Calibrate Before Use: Improving Few-Shot Performance of Language Models

Tony Z. Zhao [*1] **Eric Wallace** [*1] **Shi Feng** [2] **Dan Klein** [1] **Sameer Singh** [3]

Abstract

GPT-3 can perform numerous tasks when provided a natural language prompt that contains a few training examples. We show that this type of few-shot learning can be unstable: the choice of prompt format, training examples, and even the order of the training examples can cause accuracy to vary from near chance to near state-of-the-art. We demonstrate that this instability arises from the bias of language models towards predicting certain answers, e.g., those that are placed near the end of the prompt or are common in the pre-training data. To mitigate this, we first estimate the model's bias towards each answer by asking for its prediction when given the training prompt and a content-free test input such as "N/A". We then fit calibration parameters that cause the prediction for this input to be uniform across answers. On a diverse set of tasks, this *contextual calibration* procedure substantially improves GPT-3 and GPT-2's average accuracy (up to 30.0% absolute) and reduces variance across different choices of the prompt.

1. Introduction

Few-shot learning—the ability to learn tasks with limited examples—is an important aspect of intelligence (Lake et al., 2015; Yogatama et al., 2019). Recent work shows that large neural language models can perform few-shot learning without finetuning (Radford et al., 2019; Brown et al., 2020). Specifically, GPT-3 (Brown et al., 2020) can perform numerous tasks when provided a few examples in a natural language *prompt*. For example, to perform sentiment analysis one can condition GPT-3 on a prompt such as:

```
Input: Subpar acting. Sentiment: Negative
Input: Beautiful film. Sentiment: Positive
Input: Amazing.       Sentiment:
```

where the first two lines correspond to two training examples and the last line is a test example. To make predictions, the model predicts whether the subsequent token is more likely to be the word "Positive" or "Negative".

This style of few-shot "in-context" learning is interesting because it shows that the model can learn without parameter updates. And, more importantly, it has numerous practical advantages over the now-standard approach of finetuning (Radford et al., 2018; Devlin et al., 2019). First, it allows practitioners to "rapidly prototype" NLP models: changing the prompt *immediately* leads to a new model. Second, it provides a fully natural language interface to a machine learning model, which allows users—even those without technical expertise—to create NLP systems. Finally, since in-context learning reuses the same model for each task, it reduces memory requirements and system complexity when serving many different tasks.

However, despite these promises, we show that GPT-3's accuracy can be highly unstable across different prompts (Section 3). A prompt contains three components: a format, a set of training examples, and a permutation (ordering) for those examples. We show that different choices for these factors can lead to highly different accuracies, e.g., changing the permutation of the training examples in a sentiment analysis prompt can change accuracy from near chance (54%) to near state-of-the-art (93%). This instability implies that GPT-3 users, who typically design prompts manually, cannot expect to consistently obtain good accuracy.

We next analyze what causes this instability. We identify three pitfalls of language models that lead them to be biased toward certain answers during few-shot learning. In particular, they suffer from majority label bias, recency bias, and common token bias (Section 4). The majority label and recency biases lead the model to predict training answers that appear frequently or near the end of the prompt. For example, a prompt that ends with a Negative training example may cause a bias towards the Negative class. On the other hand, the common token bias leads the model to prefer answers that are frequent in its pre-training data, e.g.,

[*]Equal contribution [1]UC Berkeley [2]University of Maryland [3]UC Irvine. Correspondence to: Eric Wallace <ericwallace@berkeley.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

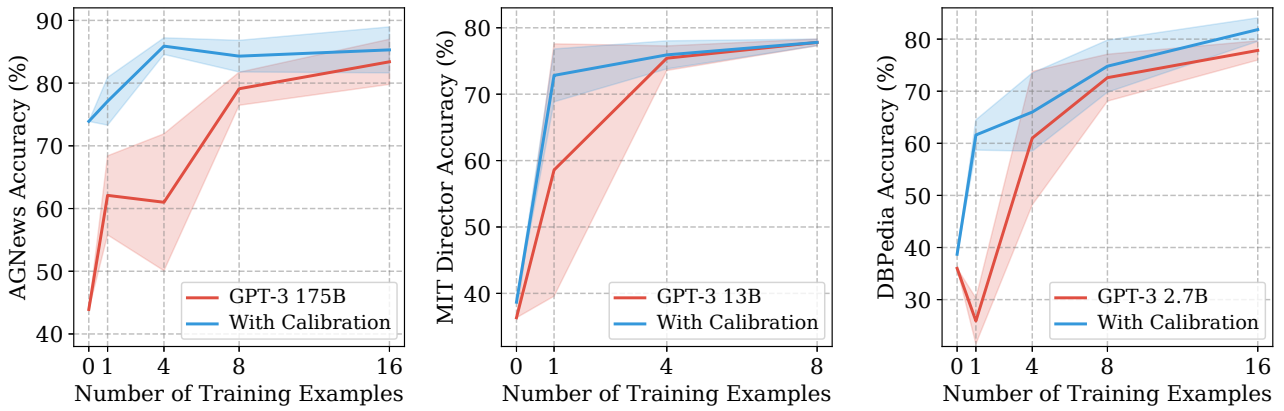

Figure 1. Few-shot learning can be highly unstable across different choices of the prompt. Above, we plot the mean accuracy (± one standard deviation) across different choices of the training examples for three different datasets and model sizes. We show that our method, *contextual calibration*, improves accuracy, reduces variance, and overall makes tools like GPT-3 more effective for end users.

it prefers "United States" over "Saint Lucia", which is likely suboptimal for the task of interest.

We identify that these biases typically result in a shift in the output distribution of the model. We can thus counteract these biases by "calibrating" the output distribution. Concretely, we estimate the model's bias towards certain answers by feeding in a dummy test input that is *content-free*. In the prompt above for example, if we replace "Amazing." with the string "N/A", the model predicts 62% Positive. We then fit the calibration parameters so that the content-free input has uniform scores for each answer. This *contextual calibration* procedure provides a good setting of the calibration parameters without additional training data.

We test the effectiveness of contextual calibration on a range of tasks (Section 5). Contextual calibration consistently improves GPT-3 and GPT-2's accuracy (up to 30.0% absolute) across different choices of the prompt format and examples (e.g., Figure 1). It also makes the accuracy more stable across different prompts, thus mitigating the need for prompt engineering. Overall, contextual calibration is a simple method that makes language models better few-shot learners: it enables end users to obtain higher accuracy with considerably less effort.

2. Background and Experimental Setup

Neural autoregressive language models (LMs) take as input a sequence of tokens and output a probability distribution over the next token. Large neural LMs can perform tasks in a zero- or few-shot manner using in-context learning (Radford et al., 2019; Brown et al., 2020). To do so, a natural language *prompt* is fed into the model. This prompt contains three components: a format, a set of training examples, and a permutation (ordering) of the training examples.

Prompt Format The prompt *format* is a template which consists of placeholders for the training and test example(s) and possibly a natural language description of the task. For example, the format of the prompt in Section 1 is a template with the style: "Input:" input "Sentiment:" label. Many alternate formats exist, e.g., one could frame the task as question answering.

Prompt Training Examples The prompt's *training examples* are used to teach the LM how to solve the task at hand. The prompt from Section 1 consists of two training examples; we refer to this as "two-shot" learning. We also consider "zero-shot" learning, where no training examples are present.

Training Example Permutation When training examples are used, they have a particular *permutation*, e.g., the "Subpar acting" example comes first in the prompt from Section 1. The permutation matters because neural language models update their hidden states in a left-to-right-fashion.

To make predictions on an input, we slot it into the test placeholder and generate from the LM. For example, see the "Amazing." test example in the prompt from Section 1. For generation tasks, we generate greedily from the LM until it produces a newline character. For classification tasks, the probability for each class is given by the probability assigned to its associated *label name*, e.g., the words "Negative" and "Positive" for sentiment classification.

2.1. Datasets and Prompt Formats

We use datasets for three tasks: text classification, fact retrieval, and information extraction. We use a fixed prompt format for each dataset unless otherwise specified. We show the format and examples from each dataset in Appendix B.

Text Classification We study text classification using six datasets: sentiment analysis using **SST-2** (Socher et al.,

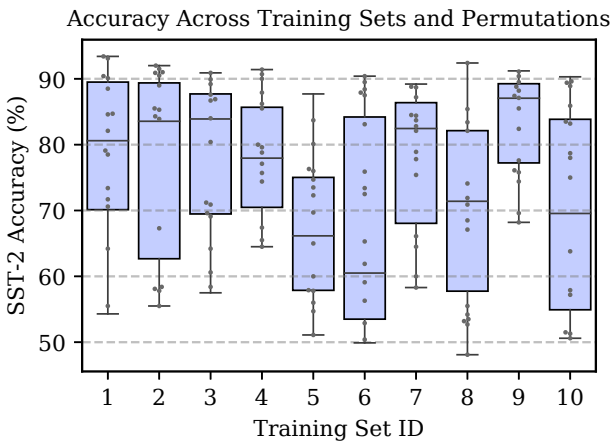

Figure 2. There is high variance in GPT-3's accuracy as we change the prompt's **training examples**, as well as the **permutation** of the examples. Here, we select ten different sets of four SST-2 training examples. For each set of examples, we vary their permutation and plot GPT-3 2.7B's accuracy for each permutation (and its quartiles).

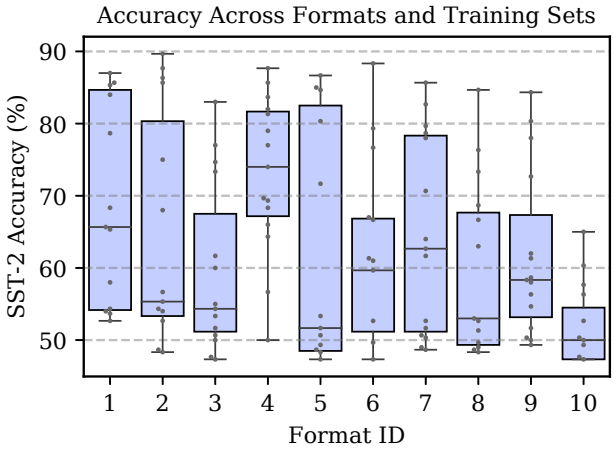

Figure 3. There is high variance in GPT-3's accuracy as we change the **prompt format**. In this figure, we use ten different prompt formats for SST-2. For each format, we plot GPT-3 2.7B's accuracy for different sets of four training examples, along with the quartiles.

2013), 6-way question classification using **TREC** (Voorhees & Tice, 2000), textual entailment using 3-way **CB** (de Marneffe et al., 2019) and binary **RTE** (Dagan et al., 2005) from SuperGLUE (Wang et al., 2019), and topic classification using the 4-way **AGNews** (Zhang et al., 2015) and 14-way **DBPedia** (Zhang et al., 2015) datasets. The prompt in Section 1 shows an example of the sentiment analysis task.

Fact Retrieval We evaluate fact retrieval with **LAMA** (Petroni et al., 2019). The dataset consists of knowledge base triples that are placed into templates with missing objects, e.g. "Obama was born in". We use these templates as our prompts, and remove the relations where the missing answer is not at the end of the template (left-to-right LMs cannot solve these). The answers are always single tokens, and we report average accuracy across all triples.

Information Extraction We consider information extraction using two slot filling datasets, **ATIS** (Hemphill et al., 1990) and **MIT Movies** trivia10k13 (Liu et al., 2012). We use two random slots for each dataset, *airline* and *departure date* for ATIS, and *director name* and *movie genre* for MIT Movies. The answer for both datasets is a span of text from the input, e.g., the ATIS airline task is to predict "american airlines" when given the sentence "list a flight on american airlines from toronto to san diego". We use Exact Match between the model's generated output and the ground-truth span as our evaluation metric.

2.2. Model Details

We run our experiments on three sizes of GPT-3 (2.7B, 13B, and 175B parameters) as well as GPT-2 (1.5B parameters). We access GPT-3 using the OpenAI API. We release code to replicate our experiments.[1]

3. Accuracy Varies Highly Across Prompts

This section studies how GPT-3's accuracy changes as we vary each aspect of the prompt (training examples, permutation, format). We focus on a subset of the datasets to simplify our analysis; in Section 5 we show that our findings hold across all of the datasets we study.

GPT-3's accuracy depends highly on both selection and permutation of training examples. Concretely, we use a fixed prompt format and choose different random sets of training examples. For each set of training examples, we evaluate the accuracy for all possible permutations.

Figure 2 shows the results for SST-2 (4-shot, GPT-3 2.7B). Surprisingly, varying the permutation can be as important, or even more important, than which training examples are chosen. For example, varying the permutation of the training examples can cause accuracy to go from near chance (54.3%) to near state-of-the-art (93.4%). For a qualitative example of the sensitivity to permutations, see Table 2 in Appendix A. This high importance on example order is in contrast to standard machine learning, where the ordering of examples during training is typically an afterthought.

The variance persists with more data and larger models. Adding more training examples into the prompt does not necessarily reduce the variance in accuracy. We sweep over the number of training examples for three different datasets in Figure 1 (red curves). The variance remains high even when we use 16 training examples. Moreover, adding more

[1] https://www.github.com/tonyzhaozh/few-shot-learning

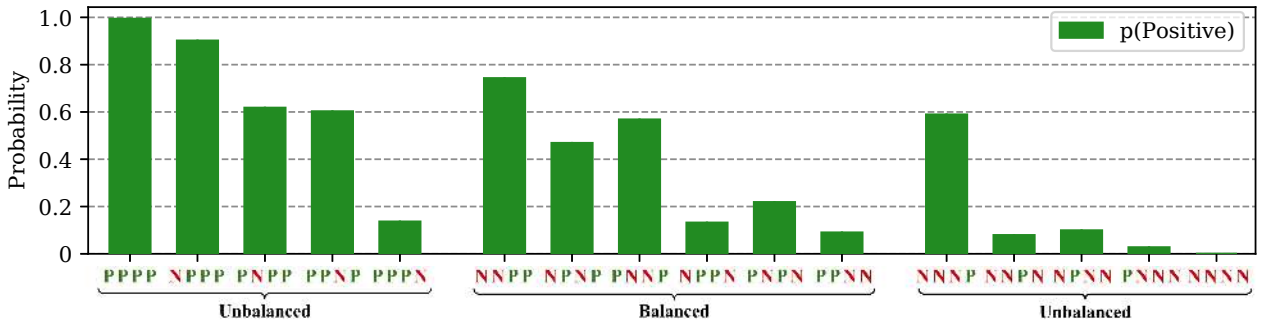

Figure 4. **Majority label and recency biases** cause GPT-3 to become biased towards certain answers and help to explain the high variance across different examples and orderings. Above, we use 4-shot SST-2 with prompts that have different class balances and permutations, e.g., [P P N N] indicates two positive training examples and then two negative. We plot how often GPT-3 2.7B predicts Positive on the balanced validation set. When the prompt is unbalanced, the predictions are unbalanced (*majority label bias*). In addition, balanced prompts that have one class repeated near the end, e.g., end with two Negative examples, will have a bias towards that class (*recency bias*).

training examples can sometimes hurt accuracy (e.g., mean accuracy drops from 36.0% to 25.9% for DBPedia 0-shot to 1-shot). The variance in accuracy can also remain high when using larger models, e.g., the left of Figure 1.

GPT-3's accuracy depends highly on prompt format. We next keep the set of training examples and permutations fixed but vary the prompt format. We focus on SST-2, and we manually design an additional 14 prompt formats. The formats include question-answer templates, conversation-style templates, prompts that resemble Web pages, and variations on the label names (all formats available in Table 7 in Appendix B). The accuracy for ten of the formats is shown in Figure 3. We find that some of the formats are better than others on average. However, all of the formats still suffer from high variance across different training sets.

4. What Causes the High Variance?

We next analyze *why* GPT-3's accuracy varies across different training examples, permutations, and prompt formats. Concretely, we show that the variance arises because LMs are biased towards outputting answers that are (1) frequent in the prompt (majority label bias), (2) towards the end of the prompt (recency bias), and (3) common in the pre-training data (common token bias).

Majority Label Bias We find that GPT-3 is biased towards answers that are frequent in the prompt. A trivial case is when a text classification prompt has a class imbalance, e.g., more Positive than Negative sentiment examples. This is demonstrated in the "unbalanced" region of Figure 4: when one class is more common, GPT-3 2.7B is heavily biased towards predicting that class. Since the SST-2 sentiment analysis dataset is balanced, this bias causes large accuracy degradations. The majority label bias also explains why we frequently observe a drop in accuracy when moving from 0-shot to 1-shot—we found that the drop is due to the model frequently repeating the class of the one training example.

The majority label bias also occurs for generation tasks. On the validation set for 4-shot LAMA with GPT-3 2.7B, 50.2% of the model predictions are a repeat of one of the four training answers (the correct repeat rate is 24.7%). Overall, the majority label bias helps to explain why different choices for the training examples heavily influence GPT-3's accuracy—it shifts the distribution of model predictions.

Recency Bias The model's majority label bias is aggravated by its *recency bias*: the tendency to repeat answers that appear towards the end of the prompt. The "balanced" region of Figure 4 demonstrates this. For instance, when two Negative examples appear at the end (P P N N), the model will heavily prefer the Negative class. Moreover, the recency bias can outweigh the majority label bias, e.g., the "P P P N" training set leads to nearly 90% of predictions being Negative, despite $\frac{3}{4}$ of the training examples being Positive.

Recency bias also affects generation tasks. For 4-shot LAMA, the training answers that are closer to the end of the prompt are more likely to be repeated by the model. Concretely, the model "overpredicts" the answer from the 1st, 2nd, 3rd, and 4th training example by 8.5%, 8.3%, 14.3%, and 16.1%, respectively.[2] Overall, recency bias helps to explain why the *permutation* of the training examples is important—the ordering of the examples heavily influences the distribution of the model predictions.

Common Token Bias Finally, we find that GPT-3 is biased towards outputting tokens that are common in its *pre-training* distribution, which is likely suboptimal for the distribution of answers on the *downstream* task. A simple case

[2]Over all relations, as well as three different sets of training examples, the model repeats the training example at a rate of 20.7%, 19.8%, 29.9%, and 26.8%. The ground-truth repeat rate is 12.2%, 11.5%, 15.6%, and 10.7%. We define "overpredicts" as the model's repeat rate minus the ground-truth repeat rate.

of this occurs for the LAMA fact retrieval dataset, where the model often predicts common entities such as "America" when the ground-truth answer is instead a rare entity.

A more nuanced case of the common token bias occurs for text classification. Recall that the model makes predictions by generating the label name associated with each class. Because certain label names appear more frequently in the pre-training data, the model will be inherently biased towards predicting certain classes. For example, on DBPedia (a balanced 14-way topic classification dataset), GPT-3 predicts the "book" class $11\times$ more often than the "artist" class. In fact, there is a moderate correlation ($r = 0.67$) between the frequency of a DBPedia label name and the rate at which GPT-3 predicts its class.[3] Overall, the common token bias helps to explain why the choice of label names is important, and why the model struggles on rare answers.

The Impact of Biases on Model Predictions We find that the end result of the above three biases is typically a simple shift in the model's output distribution. For example, Figure 5 visualizes this shift for a SST-2 sentiment prompt.

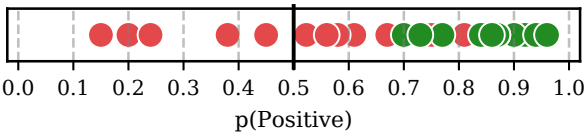

Figure 5. The Positive class probability for 25 random test inputs for a particular sentiment analysis prompt. Negative ground-truth examples are marked with ● and Positive are marked with ●.

The prompt used in Figure 5 and the model's intrinsic biases cause it to frequently predict high confidence for the Positive class. Since the default 50% threshold is used to make predictions, this results in frequent false positives. Importantly, note that if we could optimally set the classification threshold (p(Positive) = 0.68 in this case), the classifier would be highly accurate (94% on the validation set).

5. Contextual Calibration

Thus far, we have shown that GPT-3 is biased towards certain answers due to the prompt and the model's intrinsic biases. Here, we look to correct this by "calibrating" the model's output probabilities.[4] A common technique for adjusting output probabilities is to apply an affine transformation (Platt, 1999; Guo et al., 2017):

$$\hat{\mathbf{q}} = \text{softmax}(\mathbf{W}\hat{\mathbf{p}} + \mathbf{b}), \tag{1}$$

where a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}$ are applied to the original probabilities $\hat{\mathbf{p}}$ to get the new probabilities $\hat{\mathbf{q}}$.[5] For classification tasks, $\hat{\mathbf{p}}$ is the set of probabilities that are associated with each label name, renormalized to one. For generation tasks, $\hat{\mathbf{p}}$ is the entire set of probabilities for the first token.[6] In this paper, we restrict the matrix $\mathbf{W}$ to be diagonal, known as vector scaling (Guo et al., 2017), to prevent the parameters from growing quadratically in the size of $\hat{\mathbf{p}}$ (which is $\approx 50,000$ for generation tasks).

The main challenge in the zero- or few-shot setting is that we do not have data to learn $\mathbf{W}$ and $\mathbf{b}$. We thus propose a novel data-free procedure to infer a good setting of these parameters. The key idea is that the model's bias towards certain answers can be estimated by feeding in a *content-free* input such as the string "N/A". For example, consider the two-shot prompt:

```
Input: Subpar acting. Sentiment: Negative
Input: Beautiful film. Sentiment: Positive
Input: N/A            Sentiment:
```

where "N/A" serves as the test input. Ideally, GPT-3 would score this test input as 50% Positive and 50% Negative. However, the model's biases cause it to score this input as 61.8% Positive. Note that this error is *contextual*: a different choice of the training examples, permutation, and format will lead to different predictions for the content-free input.

We can correct this error by setting $\mathbf{W}$ and $\mathbf{b}$ so that the class scores for the content-free input are uniform. We first obtain $\hat{\mathbf{p}}$ for the content-free input, denoted $\hat{\mathbf{p}}_{\text{cf}}$. We then set $\mathbf{W} = \text{diag}(\hat{\mathbf{p}}_{\text{cf}})^{-1}$ and $\mathbf{b}$ to the all-zero vector.[7] To make test predictions, we compute $\mathbf{W}\hat{\mathbf{p}} + \mathbf{b}$ and take the argmax.

Implementation Details This *contextual calibration* procedure adds trivial amounts of computational overhead and is implemented in a few lines of code (compute and save $\hat{\mathbf{p}}_{\text{cf}}$, adjust output probabilities). For the content-free input, many good choices exist, including "N/A", the empty string, and gibberish tokens. In all our experiments, we average the probabilities from three content-free inputs: "N/A", "[MASK]", and the empty string.[8] One could also craft the

[3]The frequency of a token on the web is calculated using Google Ngrams https://books.google.com/ngrams. The predictions are from the 0-shot setting on the validation set.

[4]The output of GPT-3 is biased (its outputs are shifted), similar to how measurement devices such as voltage meters or weighing scales are biased. Just like how these devices require "calibration before use", where the devices' outputs are scaled/zeroed-out, we hope to apply a similar calibration procedure to LMs. This goal is distinct from statistical calibration (Brier, 1950; Guo et al., 2017), i.e., aligning a model's confidence estimate with its true accuracy.

[5]This affine transformation is usually applied to the logits, i.e., prior to the softmax. However, we only have access to GPT-3's output probabilities in the OpenAI API.

[6]We only calibrate the prediction of the first output token for generation tasks. This is reasonable because, for the tasks we consider, we found that the model's predictions are highly deterministic after generating the first token.

[7]An alternate solution is to set $\mathbf{b}$ to $-\hat{\mathbf{p}}_{\text{cf}}$ and $\mathbf{W}$ to the identity. Empirically, this alternate solution yields higher accuracy for generation tasks (where the dimensionality of $\hat{\mathbf{p}}$ is large). The solution in the main text performs better for classification.

[8]We found this simple ensemble to achieve the best results for

content-free input in a task-specific manner. We explore this for LAMA, where we replace the subject with the content-free input, e.g., we use "N/A was born in" as the input.

5.1. Results for Contextual Calibration

Here, we evaluate the effectiveness of contextual calibration across all of our datasets and LMs. We first use a fixed prompt format and select five different random sets of training examples, placing them in an arbitrary order in the prompt. We do not artificially balance the labels of the training examples for the classification tasks. We use the same sets of training examples for the baseline (standard decoding without calibration) and contextual calibration. We use labeling budgets of 0–8 examples; using more than 8-shots causes the cost of querying the OpenAI API to become prohibitively expensive.

Table 1 shows the results and Figure 1 in Section 1 plots the same data for a subset of the tasks.

Improves Mean And Worst-Case Accuracy Contextual calibration dramatically improves GPT-3's average and worst-case accuracy, by up to 30.0% absolute. These gains hold for both classification and generation tasks. Contextual calibration also sometimes allows GPT-3 2.7B to outperform the GPT-3 175B baseline—by up to 19.3%—despite being over 50x smaller.

Can Reduce Variance Across Training Sets Figure 6 plots the difference in the standard deviation between the baseline and contextual calibration for all tasks from Table 1. Contextual calibration reduces the variance considerably in a majority of cases, and it does not increase variance by much in the remaining cases.

Reduces Drop from 0-shot to 1-shot For the baseline, there are four cases where there is a drop in accuracy when moving from 0-shot to 1-shot (TREC, AGNews, DBpedia, SST-2). We attribute this drop to the majority label bias (see discussion in Section 4). Calibration removes this drop in three out of four cases.

Improves GPT-2 We also test GPT-2 1.5B (see Table 4 in Appendix A). We find that like GPT-3, GPT-2's accuracy also highly varies across different prompts. This suggests that the variance that we observe for few-shot in-context learning is a general problem for LMs. Second, contextual calibration works out-of-the-box for GPT-2—it improves the mean accuracy and reduces variance for most tasks.

Improves Accuracy Across Formats In our next set of experiments, we use a fixed set of training examples and vary the prompt format. We use the 15 prompt formats for SST-2 discussed in Section 3. We also create 15 prompt formats

AGNews, and we reuse it for all other datasets. See Section 5.2 for an ablation on the choice of content-free input.

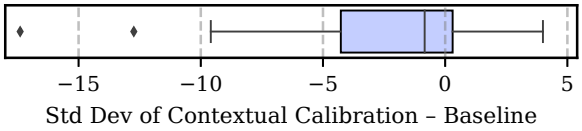

Figure 6. Aside from improving mean accuracy, contextual calibration also reduces the standard deviation of accuracy across different choices of the training examples. We plot the difference in standard deviation between contextual calibration and the baseline from Table 1.

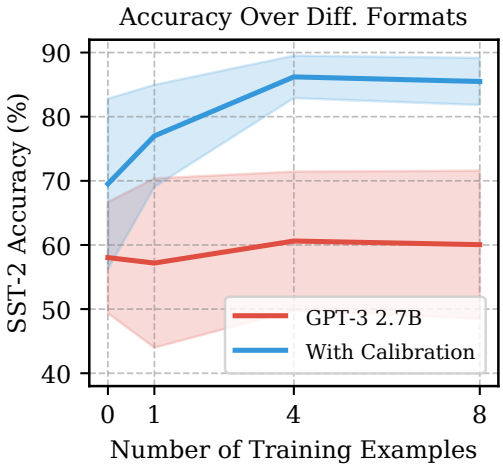

Figure 7. GPT-3 has high variance across different prompt formats; contextual calibration reduces this variance and improves mean accuracy. We show the mean accuracy (± standard deviation) over 15 different prompt formats for SST-2.

for each of three random relations in LAMA (P20, P159, P19) by using the paraphrases of the original LAMA templates generated by Jiang et al. (2020). Figure 7 shows the results before and after calibration for SST-2, and Figure 9 in Appendix A show the results for LAMA. Contextual calibration improves the average and worst-case accuracy for both datasets, and reduces the variance for SST-2.

5.2. Ablations on Contextual Calibration

We finally conduct two analyses/ablations on contextual calibration. We first analyze how effective contextual calibration is at inferring a good setting of $\mathbf{W}$. To do so, we compare its accuracy to an "oracle calibration" method that uses the validation set to find the best possible diagonal $\mathbf{W}$. We evaluate this oracle on AGNews, and find that contextual calibration is surprisingly close to it (Figure 8).

We also study how the choice of content-free input affects accuracy. In Table 3 in Appendix A, we show the accuracy for SST-2 and AGNews for different choices of the content-free input. The choice of content-free input matters, however, many good choices exist.

Dataset	LM	0-shot		1-shot		4-shot		8-shot	
		Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
Text Classification									
AGNews	2.7B	$44.7_{0.0}$	$\mathbf{63.2}_{0.0}$	$33.0_{5.1}$	$\mathbf{59.6}_{6.4}$	$43.3_{8.3}$	$\mathbf{71.1}_{8.5}$	$50.8_{7.8}$	$\mathbf{72.7}_{5.8}$
	175B	$43.9_{0.0}$	$\mathbf{73.9}_{0.0}$	$62.1_{6.3}$	$\mathbf{77.1}_{3.8}$	$61.0_{10.9}$	$\mathbf{85.9}_{1.3}$	$79.1_{2.6}$	$\mathbf{84.3}_{2.5}$
TREC	2.7B	$31.0_{0.0}$	$\mathbf{38.8}_{0.0}$	$24.3_{6.4}$	$\mathbf{36.8}_{7.7}$	$25.8_{11.5}$	$\mathbf{38.6}_{13.2}$	$29.3_{8.0}$	$\mathbf{44.3}_{11.4}$
	175B	$47.4_{0.0}$	$\mathbf{57.4}_{0.0}$	$57.7_{6.0}$	$\mathbf{75.7}_{1.4}$	$60.2_{7.6}$	$\mathbf{69.7}_{1.4}$	$45.6_{4.0}$	$\mathbf{66.9}_{6.5}$
CB	2.7B	$44.6_{0.0}$	$\mathbf{50.0}_{0.0}$	$\mathbf{33.8}_{16.6}$	$33.0_{7.3}$	$43.5_{11.9}$	$\mathbf{54.2}_{4.7}$	$43.9_{8.4}$	$\mathbf{53.0}_{7.7}$
	175B	$30.4_{0.0}$	$\mathbf{48.2}_{0.0}$	$50.9_{6.7}$	$\mathbf{51.8}_{7.2}$	$45.2_{19.4}$	$\mathbf{60.7}_{6.7}$	$59.6_{11.3}$	$\mathbf{65.0}_{7.9}$
RTE	2.7B	$44.8_{0.0}$	$\mathbf{49.5}_{0.0}$	$49.6_{2.9}$	$\mathbf{50.4}_{2.7}$	$44.0_{1.4}$	$\mathbf{54.5}_{4.7}$	$49.2_{1.9}$	$\mathbf{54.8}_{2.8}$
	175B	$\mathbf{57.8}_{0.0}$	$\mathbf{57.8}_{0.0}$	$\mathbf{62.9}_{2.7}$	$62.8_{2.3}$	$58.7_{11.9}$	$\mathbf{60.4}_{8.1}$	$\mathbf{66.2}_{5.8}$	$65.5_{2.5}$
SST-2	2.7B	$57.2_{0.0}$	$\mathbf{71.4}_{0.0}$	$67.3_{7.9}$	$\mathbf{79.1}_{8.3}$	$59.1_{10.2}$	$\mathbf{79.9}_{7.8}$	$54.0_{4.3}$	$\mathbf{82.0}_{5.5}$
	175B	$71.6_{0.0}$	$\mathbf{75.8}_{0.0}$	$93.3_{2.8}$	$\mathbf{94.7}_{1.4}$	$93.6_{3.3}$	$\mathbf{94.3}_{1.0}$	$\mathbf{95.6}_{1.0}$	$95.3_{0.7}$
DBPedia	2.7B	$36.0_{0.0}$	$\mathbf{38.7}_{0.0}$	$25.9_{4.4}$	$\mathbf{61.6}_{2.9}$	$61.0_{12.8}$	$\mathbf{66.0}_{7.5}$	$72.6_{4.5}$	$\mathbf{74.8}_{5.0}$
	175B	$22.0_{0.0}$	$\mathbf{59.7}_{0.0}$	$79.3_{3.0}$	$\mathbf{85.3}_{2.2}$	$84.6_{5.8}$	$\mathbf{86.9}_{4.0}$	$82.3_{7.8}$	$\mathbf{86.9}_{1.9}$
Fact Retrieval									
LAMA	2.7B	$14.0_{0.0}$	$\mathbf{22.7}_{0.0}$	$29.7_{1.8}$	$\mathbf{31.6}_{1.3}$	$35.8_{3.8}$	$\mathbf{37.4}_{3.4}$	$\mathbf{42.5}_{1.3}$	$\mathbf{42.5}_{1.4}$
	175B	$23.5_{0.0}$	$\mathbf{30.1}_{0.0}$	$48.9_{2.3}$	$\mathbf{49.0}_{1.4}$	$\mathbf{62.0}_{2.4}$	$61.8_{2.9}$	$\mathbf{63.8}_{1.0}$	$63.6_{1.3}$
Information Extraction									
MIT-G	2.7B	$5.0_{0.0}$	$\mathbf{5.7}_{0.0}$	$26.7_{11.4}$	$\mathbf{37.9}_{5.7}$	$53.1_{7.8}$	$\mathbf{54.7}_{6.0}$	$59.0_{4.7}$	$\mathbf{59.1}_{4.8}$
	13B	$15.0_{0.0}$	$\mathbf{18.7}_{0.0}$	$47.3_{3.9}$	$\mathbf{52.0}_{7.9}$	$57.9_{4.8}$	$\mathbf{58.9}_{4.0}$	$59.0_{4.7}$	$\mathbf{59.1}_{4.8}$
MIT-D	2.7B	$46.3_{0.0}$	$\mathbf{47.0}_{0.0}$	$42.0_{13.0}$	$\mathbf{53.5}_{13.5}$	$73.5_{4.9}$	$\mathbf{74.1}_{5.0}$	$\mathbf{75.3}_{1.0}$	$75.1_{1.3}$
	13B	$36.3_{0.0}$	$\mathbf{38.7}_{0.0}$	$58.6_{21.4}$	$\mathbf{72.8}_{4.0}$	$75.4_{1.9}$	$\mathbf{75.9}_{2.1}$	$\mathbf{77.8}_{0.5}$	$\mathbf{77.8}_{0.5}$
ATIS-A	2.7B	$10.8_{0.0}$	$\mathbf{14.0}_{0.0}$	$29.8_{12.8}$	$\mathbf{33.1}_{9.4}$	$43.0_{26.2}$	$\mathbf{47.3}_{21.3}$	$55.6_{5.0}$	$\mathbf{58.8}_{4.0}$
	13B	$49.5_{0.0}$	$\mathbf{52.7}_{0.0}$	$69.6_{17.4}$	$\mathbf{71.8}_{17.1}$	$67.5_{10.4}$	$\mathbf{69.6}_{13.4}$	$63.4_{4.6}$	$\mathbf{64.5}_{4.0}$
ATIS-D	2.7B	$6.4_{0.0}$	$\mathbf{12.9}_{0.0}$	$42.3_{28.8}$	$\mathbf{65.6}_{20.8}$	$75.0_{6.7}$	$\mathbf{83.4}_{4.2}$	$81.0_{8.8}$	$\mathbf{88.3}_{3.7}$
	13B	$4.0_{0.0}$	$\mathbf{5.0}_{0.0}$	$\mathbf{97.9}_{0.6}$	$95.5_{4.6}$	$\mathbf{98.0}_{0.6}$	$97.8_{0.7}$	$\mathbf{98.8}_{0.3}$	$\mathbf{98.8}_{0.3}$

Table 1. **Contextual calibration improves accuracy across a range of tasks.** We show the mean and standard deviation across different choices of the training examples (the prompt format is fixed). The LM column indicates the GPT-3 size (see Appendix A for GPT-2 results). The Baseline column shows the standard approach of greedy decoding (Brown et al., 2020) and *Ours* corresponds to greedy decoding after modifying the output probabilities using contextual calibration. We bold the better result of the baseline and ours. MIT-G, MIT-D, ATIS-A, and ATIS-D indicate the MIT Genre, MIT Director, ATIS Airline, and ATIS Departure Date datasets.

6. Discussion

Does Calibration Eliminate the Need to Engineer Prompts? The motivation behind "prompt engineering" is that not all prompts lead to the same accuracy. Thus, one should tune the prompt's format and examples to achieve the best possible performance (Brown et al., 2020; Gao et al., 2021). Contextual calibration does not eliminate the need to engineer prompts, however, it does mitigate it: contextual calibration makes the accuracy of the best, average, and worst-case prompts more similar (and higher).

Should You Finetune in the Few-shot Setting? We use a fixed LM with no finetuning. As mentioned in Section 1, there are numerous reasons not to finetune: it enables rapid prototyping, provides a fully natural language interface, and is more efficient in terms of memory requirements and system complexity when serving many different tasks. Moreover, like in-context learning without contextual calibration, finetuning can be unstable in the few-shot setting (Schick & Schütze, 2021a). Nevertheless, if these disadvantages are acceptable or avoidable, finetuning can improve accuracy over in-context learning in some cases (Schick & Schütze, 2021b; Gao et al., 2021). An interesting direction for future work is to study the interplay between contextual calibration and finetuning, e.g., does contextual calibration alleviate the need to finetune, or vice versa?

7. Related Work

Few-shot Learning with Language Models Recent work uses LMs to solve NLP tasks, e.g., for story cloze prediction (Schwartz et al., 2017), knowledge base comple-

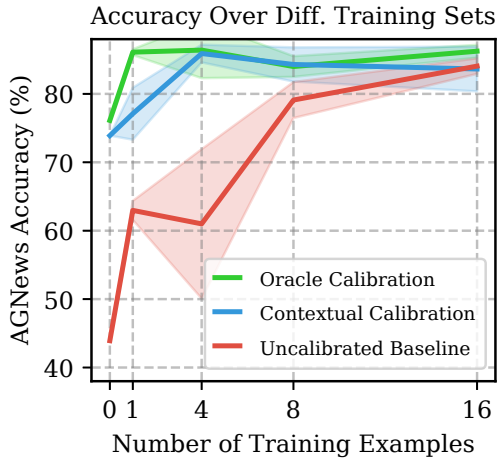

Figure 8. Contextual calibration, despite using no training data, achieves similar accuracy to an "oracle" calibration that finds the best $\mathbf{W}$ using the validation set. The plot shows GPT-3 175B's mean accuracy ($\pm$ standard deviation) on AGNews over different choices of the training examples.

tion (Petroni et al., 2019), and Winograd schemas (Trinh & Le, 2018). Radford et al. (2019) and Brown et al. (2020) show that large LMs can be used to solve a myriad of tasks in a few-shot manner via in-context learning. Our paper provides a simple modification to their setting that improves performance. Asking LMs to complete natural language prompts is also used as a method to "probe" LMs, e.g., analyzing their factual (Petroni et al., 2019; Jiang et al., 2020; Shin et al., 2020) or commonsense knowledge (Bosselut et al., 2019). Our results suggest that these probing methods may underestimate model accuracy, and we recommend that future work take advantage of contextual calibration.

Volatility of Few-shot Learning in NLP Recent work shows that when using masked language models such as BERT for zero-shot learning, the prompt format can impact accuracy (Petroni et al., 2019; Jiang et al., 2020; Shin et al., 2020). Independent and concurrent work also shows that when finetuning masked language models on few examples, the choice of training examples can impact results (Schick & Schütze, 2021b; Gao et al., 2021). We show that similar instabilities occur for in-context learning (i.e., no finetuning) with left-to-right language models. We also show a surprising instability associated with example ordering. Moreover, unlike past work, we analyze why these instabilities occur, and we use insights from this analysis to mitigate the issues.

Failures of Language Models We identify failures when LMs are used for in-context learning (e.g., recency bias). Past work identifies similar failures when LMs are used for text generation. For example, neural LMs often repeat themselves (Holtzman et al., 2020), suffer from overconfidence (Braverman et al., 2020; Jiang et al., 2021), suffer from recency bias (Khandelwal et al., 2018; Ravfogel et al., 2019), and prefer generic responses instead of rare text (Li et al., 2016; Logan et al., 2019). Past work mitigates these degeneracies by modifying the model's output probabilities or generation schemes, e.g., explicitly preventing repetitions (Paulus et al., 2018) or using sampling instead of greedy decoding (Holtzman et al., 2020).

8. Conclusion and Future Work

We show that few-shot learning can be highly volatile across different choices of the prompt. Through a detailed analysis, we identify that this volatility arises from biases in LMs, e.g., their tendency to output recent or common tokens. We use these insights to develop contextual calibration—a simple procedure to adjust the model's output probabilities—which improves accuracy, reduces variance, and overall makes tools like GPT-3 more effective for end users.

Looking at the bigger picture, our results inspire two future research directions in few-shot learning for NLP. First, on the methods side, we show that good few-shot learning requires *attention to detail*: small but non-trivial decisions such as calibration can greatly influence results. This makes it difficult to correctly develop and compare new methods (e.g., pretraining schemes or model architectures). We thus hope to make other few-shot learning methods more robust, and also expand our techniques to cover a wider ranger of tasks (e.g., calibration for open-ended generation). Second, on the analysis side, our results highlight the need to understand *what* GPT-3 learns from the prompt. The model has an impressive ability to improve with more training examples, however, we show that the model learns some superficial patterns such as repetition of common answers. We hope to better understand and analyze the dynamics of in-context learning in future work.

Acknowledgements

We thank OpenAI for providing academic access to the GPT-3 API. We thank Sewon Min, Nikhil Kandpal, Nelson Liu, Girish Sastry, Marco Tulio Ribeiro, and the members of Berkeley NLP for valuable feedback on the paper.

This work was supported by DARPA under the LwLL program/Grant No. FA8750-19-1-0504, DARPA MCS program under Contract No. N660011924033 with the United States Office Of Naval Research, DARPA and the Air Force Research Laboratory (AFRL), and NSF award #IIS-1756023.

References

Bosselut, A., Rashkin, H., Sap, M., Malaviya, C., Celikyilmaz, A., and Choi, Y. COMET: Commonsense transformers for automatic knowledge graph construction. In *ACL*, 2019.

Braverman, M., Chen, X., Kakade, S., Narasimhan, K., Zhang, C., and Zhang, Y. Calibration, entropy rates, and memory in language models. In *ICML*, 2020.

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 1950.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *NeurIPS*, 2020.

Dagan, I., Glickman, O., and Magnini, B. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, 2005.

de Marneffe, M.-C., Simons, M., and Tonhauser, J. The CommitmentBank: Investigating projection in naturally occurring discourse. In *Sinn und Bedeutung*, 2019.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

Gao, T., Fisch, A., and Chen, D. Making pre-trained language models better few-shot learners. In *ACL*, 2021.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *ICML*, 2017.

Hemphill, C. T., Godfrey, J. J., and Doddington, G. R. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language Workshop*, 1990.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration. In *ICLR*, 2020.

Jiang, Z., Xu, F. F., Araki, J., and Neubig, G. How can we know what language models know? In *TACL*, 2020.

Jiang, Z., Araki, J., Ding, H., and Neubig, G. How can we know when language models know? In *TACL*, 2021.

Khandelwal, U., He, H., Qi, P., and Jurafsky, D. Sharp nearby, fuzzy far away: How neural language models use context. In *ACL*, 2018.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. In *Science*, 2015.

Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. A diversity-promoting objective function for neural conversation models. In *NAACL*, 2016.

Liu, J., Cyphers, S., Pasupat, P., McGraw, I., and Glass, J. A conversational movie search system based on conditional random fields. In *INTERSPEECH*, 2012.

Logan, R. L., Liu, N. F., Peters, M. E., Gardner, M., and Singh, S. Barack's wife Hillary: Using knowledge-graphs for fact-aware language modeling. In *ACL*, 2019.

Paulus, R., Xiong, C., and Socher, R. A deep reinforced model for abstractive summarization. In *ICLR*, 2018.

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., and Riedel, S. Language models as knowledge bases? In *EMNLP*, 2019.

Platt, J. C. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, 1999.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. *Technical Report*, 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *Technical Report*, 2019.

Ravfogel, S., Goldberg, Y., and Linzen, T. Studying the inductive biases of RNNs with synthetic variations of natural languages. In *NAACL*, 2019.

Schick, T. and Schütze, H. Exploiting cloze questions for few-shot text classification and natural language inference. In *EACL*, 2021a.

Schick, T. and Schütze, H. It's not just size that matters: Small language models are also few-shot learners. In *NAACL*, 2021b.

Schwartz, R., Sap, M., Konstas, I., Zilles, L., Choi, Y., and Smith, N. A. The effect of different writing tasks on linguistic style: A case study of the ROC story cloze task. In *ACL*, 2017.

Shin, T., Razeghi, Y., Logan IV, R. L., Wallace, E., and Singh, S. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*, 2020.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.

Trinh, T. H. and Le, Q. V. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*, 2018.

Voorhees, E. M. and Tice, D. M. Building a question answering test collection. In *SIGIR*, 2000.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*, 2019.

Yogatama, D., d'Autume, C. d. M., Connor, J., Kocisky, T., Chrzanowski, M., Kong, L., Lazaridou, A., Ling, W., Yu, L., Dyer, C., et al. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373*, 2019.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *NeurIPS*, 2015.