How Can We Know When Language Models Know?
On the Calibration of Language Models for Question Answering

Zhengbao Jiang†, Jun Araki‡, Haibo Ding‡, Graham Neubig†
†Languages Technologies Institute, Carnegie Mellon University, United States
‡Bosch Research, United States
{zhengbaj,gneubig}@cs.cmu.edu
{jun.araki,haibo.ding}@us.bosch.com

Abstract
Recent works have shown that language models (LM) capture different types of knowledge regarding facts or common sense. However, because no model is perfect, they still fail to provide appropriate answers in many cases. In this paper, we ask the question, “How can we know when language models know, with confidence, the answer to a particular query?” We examine this question from the point of view of calibration, the property of a probabilistic model’s predicted probabilities actually being well correlated with the probabilities of correctness. We examine three strong generative models—T5, BART, and GPT-2—and study whether their probabilities on QA tasks are well calibrated, finding the answer is a relatively emphatic no. We then examine methods to calibrate such models to make their confidence scores correlate better with the likelihood of correctness through fine-tuning, post-hoc probability modification, or adjustment of the predicted outputs or inputs. Experiments on a diverse range of datasets demonstrate the effectiveness of our methods. We also perform analysis to study the strengths and limitations of these methods, shedding light on further improvements that may be made in methods for calibrating LMs. We have released the code at https://github.com/jzbjyb/lm-calibration.

1 Introduction
Language models (LMs; Church, 1988; Bengio et al., 2003; Radford et al., 2019) learn to model the probability distribution of text, and in doing so capture information about various aspects of the syntax or semantics of the language at hand. Recent works have presented intriguing results demonstrating that modern large-scale LMs also capture a significant amount of knowledge, including factual knowledge about real-world entities (Petroni et al., 2019; Jiang et al., 2020b; Roberts et al., 2020; Bouraoui et al., 2020), commonsense knowledge (Trinh and Le, 2018; Kocijan et al., 2019; Talmor et al., 2019a; Bosselut et al., 2019), and simple numerical operations (Wallace et al., 2019; Talmor et al., 2019a; Geva et al., 2020). Notably, large models trained on massive crawls of Internet text (such as T5 [Raffel et al., 2019] and GPT-3 [Brown et al., 2020]) have been shown to be able to perform quite sophisticated knowledge-based tasks simply through prompting the model to predict the next words given a particular cue.

However, at the same time, LMs are obviously not omnipotent, and still fail to provide appropriate answers in many cases, such as when dealing with uncommon facts (Poerner et al., 2019; Jiang et al., 2020a) or complex reasoning (Talmor et al., 2019a). The high performance on datasets probing factual or numerical knowledge might be achieved through modeling superficial signals in the training data that are not generalizable to unseen test cases (Poerner et al., 2019; Zhou et al., 2020; Wallace et al., 2019; Talmor et al., 2019a). Thus, if such models are to be deployed in real applications it is of crucial importance to determine the confidence with which they can provide an answer. This is especially true if these models are deployed to safety-critical domains such as healthcare and finance, where mistaken answers can have serious consequences.1

In this paper, we ask the question, “How can we know when language models know, with confidence, the answer to a particular knowledge-based query?” Specifically, we examine this from the

1For example, a mocked-up medical chatbot based on GPT-3 answered the question of “should I kill myself?” with “I think you should” (Quach, 2020).
Multiple-choice

Oxygen and sugar are the products of (A) cell division. (B) digestion. (C) photosynthesis. (D) respiration.

Extractive
What type of person can not be attributed civil disobedience? Civil disobedience is usually defined as pertaining to a citizen’s relation ...

<table>
<thead>
<tr>
<th>Format</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple-choice</td>
<td>Oxygen and sugar are the products of (A) cell division. (B) digestion. (C) photosynthesis. (D) respiration.</td>
</tr>
<tr>
<td>Extractive</td>
<td>What type of person can not be attributed civil disobedience? Civil disobedience is usually defined as pertaining to a citizen’s relation ...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Candidate Answers</th>
<th>Original</th>
<th>Calibrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell division</td>
<td>0.00</td>
<td>0.02</td>
</tr>
<tr>
<td>digestion</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>photosynthesis</td>
<td>0.00</td>
<td>0.83</td>
</tr>
<tr>
<td>respiration</td>
<td>1.00</td>
<td>0.14</td>
</tr>
<tr>
<td>head of government</td>
<td>0.07</td>
<td>0.49</td>
</tr>
<tr>
<td>public official</td>
<td>0.91</td>
<td>0.26</td>
</tr>
<tr>
<td>head of government of a country</td>
<td>0.01</td>
<td>0.16</td>
</tr>
<tr>
<td>public officials</td>
<td>0.01</td>
<td>0.09</td>
</tr>
</tbody>
</table>

Table 1: LM calibration examples for the T5 model with correct answers in bold. ‘‘Original’’ and ‘‘Calibrated’’ indicate the normalized probability before and after fine-tuning to improve calibration.

point of view of calibration, whether the model’s probability estimates are well-aligned with the actual probability of the answer being correct. We apply the largest publicly available LMs, T5, BART, and GPT-2, over a wide range of question answering (QA) datasets (Khashabi et al., 2020) covering diverse domains. We first observe that despite the models’ high performance (e.g., T5 eclipses other alternatives such as GPT-3 on some datasets), the models tend to not be well calibrated; their probability estimates over candidates have far-from-perfect correspondence with the actual probability that the answer they provide is correct. Some examples of this are demonstrated in the ‘‘Original’’ column of Table 1.

To alleviate this problem, we propose methods to make LMs’ confidence scores correlate better with the likelihood of model prediction being correct. We examined both fine-tuning methods that modify LMs’ parameters and post-hoc methods that keep LMs fixed and only manipulate the confidence values or inputs. Specifically, we fine-tune the LM using softmax- or margin-based objective functions based on multiple candidate answers. For post-hoc calibration, we examine temperature-based scaling and feature-based decision trees that take prediction probability and input-related features as input and produce calibrated confidence (Jagannatha and Yu, 2020; Desai and Durrett, 2020; Kamath et al., 2020). We also study the sensitivity of LMs’ confidence estimation with respect to language variation by paraphrasing candidate answers and augmenting questions using retrieved context.

Experimental results demonstrate that both fine-tuning and post-hoc methods can improve calibration performance without sacrificing accuracy. We further perform analysis and ablation studies on our methods, inspecting different aspects that may affect calibration performance. We found that like other neural models, LMs are over-confident much of the time with confidence close to either 0 or 1. As a result, post-processing confidence with temperature-based scaling and feature-based decision trees is universally helpful. We also found that LMs become better calibrated if we phrase each answer multiple ways and provide more evidence through retrieval, indicating that current LMs are sensitive to both input and output.

2 LM-based Question Answering

LMs are now a ubiquitous tool in not only natural language generation, but also natural language understanding (NLU), where they are largely used for unsupervised representation learning in pre-trained models such as BERT (Devlin et al., 2019). However, recent work has demonstrated that LMs can also be used as-is to solve NLU tasks, by predicting the missing words in cloze-style questions (Petroni et al., 2019), or by predicting the continuation to prompts (Bosselut et al., 2019; Brown et al., 2020).

Previous works that purport to calibrate LMs (Desai and Durrett, 2020; Jagannatha and Yu, 2020; Kamath et al., 2020; Kong et al., 2020) mainly focus on the former use case, using representations learned by LMs to predict target classes (for tasks such as natural language inference, part-of-speech tagging, or text classification) or identify answer spans (for tasks such as extractive QA). In contrast, we focus on the latter case, calibrating LMs themselves by treating them as natural language generators that predict the next words given a particular input.

To make our observations and conclusions as general as possible, we experiment over a diverse range of QA datasets with broad domain coverage...
Datasets and Domains

<table>
<thead>
<tr>
<th>Format</th>
<th>Datasets and Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-choice</td>
<td>ARC (science (Clark et al., 2018)), A12 Science Questions (science (Clark et al., 2018)), OpenbookQA (science (Mihaylov et al., 2018)), Winogrande (commonsense (Sakaguchi et al., 2020)), CommonsenseQA (commonsense (Talmor et al., 2019b)), MCTest (fictional stories (Richardson et al., 2013)), PIQA (physical (Bisk et al., 2020)), SIQA (social (Sap et al., 2019)), RACE (English comprehension (Lai et al., 2017)), QASC (science (Khot et al., 2018)), NewsQA (news (Trischler et al., 2017)), MT-test (mixed (Hendrycks et al., 2020))</td>
</tr>
<tr>
<td>Extractive</td>
<td>SQuAD 1.1 (wikipedia (Rajpurkar et al., 2016)), SQuAD 2 (Wikipedia (Rajpurkar et al., 2018)), NewsQA (news (Trischler et al., 2017)), Quoref (wikipedia (Dasigi et al., 2019)), ROPES (situation understanding (Lin et al., 2019))</td>
</tr>
</tbody>
</table>

Table 2: Datasets used in this paper and their domains.

over questions regarding both factual and commonsense knowledge (Khashabi et al., 2020). We list all the datasets we used in Table 2 along with their corresponding domain. Since we focus on calibrating LMs as generators, we follow Khashabi et al. (2020) in converting QA datasets of different formats to a unified sequence-to-sequence format that takes a question \( X \) as input and calculates the probability of a continuation \( \hat{Y} \) that corresponds to the answer:

\[
P_{LM}(Y|X) = \prod_{i=1}^{|Y|} P_{LM}(y_i|X, y_{<i}).
\]

Specifically, we focus on two varieties of QA: multiple-choice and extractive, with examples shown in Table 1.²

**Multiple-choice QA** For multiple-choice QA, we assume a question and a set of candidate answers \( \mathcal{I}(X) = \{Y^{(i)}\} \). Inputs \( X \) to LMs are questions concatenated with multiple candidate answers (with each answer prefaced by (A), (B), etc.), and context such as a passage that can be used to help answer the question if any exists. To find the answer the model will return, we calculate the highest-probability answer among the answer candidates:

\[
\hat{Y} = \arg \max_{Y^* \in \mathcal{I}(X)} P_{LM}(Y^*|X).
\]

We can also calculate the normalized probability

\[
P_N(\hat{Y}|X) = \frac{P_{LM}(\hat{Y}|X)}{\sum_{Y^* \in \mathcal{I}(X)} P_{LM}(Y^*|X)},
\]

which provides some idea of the confidence of answer \( \hat{Y} \) with respect to the candidate list.

**Extractive QA** For extractive QA, inputs \( X \) to LMs are questions concatenated with context passages from which the answer must be extracted. In this case, every span within the passage is a candidate answer in \( \mathcal{I}(X) \). However, enumerating over all possible spans of the context passage is computationally costly. Thus, we follow Jagannatha and Yu (2020) in using a manageable set of candidate outputs to perform calibration. Specifically, we develop a method to efficiently calculate probabilities over promising spans that exist in the input. First, we calculate the probability of the first token in output \( Y^* \), masking out any tokens that are not included in the input passage at all. Then, for the top \( R \) scoring tokens, we find their location in the input passage, and calculate the probability of all continuing spans up to a certain length (e.g., 20 tokens). We finally keep the top \( K \) spans as candidates \( \mathcal{I}(X) \) and use all candidates to calculate the probability in a manner similar to that of multiple-choice QA.

### 3 Background on Calibration

A model is considered well calibrated if the confidence estimates of its predictions are well-aligned with the actual probability of the answer being correct. Given an input \( X \) and true output \( Y \), a model output \( \hat{Y} \), and a probability \( P_N(\hat{Y}|X) \) calculated over this output, a perfectly calibrated model satisfies the following condition:

\[
P(\hat{Y} = Y|P_N(\hat{Y}|X) = p) = p, \forall p \in [0, 1].
\]

---

²We also considered using free-form (abstractive) QA datasets, where the answers are not constrained to be one of several choices and can instead be any text. However, we found it hard to evaluate the correctness of generated outputs, as paraphrases of the correct answer are still correct, so we do not report results on these datasets in this paper. Solving this evaluation problem and evaluating calibration on these tasks is an enticing direction for future work.
In practice, we approximate this probability by bucketing predictions into $M$ disjoint equally sized interval bins based on confidence. Guo et al. (2017) examined the calibration properties of neural network classifiers, and proposed a widely used measure of calibration called expected calibration error (ECE), which is a weighted average of the discrepancy between each bucket’s accuracy and confidence:

$$\sum_{m=1}^{M} \frac{|B_m|}{n} \left| \text{acc}(B_m) - \text{conf}(B_m) \right|,$$

where $B_m$ is the $m$-th bucket containing samples whose prediction confidence falls into the interval $(\frac{m-1}{M}, \frac{m}{M}]$, acc($B_m$) is the average accuracy of this bucket, and conf($B_m$) is the average confidence of this bucket. The above equation can be visualized using reliability diagrams (e.g., Figure 1 in the experiments), where each bar corresponds to one bucket, and the height is equal to the average accuracy. The diagram of a perfectly calibrated model should have all bars aligned with the diagonal.

Unfortunately, we found that state-of-the-art LM-based methods for question answering (such as the UnifiedQA model of Khashabi et al. [2020]) were extraordinarily poorly calibrated, with the normalized probability estimates barely being correlated with the likelihood of the outputs being correct. For the two examples in Table 1, for instance, we can see that the language model assigns a very high probability to answers despite the fact that they are wrong. This is particularly important because with T5 (Raffel et al., 2019), GPT-3 (Brown et al., 2020), and others (Guu et al., 2020; Lewis et al., 2020c) being provided as a potential answer to complex knowledge-based tasks, for models to actually be used in practical scenarios they must also be able to know when they cannot provide correct information. In the following section, we examine methods to improve the calibration of pre-trained models through a number of methods.

4 Calibrating LMs for Question Answering

Our calibration methods can be grouped into two categories: methods that fine-tune LMs and post-hoc methods that keep LMs fixed and only manipulate confidence or inputs.

4.1 Fine-tuning-based Calibration

Existing LMs mainly use maximal likelihood estimation (MLE) during training, which maximizes the probability of ground truth output given the input. However, it is well attested that MLE-trained language generators are biased, tending to prefer shorter outputs (Murray and Chiang, 2018), or being biased towards more frequent vocabulary (Ott et al., 2018). However, in the case where we know a set of reasonable candidates $I(X)$, one straightforward way to fine-tune LMs is to only consider candidates in $I(X)$ and directly tune $P_N(Y|X)$ to be a good probability estimate of the actual outputs. We propose two fine-tuning objective functions based on the candidate set.

Softmax-based objective functions model candidates in a one-vs-all setting, where we use the softmax function to normalize the confidence of candidates and maximize the probability corresponding to the correct candidate. We use the negative log likelihood as the loss function:

$$L(X, Y) = -\log \frac{\exp(s(Y))}{\sum_{Y' \in I(X)} \exp(s(Y'))},$$

where the ground truth $Y$ is one of the candidates in $I(X)$, and $s(\cdot)$ is the logit of the corresponding
output (omit condition $X$ for simplicity), which is computed as the log probabilities of all tokens in the output: $s(Y) = \log P_{LM}(Y|X)$.

**Margin-based** objective functions try to maximize the confidence margin between ground truth output and negative results. This is motivated by the fact that non-probabilistic objectives such as those used by support vector machines provide reasonably good probabilistic estimates after appropriate scaling and adjustment (Platt et al., 1999). Specifically, we use the following objective:

$$L(X, Y) = \sum_{Y' \in \mathcal{I}(X)} \max(0, \tau + s(Y') - s(Y)).$$

### 4.2 Post-hoc Calibration

Comparing to fine-tuning methods that optimize the parameters in the model, post-hoc calibration methods keep the model as-is and manipulate various types of information derived from the model to derive good probability estimates (Guo et al., 2017; Jagannatha and Yu, 2020; Desai and Durrett, 2020). In this section, we consider two aspects of the model: model probabilities $P_N(Y' | X)$ and features of the model inputs $X$ or outputs $Y$.

We attempted two representative methods, namely, temperature-based scaling (Guo et al., 2017) and feature-based decision trees (Jagannatha and Yu, 2020), to study whether post-processing probabilities is an effective method for calibration of LMs in the context of QA.

**Temperature-based scaling** methods have been proposed for classification tasks (Guo et al., 2017; Desai and Durrett, 2020), where a positive scalar temperature hyperparameter $\tau$ is introduced in the final classification layer to make the probability distribution either more peaky or smooth: $\text{softmax}(z/\tau)$. If $\tau$ is close to 0, the class with the largest logit receives most of the probability mass, while as $\tau$ approaches $\infty$, the probability distribution becomes uniform. When applying this method to our setting, we use log probabilities of the candidates in $\mathcal{I}(X)$ as logits in computing the softmax function: $z = \log P_{LM}(Y' | X), Y' \in \mathcal{I}(X)$, and $\tau$ is optimized with respect to negative log likelihood on the development split.

**Feature-based decision tree** methods explore non-linear combinations of features to estimate the confidence compared to temperature-based scaling which only considers the raw confidence. We follow previous works (Jagannatha and Yu, 2020; Dong et al., 2018) and use gradient-boosted decision trees (Chen and Guestrin, 2016) as our regressor to estimate the confidence based on features. Besides the raw confidence, we consider the following features and explain their intuitions:

- **Model Uncertainty**: We use the entropy of the distribution over the candidate set $\mathcal{I}(X)$ to inform the regressor of how uncertain the LM is with respect to the question.
- **Input Uncertainty**: We use the perplexity of the LM on the input to indicate the uncertainty over the input. The intuition is that high perplexity might indicate that the input comes from a distribution different from the training distribution of the LM.
- **Input Statistics**: We also use the length of the input and output as features, motivated by our hypothesis that longer text may provide more information to LMs than shorter text.

We train the regressor on the development set similarly to temperature-based scaling by minimizing negative log likelihood.

### 4.3 LM-specific Methods

In addition to standard methods that are applicable to most prediction models, we also examine several methods that are specific to the fact that we are using LMs for the task of QA.

**Candidate Output Paraphrasing** Motivated by the fact that LMs are sensitive to language variation (Jiang et al., 2020b) in tasks like question answering and factual prediction, we hypothesize that one potential reason why the confidence estimation of LMs is not accurate is that the candidate output is not worded in such a way that the LM would afford it high probability. As shown by the example in Table 3, paraphrasing the correct answer from “devoted” to “dedicated” increases the probability from 0.04 to 0.94. Motivated by this, we use a round-trip translation model to paraphrase each candidate output $Y' \in \mathcal{I}(X)$ into several other expressions by first translating it into another language and then back-translating it to generate a set of paraphrases $\text{para}(Y')$. We then
How would you describe Addison? (A) excited (B) careless (C) devoted. Addison had been practicing for the driver’s exam for months. He finally felt he was ready, so he signed up and took the test.

| Paraphrases & Probabilities | devoted (0.04), dedicated (0.94), commitment (0.11), dedication (0.39) |

Table 3: An example question with the correct answer in bold. Different paraphrases of the correct answer have different probabilities.

calculate the probability of each candidate output by summing the probability of all paraphrases $P(Y') = \sum_{Q \in \text{para}(Y')} P_{LM}(Q|X)$ and normalize it following Equation 1. By collectively considering multiple paraphrases, the issue of sensitivity to the wording can be alleviated somewhat, as there will be a higher probability of observing a paraphrase that is afforded high probability by the model.

Input Augmentation Previous work has found that LMs’ factual predictions can be improved if more context is provided (Petroni et al., 2020a), which has inspired many retrieval-augmented LMs that retrieve evidence from external resources and condition the LMs’ prediction on this evidence (Guu et al., 2020; Lewis et al., 2020a,c). We hypothesize that retrieving extra evidence to augment the input also has the potential to improve the confidence estimation of LMs as it will provide the model more evidence upon which to base both its predictions and its confidence estimates. We follow (Petroni et al., 2020a) to retrieve the most relevant Wikipedia article using TF-IDF-based retrieval systems used in DrQA (Chen et al., 2017) and append the first paragraph of the article to the input.

5 Experiments

5.1 Experimental Settings

Datasets We evaluate the calibration performance on both multiple-choice QA datasets and extractive QA datasets listed in Table 2. To test whether our calibration methods can generalize to out-of-domain datasets, we use a subset of datasets of multiple-choice/extractive QA to train our methods, and the remaining subset of datasets to evaluate the performance. Specifically, we use ARC (easy), AI2 Science Question (elementary), OpenBookQA, QASC, Winogrande, CommonsenseQA, and PhysicalIQ for the training subset for multiple-choice QA (denoted as MC-train), and SQuAD 1.1, NewsQA for the training subset for extractive QA (denoted as Ext-train). The remaining subsets used for evaluation are denoted as MC-test and Ext-test, respectively. We also included a much harder multiple-choice QA dataset (denoted as MT-test; Hendrycks et al. [2020]) regarding common sense in a number of genres, in which the largest GPT-3 model and UnifiedQA both display only low to moderate accuracy. For fine-tuning methods, we use the train split of MC-train/Ext-train to fine-tune the LMs. For post-hoc methods like temperature-based scaling and decision trees, we follow Guo et al. (2017) and use the development split of MC-train/Ext-train to optimize the parameters.3

LMs One clear trend of the past several years is that the parameter size and training data size of pre-trained models plays a significant role in the accuracy of models; pre-trained LMs such as BERT (Devlin et al., 2019) tend to underperform more recently released larger LMs like Turing-NLG and GPT-3 (Brown et al., 2020). Thus, we use the largest publicly available LM, which at the time of this writing is Raffel et al.’s (2019) T5 model. The T5 model is a sequence-to-sequence model with both encoder and decoder using transformers (Vaswani et al., 2017), and the largest version has 11 billion parameters, allowing it to realize state-of-the-art performance on tasks such as question answering and natural language understanding (Roberts et al., 2020; Khashabi et al., 2020).

Specifically, we use two varieties of this model. The original T5 model is a sequence-to-sequence model trained on a large corpus of Web text, specifically trained on a denoising objective that generates missing tokens given inputs with some tokens masked out. The UnifiedQA model uses the initial T5 model and fine-tunes on a variety of QA datasets by converting multiple-choice, extractive QA formats into a unified sequence-to-sequence format, similar to the one that we show in Table 1. We use the 3-billion version in our

---

3Since not all datasets in MC-test and Ext-test have a test split, we report the performance on the development split.

4https://msturing.org/.
main experiments in subsection 5.3 (for efficiency purposes), but also report the performance of the largest 11-billion version in ablation studies subsection 5.5.

For comparison with LMs of different architectures trained on different datasets, we also report the performance of two other LMs in Section 5.5: the 0.4-billion BART model (Lewis et al., 2020b), which is a sequence-to-sequence model and the 0.7-billion GPT-2 large model (Radford et al., 2019), which is a conventional language model. We fine-tune them following the same recipe as UnifiedQA (Khashabi et al., 2020).

### Evaluation Metrics

We use accuracy to measure the prediction performance of our methods, and ECE to measure the calibration performance. Accuracy is computed as the ratio of question-answer pairs for which the correct answer has the highest probability among all the candidates in $I(x)$. ECE is computed using Equation 2 by bucketing all candidate answers in $I(x)$ based on confidence. For MC-test and Ext-test which include multiple datasets, we compute accuracy and ECE on each dataset separately and average across them to avoid the metrics being dominated by large datasets.

### Implementation Details

We fine-tune UnifiedQA-3B with a batch size of 16 for 3k steps and UnifiedQA-11B with a batch size of 3 for 15k steps on a v3-8 TPU. The maximal length of input and output are set to 512 and 128 respectively, following the setting of UnifiedQA (Khashabi et al., 2020). For extractive QA datasets, we use top $R = 10$ first tokens and finally $K = 5$ spans are used as candidates. For the paraphrasing-based method, we use the WMT-19 English-German and German-English transformer models to perform back translation (Ng et al., 2019). The beam size is set to 10 for both directions, which will yield $10 \times 10 = 100$ paraphrases in the end. Since some paraphrases are duplicated, we count the frequency and use the top 5 unique paraphrases in our main experiments subsection 5.3. We also report the performance of using different numbers of paraphrases in subsection 5.5. For the retrieval-based augmentation, we use the KILT toolkit (Petroni et al., 2020b) to retrieve the most relevant article from the Wikipedia dump, and append the first three sentences of the first paragraph of the retrieved article to the input. For the feature-based decision trees model, we use XGBoost (Chen and Guestrin, 2016) with logistic binary objective, max depth of 4, number of parallel trees of 5, and subsample ratio of 0.8. We use Temp. to denote temperature-based scaling, XGB to denote feature-based decision trees, Para. to denote paraphrasing, Aug. to denote input augmentation, and Combo to denote the combination of Temp., Para., and Aug. in the experimental section. We use the model with the best calibration performance in post-hoc calibration experiments. For multiple-choice QA, we use the UnifiedQA model after margin-based fine-tuning. For extractive QA, we use the original UnifiedQA model.

### 5.2 Are LM-based QA Models Well Calibrated?

As shown in Table 4, our baseline models (i.e., T5 and UnifiedQA) are strong, achieving state-of-the-art accuracy on a diverse range of QA datasets. On the MT-test datasets, the UnifiedQA model even outperforms the largest version of GPT-3 with 175 billions parameters (Hendrycks et al., 2020). Despite the impressive performance, these models are not well calibrated, with ECE higher than 0.2 on the MT-test dataset. We found that LMs tend to be over-confident about cases they do not know, as shown in the confidence distribution in the first row of Figure 2 that most predictions have aggressive confidence being close to 0 or 1. The UnifiedQA model assigns high confidence to the wrong answer for examples in Table 1, indicating that its confidence estimates are not trustworthy.

### 5.3 Can LM-based QA Models be Calibrated?

We calibrate the UnifiedQA model using both fine-tuning-based methods and post-hoc methods

<table>
<thead>
<tr>
<th>Method</th>
<th>MC-test</th>
<th>MT-test</th>
<th>Ext-test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>ECE</td>
<td>ACC</td>
</tr>
<tr>
<td>T5</td>
<td>0.313</td>
<td>0.231</td>
<td>0.268</td>
</tr>
<tr>
<td>UnifiedQA</td>
<td>0.769</td>
<td>0.095</td>
<td>0.437</td>
</tr>
<tr>
<td>+ softmax</td>
<td>0.767</td>
<td>0.065</td>
<td>0.433</td>
</tr>
<tr>
<td>+ margin</td>
<td>0.769</td>
<td><strong>0.057</strong></td>
<td>0.431</td>
</tr>
</tbody>
</table>

Table 4: Performance of different fine-tuning methods.
Figure 2: The ratio of predictions with respect to confidence of the T5 model (top-left), the UnifiedQA model (top-right), the UnifiedQA model after temperature-based calibration (bottom-left), and the UnifiedQA model after feature-based calibration (bottom-right) on the MC-test datasets.

and show their performance in Table 4 and Table 5 respectively.

Overall, on multi-choice QA datasets (i.e., MC-test and MT-test), both fine-tuning-based methods and post-hoc methods can improve ECE while maintaining accuracy compared to the baseline UnifiedQA model. The best-performing method (i.e., Combo), which combines margin-based fine-tuning, temperature-based scaling, paraphrasing, and input augmentation, improves ECE from 0.095 to 0.044—that is, by over 53%. As shown in the reliability diagrams of the original UnifiedQA model (top-right) and the UnifiedQA model calibrated with Combo (bottom-left) in Figure 1, calibration using our methods makes the confidence estimates of predictions better aligned with their correctness. Comparing those two diagrams, an interesting observation is that our method seems to over-calibrate the LM, making over-estimated bars on the right-hand side of the top-right diagram (bars lower than the diagonal) under-estimated, and vice versa. This is probably caused by the temperature being too aggressive (i.e., too large), making the distribution too flat. Note that the datasets used to learn the temperature (MC-train) and used in evaluation (MC-test) are different, which we hypothesize is the reason why the temperature is too aggressive. We verify this by learning an oracle temperature on the evaluation datasets (MC-test). The learned temperature indeed becomes smaller (1.35 → 1.13), and the reliability diagram (bottom-right in Figure 1) is almost perfectly aligned. This demonstrates the challenge of calibrating LMs across different domains.

However, on extractive QA datasets, the improvement brought by different calibration methods is smaller. We hypothesize that this is because the candidate set \( \mathcal{I}(X) \) generated by the span-based decoding method for extractive QA are harder to calibrate than the manually curated candidate answers for multiple-choice QA. We compute the average entropy of the confidence of the UnifiedQA model over \( \mathcal{I}(X) \) on both extractive QA (Ext-test) and multiple-choice QA datasets (MC-test), and found that Ext-test indeed has much higher entropy compared to MC-test (0.40 vs 0.13), which partially explains the difficulty of calibration on extractive QA datasets.

### 5.4 Analysis of Individual Calibration Methods

In this section, we discuss each method in detail and analyze why they can improve calibration performance.

**Objective Function Matters.** The original UnifiedQA model is fine-tuned based on MLE, which maximizes the probability of the gold answer given the question. Both softmax-based and margin-based fine-tuning, which explicitly compare and adjust the probability of candidate answers, can further improve ECE on multiple-choice datasets. We argue that the softmax-based and margin-based objective functions are better suited for questions with potential candidates.
Post-processing Confidence is Effective Universally. Post-processing the raw confidence either solely based on confidence or other features is effective across all datasets, which is consistent with the conclusion on other tasks such as structured prediction and natural language inference (Jagannatha and Yu, 2020; Desai and Durrett, 2020). We demonstrate the histogram of confidence before and after applying temperature-based scaling or feature-based decision trees in Figure 2. LMs tend to be over-confident, with most predictions having either extremely high or low confidence. Both methods can successfully re-scale the confidence to reasonable ranges, thus improving the calibration performance.

Paraphrasing Answers and Input Augmentation can Improve Confidence Estimation. The improvement brought by using paraphrasing is significant on multiple-choice datasets, demonstrating that using diverse expressions can indeed improve confidence estimation. To better understand under what circumstances paraphrasing works, we group candidate answers into two categories: The first group includes candidate answers that become better calibrated using paraphrases; the second group includes candidate answers whose confidence remains the same using paraphrases. We say that a candidate becomes better calibrated if its confidence increases/decreases by 20% if it is a correct or incorrect answer respectively. We found that the average length of questions for better calibrated candidates (187) is much shorter than that of candidates without improvement (320), indicating that paraphrasing is useful mainly for short questions. We also compute the diversity of word usage in paraphrases using the number of unique words divided by the total length of paraphrases. We found that better calibrated candidates have slightly higher diversity (0.35 vs 0.32), which is consistent with our intuition. Retrieval-based augmentation can also improve calibration performance on multiple-choice datasets, which is probably because the retrieved documents can provide extra evidence about the question, making LMs more robust at confidence estimation.

Calibration Methods are Complementary. By combining margin-based fine-tuning, temperature-based scaling, paraphrasing, and input augmentation, we achieve the best ECE on MC-test, demonstrating that these calibration methods are complementary to each other.

5.5 Ablation Study

In this section, we perform an ablation study to examine different aspects of LM calibration, including calibration performance of different LMs, across LMs with different sizes, using different numbers of paraphrases, and across datasets with potential domain shift.

Performance of Different LMs. We report the performance of two other LMs in Table 6. Both the BART and GPT-2 models are smaller than T5, thus the overall accuracy and calibration performance are lower than that of T5. Both fine-tuning and post-hoc calibration methods can improve ECE, indicating that our methods are applicable to LMs trained with different datasets and architectures.

Performance of LMs with Different Sizes. We conduct experiments using the largest version (i.e., 11B) of the T5 and UnifiedQA model to analyze how calibration performance varies with respect to the size of the LM in Table 7. We found that larger LMs usually achieve both higher accuracy and better calibration performance, which is contradictory to the observation in image classification (Guo et al., 2017) where larger models such as ResNet (He et al., 2016) are no longer well calibrated compared to smaller models like LeNet (Lecun et al., 1998). Given the fact that the size of both the pre-training corpus and LMs are...
Table 7: Performance of the 11B LMs.

<table>
<thead>
<tr>
<th>Method</th>
<th>MC-test ACC</th>
<th>MC-test ECE</th>
<th>MT-test ACC</th>
<th>MT-test ECE</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5</td>
<td>0.359</td>
<td>0.206</td>
<td>0.274</td>
<td>0.235</td>
</tr>
<tr>
<td>UnifiedQA</td>
<td>0.816</td>
<td>0.067</td>
<td>0.479</td>
<td>0.175</td>
</tr>
<tr>
<td>+ softmax</td>
<td>0.823</td>
<td>0.041</td>
<td>0.488</td>
<td>0.129</td>
</tr>
<tr>
<td>+ margin</td>
<td>0.819</td>
<td>0.034</td>
<td>0.485</td>
<td>0.107</td>
</tr>
<tr>
<td>+ Temp.</td>
<td>0.819</td>
<td>0.036</td>
<td>0.485</td>
<td>0.098</td>
</tr>
<tr>
<td>+ XGB</td>
<td>0.818</td>
<td>0.065</td>
<td>0.486</td>
<td>0.108</td>
</tr>
<tr>
<td>+ Para.</td>
<td>0.820</td>
<td>0.035</td>
<td>0.484</td>
<td>0.092</td>
</tr>
<tr>
<td>+ Aug.</td>
<td>0.812</td>
<td>0.031</td>
<td>0.493</td>
<td>0.090</td>
</tr>
<tr>
<td>+ Combo</td>
<td>0.807</td>
<td>0.032</td>
<td>0.494</td>
<td>0.085</td>
</tr>
</tbody>
</table>

Table 8: Performance comparison between training and evaluation datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>MC-train ACC</th>
<th>MC-train ECE</th>
<th>MC-test ACC</th>
<th>MC-test ECE</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5</td>
<td>0.334</td>
<td>0.228</td>
<td>0.313</td>
<td>0.231</td>
</tr>
<tr>
<td>UnifiedQA</td>
<td>0.727</td>
<td>0.133</td>
<td>0.769</td>
<td>0.095</td>
</tr>
<tr>
<td>+ softmax</td>
<td>0.735</td>
<td>0.084</td>
<td>0.767</td>
<td>0.065</td>
</tr>
<tr>
<td>+ margin</td>
<td>0.737</td>
<td>0.069</td>
<td>0.769</td>
<td>0.057</td>
</tr>
<tr>
<td>+ Temp.</td>
<td>0.737</td>
<td>0.051</td>
<td>0.769</td>
<td>0.049</td>
</tr>
<tr>
<td>+ XGB</td>
<td>0.737</td>
<td>0.074</td>
<td>0.771</td>
<td>0.055</td>
</tr>
<tr>
<td>+ Para.</td>
<td>0.742</td>
<td>0.053</td>
<td>0.767</td>
<td>0.051</td>
</tr>
<tr>
<td>+ Aug.</td>
<td>0.721</td>
<td>0.059</td>
<td>0.744</td>
<td>0.051</td>
</tr>
<tr>
<td>+ Combo</td>
<td>0.722</td>
<td>0.042</td>
<td>0.748</td>
<td>0.044</td>
</tr>
</tbody>
</table>

 extremely larger compared to previous practice, we might have completely different observations with respect to confidence estimation. Unlike ResNet trained on CIFAR-100, the training of LMs is not bottlenecked by the dataset, and larger LMs have a stronger capacity to model text distribution and memorize facts, which leads to better calibration performance overall (Kaplan et al., 2020). Overall, our methods can improve ECE from 0.067 to 0.032 using the 11B UnifiedQA model on the MC-test dataset, and from 0.175 to 0.085 on the MT-test dataset. However, compared to the 3B version, improvement brought by post-hoc calibration methods is smaller, which is probably because the 11B version is better optimized and more knowledgeable.

Performance on Training and Evaluation Datasets. As introduced in the experimental section, we perform calibration on the MC-train dataset and evaluate the final performance on the MC-test dataset to study whether our calibration methods can generalize to out-of-domain dataset. We compare the performance on the training dataset and the evaluation dataset in Table 8. We found that on both datasets, each individual method can improve ECE, indicating that our method can generalize to out-of-domain datasets. Note that the improvement on the training dataset (0.133 → 0.042) is larger than on improvement on the evaluation dataset (0.095 → 0.044), which is probably caused by the domain shift between the two datasets.

6 Related Work

Calibration Calibration is a well-studied topic in other tasks such as medical diagnosis (Jiang et al., 2012) and image recognition (Guo et al., 2017; Lee et al., 2018). Previous works in NLP have examined calibration in structured prediction problems such as part-of-speech tagging and named entity recognition (Jagannatha and Yu, 2020), natural language understanding tasks such as natural language inference, paraphrase detection, extractive question answering, and text classification (Desai and Durrett, 2020; Kamath et al., 2020; Kong et al., 2020). In contrast, we focus on calibrating LMs themselves by treating them as natural language generators that predict the next words given a particular input.

LM Probing Previous works probe pre-trained LMs with respect to syntactic and semantic properties (Hewitt and Manning, 2019; Tenney et al.,
factual knowledge (Petroni et al., 2019; Poerner et al., 2019; Jiang et al., 2020b), commonsense knowledge (Trinh and Le, 2018; Kocijan et al., 2019), and other properties (Talmor et al., 2019a). These works usually focus on what LMs know, while in this paper we also consider the cases when LMs do not know the answer with confidence.

7 Conclusion

In this paper, we examine the problem of calibration in LMs used for QA tasks. We first note that despite the impressive performance state-of-the-art LM-based QA models tend to be poorly calibrated in their probability estimates. To alleviate this problem, we attempted several methods to either fine-tune the LMs, or adjust the confidence by post-processing raw probabilities, augmenting inputs, or paraphrasing candidate answers. Experimental results demonstrate the effectiveness of these methods. Further analysis reveals the challenges of this problem, shedding light on future work on calibrating LMs.

Some future directions could be developing calibration methods for LMs on a more fine-grained level than simply holistic calibration across the entire dataset. For example, there has been significant interest in how models perform across diverse subsets of the entire training data (Hashimoto et al., 2018) and how they reflect dataset biases (Rudinger et al., 2018), and the interaction of model confidence with these phenomena is of significant interest. It is also interesting to investigate the effect of calibration on users or downstream tasks. For instance, providing users with model confidences can influence downstream decisions (Zhang et al., 2020), and users may want to adjust required confidence thresholds on critical domains (e.g., health, safety, medicine). All of these are interesting paths of inquiry for future research.

Acknowledgments

This work was supported in part by a gift from Bosch research. The authors thank the Google Cloud and TensorFlow Research Cloud for computation credits that aided in the execution of this research.

References


