A Single Example Can Improve Zero-Shot Data Generation

Pavel Burnyshev[1], Valentin Malykh[1,2], Andrey Bout[1], Ekaterina Artemova[1,3], and Irina Piontkovskaya[1]

[1]Huawei Noah's Ark Lab, Moscow, Russia
[2]Kazan Federal University, Kazan, Russia
[3]HSE University, Moscow, Russia
{*burnyshev.pavel, malykh.valentin, bout.andrey, artemova.ekaterina, piontkovskaya.irina*}*@huawei.com*

Abstract

Sub-tasks of intent classification, such as robustness to distribution shift, adaptation to specific user groups and personalization, out-of-domain detection, require extensive and flexible datasets for experiments and evaluation. As collecting such datasets is time- and labor-consuming, we propose to use text generation methods to gather datasets. The generator should be trained to generate utterances that belong to the given intent. We explore two approaches to generating task-oriented utterances. In the **zero-shot approach**, the model is trained to generate utterances from seen intents and is further used to generate utterances for intents unseen during training. In the **one-shot approach**, the model is presented with a single utterance from a test intent. We perform a thorough automatic, and human evaluation of the dataset generated utilizing two proposed approaches. Our results reveal that the attributes of the generated data are close to original test sets, collected via crowd-sourcing.

1 Introduction

Training dialogue systems used by virtual assistants in task-oriented applications requires large annotated datasets. The core machine learning task to every dialogue system is *intent detection*, which aims to detect what the intention of the user is. New intents emerge when new applications, supported by the dialogue systems, are launched. However, an extension to new intents may require annotating additional data, which may be time-consuming and costly. What is more, when developing a new dialogue system, one may face the cold start problem if little training data is available. Open sources provide general domain annotated datasets, primarily collected via crowd-sourcing or released from commercial systems, such as Snips NLU benchmark (Coucke et al., 2018). However, it is usually problematic to gather more specific data from any source, including user logs, protected by the privacy policy in real-life settings.

For all these reasons, we suggest a learnable approach to create training data for intent detection. We simulate a real-life situation in which no annotated data but rather only a short description of a new intent is available. To this end, we propose to use methods for zero-shot conditional text generation to generate plausible utterances from intent descriptions. The generated utterances should be in line with the intent's meaning.

Our contributions are:

1. We propose a zero-shot generation method to generate a task-oriented utterance from an intent description;
2. We evaluate the generated utterances and compare them to the original crowd-sourced datasets. The proposed zero-shot method achieves high scores in fluency and diversity as per our human evaluation;
3. We provide experimental evidence of a semantic shift when generating utterances for unseen classes using the zero-shot approach;
4. We apply reinforcement learning for the one-shot generation to eliminate the semantic shift problem. The one-shot approach retains semantic accuracy without sacrificing fluency and diversity.

2 Related work

Conditional language modelling generalizes the task of language modelling. Given some conditioning context z, it assigns probabilities to a sequence of tokens (Mikolov and Zweig, 2012). Machine translation (Sutskever et al., 2014; Cho et al., 2014) and image captioning (You et al., 2016) are seen as typical conditional language modelling tasks. More sophisticated tasks include text abstractive summarization (Nallapati et al., 2017; Narayan et al., 2019) and simplification (Zhang

and Lapata, 2017), generating textual comments to source code (Richardson et al., 2017) and dialogue modelling (Lowe et al., 2017). Structured data may act as a conditioning context as well. Knowledge base (KB) entries (Vougiouklis et al., 2018) or DBPedia triples (Colin et al., 2016) serve as condition to generated plausible factual sentences. Neural models for conditional language modelling rely on encoder-decoder architectures and can be learned both jointly from scratch (Vaswani et al., 2017) or by fine-tuning pre-trained encoder and decoder models (Budzianowski and Vulić, 2019; Lewis et al., 2020).

Zero-shot learning (ZSL) has formed as a recognized training paradigm with neural models becoming more potent in the majority of downstream tasks. In the NLP domain, the ZSL scenario aims at assigning a label to a piece of text based on the label description. The learned classifier becomes able to assign class labels, which were unseen during the training time. The classification task is then reformulated in the form of question answering (Levy et al., 2017) or textual entailment (Yin et al., 2019). Other techniques for ZSL leverage metric learning and make use of capsule networks (Du et al., 2019) and prototyping networks (Yu et al., 2019).

Zero-shot conditional text generation implies that the model is trained in such a way that it can generalize to an unseen condition, for which only a description is provided. A few recent works in this direction show-case dialog generation from unseen domains (Zhao and Eskenazi, 2018) and question generation from KB's from unseen predicates and entity types (Elsahar et al., 2018). CTRL (Keskar et al., 2019), pre-trained on so-called control codes, which can be combined to govern style, content, and surface form, provides for zero-shot generation for unseen codes combinations. PPLM (Dathathri et al., 2019) uses signals, representing the class, e.g., bag-of-words, during inference, and can generate examples with given semantic attributes without pre-training.

Training data generation can be treated as form of data augmentation, a research direction being increasingly in demand. It enlarges datasets for training neural models and help avoid labor-intensive and costly manual annotation. Common techniques for textual data augmentation include back-translation (Sennrich et al., 2016), sampling from latent distributions (Xia et al., 2021), simple heuristics, such as synonym replacement (Wei and Zou, 2019) and oversampling (Chawla et al., 2002). Few-shot text generation has been applied to natural language generation from structured data, such as tables (Chen et al., 2020) and to intent detection data augmentation (Xia et al., 2021). However, these methods are incompatible with ZSL, requiring at least a few labeled examples for the class being augmented. An alternative approach suggests to use a model to generate data for the target class based on task-specific world knowledge (Chen et al., 2017) and linguistic features (Iyyer et al., 2018).

Deep reinforcement learning (RL) methods prove to be effective in a variety of NLP tasks. Early works approach the tasks of machine translation (Grissom II et al., 2014), image captioning (Rennie et al., 2017) and abstractive summarization (Paulus et al., 2017), assessed with not differentiable metrics. (Wu et al., 2021) tries to improve the quality of transformer-derived pre-trained models for generation by leveraging proximal policy optimization. Other applications of deep RL include dialogue modeling (Li et al., 2016b) and open-domain question answering (Wang et al., 2018).

3 Methods

Our main goal is to generate plausible and coherent utterances, which relate to unseen intents, leveraging the description of the intent only. These utterances should clearly express the desired intent. For example, if conditioned on the intent *"delivery from the grocery store"* the model should generate an utterance close to *"Hi! Please bring me milk and eggs from the nearest convenience store"* or similar.

Two scenarios can be used to achieve this goal. In the **zero-shot scenario**, we train the model on a set of seen intents $\mathcal{S}$ to generate utterances. If the generation model generalizes well, the utterances generated for unseen intents $\mathcal{U}$ are diverse and fluent and retain intents' semantics. In the **one-shot scenario**, we utilize one utterance per unseen intent $\mathcal{U}$ to train the generation model and learn the semantics of this particular intent.

3.1 Zero-shot generation

Our model as depicted in Figure 1) aims to generate plausible utterances conditioned on the intent description. We fine-tune the GPT-2 medium model

([Radford et al., 2019](#)) on task-oriented utterances, collected from several NLU benchmarks (see Section 5.1 for more details on the dataset).
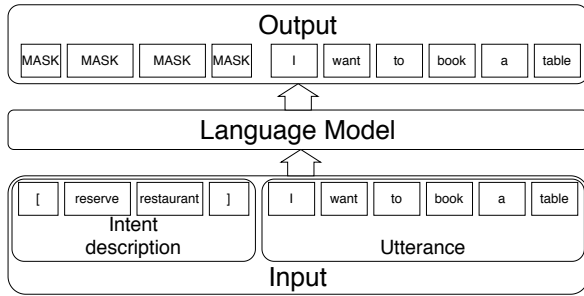

Figure 1: Training setup. The input an intent description and an utterance concatenated, the output is the utterance.

Our approach to fine-tuning the GPT-2 model follows ([Budzianowski and Vulić, 2019](#)). Two pieces of information, the intent description and the utterance are concatenated to form the input. More precisely, the input has the following format: *[intent description] utterance*. During the training phase, the model is presented with the output obtained from the input by masking the intent description. The output has the following format: <MASK>, ..., <MASK> *utterance*. The full list of intents is provided in Table 4 in Appendix.

Such input allows the model to pay attention to intent tokens while generating. The standard language modeling objective, negative log-likelihood loss, is used to train the model:

$$\mathcal{L}\left(\theta\right) = -\sum_{i}\sum_{t=1}^{\left|\mathbf{x}^{(i)}\right|} \log p_{\theta}\left(x_t^{(i)} | \text{intent}, x_{<t}^{(i)}\right).$$

We fine-tuned the model for one epoch to avoid over-fitting. Otherwise, the model tends to repeat redundant semantic constructions of the input utterances. At the same time, a bias towards the words from the training set gets formed. The parameters of the training used were set to the following values: batch size equals to 32, learning rate equals to 5e-5, the optimizer chosen is Adam ([Kingma and Ba, 2015](#)) with default parameters.

3.2 One-shot Generation

Motivation. The zero-shot approach to conditional generation may degrade or even fail if (i) the intent description is too short to properly reflect the semantics of the intent, (ii) the intent description is ambiguous or contains ambiguous words. Produced utterances may distort the initial meaning of the intent or be meaningless at all. The model may generate an utterance *"Count the number of people in the United States"* for the intent "calculator", or *"Add a book by Shakespeare to the calendar"* for a "book reading" service. Although such examples can be treated not as outliers but rather as real-life whimsical utterances, this is not the desired behavior for the generation model. We address this phenomenon as *Semantic Shift* and provide experimental evidence of it in Section 5.4.

Based on these observations, we hypothesize that the problem could be solved if we provide a single training example to improve models' generalization abilities. A single example can give the model a clue about what the virtual assistant can do with books and which entities our calculator is designed to calculate by gaining better world knowledge. For this purpose, we are moving from the zero-shot to the one-shot setting. We propose a method for improving zero-shot generation by leveraging just one example.

Our approach is inspired by the recent TextGAIL ([Wu et al., 2021](#)) approach. It addresses the problem of exposure bias in pre-trained language models and proposes a GAN-like style scheme for fine-tuning GPT-2 to produce appropriate story endings using a reinforcement algorithm. As a reward, TextGAIL uses a discriminator output trained to distinguish real samples from generated samples. As we are limited in using learnable discriminators because of the lack of training data, we propose an objective function based on a similarity score. Our objective function produces utterances, which are close to the reference example. At the same time, it forces the model to generate more diverse and plausible utterances. Table 5 in Appendix provides reference examples used for the one-shot generation method.

Method. After zero-shot fine-tuning, we perform a one-shot model update for each intent separately. We perform several steps of the Proximal Policy Optimization algorithm ([Schulman et al., 2017](#)) with the objective function described further.

Reward. Our reward function is based on BERTScore ([Zhang et al., 2019](#)), which serves as the measure of contextual similarity between generated sentences and the reference example.

BERTScore correlates better with human judgments than other existing metrics, used to control semantics of generated texts and detect paraphrases. Given a reference and a candidate sentence, we embed them using RoBERTa model (Liu et al., 2019). The BERTScore F1 calculated on top of these embeddings is used as a part of the final reward.

It is not enough to reward the model only for the similarity of the generated utterance to the reference one. If so, the model tends to repeat the reference example and receives the maximal reword. We add the negative sum of frequencies of all n-grams in the utterance to the reward function, forcing the model to generate less frequent sequences.

Given an intent I and a reference example x_{ref}^I, the reward for the sentence x is calculated by the formula:

$$R_I(x) = R_{sim}(x_{\text{ref}}^I, x) + R_{div}(x)$$
$$R_{sim}(x_{\text{ref}}^I, x) = \text{BERTScore}(x_{\text{ref}}^I, x)$$
$$R_{div}(x) = \sum_{s \in \text{n-grams}(x)} (-\nu_s)$$

where ν_s is the n-gram frequency, calculated from all the generated utterances inside one batch.

Objective function. First, we plug this reward into standard PPO objective function, getting intent-specific term $L_I^{\text{policy}}(\theta)$. Following the TextGAIL approach, we add **KL** divergence with the model without zero-shot fine-tuning to prevent forgetting the information from the pre-trained model. We add an entropy regularizer, making the distribution smoother, which leads to more diverse and fluent sentences. According to our experiments, this term helps avoid similar prefixes for all generated sentences as n-gram reward only does not cope with this issue. The final generator objective for maximization in the one-shot scenario for the intent I can be written as follows:

$$L(I; \theta) = L_I^{\text{policy}}(\theta) + \hat{\mathbb{E}}_t[\beta\mathbf{H}(p_{\theta;I}(\cdot|s_t))$$
$$-\alpha\mathbf{KL}[p_{\theta;I}(\cdot|s_t), q(\cdot|s_t)]],$$

where s_t is intent description, $p_{\theta;I}$ is the conditional distribution $p_\theta(\cdot|I)$(distribution, derived from model with updates from PPO policy), q is an unconditional LM distribution, calculated by GPT-2 language model without fine-tuning. The entropy and **KL** are calculated per each token, while the L^{policy} term is calculated for the whole sentence.

3.3 Decoding strategies

Recent studies show that a properly chosen decoding strategy significantly improves consistency and diversity metrics and human scores of generated samples for multiple generation tasks, such as story generation (Holtzman et al., 2019), open-domain dialogues, and image captioning (Ippolito et al., 2019). However, to the best of our knowledge, no method proved to be a one-size-fits-all one. We perform experiments with several decoding strategies, which improve diversity while preserving the desired meaning. We perform an experimental evaluation of different decoding parameters.

Beam Search, a standard decoding mechanism, keeps the top b partial hypotheses at every time step and eventually chooses the hypothesis that has the overall highest probability.

Random Sampling (top-k) (Fan et al., 2018) greedily samples at each time step one of the top-k most likely tokens in the distribution.

Nucleus Sampling (top-p) (Holtzman et al., 2019) samples from the most likely tokens whose cumulative probability does not exceed p.

Post Decoding Clustering (Ippolito et al., 2019) (i) clusters generated samples using BERT-based similarity and (ii) selects samples with the highest probability from each cluster. It can be combined with any decoding strategy.

4 Performance evaluation

We use several quality metrics to assess the generated data: (i) we use multiple fluency and diversity metrics, (ii) we account for the performance of the classifiers trained on the generated data.

Fluency. We consider fluency dependent upon the number of spelling and grammar mistakes: the utterance is treated as a fluent one if there are no misspellings and no grammar mistakes. We utilize LanguageTool (Miłkowski, 2010), a free and open-source grammar checker, to check spelling and correct grammar mistakes.

Diversity. Following (Ippolito et al., 2019), we consider two types of diversity metrics:

$Dist$-k (Li et al., 2016a) is the total number of distinct k-grams divided by the total number of produced tokens in all of the utterances for an intent;

Ent-k (Zhang et al., 2018) is an entropy of k-grams distribution. This metric takes into consideration that infrequent k-grams contribute more to diversity than frequent ones.

Accuracy. After we obtain a large amount of generated data, we train a RoBERTa-based classifier (Liu et al., 2019) to distinguish between different intents, based on the generated utterances. As usual, we split the generated data into two parts so that the first part is used for training, and the second part serves as the held-out validation set to compute the classification accuracy acc_{clsf}. High acc_{clsf} values mean that the intents are well distinguishable, and the utterances that belong to the same intent are semantically consistent.

Human evaluation We perform two crowd-sourcing studies to evaluate the quality of generated utterances, which aim at the evaluation of semantic correctness and fluency.

First, we asked crowd workers to evaluate semantic correctness. We gave crowd workers an utterance and asked them to assign one of the four provided intent descriptions; a correct option was among them (i.e., the one used to generate this very utterance). For the sake of completeness, we added a fifth option, "none of above". We assess the results of this study by two metrics, accuracy and $recall@4$. Accuracy acc_{crowd} measures the number of correct answers, while $recall@4$ measures the number of answers which are different from the last "none of above" option.

Second, we asked crowd workers to evaluate the fluency of generated utterances. Crowd workers were provided with an utterance and were asked to score it on a Likert-type scale from 1 to 5, where (5) means that the utterance sounds natural, (3) means that the utterance contains some errors, (1) means that it is hard or even impossible to understand the utterance. We assess the results of this study by computing the average score.

5 Zero-shot generation experiments

5.1 Data preparation

Data for fine-tuning. We combined two NLU datasets, namely The Schema-Guided Dialogue Dataset (SGD) (Rastogi et al., 2020) and Natural Language Understanding Benchmark (NLU-bench) (Coucke et al., 2018) for the fine-tuning stage. Both datasets have a two-level hierarchical

structure: they are organized according to services (in SGD) or scenarios (in NLU-Bench). Each service/scenario contains several intents, typically 2-5 intents per high-level class. For example, the service *Buses_1* is divided into two intents *FindBus* and *BuyBusTickets*.

SGD dataset consists of multi-turn task-oriented dialogues between user and system; each user utterance is labeled by service and intent. We adopted only those utterances from each dialog in which a new intent arose, which means the user clearly announced a new intention. This is a common technique to remove sentences that do not express any intents. As a result, we got three utterances per dialog on average.

As NLU-Bench consists of user utterances, each marked up with a scenario and intent label, we used it without filtering. Summary statistics of the dataset used is provided in Table 1.

	SGD	NLU-bench	Total
No. of utterances	49986	25607	75593
No. of services	32	18	50
No. of intents	67	68	135
Total tokens	~550k	~170k	~720k
Unique tokens	~10.8k	~8.3k	~17.4k

Table 1: The total number of utterances, intents, services and words across datasets and final statistics of our fine-tuning data.

Intent set for generation. For the evaluation of our generation methods, we created a set of 38 services and 105 intents[1] covering the most common requirements of a typical user of a modern dialogue system. The set includes services dedicated to browsing the Internet, adjusting mobile device settings, searching for vehicles, and others. To adopt a zero-shot setup, we split the data into train and test sets in the following way. Some of the services are unseen ($s \in \mathcal{U}$), i.e., are present in the test set only. There are no seen services in the train set related to unseen services. The rest of the services are seen, i.e., present in both train and test set ($s \in \mathcal{S}$), but different intents put in train and test sets. For example, *Flight* services are present in the train data and *Plane* service is

---

[1] The full list of services and intents in both sets presented in the Appendix

Zero-shot generation						
Decoding strategy	Automated metrics			Human evaluation		
	acc_{clsf}	$Dist$-4	Ent-4	acc_{crowd}	$recall@4$	Fluency score
Random Sampling ($b = 4$)	0.82	**0.50**	**6.20**	0.63	0.87	4.77
Nucleus Sampling ($p = 0.6$) + PDC	0.82	0.40	5.77	0.68	0.85	**4.95**
Beam Search ($b = 3$) + PDC	0.85	0.22	4.92	0.67	0.85	4.88
Beam Search ($b = 3$)	0.88	0.15	4.76	0.60	0.80	4.76
Nucleus Sampling ($p = 0.4$)	0.89	0.25	4.95	0.72	0.90	4.81
One-shot generation						
Nucleus Sampling ($p = 0.4$)	**0.94**	0.39	5.88	**0.78**	**0.91**	4.86

Table 2: Decoding strategies for zero-shot and one-shot generation. PDC stands for Post Decoding Clustering.

	acc_{crowd}	$recall@4$	$Dist$-4	Ent-4
SGD+NLU-bench	0.83	0.95	0.53	5.92

Table 3: Evaluation of the test dataset, created by merging and re-splitting two datasets under consideration.

used in the test set; from *Music* services, intents *Lookup song* and *Play song* were used for training, and *Create playlist* and *Turn on music* for a testing. To form the intent description for fine-tuning and generation, we join service and intent labels.

5.2 Evaluation

We generated 100 examples per intent using different decoding strategies and their parameters. For the more detailed evaluation, we picked up the generation methods of different decoding strategies that achieved good scores ($acc_{clsf} > 80\%$ and Ent-4 > 4). For these utterances, we performed a human evaluation of semantic correctness and diversity; Table 2 compares the decoding strategies according to various quality metrics. For a more detailed evaluation of decoding strategies, see Table 2 in Appendix.

To compare the diversity of human-generated utterances to our generated utterances, we evaluate the fine-tuning dataset with Ent-4 and $Dist$-4 metrics. The semantics of generated data is assessed by acc_{crowd} and $recall@4$. We present metrics for this dataset in Table 3.

5.3 Analysis and model comparison

Fluency. Spell checking results reveal the following issues of the generated utterances. The major issues are related to casing: an utterance may start in lower case, the first-person singular personal pronoun "I" is frequently generated in lower case,

too. Punctuation issues include missing quotes, question marks, periods, or repeated punctuation marks. Common mistakes are omitting of a hyphen in the word *"Wi-Fi"* and *"e-mail"* and confusing definite and indefinite articles, as well as confusing *"a"/"an"*. These issues are more or less natural to humans and thus do not prevent further use of generated utterances. The only unnatural issues found by LanguageTool are phrase repetition in small numbers (4 errors of this type per 10000 utterances). For examples of fluency issues in generated data, see Table 1 in Appendix.

Diversity. Table 4 shows examples of the phrases generated by means of different decoding strategies, conditioning on the intent *Show message*, along with diversity metrics, $Dist$ and Ent. Higher Ent and $Dist$ scores indeed correspond to a more diverse decoding strategy. At the same time, extremely high diversity may generate utterances unrelated to the intent, expressing non-clear meaning and lack of common sense.

Diversity / Accuracy trade-off. Figure 2 shows the trade-off between the diversity (Ent-4) and the accuracy (acc_{clsf}) of the generated data.

Every point corresponds to sentences generated using different zero-shot strategies. The human level stands for the diversity and accuracy metrics computed for the test set as is. The beam search scores are mainly in the top-left corner of the plane, leading to high accuracy and low diversity values.

Beam Search (3) $Ent\text{-}4 = 4.26$	Random Sampling (3) $Ent\text{-}4 = 5.93$	Nucleus Sampling (0.98) $Ent\text{-}4 = 6.86$
i need to know what's going on with my phone	i want to see my messages in the phone book	show me a message from jean lee for my favorite apple company
i want you show me the message from my phone	show me my most recent messages from my phone number	how can you tell me mike with the message
i want you show me my messages on my phone	show me the messages from the device i was using	could you check to see if my friends are in a group that is gossiping
i want you to show my messages on my smart phone	show me the message from my friend jane that i sent to her	list all messages in my bbq menu from ausy
i want to read a new message from my friend	can you please show me the messages from my phone	just turn on the smart mute this monday night

Table 4: Utterances, generated by different decoding strategies and the diversity scores of the decoding strategies.
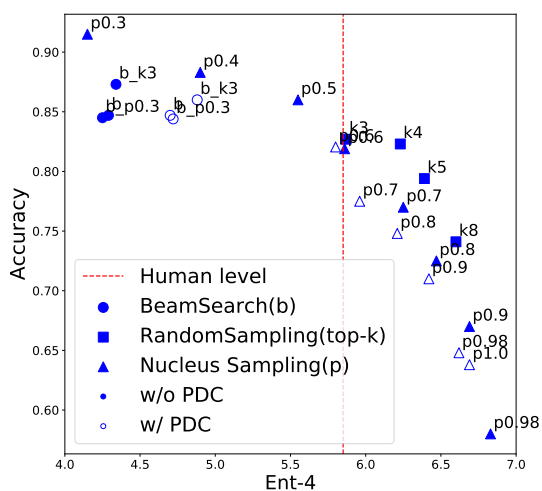

Figure 2: The trade-off between diversity ($Ent\text{-}4$) and accuracy.

Top-k Random Sampling strategy does not achieve the highest levels of accuracy. Nucleus Sampling can generate datasets with a large range of diversity and accuracy scores, depending on the chosen parameter. Post-decoding clustering increases diversity for low-diverse decoding strategies and decreases it for high-diverse ones, moving the generator closer to the human level.

Two ways to assess accuracy. Table 2 shows that there is no clear correspondence between automated accuracy acc_{clsf} and human accuracy acc_{crowd}. Therefore acc_{clsf} cannot serve as the final measure for the semantic consistency of the generator. The *Semantic shift* problem cannot be captured by the automated accuracy acc_{clsf}: the

model generates examples which are consistent inside each class, and classes are well-separated, but the generated examples do not correspond well to the intent descriptions.

5.4 Semantic shift problem

The semantic consistency is crucial: how well do the generated utterances correspond to the intent description? In most cases, zero-shot generation is quite reliable: $acc_{crowd} > 0.8$ for 57% of intents, $recall@4 > 0.9$ for 72% of intents. However, generated utterances are distinguishable from other classes for some intents, but they do not completely correspond to the intent description. Several generated utterances below illustrate this issue.

Intent: *Buy train tickets*
Utterance: I want to buy a bus ticket. I want to leave on the 12th of this month.
Intent: *Put default wallpapers*
Utterance: Put the default wallpaper for the bedroom. I want to see it on the wall.
Intent: *Calculator Find sum*
Utterance: I need to find a calculator. I need to know the value of one dollar.

For example, The bias in the fine-tuning data causes this issue. For example, travel-related intents mainly correspond to bus travel. So the model confuses buses and trains. In other cases, the model gets wrong the intent description due to the lack of world knowledge. E. g. the generated phrases for *Wallpaper* may be related to wallpapers in a house; utterances for *Calculator* may be related to finding some numbers like the average price of houses in the area.

Intent description and reference examples	Undesirable meaning	Zero-shot	One-shot
Intent description Train Buy train ticket **Reference** Make a purchase of the train ticket, not bus. Buy a train ticket for a specific date to some location	**Meaning** Get bus ticket **Example** I need a bus to go there. I need to leave on the 3rd of this month.	97	23
Intent description Wallpapers Put default wallpaper **Reference** Change the background picture of the device display to the default one. Replace current background on the device with the default one	**Meaning** Put new wall cover in a house **Example** I want to put the wallpaper for my bedroom on the wall.	74	1
Intent description Calculator Find sum **Reference** Compute, calculate the sum of the given numbers. Open the calculator and compute the sum of the following numbers	**Meaning** Find some amount of money **Example** I need to find the average price of a house.	57	0

Table 5: Evaluation of semantic shift reduction by one-shot generation. The first column contains intent description and reference utterances used for one-shot generation. The second column shows examples of typical undesirable meaning. The last two columns show the percentage of examples with given incorrect meaning among 100 generated utterances by zero-shot and one-shot generation. Nucleus sampling ($p = 0.4$) is used for both methods.

6 One-shot generation experiments

Based on human evaluation of zero-shot generated data, we select Nucleus Sampling ($p = 0.4$) as the best decoding strategy and apply it further in the one-shot scenario. Indeed, Table 2 confirms that the one-shot generation improves all evaluation metrics, both human and automated. The resulting one-shot utterances are more fluent than zero-shot utterances. The classifier trained on one-shot utterances has higher accuracy values when compared to the one trained on zero-shot utterances.

At the same time, one-shot generation restricts the semantics of the generated utterances and reduces the semantic shift. To illustrate, how the problem of semantic shift diminishes, we study several cases where the zero-shot model tends to generate utterances with undesirable meaning (see Section 5.4): **bus** instead of **train**; **wallpaper** as a *wall cover* instead of *background picture*; **sum** as *amount of money* instead of *number*. Table 5 shows that after one-shot fine-tuning, the number of utterances with undesirable meaning becomes drastically lower; for more examples, see Table 3 in Appendix.

7 Conclusion

In this paper, we have introduced zero-shot and one-shot methods for generating utterances from intent descriptions. We ensure the high quality of the

generated dataset by a range of different measures for diversity, fluency, and semantic correctness, including a crowd-sourcing study. We show that the one-shot generation outperforms the zero-shot one based on all metrics considered. Using only a single utterance for an unseen intent to fine-tune the model increases diversity and fluency. Moreover, fine-tuning on a single utterance diminishes the semantic shift problem and helps the model gain better world knowledge.

Virtual assistants in real-life setup should be highly adaptive. In some tasks, we need much more data than is currently available: exploring model robustness to distribution change, finding the best architecture, dealing with a fast-growing set of intents (the number of intents could be thousands). If the intents to support come from different providers, they pose diverse semantics, style, and noises. Adaptation to different user groups and individual users, having different intent usage distribution, is another crucial problem. We need large-scale and flexible datasets to approach these tasks, which can hardly be collected via crowd-sourcing from external sources.

Zero- or one-shot generation is an appealing technique. The model obtains the background knowledge about the world and the domain during pretraining. Next, only small amounts of data are

needed to fine-tune the model. State-of-the-art pre-trained language models, fine-tuned in a zero- or one-shot fashion, generate fluent and diverse phrases close to real-life utterances. The meaning of the intent and essential details, such as book titles, movie genres, expression of speech acts, or emoticons, are preserved. What is more, manipulating a decoding strategy makes it possible to balance the generated utterances' diversity, semantic consistency, and correctness.

Our future work directions include assessing the downstream performance of proposed generation methods for an end-user application and evaluating slot-filling performance. The proposed approach can be tested to generate utterances specific to interest groups.

Acknowledgements

References

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically Labeled Data Generation for Large Scale Event Extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot nlg with pretrained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190.

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. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The webnlg challenge: Generating text from dbpedia data.

In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. SNIPS Voice Platform: an Embedded Spoken Language Understanding System for Private-by-Design Voice Interfaces. *arXiv preprint arXiv:1805.10190*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and Play Language Models: A Simple Approach to Controlled Text Generation. In *International Conference on Learning Representations*.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Chun Wang, and Bing Ma. 2019. Investigating Capsule Network and Semantic Feature on Hyperplanes for Text Classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 456–465.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 218–228.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't Until the Final Verb Wait: Reinforcement learning for Simultaneous Machine Translation. In *Proceedings of the 2014 Conference on empirical methods in natural language processing (EMNLP)*, pages 1342–1352.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*.

Daphne Ippolito, Reno Kriz, Maria Kustikova, João Sedoc, and Chris Callison-Burch. 2019. Comparison of Diverse Decoding Methods from Conditional Language Models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Proceedings of NAACL-HLT*, pages 1875–1885.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL:

A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A Diversity-Promoting Objective Function for Neural Conversation Models. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016b. Deep Reinforcement Learning for Dialogue Generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ryan Thomas Lowe, Nissan Pow, Iulian Vlad Serban, Laurent Charlin, Chia-Wei Liu, and Joelle Pineau. 2017. Training end-to-end dialogue systems with the ubuntu dialogue corpus. *Dialogue & Discourse*, 8(1):31–65.

Tomas Mikolov and Geoffrey Zweig. 2012. Context Dependent Recurrent Neural Network Language Model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239. IEEE.

Marcin Miłkowski. 2010. Developing an Open-source, Rule-based Proofreading Tool. *Software: Practice and Experience*, 40(7):543–566.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2019. What is this article about? extreme summarization with topic-aware convolutional neural networks. *Journal of Artificial Intelligence Research*, 66:243–278.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A Deep Reinforced Model for Abstractive Summarization. *arXiv preprint arXiv:1705.04304*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical Sequence Training for Image Captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024.

Kyle Richardson, Sina Zarrieß, and Jonas Kuhn. 2017. The code2text challenge: Text generation in source libraries. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 115–119.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27:3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Pavlos Vougiouklis, Hady Elsahar, Lucie-Aimée Kaffee, Christophe Gravier, Frédérique Laforest, Jonathon Hare, and Elena Simperl. 2018. Neural wikipedian: Generating textual summaries from knowledge base triples. *Journal of Web Semantics*, 52:1–15.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced Ranker-Reader for Open-domain Question Answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jason Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint*

Conference on Natural Language Processing (EMNLP-IJCNLP), pages 6383–6389.

Qingyang Wu, Lei Li, and Zhou Yu. 2021. Textgail: Generative adversarial imitation learning for text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14067–14075.

C Xia, C Xiong, and PS Yu. 2021. Pseudo siamese network for few-shot intent generation. In *ACM SIGIR*.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3905–3914.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659.

Yunlong Yu, Zhong Ji, Zhongfei Zhang, and Jungong Han. 2019. Episode-based prototype generating network for zero-shot learning.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594.

Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018. Generating Informative and Diverse Conversational Responses via Adversarial Information Maximization. *Advances in Neural Information Processing Systems*.

Tiancheng Zhao and Maxine Eskenazi. 2018. Zero-shot dialog generation with cross-domain latent actions. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 1–10.