Data Augmentation for Intent Classification with Off-the-shelf Large Language Models

Anonymous ACL submission

Abstract

Data augmentation is a widely employed technique to alleviate the problem of data scarcity. In this work, we propose a prompting-based approach to generate labelled training data for intent classification with off-the-shelf language models (LMs) such as GPT-3. An advantage of this method is that no task-specific LM-fine-tuning for data generation is required; hence the method requires no hyper-parameter tuning and is applicable even when the available training data is very scarce. We evaluate the proposed method in a few-shot setting on four diverse intent classification tasks. We find that GPT-generated data significantly boosts the performance of intent classifiers when intents in consideration are sufficiently distinct from each other. In tasks with semantically close intents, we observe that the generated data is less helpful. Our analysis shows that this is because GPT often generates utterances that belong to a closely-related intent instead of the desired one. We present preliminary evidence that a prompting-based GPT classifier could be helpful in filtering the generated data to enhance its quality.

1 Introduction

A key challenge in creating task-oriented conversational agents is gathering and labelling training data. Standard data gathering options include manual authoring and crowd-sourcing. Unfortunately, both of these options are tedious and expensive. *Data augmentation* is a widely used strategy to alleviate this problem of data acquisition.

There are two particularly promising paradigms for data augmentation in natural language processing that use pretrained language models (LMs) (Peters et al., 2018; Devlin et al., 2018). The first family of methods fine-tune an LM on task-specific data and generate new examples using the fine-tuned LM (Wu et al., 2018; Kumar et al., 2019, 2021; Anaby-Tavor et al., 2020; Lee et al., 2021).

Input Prompt:

```
The following sentences belong to the same
category music_likeness:

Example 1: i like soft rock music
Example 2: current song rating three stars
Example 3: save this song as a favorite
Example 4: remind me that i like that song
Example 5: save my opinion on the currently
           playing song
Example 6: i love the song do you
Example 7: add the song to my favorites
Example 8: store opinion on song
Example 9: the song in background is cool
Example 10: i am the living blues
Example 11:
```

Completions:

```
i dislike classical music
she is a music lover
i am a lover of painting
this is the best song ever
video that looks like the other video
save preference on my profile
express negative opinion on the song
i am a great blues follower
the song is better than i thought
this song is also fun
```

Figure 1: **Generation Process.** Given a seed intent (here, music_likeness) and K(=10) available examples for that intent, we construct a prompt following the shown template. Note that the last line of the prompt is incomplete (there is no new line character.) We then feed this prompt to a GPT-3 engine, which generates some completions of the prompt. In this example, `red text` denotes unfaithful examples and `blue text` denotes faithful examples. **Note:** For brevity, we only show ten generated sentences.

A limitation of these methods is that, in a real-world scenario, task-specific data is scarce, and fine-tuning an LM can quickly become the bottleneck. The second family of methods sidesteps this bottleneck by employing off-the-shelf pretrained LMs such as GPT-3 (Brown et al., 2020) to directly generate text without any task-specific fine-tuning.

In particular, data generation by the GPT3Mix approach by Yoo et al. (2021) boosts performance on multiple classification tasks; however, they only consider tasks with few (up to 6) classes and easy-to-grasp class boundaries (e.g., *positive* and *negative*).

This work studies the applicability of massive off-the-shelf LMs, such as GPT-3 and GPT-J (Wang and Komatsuzaki, 2021) to perform effective data augmentation for intent classification (IC) tasks. In IC, the end goal is to predict a user's intent given an utterance, i.e., what the user of a task-oriented chatbot wants to accomplish. Data augmentation for IC is particularly challenging because the generative model must distinguish between a large number (in practice up to several hundreds) of fine-grained intents that can be semantically very close to each other. Prior methods such as GPT3Mix prompt the model with the names of all classes as well as a few examples from randomly chosen classes. Such approaches are poorly suitable for intent classification tasks with tens or hundreds of possible intents. Instead, in this study, we use a simple prompt structure that focuses on a single seed intent (see Figure 1) as it combines the intent's name and all available examples.

Our experiments primarily focus on few-shot IC on four prominent datasets: CLINC150 (Larson et al., 2019), HWU64 (Xingkun Liu and Rieser, 2019), Banking77 (Casanueva et al., 2020), and SNIPS (Coucke et al., 2018). We also consider a partial few-shot setup to compare to the Example Extrapolation (Ex2) approach by Lee et al. (2021) who use a similar prompt but fine-tune the LM instead of using it as is. The main findings of our experiments are as follows: **(1)** GPT-generated samples boost classification accuracy when the considered intents are well-distinguished from each other (like in CLINC150, SNIPS). **(2)** On more granular datasets (namely HWU64 and Banking77), we find that GPT struggles in distinguishing between different confounding intents. **(3)** A small-scale study to further understand this behaviour suggests that GPT could be used as a classifier to filter out unfaithful examples and enhance the quality of the generated training set. Additionally, we investigate how valuable the generated data could be if relabelled by a human. Using an oracle model, we show that **(4)** the human labelling of GPT-generated examples can further improve the performance of intent classifiers, and that **(5)** LM-generated data has a higher relabelling potential compared to edit-based augmentation techniques, such as Easy Data Augmentation (EDA) (Wei and Zou, 2019).

2 Method

We consider training an intent classifier, where an intent is a type of request that the conversational agent supports; e.g. the user may want to change the language of the conversation, play a song, transfer money between accounts, etc. However, collecting many example utterances that express the same intent is difficult and expensive. Therefore, this paper experiments with a straightforward method to augment the training data available for an intent: creating prompts from the available examples and feeding them to a large language model such as GPT-3 (Brown et al., 2020). Figure 1 illustrates the process of data generation for an intent with K available examples.

3 Experimental Setup

3.1 Datasets

We use four intent classification datasets in our experiments with varying levels of granularity among intents. CLINC150 (Larson et al., 2019), HWU64 (Xingkun Liu and Rieser, 2019) are multi-domain datasets, each covering a wide range of typical task-oriented chatbot domains, such as playing music and setting up alarms. Importantly, the CLINC150 task also contains examples of out-of-scope (OOS) utterances that do not correspond to any of CLINC's 150 intents. Banking77 (Casanueva et al., 2020) is a single domain dataset with very fine-grained banking-related intents. Finally, the SNIPS (Coucke et al., 2018) dataset contains 7 intents typical for the smart speaker usecase. We refer the reader to Table 1 for exact statistics of all used datasets.

3.2 Setup

The main data-scarce setup that we consider in this work is the *few-shot setup*, where only $K = 10$ training examples are available for every intent of interest. Additionally, to compare to example extrapolation with fine-tuned language models as proposed by Lee et al. (2021), we consider a *partial few-shot setup*. In the latter setting, we limit the amount of training data only for a handful of *few-*

	CLINC150	SNIPS	HWU64	Banking77
domains	10	1	18	1
intents	150	7	64	77
train examples	15000 (100)	13084	8954*	9002*
val. examples	3000 (100)	700	1076*	1001*
test examples	4500 (1000)	700	1076	3080

Table 1: Statistics of the intent classification datasets that we use in our experiments. * indicates that we split the original data into training and validation instead of using a split provided by the dataset authors. For CLINC150, the number of out-of-scope examples in different data partitions is given in parenthesis.

shot intents[1] and use the full training data for others. When data augmentation is performed, we augment the few-shot intents to have N examples, where N is the median number of examples per intent of the original data.

To precisely describe the training and test data in all settings, we will use D_{part} to refer to dataset parts, i.e. train, validation, and test. In addition, we use D_F and D_M to refer to data-scarce and data-rich intents (the latter only occur in the partial few-shot setting). This notation is defined for all parts, therefore, $D_{part} = D_{\{F,part\}} \cup D_{\{M,part\}}$, $\forall part \in \{train, val, test\}$. When GPT-3 or a baseline method is used to augment the training data we generate $N - K$ examples per intent and refer to the resulting data as $\tilde{D}_{F,train}$. We experiment with four different-sized GPT-3 models[2] by OpenAI and GPT-J by EleutherAI[3] to obtain $\tilde{D}$. The four GPT-3 models are: Ada, Babbage, Curie, and Davinci. In order, Ada is the smallest model and Davinci is the largest. Model sizes of GPT-3 engines are not known precisely but are estimated by Eleuther AI to be between 300M and 175B parameters[4].

3.3 Training and Evaluation

We fine-tune BERT-large (Devlin et al., 2018) on the task of intent classification by adding a linear layer on top of the `[CLS]` token (Wolf et al., 2019).

[1]We use the truncation heuristic provided by Lee et al. (2021): https://github.com/google/example_extrapolation/blob/master/preprocess_clinc150.py

[2]https://beta.openai.com/docs/engines

[3]https://github.com/kingoflolz/mesh-transformer-jax/

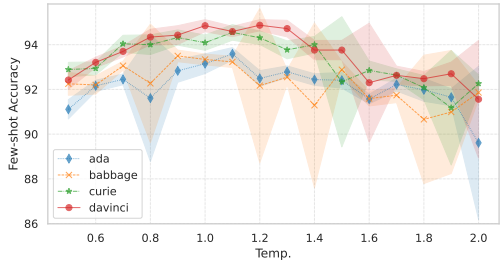
[4]https://blog.eleuther.ai/gpt3-model-sizes/

In all setups we use the original validation set for tuning the classifier's training hyperparameters. We chose to use the full validation set as opposed to a few-shot one to avoid issues with unstable hyperparameter tuning and focus on assessing the quality of the generated data.

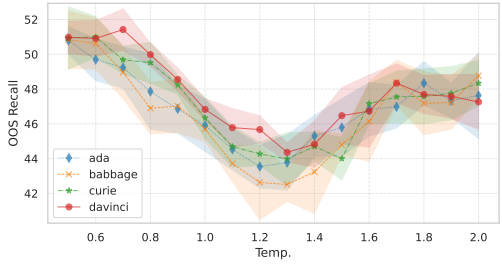
Full few-shot. In this setup, we treat *all* the intents as few-shot and evaluate our method on the following three scenarios: (i) **Baseline**: all the intents are truncated to $K = 10$ samples per intent, (ii) **Augmented**: $\tilde{D}_{\{F,train\}}$ is generated using GPT and models are trained on $D_{\{F,train\}} \cup \tilde{D}_{\{F,train\}}$ and (iii) **EDA-baseline**: same as above, but $\tilde{D}_{\{F,train\}}$ is generated using Easy Data Augmentation (EDA) by Wei and Zou (2019). For each scenario, we report the 1) overall in-scope accuracy on the complete test set D_{test}, i.e. intent classification accuracy excluding OOS samples in the test set, and 2) few-shot classification accuracy of the models on $D_{\{F,test\}}$. For CLINC150, we also report out-of-scope recall (OOS recall) on D_{test} that we compute as the percentage of OOS examples that the model correctly labelled as such.

The purpose of this setting is to estimate what further gains can be achieved if the data generated by GPT were labelled by a human. We train an oracle $\mathcal{O}$ on the full training data D_{train}. We also use $\mathcal{O}$ to assess the quality of the generated data. Namely, we compute *fidelity* of the generated data as the ratio of generated utterances that the oracle labels as indeed belonging to the intended seed intent. A higher fidelity value means that the generated samples are more faithful to original data distribution.

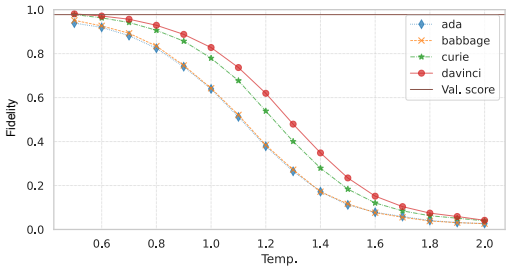
Partial few-shot. In this setup, we train $\mathcal{S}$ intent classifiers, choosing different *few-shot intents* every time to obtain D_F. We then average the metrics across these $\mathcal{S}$ runs. For CLINC150, $\mathcal{S} = 10$ corresponding to the 10 different domains, whereas for SNIPS, $\mathcal{S} = 7$ corresponding to the 7 different intents. We evaluate our method on the following three scenarios introduced by Lee et al. (2021): (i) **Baseline**: models are trained without data augmentation on $D_{\{F,train\}} \cup D_{\{M,train\}}$. (ii) **Upsampled**: $D_{\{F,train\}}$ is upsampled to have N examples per intent. Then models are trained on upsampled $D_{\{F,train\}} \cup D_{\{M,train\}}$. (iii) **Augmented**: models are trained on $D_{\{F,train\}} \cup \tilde{D}_{\{F,train\}} \cup D_{\{M,train\}}$. For each scenario in this setup, we report the overall in-scope classification accuracy (and OOS Recall



(a) Temperature v/s Few-shot accuracy

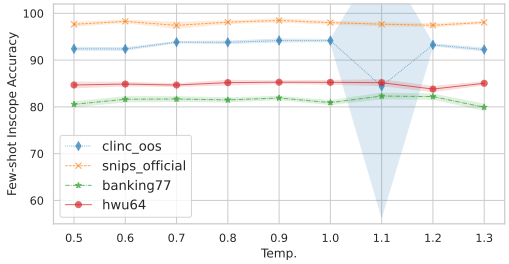


(b) Temperature v/s OOS recall

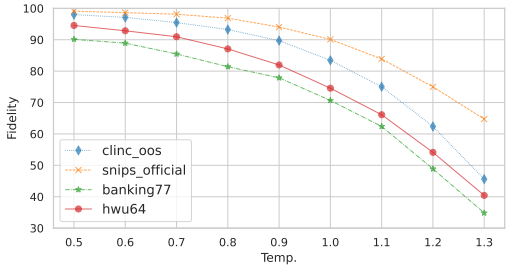


(c) Temperature v/s Fidelity

Figure 2: **Partial few-shot validation performance for different GPT-3 models and temperatures.** (a) few-shot accuracy, (b) OOS recall of intent classifiers trained on augmented sets, and (c) fidelity measured as the accuracy of the oracle on the augmented sets.



(a) Temperature v/s Few-shot inscope accuracy



(b) Temperature v/s Fidelity

Figure 3: **Full few-shot validation performance for different GPT-J temperatures on different datasets.** (a) few-shot inscope accuracy of intent classifiers trained on augmented sets, and (b) fidelity (oracle accuracy) of augmented sets generated by GPT-J with different temperatures.

for CLINC150).

For both partial few-shot and full few-shot settings, we report means and standard deviations over 10 repetitions of each experiment.

4 Experimental Results

Full few-shot. Table 2 shows the results of our few-shot experiments. For CLINC150 and SNIPS, data augmentation with GPT-3 is very effective as it leads to respective accuracy improvements of up to approximately 3.7% and 6% on these tasks. These improvements are larger than what the baseline EDA method brings, namely 2.4% and 2.9% for CLINC150 and SNIPS. Importantly, using larger GPT models for data augmentation brings significantly bigger gains. Data augmentation results on Banking77 and HWU64 are, however, much worse, with no or little improvement upon the plain few-shot baseline. We present a thorough investigation of this behaviour in Section 4.1. One can also see that data augmentation with GPT models lowers the OOS recall.

Next, we observe that relabelling EDA and GPT-generated sentences by the oracle gives a significant boost to accuracies across the board, confirming our hypothesis that human inspection of generated data could be fruitful. Importantly, we note that the magnitude of improvement for EDA is less than for GPT models. This suggests that GPT models generate more diverse data that can eventually be more useful after careful human inspection. Lastly, relabelling also improves OOS recall on CLINC150, which is due to the fact that much of the generated data was labelled as OOS by the oracle.

Partial few-shot. Table 3 shows the results of our partial few-shot experiments on CLINC150 and SNIPS. By augmenting the dataset with GPT-generated samples, the few-shot accuracy improves by up to 2.92% on CLINC150 and 18.14% on SNIPS compared to the baseline setting. Our method achieves competitive results compared to

Model	CLINC150		HWU64	Banking77	SNIPS
	IA (96.93)	OR (42.9)	IA (92.75)	IA (93.73)	IA (98.57)
EDA	92.66 (0.40)	43.81 (2.03)	83.67 (0.48)	83.96 (0.66)	92.50 (1.61)
Baseline (Ours)	90.28(0.49)	50.18(1.14)	81.43 (0.57)	83.35 (0.59)	89.69 (1.63)
Augmented					
Ada (Ours)	91.31 (0.34)	21.69 (1.57)	79.68 (0.83)	79.30 (0.42)	94.27 (0.52)
Babbage (Ours)	92.72 (0.33)	22.99 (2.39)	81.86 (0.78)	80.31 (0.41)	94.74 (0.67)
Curie (Ours)	93.37 (0.21)	25.85 (1.49)	82.85 (0.70)	83.50 (0.44)	94.73 (0.62)
GPT-J (Ours)	93.25 (0.19)	24.02 (1.45)	81.78 (0.56)	82.32 (0.90)	95.19 (0.61)
Davinci (Ours)	94.07 (0.18)	27.36 (1.08)	82.79 (0.93)	83.60 (0.45)	95.77 (0.86)
Augmented + Relabelled					
EDA	93.43 (0.22)	48.56 (1.84)	85.58 (0.73)	84.82 (0.57)	94.91 (0.66)
Ada (Ours)	95.09 (0.16)	41.38 (1.77)	88.53 (0.61)	88.45 (0.19)	97.03 (0.18)
Babbage (Ours)	95.39 (0.17)	40.58 (1.63)	89.49 (0.32)	88.86 (0.26)	96.89 (0.49)
Curie (Ours)	95.08 (0.19)	40.09 (2.38)	89.78 (0.47)	88.30 (4.64)	96.86 (0.31)
GPT-J (Ours)	95.11 (0.13)	43.94 (1.76)	89.52 (0.54)	88.94 (0.40)	97.33 (0.38)
Davinci (Ours)	95.08 (0.13)	40.76 (1.37)	89.53 (0.45)	88.89 (0.31)	97.03 (0.38)

Table 2: **Full few-shot results on CLINC150, HWU64, Banking77, and SNIPS datasets. IA**: Inscope Accuracy (mean (std)). **OR**: OOS-Recall (mean (std)). Towards the top of the table, we also report the test set performance (enclosed in parentheses) when all examples are used for fine-tuning (without any augmentation.)

		CLINC150			SNIPS	
		Overall		Few-shot	Overall	Few-shot
	Classifier	IA	OR	A	IA	A
Baseline♠	T5	97.4	-	93.7	95.2	74.0
Upsampled♠	T5	97.4	-	94.4	95.9	80.0
Augmented (Ex2)♠	T5	97.4	-	95.6	97.8	94.0
Baseline (ours)	BERT	96.28 (0.06)	39.14 (0.82)	91.36 (0.47)	95.47 (0.45)	78.38 (3.34)
Upsample (ours)	BERT	96.20 (0.05)	40.21 (0.59)	90.93 (0.19)	95.29 (0.37)	79.28 (2.05)
Augmented (Ada)	BERT	96.16 (0.05)	34.37 (0.27)	92.60 (0.15)	97.30 (0.24)	94.41 (0.72)
Augmented (Babbage)	BERT	96.39 (0.06)	35.71 (0.46)	93.66 (0.21)	97.46 (0.25)	95.31 (0.74)
Augmented (Curie)	BERT	96.41 (0.06)	36.77 (0.93)	93.90 (0.21)	97.37 (0.19)	94.79 (0.64)
Augmented (GPT-J)	BERT	96.38 (0.05)	35.91 (0.94)	93.85 (0.25)	97.59 (0.21)	96.08 (0.39)
Augmented (Davinci)	BERT	96.45 (0.03)	37.52 (0.54)	94.28 (0.24)	97.66 (0.21)	96.52 (0.35)

Table 3: **Partial few-shot results on CLINC150 and SNIPS datasets.** Refer to Section 3.3 for more details. **IA**: Inscope accuracy (mean (std)). **OR**: OOS Recall (mean (std)). **A**: Accuracy (mean (std)). ♠ (Lee et al., 2021).

Ex2 (Lee et al., 2021), both in terms of absolute accuracies and the relative gains brought by data augmentation. Note that Ex2 uses T5-XL (Roberts et al., 2020) with nearly 3 billion parameters as its base intent classifier, while our method uses BERT-large with only 340 million parameters.

4.1 Analysis

Effect of GPT sampling temperature. We investigate the impact of generation temperature on the quality and fidelity of generated data. We perform this investigation on the CLINC150 dataset using the partial few-shot setup. Results in Figure 2 show that, for all engines, the generated data leads to the highest classification accuracy when the generation temperature is around 1.0, although lower tempera-

Davinci generated sentences	Seed Intent	Oracle Prediction
HWU64		
play a song with the word honey	music_likeness	play_music
you are playing music	music_likeness	play_music
'let me hear some of that jazz!'	music_likeness	play_music
i really like myspace music	play_music	music_likeness
i love the start lucky country music	play_music	music_likeness
thank you for the music	play_music	music_likeness
please play the next song	music_settings	play_music
play background music	music_settings	play_music
play the hour long loop of rock song	music_settings	play_music
need you to play that song one more time	play_music	music_settings
skip that song, its turkish	play_music	music_settings
pickup the beat or a temp track or audio plugin	play_music	music_settings
Banking77		
My last attempt to top up didn't seem to work, any success?	topping_up_by_card	top_up_failed
I tried to top off my wallet using my card but it says "top up failed".	topping_up_by_card	top_up_failed
I cannot top-up by my cellular phone number? How do I do that?	topping_up_by_card	top_up_failed
Can you transfer money to my Ola prepaid option? Or help me top up my card to money. They never accept my card so I always have to suffer	top_up_failed	topping_up_by_card
Hi my app is activated on activate.co.in, but unable to top up my phone. I tried credit card, debit card and Paytm but fails	top_up_failed	topping_up_by_card
I try to top up my card but it's not going through. It's still on pending status. Do I need to wait or did I do something wrong	top_up_failed	pending_top_up
I tried top-up with my card but notification shows that 'Pending'. This has been happening since last night. Can you tell me what's going on	top_up_failed	pending_top_up
Top up didn't go through.	pending_top_up	top_up_failed
Did my master card top-up fail?	pending_top_up	top_up_failed

Table 4: **Davinci-generated sentences for closely-related intents in HWU64 and Banking77 datasets.** Highlighted sub-strings indicate a difference with respect to the seed intent.

tures result in higher OOS recall. We also observe that the fidelity of the generated samples decreases as we increase the temperature (i.e. higher diversity, see Figure 2c). This suggests that higher fidelity does not always imply better quality samples as the language model may simply copy or produce less diverse utterances at lower temperatures. In Appendix A, we perform a human evaluation, reaching similar conclusions as when using an oracle to approximate fidelity.

Fidelity on different datasets. Our results in Section 4 show that data augmentation gains are much higher on CLINC150 and SNIPS than on HWU64 and Banking77. To contextualize these results, we report the fidelity of GPT-J-generated data for all these tasks in Figure 3b. Across all generation temperatures, the fidelity of the generated data is higher for CLINC150 and SNIPS than for HWU64 and Banking77. For all datasets, the fidelity is higher when the generation temperature is lower; however, Figure 3a shows that low-temperature data also does improve the model's performance.

Data generation for close intents. To better understand the lower fidelity and accuracy on HWU64 and Banking77 datasets, we focus on intents with the lowest fidelities. Here, by intent fidelity, we mean the percentage of the intent's generated data that the oracle classified as indeed belonging to the seed intent. In the Banking77 dataset, the lowest-fidelity intent is "topping_up_by_card." For this intent, only 33% of the Davinci-generated sentences were labelled as "topping_up_by_card" by the oracle, implying that two-thirds of the sentences did not fit that intent, "top_up_failed"

Fidelity (3 intents)	HWU64	Banking77
w/o filtering (468)	60.26	57.31
w/ filtering (371)	72.51	65.54
3-way accuracy		
Davinci	86.36	78.75
10-shot BERT-large	82.95	65.54
Full data BERT-large	94.32	95.00

Table 5: The impact and the accuracy of using GPT-3 as a 3-way classifier on close intent triplets from HWU64 and Banking77 datasets. For fidelity, generated examples are rejected if the GPT-3 classifier labels them as not belonging to the seed intent. Classification accuracies are reported on the reduced validation+test sets where we only consider examples from the three confounding intents.

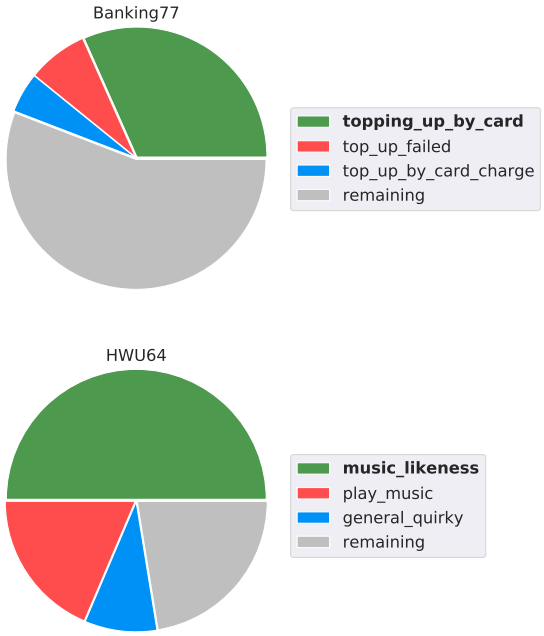


Figure 4: Distribution of labels as predicted by the oracle for lowest-fidelity intents in Banking77 and HWU64 datasets ("topping_up_by_card" and "music_likeness," respectively). Green areas denote the portion of generated sentences deemed fit by the oracle for the lowest-fidelity intents in the two datasets. Red and Blue areas respectively correspond to the most common and the second most common alternative intent predicted by the oracle.

and "top_up_card_charge" being the two most common alternatives chosen by the oracle. Similarly, only 50% of the Davinci-generated sentences abide by the lowest-fidelity "music_likeness" intent in the HWU64 dataset, "play_music" and "general_quirky" being the most common intents among the "unfaithful" sentences. Figure 4 visualizes this high percentage of unfaithful generated sentences. It also shows the proportion of the two most common alternatives that the oracle preferred over the seed intent. Table 4 presents generated sentences for confounding intents in the HWU64 and Banking77 datasets. There are clear indications of mix-up of intents, e.g., Davinci generates, "play a song with the word honey," which should belong to "play_music" rather than "music_likeness." There are also instances where the LM mixes two intents; for instance, Davinci generates "Hi my app is activated on activate.co.in, but unable to top up my phone. I tried credit card, debit card and Paytm but fails," which could belong to either "topping_up_by_card" intent (as it mentions about using credit card in the context of a top up) or "top_up_failed" (as the top up ultimately fails).

4.2 Can GPT Models Understand Close Intents?

We perform extra investigations to better understand what limits GPT-3's ability to generate data accurately. We hypothesize that one limiting factor can be GPT-3's inability to understand fine-grained differences in the meanings of utterances. To verify this hypothesis, we evaluate how accurate GPT-3 is at classifying given utterances as opposed to generating new ones. Due to the limited prompt size of 2048 tokens, we can not prompt GPT-3 to predict all the intents in the considered datasets. We thus focus on the close intent triplets from HWU64 and Banking77 datasets that we use in Table 4. We compare the 3-way accuracy of a prompted GPT-3 classifier to the similarly-measured 3-way performance of conventional BERT-large classifiers. We prompt GPT-3 with 10 examples per intent (see Figure 5). For comparison, we train BERT-large classifiers on either the same 10 examples or the full training set. Table 5 shows that the Davinci version of GPT-3 performs in between the 10-shot and the full-data conventional classifiers. This suggests that while GPT-3's understanding of nuanced intent differences is imperfect, it could still be sufficient to improve the performance of the downstream few-shot model. Inspired by this finding, we experiment with using GPT-3's classification abilities to improve the quality of generated data. Namely, we reject the generated utterances that GPT-3 classifies as not belonging to the seed intent. For both

Input Prompt:

```
Each example in the following list contains
a sentence that belongs to a category. A
category is one of the following:
music_likeness, play_music, music_settings:

sentence: next i want to hear shinedown ;
category: play_music
sentence: i am the living blues ;
category: music_likeness
sentence: open music player settings ;
category: music_settings
sentence: play hopsin from my latest
playlist ; category: play_music
sentence: i like this song ;
category:
```

GPT-3 Predictions:

```
play_music,music_likeness,music_settings,
music_likeness,music_likeness,help_command
```

Figure 5: **Using GPT-3 as a classifier.** Given a triplet of close intents, we mix and shuffle the multiple seed examples available for each of them. Then, we append an incomplete line to the prompt with just the generated sentence and feed it to GPT-3 multiple times. Among the responses, we choose the most generated in-triplet intent as the predicted intent ("music_likeness" in the above example). **Note:** For brevity, we don't show all the seed examples and predictions.

HWU64 and Banking77, this filtering method significantly improves the fidelity of the generated data for the chosen close intent triplets.

5 Related Work

The natural language processing literature features diverse data augmentation methods. Edit-based methods such as Easy Data Augmentation apply rule-based changes to the original utterances to produce new ones (Wei and Zou, 2019). In back-translation methods (Sennrich et al., 2016) available examples are translated to another language and back. Recently, data augmentation with fine-tuned LMs has become the dominant paradigm (Wu et al., 2018; Kumar et al., 2019, 2021; Anaby-Tavor et al., 2020; Lee et al., 2021). Our simpler method sidesteps LM-fine-tuning and directly uses off-she-shelf LMs as is.

The data augmentation approach that is closest to the one we use here is GPT3Mix by Yoo et al. (2021). A key part of the GPT3Mix prompt is a list of names of all possible classes (e.g. "The sentiment is one of 'positive' or 'negative'"). The LM is then expected to pick a random class from the list and generate a new example as well as the corresponding label. We assume that a list of 150 intent names would be difficult for the model to handle and instead choose a different prompt that encourages the model to extrapolate between examples of a seed intent similarly to (Lee et al., 2021).

Other work on few-shot intent classification explores fine-tuning dialogue-specific LMs as classifiers as well as using similarity-based classifiers instead of MLP-based ones on top of BERT (Vulić et al., 2021). We believe that improvements brought by data augmentation would be complementary to the gains brought by these methods.

Lastly, our method to filter out unfaithful GPT generations is related to the recent work by Wang et al. (2021) that proposes using GPT3 for data labelling. A crucial difference with respect to our work, however, is that we use GPT-3 for rejecting mislabelled samples rather than proposing labels for unlabelled samples.

6 Conclusion

We propose a prompt-based method to generate intent classification data with large pretrained language models. Our experiments show that generated data can be helpful as additional labelled data for some tasks, whereas, for other tasks, the generated data needs to be either relabelled or filtered to be helpful. We show that a filtering method that recasts the same GPT model as a classifier can be effective. Our filtering method, however, requires knowing the other intents that the generated data is likely to belong to instead of the seed intent. Future work can experiment with heuristics for approximately identifying the most likely actual intents for the generated utterances. This would complete a data generation and filtering pipeline that, according to our preliminary results in Section 4.2 here, could be effective. Other filtering methods could also be applied, such as looking at the likelihood of the generated utterances as explored in a concurrent work by Meng et al. (2022). Lastly, an interesting future work direction is identifying which generated utterances most likely need a human inspection.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do Not Have Enough Data? Deep Learning to the Rescue! *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7383–7390.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]*. ArXiv: 2005.14165.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips Voice Platform: an embedded Spoken Language Understanding system for private-by-design voice interfaces. *arXiv:1805.10190 [cs]*. ArXiv: 1805.10190.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL 2019*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2021. Data Augmentation using Pre-trained Transformer Models. *arXiv:2003.02245 [cs]*. ArXiv: 2003.02245.

Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. 2019. A Closer Look At Feature Space Data Augmentation For Few-Shot Intent Classification. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction. 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 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Kenton Lee, Kelvin Guu, Luheng He, Tim Dozat, and Hyung Won Chung. 2021. Neural Data Augmentation via Example Extrapolation. *arXiv:2102.01335 [cs]*. ArXiv: 2102.01335.

Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Generating training data with language models: Towards zero-shot language understanding. *arXiv preprint arXiv:2202.04538*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL-HLT 2018*. ArXiv: 1802.05365.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

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, Berlin, Germany. Association for Computational Linguistics.

Ivan Vulić, Pei-Hao Su, Samuel Coope, Daniela Gerz, Paweł Budzianowski, Iñigo Casanueva, Nikola Mrkšić, and Tsung-Hsien Wen. 2021. ConvFiT: Conversational fine-tuning of pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1151–1168, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want To Reduce Labeling Cost? GPT-3 Can Help. *arXiv:2108.13487 [cs]*. ArXiv: 2108.13487.

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 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2018. Conditional BERT Contextual Augmentation. *arXiv:1812.06705 [cs]*. ArXiv: 1812.06705.

Pawel Swietojanski Xingkun Liu, Arash Eshghi and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, pages xxx–xxx, Ortigia, Siracusa (SR), Italy. Springer.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. GPT3Mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Appendix

A Human Evaluation

In Figure 2 we evaluate the fidelity of the samples generated by GPT-3 with respect to the original set of sentences used to prompt it. Fidelity is approximated by the classification performance of an "oracle" intent classifier trained on the whole dataset ($D_{train} \cup D_{test}$) and evaluated over the generated samples. In order assess whether the oracle predictions are comparable to those of a human, we perform a human evaluation study.

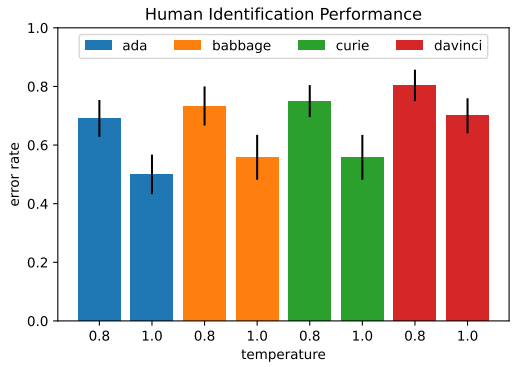


Figure 6: **Human evaluation.** Error rate of human evaluators at the task of finding whether any sentence in a group of 5 was generated by GPT-3 or not. Each color represents a different GPT-3 engine. Higher error rate indicates that humans could not correctly identify generated samples and thus it also indicates higher fidelity. The standard error is displayed as a vertical line on top of each bar.

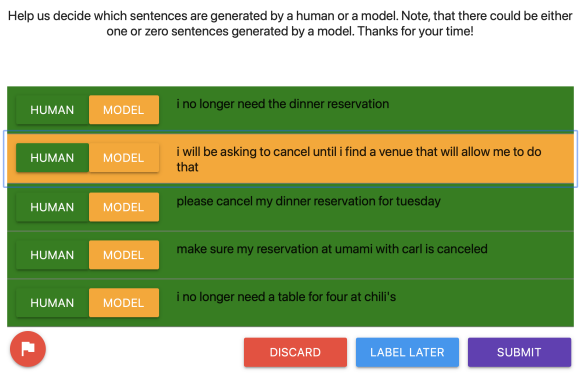


Figure 7: **Human evaluation tool.** Example of a question for the human evaluators. Human evaluators are asked to flag which example is GPT-3 generated if any among the 5 presented ones.

We consider that a model produces sentences with high fidelity if a human is unable to distinguish them from a set of human-generated sentences belonging to the same intent. Therefore, for each intent in the CLINC150 dataset, we sample five random examples and we randomly choose whether to replace one of them by a GPT-3 generated sentence from the same intent. We generate sentences with each of the four GPT-3 models considered in the main text with two different temperatures (0.8 and 1.0). The sentence to replace is randomly selected. Finally, the five sentences are displayed to a human who has to choose which of the sentences is generated by GPT-3, if any.

The task is presented to human evaluators in the form of a web application (see Figure 7). We placed a button next to each sentence in order to force human evaluators to individually consider each of the examples. Once annotated, the evaluator can either *submit*, *discard*, or leave the task to *label later*. We used a set of 15 voluntary evaluators from multiple backgrounds, nationalities, and genders. Each evaluator annotated an average of 35 examples, reaching a total of 500 evaluated tasks.

For each model and temperature, we report the error rate of humans evaluating whether a task contains a GPT-generated sample. We consider that evaluators succeeds at a given task when they correctly find the sentence that was generated by GPT or when they identify that none of them was generated. Thus, the error rate for a given model and temperature is calculated as #failed / total_evaluated.

Results are displayed in Figure 6. We find that human evaluators tend to make more mistakes when the temperature used to sample sentences from GPT-3 is smaller. This result is expected since lowering the temperature results in sentences closer to those prompted to GPT-3, which are human-made. We also observe that models with higher capacity such as `Davinci` tend to generate more indistinguishable sentences than lower-capacity models such as `Ada`, even for higher temperatures. These results are in agreement with the "oracle" fidelity results introduced in Figure 2.