Controlled Text Generation for Data Augmentation in Intelligent Artificial Agents

Nikolaos Malandrakis\(^1\), Minmin Shen\(^2\), Anuj Goyal\(^2\), Shuyang Gao\(^2\), Abhishek Sethi\(^2\), Angeliki Metallinou\(^2\)

\(^1\) Signal Analysis and Interpretation Laboratory (SAIL), USC, Los Angeles, CA 90089
\(^2\) Amazon Alexa AI

malandra@usc.edu, \{shenm, anujgoya, shuyag, abhsethi, ametalli\}@amazon.com

Abstract

Data availability is a bottleneck during early stages of development of new capabilities for intelligent artificial agents. We investigate the use of text generation techniques to augment the training data of a popular commercial artificial agent across categories of functionality, with the goal of faster development of new functionality. We explore a variety of encoder-decoder generative models for synthetic training data generation and propose using conditional variational auto-encoders. Our approach requires only direct optimization, works well with limited data and significantly outperforms the previous controlled text generation techniques. Further, the generated data are used as additional training samples in an extrinsic intent classification task, leading to improved performance by up to 5\% absolute f-score in low-resource cases, validating the usefulness of our approach.

1 Introduction

Voice-powered artificial agents have seen widespread commercial use in recent years, with agents like Google’s Assistant, Apple’s Siri and Amazon’s Alexa rising in popularity. These agents are expected to be highly accurate in understanding the users’ requests and to be capable of handling a variety of continuously expanding functionality. New capabilities are initially defined via a few phrase templates. Those are expanded, typically through larger scale data collection, to create datasets for building the machine learning algorithms required to create a serviceable Natural Language Understanding (NLU) system. This is a lengthy and expensive process that is repeated for new functionality expansion and can significantly slow down development time.

We investigate the use of neural generative encoder-decoder models for text data generation. Given a small set of phrase templates for some new functionality, our goal is to generate new semantically similar phrases and augment our training data. This data augmentation is not necessarily meant as a replacement for large-scale data collection, but rather as a way to accelerate the early stages of new functionality development. This task shares similarities with paraphrasing. Therefore, inspired by work in paraphrasing (Prakash et al., 2016) and controlled text generation (Hu et al., 2018), we investigate the use of variational autoencoder models and methods to condition neural generators.

For controlled text generation, (Hu et al., 2018) used a variational autoencoder with an additional discriminator and trained the model in a wake-sleep way. (Zhou and Wang, 2018) used reinforcement via an emoji classifier to generate emotional responses. However, we found that when the number of samples is relatively small compared to the number of categories, such an approach might be counter-productive, because the required classifier components cannot perform well. Inspired by recent advantages of connecting information theory with variational auto-encoders and invariant feature learning (Moyer et al., 2018), we instead use this approach to our controlled text generation task, without a discriminator.

Furthermore, our task differs from typical paraphrasing in that semantic similarity between the output text and the NLU functionality is not the only objective. The synthetic data should be evaluated in terms of its lexical diversity and novelty, which are important properties of a high quality training set.

Our key contributions are as follows:

- We thoroughly investigate text generation techniques for NLU data augmentation with sequence to sequence model and variational auto-encoders, in an atypically low-resource
• We validate our method in an extrinsic intent classification task, showing that the generated data brings considerable accuracy gains in low resource settings.

2 Related Work

Neural networks have revolutionized the field of text generation, in machine translation (Sutskever et al., 2014), summarization (See et al., 2017) and image captioning (You et al., 2016). However, conditional text generation has been relatively less studied as compared to conditional image generation and poses some unique problems. One of the issues is the non-differentiability of the sampled text that limits the applicability of a global discriminator in end-to-end training. The problem has been relatively addressed by using CNNs for generation (Rajeswar et al., 2017), policy gradient reinforcement learning methods including SeqGAN (Yu et al., 2017), LeakGAN (Guo et al., 2018), or using latent representation like Gumbel softmax ((Jang et al., 2016)). Many of these approaches suffer from high training variance, mode collapse or cannot be evaluated beyond a qualitative analysis.

Many models have been proposed for text generation. Seq2seq models are standard encoder-decoder models widely used in text applications like machine translation (Luong et al., 2015) and paraphrasing (Prakash et al., 2016). Variational Auto-Encoder (VAE) models are another important family (Kingma and Welling, 2013) and they consist of an encoder that maps each sample to a latent representation and a decoder that generates samples from the latent space. The advantage of these models is the variational component and its potential to add diversity to the generated data. They have been shown to work well for text generation (Bowman et al., 2016). Conditional VAE (CVAE) (Kingma et al., 2014) was proposed to improve over seq2seq models for generating more diverse and relevant text. CVAE based models (Serban et al., 2017; Zhao et al., 2017; Shen et al., 2017; Zhou and Wang, 2018) incorporate stochastic latent variables that represents the generated text, and append the output of VAE as an additional input to decoder.

Paraphrasing can be performed using neural networks with an encoder-decoder configuration, including sequence to sequence (S2S) (Luong et al., 2015) and generative models (Bowman et al., 2016) and various modifications have been proposed to allow for control of the output distribution of the data generation (Yan et al., 2015; Hu et al., 2018). Unlike the typical paraphrasing task we care about the lexical diversity and novelty of the generated output. This has been a concern in paraphrase generation: a generator that only produces trivial outputs can still perform fairly well in terms of typical paraphrasing evaluation metrics, despite the output being of little use. Alternative metrics have been proposed to encourage more diverse outputs (Shima and Mitamura, 2011). Typically evaluation of paraphrasing or text generation tasks is performed by using a similarity metric (usually some variant of BLEU (Papineni et al., 2002)) calculated against a held-out set (Prakash et al., 2016; Rajeswar et al., 2017; Yu et al., 2017).

3 Methodology

3.1 Problem Definition

New capabilities for virtual agents are typically defined by a few phrases templates, also called carrier phrases, as seen in Fig. 1. In carrier phrases the entity values, like the movie title ‘Batman’, are replaced with their entity types, like movie_title. These are also called slot values and slot types, respectively, in the NLU literature. For our generation task, these phrases define a category: all carrier phrases that share the same domain, intent and slot types are equivalent, in the sense that they prompt the same agent response. For the remainder of this paper we will refer to the combination of domain, intent and slot types as the signature of a
phrase. Given a small amount of example carrier phrases for a given signature of a new capability (typically under 5 phrases), our goal is to generate additional semantically similar carrier phrases for the target signature.

The core challenge lies in the very limited data we can work with. The low number of phrases per category is, as we will show, highly problematic when training some adversarial or reinforcement structures. Additionally the high number of categories makes getting an output of the desired signature harder, because many similar signatures will be very close in latent space.

3.2 Generation models

Following is a short description of the models we evaluated for data generation. For all models we assume we have training carrier phrases $c_i \in D_{tr}$ across signatures $s$, and we pool together the data from all the signatures for training. The variational auto-encoders we used can be seen in Fig 2.

Sequence to Sequence with Attention Here, we use the seq2seq with global attention proposed in (Luong et al., 2015) as our baseline generation model. The model is trained on all input-output pairs of carrier phrases belonging to the same signature $s$, e.g., $c_1, c_2 \in D_{tr}$. At generation, we aim to control the output by using an input carrier of the target signature $s$.

Variational Auto-Encoders (VAEs) The VAE model can be trained with a paraphrasing objective e.g., $c_1 \in D_{tr}$ can be both the input and the output. However, if we train with a reconstruction objective, during generation, we ignore the encoder and randomly sample the VAE prior $z$ (typically from a normal distribution). As a result, we have no control over the output signature distribution, and we may generate any of the signatures $s$ in our training data. This disadvantage motivates the investigation of two controlled VAE models.

VAE with discriminator is a modification of a VAE proposed by (Hu et al., 2018) for a similar task of controlled text generation. In this case, adversarial type of training is used by training a discriminator, i.e., a classifier for the category (signature $s$), to explicitly enforce control over the generated output. The network is trained in steps, with the VAE trained first, then the discriminator is attached and the entire network re-trained using a sleep-wake process. We tried two variations of this, one training a VAE, another training a CVAE, before adding the discriminator. Note that control over the output depends on the discriminator performance. While this model worked well for controlling between a small number of output categories as in (Hu et al., 2018), our setup includes hundreds of signatures $s$, which posed challenges in achieving accurate control over the output phrases (Sec. 5.2).

Conditional VAE (CVAE) Inspired by (Moyer et al., 2018) for invariant feature learning, we propose to use a CVAE based controlled model structure. Such structure is a modification on the VAE, where we append the desired category label, here signature $s$, in 1-hot encoding, to each step of the decoder without an additional discriminator as shown in (Hu et al., 2018). Note that the original conditional VAE has already been applied to controlled visual settings (Yan et al., 2015). It has been shown that by direct optimizing the loss, this model automatically learns a invariant representation $z$ that is independent of the category (signature $s$ (Moyer et al., 2018)) although no explicit constraint is forced. We propose to use this model in our task, because it is easy to train (no wake-sleep or adversarial training), requires less data, and provides us a way to control the desired VAE output signature, by setting the desired signature encoding to $s$. Like the standard VAE, the CVAE can be trained either with a paraphrasing or with a reconstruction objective. If training with reconstruction,
during generation we randomly sample from \( z \) but can control the output signature by setting \( s \).

All model encoders and decoders are GRUs. For the discriminator we tried CNN and LSTM with no significant performance differences.

4 Datasets
We experiment on two datasets collected for Alexa, a commercial artificial agent.

**Movie dataset** It contains carrier phrases that are created as part of developing new movie-related functionality. It is composed of 179 signatures defined with an average of eight carrier phrases each. This data represents a typical new capability that starts out with few template carrier phrases, and we use it to examine if this low resource dataset can benefit from synthetic data generation.

**Live entertainment dataset** It contains live customer data from deployed entertainment related capabilities (music, books, etc), selected for their semantic relevance to movies. These utterances were de-lexicalized by replacing slot values with their respective slot types. We used a frequency threshold to filter out rare carrier phrases, and ensure a minimum number of three carrier phrases per signature.

Table 1 shows the data splits for the movie, live entertainment and ‘all’ datasets, the latter containing both movies and live entertainment data, including the number of signatures, slot types and unique non-slot words in each set. While the data splits were stratified, signatures with fewer than four carriers were placed only in the train set, leading to the discrepancy in signature numbers across partitions.

5 Experiments

5.1 Experimental setup
At the core of our data augmentation task lies the question “what defines a good training data set?”. We can evaluate aspects of the generated data via synthetic metrics, but the most reliable method is to generate data for an extrinsic task and evaluate any improvements in performance. In this paper we employ both methods are reporting results for intrinsic and extrinsic evaluation metrics.

For the intrinsic evaluation, we train the data generator either only on movie data or on ‘all’ data (movies and entertainment combined), using the respective dev sets for hyper-parameter tuning. During generation, we similarly consider either the movies test set, or the ‘all’ test set, and aim to generate ten synthetic phrases per test set phrase. VAE type generators can be trained for paraphrasing \((c_1 \rightarrow c_2)\) or reconstruction \((c_1 \rightarrow c_1)\). During generation, sampling can be performed either from the prior, e.g., by ignoring the encoder and sampling \( z \sim \mathcal{N}(0, I) \) to generate an output, or from the posterior e.g., using \( c_1 \) as input to the encoder and producing the output \( c_2 \). Note that not all combinations are applicable to all models. Those applicable are shown in Table 3, where ‘para’, ‘recon’, ‘prior’ and ‘post’ denote paraphrasing, reconstruction, prior and posterior respectively. Special handling was required for a VAE with reconstruction training and prior sampling, where we have no control over the output signature. To solve this, we compared each output phrase to every signature in the train set (via BLEU4 (Papineni et al., 2002)) and assigned it to the highest scoring signature. Some sample output phrases can be seen in Fig. 3.

To examine the usefulness of the generated data for an extrinsic ask, we perform intent classification, a standard task in NLU. Our classifier is a Bi-LSTM model. We use the same data as for the data generation experiments (see Table 1), and group our class labels into intents (as opposed to signatures), which leads to classifying 136 intents in the combined movies and entertainment data (‘all’). Our setup follows two steps: First, the data generators are trained on ‘all’ train sets, and used to generate phrases for the dev sets (‘all’ and movies). Second, the intent classifier is trained on the ‘all’ train and dev sets (baseline), vs the combination of ‘all’ train, dev and generated synthetic data, which is our proposed approach. We evaluate on the ‘all’ and movies test sets, and use macro-averaged F-score across all intents as our metric.

5.2 Intrinsic evaluation
To evaluate the generated data we use an ensemble of evaluation metrics attempting to quantify three important aspects of the data: (1) how accurate or relevant the data is to the task, (2) how diverse the set of generated phrases is and (3) how novel these synthetic phrases are. Intuitively, a NLG system can be very accurate - generate valid phrases of the correct signature - while only generating phrases from the train set or while generating the same phrase multiple times for the same signature; either of these scenarios would not lead to useful data. To
Table 1: Data distribution and splits. ‘All’ contains the combined Movie and Entertainment live datasets.

| domain         | subset | carriers | signatures | slots | words |
|----------------|--------|----------|------------|-------|-------|
| Movies         | train  | 1,382    | 179        | 21    | 353   |
|                | dev    | 622      | 109        | 15    | 292   |
|                | test   | 520      | 69         | 10    | 254   |
| Live Entertainment | train | 4269     | 588        | 120   | 332   |
|                | dev    | 1236     | 244        | 77    | 194   |
|                | test   | 1335     | 271        | 74    | 217   |
| All            | train  | 5,651    | 767        | 141   | 685   |
|                | dev    | 1,858    | 353        | 92    | 486   |
|                | test   | 1,855    | 340        | 84    | 471   |

Table 2: Data distribution and splits for the extrinsic task.

| domain | subset | carriers | intents |
|--------|--------|----------|---------|
| All    | train  | 5651     | 136     |
|        | dev    | 1858     | 101     |
|        | train+dev | 7509   | 136     |
|        | test   | 1855     | 94      |

evaluate accuracy we compare the generated data to a held out test set using BLEU4 (Papineni et al., 2002) and the slot carry-over rate, the probability that a generated phrase contains the exact same slot types as the target signature $s$. To evaluate novelty we compare the generated data to the train set of the generator, using 1-BLEU4 (where higher is better) and 1-Match rate, where the match rate is the chance that a perfect match to a generated phrase exists in the train set. These scores tell us how different, at the lexical level, the generated phrases are to the phrases that already exist in the train set. Finally, to evaluate diversity we compare the phrases in the generated data to each other, using again 1-BLEU4 and the unique rate, the number of unique phrase produced over the total number of phrases produced. These scores indicate how lexically different the generated phrases are to each other. Figure 4 shows the set comparisons made to generate the intrinsic evaluation metrics. Note that these metrics mostly evaluate surface forms; we expect phrases generated for the same signature to be semantically similar to phrases with the same signature in the train set and to each other, however we would like them to be lexically novel and diverse.

Table 3 presents the intrinsic evaluation results, where generators are trained and tested on ‘all’ data, for the best performing model per case, tuned on the dev set. First, note the slot carry over (slot c.o.), which can be used as a sanity check measuring the chance of getting a phrase with the desired slot types. Most models reach 0.8 or higher slot c.o. as expected, but some fall short, indicating failure to produce the desired signature. The failure for VAE and CVAE models with discriminators is most notable, and can be explained by the fact that we have a large number of train signatures (~800) and too few samples per signature (mean 8, median 4), to accurately train the discriminator. We verified that the discriminator overall accuracy does not exceed 0.35. The poor discriminator performance leads to the decoder not learning how to use signature $s$. The failure of VAE with posterior sampling is similarly explained by the large number of signatures: the signatures are so tightly packed in the latent space, that the variance of sampling $z$ is likely to result in phrases from similar but different signatures.

This sanity check leaves us with five reasonably performing models: S2S, VAE trained for reconstruction and sampled from the prior and CVAE with multiple training and sampling strategies. Overall, these models achieve high accuracy with respect to the slot c.o. and BLEU4 metrics, assisted by the rather limited vocabulary of the data. To examine the trade-offs between the models, in Fig. 5, we show the accuracy BLEU4 as a function of diversity unique rate, i.e., how many different phrases we generated. Each point is a model trained with different hyper-parameter settings, across relevant hyper-parameters, network component dimensionalities etc. As expected, diversity is negatively correlated with accuracy. We make similar observations for novelty metrics (plots omitted for brevity), i.e., diversity and novelty are negatively correlated.
Figure 3: Sample output phrases. The S2S model and all posterior sampling models use a phrase as an input. For prior sampling the desired signature is the model input.

| Model: S2S                  | Input: i\_negation feel like watching a movie with person\_name |
|-----------------------------|---------------------------------------------------------------|
| i\_negation                 | like movies by person\_name                                  |
| i\_negation                 | feel like watching movies by person\_name                   |
| i\_negation                 | feel like watching a movie by person\_name                  |
| i\_negation                 | like person\_name                                           |
| i\_negation                 | feel like watching a movie                                  |
| i\_negation                 | want to talk about person\_name                             |
| no i\_negation              | like person\_name                                           |

| Model: V AE, sampling from prior distribution | Input: GetMovieAwards (intent) - award\_title, movie\_title (slots) |
|-----------------------------------------------|------------------------------------------------------------------|
| did movie\_title win an award\_title          |                                                                  |
| any award\_title won by movie\_title          |                                                                  |
| tell me any award\_title which movie\_title won |                                                                |
| was movie\_title nominated for an award\_title the movie movie\_title |                          |
| any award\_title for movie\_title             |                                                                  |
| what are the award\_title which movie\_title won |                                                               |
| give me any award\_title the movie was nominated for |                                                                               |

| Model: CVAE, sampling from prior distribution | Input: GetActorMovies (intent) - genre, person\_name (slots) |
|-----------------------------------------------|---------------------------------------------------------------|
| give me genre movies starring person\_name    |                                                                  |
| show me other genre movies with person\_name in it |                                                               |
| what are the genre movies that person\_name starred in |                                         |
| tell me genre movies starring person\_name     |                                                                  |
| what are genre movies with person\_name        |                                                                  |
| suggest genre movies starring person\_name     |                                                                  |

Figure 4: Intrinsic evaluation overview

Table 4 shows intrinsic results on the movies test set. For brevity, we show the mean relative change for the best performing models for each metric, computed between using only movie data to train the generators vs using the combined ‘all’ data. In the latter case, the live entertainment data is added to train a more robust generator for movies. As expected, we notice a small loss in accuracy (-1.9 % rel. change on average for BLEU4) when using the ‘all’ data for generator training, but also a significant gain in diversity and novelty of the movie generated data (121 % and 153 % rel. change on average respectively for 1-BLEU4). Overall, the reconstruction VAE and CVAE models achieve the best results and have favorable performance trade-offs when using ‘all’ data to enrich movie data generation.
Table 3: Best performance per metric for each model when applied to ‘all’ domains.

| metric          | S2S | VAE | VAE+DISC | CVAE | CVAE+DISC |
|-----------------|-----|-----|----------|------|-----------|
| training        |     |     |          |      |           |
|      | para | recon |       | recon | recon | recon | recon |
| sampling        |     |     |          |      |           |
|      | post | prior | post     | prior | post     | prior |
| accuracy        |     |     |          |      |           |
| BLEU4           | 0.86| 0.91| 0.24     | 0.42 | 0.91     | 0.88  | 0.90  | 0.11  |
| slot c.o.       | 0.84| 0.95| 0.02     | 0.12 | 0.98     | 0.93  | 0.95  | 0.01  |
| diversity       |     |     |          |      |           |
| 1-BLEU4         | 0.06| 0.19| 0.83     | 0.84 | 0.14     | 0.23  | 0.33  | 0.19  |
| uniq. rate      | 0.58| 0.68| 0.98     | 0.76 | 0.44     | 0.56  | 0.68  | 0.97  |
| novelty         |     |     |          |      |           |
| 1-BLEU4         | 0.25| 0.07| 0.75     | 0.98 | 0.04     | 0.12  | 0.21  | 0.99  |
| 1-match rate    | 0.89| 0.76| 0.99     | 1.00 | 0.32     | 0.50  | 0.59  | 1.00  |

Table 4: Relative change when adding more training data for generator training (movies only vs ‘all’) across evaluation metrics on the movies test set

| metric          | CVAE | CVAE | CVAE | VAE |
|-----------------|------|------|------|-----|
|                | prior | posterior | s2s | prior |
| accuracy        |     |     |      |     |
| BLEU4           | -0.7%| -0.7%| -4.3%| -2.0%|
| slot c.o.       | -2.7%| -3.1%| -9.5%| -5.9%|
| diversity       |     |     |      |     |
| 1-BLEU4         | 112.1%| 103.6%| 133.7%| 134.6%|
| uniq. rate      | 19.6%| 19.7%| 32.7%| 37.1%|
| novelty         |     |     |      |     |
| 1-BLEU4         | 147.1%| 40.7%| 222.1%| 201.2%|
| 1-match rate    | 113.5%| 36.7%| 145.4%| 90.1%|

Figure 5: Intrinsic accuracy BLEU4 as a function of intrinsic diversity for the ‘all’ test set for multiple hyper-parameter combinations of each model.

5.3 Extrinsic Evaluation

In Figure 6 we present the change in the F1 score for intent classification when adding the generated data into the classifier training (compared to the baseline classifier with no generated data) as a function of the intrinsic BLEU4 accuracy metric. The plot presents results on the movies test set. Each point is a model trained with different hyper-parameters and the line $y = 0$ represents zero change from baseline, while models over this line represent improvement. Some hyper-parameter choices clearly lead to sub-optimal results, but they are included to show the relationship between intrinsic and extrinsic performance across a wider range of conditions. We notice that many generators produce useful synthetic data that lead to improvement in intent classification, with the best performing ones being the CVAE models with around 5% absolute improvement in F-score on the movie test set ($p < 0.01$). This is an encouraging results, as it verifies the usefulness of the generated data for improving the extrinsic low resource task. For the ‘all’ test set experiments, the improvement is less pronounced, with maximum gain from synthetic data being around 2%, again for the CVAE models. This smaller improvement could be because this test set is not as low resource (roughly twice as many train carriers phrases per
6 Conclusions

We described a framework for controlled text generation for enriching training data for new NLU functionality. Our challenging text generation setup required control of the output phrases over a large number of low resource signatures of NLU functionality. We used intrinsic metrics to evaluate the quality of the generated synthetic data in terms of accuracy, diversity and novelty. We empirically investigated variational encoder-decoder type models and proposed to use a CVAE based model, which yielded the best results, being able to generate phrases with favorable accuracy, diversity and novelty trade-offs. We also demonstrated the usefulness of our proposed methods by showing that the synthetic data can improve the accuracy of an extrinsic low resource classification task.

7 Acknowledgments

This work was performed while Nikolaos Malandrakis was at Amazon Alexa AI, Sunnyvale.

References

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In Proceedings of the SIGNLL Conference on Computational Natural Language Learning (CONLL), pages 10–21.

Jiaxian Guo, Weinan Zhang Yong Yu Sidi Lu, Han Cai, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In Proceedings of AAAI, pages 5141–5148.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2018. Toward controlled generation of text. arXiv:1703.00955.
