specifies a predefined template pool and an encoder model is used to score pairs of conversation history and candidate template response (Liu et al., 2018; Zhou et al., 2018; Kannan et al., 2016). Using template ranking rather than direct sequence generation allows model developers to limit generated responses to pre-approved messages, preventing the model from producing impolite or ungrammatical responses. In addition, sequence generation models have a tendency to favor safe, generic responses (Baheti et al., 2018; Shao et al., 2017; Zhang et al., 2018; Li et al., 2016), and template ranking models can be used to ensure that the system generates information-rich responses that drive the conversation towards an end goal. However, manually creating templates is time-consuming and requires domain expertise. For certain use cases such as customer service, templates need to be continually updated to reflect policy changes, further adding to this cost. In addition, manually created templates may differ subtly from actual agent utterances in model training data and thus may not be selected by the ranking model.

In this paper, we explore automating the creation of a template pool for a customer service chat application through clustering historical agent utterances and choosing representative utterances from each cluster. To the best of our knowledge, research on automatic template creation using utterance clustering has been limited.

The structure of this paper is as follows. In section 2, we describe the data and text preprocessing methods we used to extract template candidates from historical chat transcripts. In section 3, we describe our proposed approach for template generation: an end-to-end approach that uses an auto-encoder structure to jointly learn the utterance representation and construct template clusters. In addition, we describe a strong baseline that we propose for comparison: a sequential approach in which we first learn the utterance representation and then construct template clusters. In section 4, we describe the automatic and human-in-the-loop evaluations that we conducted and our findings, and in section 5 we draw conclusions and propose future research directions.

2 Data

We select template responses from a dataset of agent utterances extracted from historical chat transcripts. To construct this dataset, we collect anonymized transcripts of conversations between customers and customer service agents (CSAs) in two domains: (1) Cancel Membership (CM), and (2) Tracking shows delivered but order not received (DNR). In the anonymized transcripts, all unique customer identifiers (UCI) are replaced with a special token: "*GENERIC_SLOT*". We further extract all agent utterances[1] in these transcripts and exclude those occurring only once in the data. The intuition behind this is that if an utterance only occurred once, it is not likely to be useful as a template. We end up with approximately 550K agent utterances in each domain. The DNR domain contains longer utterances than the CM domain (an average of 12 words per sentence vs. 11 for CM) and a larger vocabulary size (22.9K for DNR vs. 19.2K for CM).

2.1 Annotation Guidelines

To create our evaluation data, we select a random sample of approximately 1,000 utterances from each domain and have it annotated for "Cluster ID". For the annotation task, we ask the annotators to come up with cluster IDs as they are annotating the utterances and then consolidate these clusters after they are done assigning all utterances to clusters. We have one annotator per domain and a gold annotator that further refines the clusters for both domains. For each domain we ask the annotator to do the following:

1. Starting with the first utterance, define the first cluster to convey the semantic meaning of this utterance and give a descriptive name for the cluster.
2. For each utterance in the list, either assign it to an existing cluster (if appropriate) or define a new cluster.
3. When assigning utterances to clusters, ignore the tense and utterance type (statement versus question).
   E.g., *"I canceled your membership"*, *"I will cancel your membership"*, and *"Should I cancel your membership?"* will all belong to the same cluster.
4. All noisy/unneeded utterances that are not related to the current domain or that do not contain information that can be useful for resolving the customer's issue should be excluded.
5. After finishing all of the utterances, go through the list of clusters to merge redundant ones and map the utterances to the new list of cluster IDs.

The annotation process resulted in 44 and 43 clusters for the CM and DNR domains respectively. Table 1 shows sample utterances from some clusters.

3 Approach

We cluster agent utterances using a novel end-to-end approach, Deep Sentence Encoder Clustering (DSEC), in which the utterance representation and the clustering model are jointly learned. We compare this against two baselines: (1) a weak baseline in which templates are sampled randomly from the dataset, and (2) a sequential baseline in which the utterance representation and the clustering model are learned sequentially. For the baseline system, we use dense features to represent each utterance and explore the use of different embedding types—GloVe (Pennington et al., 2014), ELMo (Peters et al., 2018b,a), and BERT (Devlin et al., 2018)—as well as the effect of using in-domain data on the performance of the system.

For both DSEC and the sequential baseline, after the clusters have been obtained, we create the template pool by selecting the highest-confidence utterance in each cluster. The confidence is either the probability that the utterance falls in the cluster (for DSEC), or the distance between the utterance and its cluster centroid (for the sequential baseline).

[1]"Utterance" is defined as all that is typed before sending the message to the customer.

Domain	Cluster Description	Utterances
CM	Informing the customer of confirmation e-mail	Your refund of GENERIC_SLOT will be credited to your original payment method within 7 to 10 business days.
CM	Confirming refund request	I see that you have used the membership benefits, and because of that I can offer GENERIC_SLOT refund. Sounds good?
CM	Greeting	Good afternoon.
CM	Asking for confirmation	Can you please confirm the last four digits and the expiration date of the payment method that has been charged?
DNR	Confirming refund options	Would you like the refund to be back on your gift or credit card?
DNR	Apology	I do apologize for the inconvenience if it was tagged as delivered but nowhere to be found.
DNR	Confirming order status	It seems that the package was already lost and mismarked as delivered.

Table 1: Sample agent utterances for our two domains: Cancel membership (CM) and Tracking shows delivered but item not received (DNR)

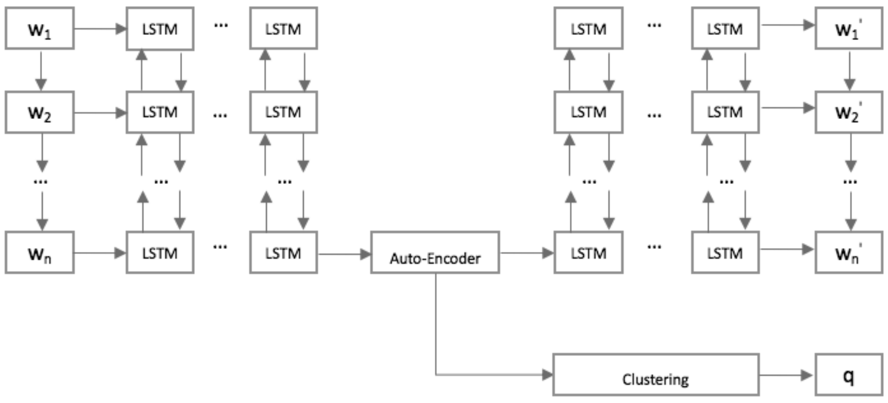

Figure 1: DSEC: An LSTM auto-encoder representing the sentence encoder and a clustering layer

3.1 Deep Sentence Encoder Clustering (DSEC)

We propose an end-to-end auto-encoder structure (Figure 1) that learns a sentence encoding layer that aims to achieve two goals simultaneously: (1) generate a feature representation from which the input utterance can be reconstructed as accurately as possible, and (2) construct template clusters by introducing a clustering-oriented loss. To achieve these two goals, we minimize a weighted (w) sum of reconstruction loss (L_r) and clustering loss (L_c).

$$L = L_r + \omega * L_c$$

To build the auto-encoder structure, we utilize a deep bi-directional Long Short-Term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997). We first use a word embedding layer, and then train a multi-layer bi-directional LSTM as the encoder. We choose a bi-directional network since subsequent words can sometimes facilitate the prediction of previous words. For example, it is easy to infer that the previous word has a high probability of being "*I*" if we know that the current word is "*am*". The final output of the hidden layer is then used as the input to the decoder. Padding is used to normalize sentence length, and a softmax function is added on top of the decoder to reconstruct the input. It is intuitive that the vectors generated by the encoder are good representations of the sentences they encode if they contain enough information to reconstruct these sentences.

For clustering, we define the loss using a soft assignment between the sentence embedding and the cluster centroids, similar to Xie et al. (2016). In particular, we first use the Student's *t-distribution* as a kernel to measure the similarity between the sentence encoder z_i and each of the centroid points

μ_j:

$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2/\alpha)^{-(\alpha+1)/2}}{\sum_{j'}(1 + \|z_i - \mu_{j'}\|^2/\alpha)^{-(\alpha+1)/2}}$$

where q_{ij} indicates the probability of assigning sentence i to cluster j. The degree of freedom α is set to be 1. The sentence clustering loss is defined as:

$$L = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

in which the soft target distribution P is defined as:

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_j (q_{ij}^2 / \sum_i q_{ij})}$$

One potential deficiency of using the target distribution, as Guo et al. (2017) pointed out, is that such a loss emphasizes data points with large p_{ij} (i.e. high confidence) hence is less impacted by mistakes for the points farther away from the centroid or ones that are close to the decision boundary hence can lead to underfitting if many such points exist. This problem can be more severe in sentence clustering than image clustering since image clustering usually has a more well defined objective whereas sentence clustering can be ambiguous and subjective. We find that different annotators often suggest different cluster labels for many of the sentences. To alleviate this issue, we suggest setting a threshold on the probability q_{ij} to filter out utterances with weak cluster signals when tuning or evaluating the model. Note that our goal is to select representative utterances from each cluster to form a reliable template pool. In this way it is most important to maximize the quality of utterances with high estimated confidences.

In practice, we initialize the reconstruction coefficients by first training the auto-encoder separately, i.e. setting $\omega = 0$. This "*warm-start*" approach helps accelerate the convergence rate.

Our proposed method borrows the loss from Xie et al. (2016) but addresses a different problem. First, Xie et al. (2016) use a convolutional network to learn an image representation. We target sentence reconstruction along with clustering, and thus propose an LSTM structure to capture the time series aspect of the sequence. Second, we use a pre-trained model fit on our own customer service data to initialize the parameters, and thus our model does not have to be very deep, which makes it less computationally intensive to train.

3.2 Sequential Baseline

Since there is no prior research targeting the task of automating template creation for ranking-based dialogue models, we propose a strong baseline that embeds the utterances and clusters them sequentially. To ensure that the baseline we are comparing against is effective, we explore the use of publicly available/pretrained embedding models versus models that are trained on in-domain customer service data. Additionally, we experiment with a traditional word embedding model, GloVe, in which the representation of each word in the vocabulary is the same regardless of the context it is appearing in, as well as contextual embeddings in which the representation depends on the entire context in which a word is used, namely ELMo and BERT. For in-domain data, we use approximately 118 million utterances to train a customer service (CS) GloVe model and an attention-based ELMo model. Once we obtain the representation for each utterance using a specific embedding model, we then use a pooling layer to obtain the utterance representation. For the pooling layer, we use weighted-mean pooling, in which each word is weighted by the "Term Frequency Inverse Document Frequency" (tf-idf) score (Aizawa, 2003), with documents defined as utterances in this case.[2]

Finally, we cluster the utterance representations. We experiment with K-means (MacQueen et al., 1967), AffinityPropagation (Frey and Dueck, 2007), spectral (Shi and Malik, 2000), Ward's (Murtagh and Legendre, 2014), Agglomerative (Müllner, 2011) and Birch (Zhang et al., 1997) clustering. For K-means, we use the centroid as the representation of the cluster, while for other algorithms, we take the mean pooling for all templates in the cluster as the centroid, compute the distance from each template to the centroid, and choose the template that is the shortest distance from the centroid. In the experiments described in Section 4, we select the clustering method with the best normalized mutual information score (NMI) as our baseline. We find that this is always achieved by either Ward's or Birch.

[2]We also experimented with max and unweighted-mean pooling but achieved better results using weighted-mean pooling.

4 Experiments & Results

We evaluate clustering performance using both automatic and human-in-the-loop evaluations. For all experiments, we fix the cluster number at 50 for all models to ensure that the template pool has good coverage of common situations.

4.1 Automatic Evaluation

To evaluate the quality of the generated clusters, we compare the ground truth—from our gold labeled data—with predicted labels using normalized mutual information score (NMI), unsupervised clustering accuracy (ACC; Xie et al. (2016)), and Rand index adjusted for chance (ARI; Hubert and Arabie (1985)). We evaluate the performance of DSEC when compared to (1) the sequential baseline and (2) a weak baseline that randomly assigns each utterance to one of the clusters.

Tables 2 and 3 show the results of the automatic evaluation on the labeled CM and DNR datasets. For DSEC, the validation accuracy of reconstruction is approximately 93% for both datasets, indicating that the auto-encoder vector extracts the sentence information well. On CM, DSEC achieves the best NMI and ACC, while the sequential method, with the ELMo-CS embedding and weighted mean pooling of tf-idf features, has the best ARI results overall. The models using in-domain embeddings outperform others with pre-trained embeddings. Note that the metrics NMI, ACC, and ARI are not always consistent when compared across different methods. For example, Glove-CS has a high ARI score but under-performs with all the other automatic metrics.

In addition, clustering performs better on DNR dataset than on CM. This is potentially because the CM domain contains a broader range of customer issues corresponding to different membership types and hence is more challenging to represent using utterance clustering. For example, the templates can be quite different for canceling a free trial, a regular subscription, and certain memberships with an additional subscription attached.

Overall, none of the proposed methods achieve the accuracy of some image clustering work, such as Guo et al. (2017). As discussed before, image clustering and text clustering are very different tasks, and sentence clustering can be quite subjective. Rephrasing or adding content to sentences can make such clustering challenging even for humans. For example, it is non-trivial to decide whether the following sentences should be clustered together: "I will cancel your membership", "I'll cancel your membership and issue a refund", and "The membership will be canceled starting today and you will not be able to use the free subscription". Note that the second and third sentences both contain additional information as opposed to the first sentence. In practice, we encourage annotators to define each cluster as precisely as possible, even if it results in a large number of clusters. This can increase the coverage of the generated template pool but decrease the performance of clustering in the automatic evaluation. To determine the true impact of clustering on our downstream task, response generation, we conduct a human-in-the-loop evaluation in which we use the generated template pool along with a neural response ranking model to recommend responses to CSAs handling customer service contacts.

4.2 Human-in-the-Loop Evaluation

To evaluate the effectiveness of clustering for the downstream task of response generation, we use a human-in-the-loop research platform through which CSAs handle live customer contacts. Specifically, we train template-based neural response ranking models for CM and DNR similar to the model proposed by Lu et al. (2019), and then use them to select responses from the template pools generated using the methods proposed above. Note that training the response ranking model is independent of template creation. We then test the resulting model and template pool using this platform. Instead of showing CSAs the standard chat box, the platform presents ten suggested responses chosen by the trained model from the pool generated by one of the clustering approaches. These 10 suggestions come from different clusters since we only send one template per cluster to the ranking model. They are based on the complete conversation history up to this point and are updated each time the customer or the agent sends a response. The CSA can pick any of the suggested templates as a response, or type their own text if none of the templates appears appropriate. An ideal template pool should minimize the chance that CSAs need to type their own text, and also have no overlapping templates in it.

We choose the following metrics for the human-in-the-loop evaluation, reported in Table 4[3]:

[3]For the human-in-the-loop experiment, we only include

	Rand-BL	Glove	GloVe-CS	BERT	ELMo	ELMo-CS	DSEC
NMI	0.22	0.43	0.54	0.31	0.30	0.54	**0.60**
ACC	0.10	0.24	0.33	0.16	0.16	0.31	**0.39**
ARI	0.00	0.11	**0.20**	0.04	0.03	0.18	0.12

Table 2: Results of Automatic Evaluation on CM Data

	Rand-BL	Glove	GloVe-CS	BERT	ELMo	ELMo-CS	DSEC
NMI	0.23	0.49	0.62	0.39	0.40	0.61	**0.63**
ACC	0.10	0.31	0.47	0.2	0.24	0.41	**0.51**
ARI	0.00	0.15	0.32	0.12	0.09	0.26	**0.34**

Table 3: Results of Automatic Evaluation on DNR Data

1. Top-10 acceptance rate: The percentage of utterances for which the CSA selects one of the suggested responses.
2. Top-1 acceptance rate: The percentage of utterances for which the CSA selects the first suggested response.
3. All suggestions accepted: The percentage of contacts that are handled using only suggested utterances.
4. Average depth of first rejection: The percentage of utterances in the conversation that occur before the agent rejects all suggestions and types their own text.
5. Unique rate: This measures the variation of the template pool, calculated as one minus the percentage of templates that can be removed without reducing the coverage. Ideally, this number would be 1.0.
6. Number of missing templates: The number of utterances that are reported missing from agents. Ideally, this number would be 0.

In this experiment, we compare the performance of (1) the end-to-end approach (DSEC), (2) the sequential setup that performs best in the automatic evaluation (GloVe-CS with weighted-mean pooling and Ward's clustering), and (3) a random baseline in which we randomly select 50 utterances from the dataset to be used as templates. Additionally, we include a human/gold baseline for which the template pool is manually created and refined by collecting feedback from agents over the course of one month.

The utterance acceptance rate indicates that DSEC outperforms both the random and the sequential baseline and performs only slightly worse than the human template pool. As expected, the "all suggestions accepted" rate is much lower for DSEC than for the gold/human pool, but better than for the other automated methods. We find that the sequential approach manages to minimize the length of the conversation (i.e. the number of CSA utterances). One possibility is that it results in a better coverage rate so that it can guide the agents to solve contacts more efficiently than the other methods.

We measure coverage by asking agents to report missing templates. Agents reported a few missing templates for all of the automatically generated pools. The variance in this metric is high because the experiment is only run for about 200 contacts for each experimental configuration. In this way, corner examples may not show up for all of the configurations, and a larger experiment is needed to determine exactly how many templates are missing.

Lastly, the sequential baseline results in a higher depth of first rejection than the manual approach. A possible cause is that this approach leads to a larger proportion of shorter contacts: The sequential approach has 4% more contacts that have less than 10 CSA utterances than the manual one. This could indicate that automatically generated templates can increase the efficiency of contact handling by steering CSAs away from utterances that could lead to longer conversations.

5 Conclusion & Future Work

We present DSEC, an end-to-end sentence encoding and clustering approach that can help automate template creation for template-based conversational models. The purpose is to avoid the human effort required to manually create a template pool when training a response generation model for a conversational system. We evaluate the proposed approach on two customer service datasets and find that it outperforms both a strong sequential baseline and a random baseline in most cases. In addition,

the CM dataset due to limited agent resources.

Metric	Rand	Gold	Seq	DSEC
Total contacts	187	250	211	209
Average number of CSA turns per contact	12.0	12.5	10.1	12.0
Top-10 acceptance rate (% utterances)	48.8	56.4	50.5	52.0
Top-1 acceptance rate (% utterances)	20.9	26.7	22	23.8
All suggestions accepted (% contacts)	2.7	10.3	4.2	6.3
All but 1 accepted (% contacts)	7.8	13.1	10.3	12.4
Average depth of 1st rejection (% contacts)	30.7	31.7	32.3	31.2
Unique rate	0.67	1.0	0.77	0.71
Number of missing templates	4	0	2	2

Table 4: Results of Human-in-the-Loop Experiment on CM Data

we use the resulting template pools in a human-in-the-loop experiment and observe that the template pool created using DSEC performs only slightly worse than a manually created template pool that takes over a month of human effort to develop. In future work, we plan on exploring building a pipeline that can automatically polish and update the generated template pool using feedback from agents.

Acknowledgements

We thank the three anonymous reviewers for their feedback and insights in improving the work.

References

Akiko Aizawa. 2003. An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, 39(1):45–65.

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating more interesting responses in neural conversation models with distributional constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

B. J. Frey and D. Dueck. 2007. Clustering by Passing Messages Between Data Points. *Science*, 315(5814):972–976.

Xifeng Guo, Xinwang Liu, En Zhu, and Jianping Yin. 2017. Deep clustering with convolutional autoencoders. In *International Conference on Neural Information Processing*, pages 373–382. Springer.

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

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*, 2(1):193–218.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, et al. 2016. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964. ACM.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Bing Liu, Gokhan Tur, Dilek Hakkani-Tur, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2060–2069.

Yichao Lu, Manisha Srivastava, Jared Kramer, Heba Elfardy, Andrea Kahn, Song Wang, and Vikas Bhardwaj. 2019. Goal-oriented end-to-end conversational models with profile features in a real-world setting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 48–55.

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on* *mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Daniel Müllner. 2011. Modern hierarchical, agglomerative clustering algorithms. *arXiv:1109.2378 [cs, stat]*. ArXiv: 1109.2378.

Fionn Murtagh and Pierre Legendre. 2014. Wards Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Wards Criterion? *Journal of Classification*, 31(3):274–295.

Jeff Pasternack, Nimesh Chakravarthi, Adam Leon, Nandeesh Rajashekar, Birjodh Tiwana, and Bing Zhao. 2017. Building smart replies for member messages.

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

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018a. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018b. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Minghui Qiu, Feng-Lin Li, Siyu Wang, Xing Gao, Yan Chen, Weipeng Zhao, Haiqing Chen, Jun Huang, and Wei Chu. 2017. Alime chat: A sequence to sequence and rerank based chatbot engine. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 498–503.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating high-quality and informative conversation responses with sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Jianbo Shi and Jitendra Malik. 2000. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):18.

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

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, Jun Xu, and Xueqi Cheng. 2018. Learning to control the specificity in neural response generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1997. BIRCH: A New Data Clustering Algorithm and Its Applications. *Data Mining and Knowledge Discovery*, 1(2):141–182.

Xiangyang Zhou, Lu Li, Daxiang Dong, Yi Liu, Ying Chen, Wayne Xin Zhao, Dianhai Yu, and Hua Wu. 2018. Multi-turn response selection for chatbots with deep attention matching network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1.