Unsupervised Calibration through Prior Adaptation for Text Classification using Large Language Models

Lautaro Estienne
ICC, CONICET-UBA, Argentina
Faculty of Engineering, University of Buenos Aires, Argentina
lestienne@fi.uba.ar

Abstract

A wide variety of natural language tasks are currently being addressed with large-scale language models (LLMs). These models are usually trained with a very large amount of unsupervised text data and adapted to perform a downstream natural language task using methods like fine-tuning, calibration or in-context learning. In this work, we propose an approach to adapt the prior class distribution to perform text classification tasks without the need for labelled samples and only a few in-domain sample queries. The proposed approach treats the LLM as a black box, adding a stage where the model posteriors are calibrated to the task. Results show that these methods outperform the un-adapted model for different number of training shots in the prompt and a previous approach where calibration is performed without using any adaptation data.

1 Introduction

In the last years, Large Language Models (LLMs) like GPT-3 (Brown et al., 2020), FLAN-T5 (Chung et al., 2022), InstructGPT (Ouyang et al., 2022) have proven to be useful for a large variety of complex natural language understanding tasks, showing outstanding performance on many benchmarks related to reading comprehension, summarization, information retrieval, and generative question-answering, among others (Narayan et al., 2018; Zellers et al., 2019; Khattab et al., 2022; Omar et al., 2023). LLMs are pre-trained on a large amount of unsupervised text data following a cost function that is usually self-supervised (autoregressive, denoising, etc.) (Yang et al., 2019; Chung et al., 2022; Devlin et al., 2018).

Notably, LLMs achieve competitive results in a zero-shot scenarios, i.e., without being adapted to the downstream task of interest (Wei et al., 2021; Chung et al., 2022; OpenAI, 2023). Nevertheless, when data is available for adaptation, significant gains can be achieved over the zero-shot scenario. In these cases, the adaptation is done, for example, through (full or selective) fine-tuning (Devlin et al., 2018; Chung et al., 2022), in-context learning (Brown et al., 2020; Wei et al., 2021), or post-processing of the model's outputs (Zhao et al., 2021; Jiang et al., 2021), depending on the size of adaptation dataset, whether this data is labelled or not, and the amount of computational resources available.

In this work, we propose an approach to adapt LLMs to text classification tasks using unlabelled in-domain data. That is, we assume we have examples of the type of text that needs to be classified, but we do not have the actual class of these examples. We propose a light-weight method inspired by the theory of calibration which, for the datasets we experimented with, required only a few dozen of in-domain samples to achieve optimal performance. We call this method **UCPA** (**U**nsupervised **C**alibration through **P**rior **A**daptation). We compare our proposed approach with a previously proposed approach which does not rely on any in-domain data (Zhao et al., 2021) and show that the additional information provides significant performance improvements. Further, we compare our method, theoretically and empirically, with supervised calibration of the posteriors using logistic regression. We show that our approach performs similarly to supervised calibration, without the need for labelled data. Finally, another version of the method is presented where we assume that, even though no labelled data is available, the class priors can be estimated from knowledge of the task. We call this variant **SUCPA** (**S**emi-**U**nsupervised **C**alibration through **P**rior **A**daptation) since some information about the task is needed to estimate the priors.

2 Related Work

Large Language Models (LLMs) LLMs are language models with a large number of parameters, in the order of billions, trained with a massive amount of text to minimize a cost function that can vary from model to model. Models like GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020) or LLaMA (Touvron et al., 2023) are decoder-only transformer-based (Vaswani et al., 2017) architectures trained with an autoregressive loss. In contrast, models like BERT (Devlin et al., 2018) or T5 (Raffel et al., 2019) are trained to denoise the input to obtain the output.

Fine-tuning Pre-trained LLMs can be adapted to a specific task of interest using finetuning techniques like Parameter Efficient Finetuning (PEFT) (Liu et al., 2022), soft-prompt (Lester et al., 2021) and Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022). Despite the attempt of some methods to reduce the number of trainable parameters without sacrificing performance, finetuning is generally an expensive way of adapting a LLM to a certain task since it requires significant amounts of in-domain data, as well as computational resources to load and train the LLM.

In-context Learning Given the large computational and data requirements of the fine-tuning approach, alternative approaches to adapt LLMs to a certain task of interest have been proposed. In-context learning refers to the practice of prepending instructions about the task of interest before the text to be classified, summarized or continued in some other way. Besides these instructions, examples (usually called "shots") on how to perform the task can be added to the prompt. The GPT-3 paper (Brown et al., 2020) showed that close to the state-of-the-art performance in many tasks could be obtained by providing instructions in the prompt without any further adaptation to the task. This work led to the study of good prompt design practices (Zhao et al., 2021).

Calibration There is a large body of literature regarding calibration of classifiers' outputs (Filho et al., 2021), with some recent applications to Natural Language Processing Tasks (Jagannatha and Yu, 2020; Braverman et al., 2019). Recently, a work from Zhao et al. (2021) used the concept of "content-free" input to perform an ad-hoc unsupervised form of calibration for different tasks carried out by a LLM. Our work can be seen as a generalization and formalization of this work, where we derive the approach as unsupervised calibration with an affine expression where the parameters are obtained through the minimization of the cross-entropy.

3 UCPA: Unsupervised Calibration through Posterior Adaptation

An LLM produces posterior probabilities $P(n|\mathbf{h}, \mathbf{q}, \mathbf{e})$ for the next token, n, given the history of previously generated words or tokens, $\mathbf{h}$, and the query, $\mathbf{q}$, and preface, $\mathbf{e}$, which together form the prompt $(\mathbf{e}, \mathbf{q})$. The bolded variables indicate sequences of one or more tokens, while n is a single token. In our case, the preface contains instructions on the classification task to be solved (Chung et al., 2022) and, optionally, a set of training examples for in-context learning (Brown et al., 2020).

When using a LLM to do classification, we need to use the model to obtain $P(y|\mathbf{q}, \mathbf{e})$ from a prompt $(\mathbf{e}, \mathbf{q})$, where $y \in \mathcal{Y}$ and $\mathcal{Y} = \{y_1, \ldots, y_K\}$ is the set of possible classes. To do this, we first define an *ad-hoc* label name w_k for every label $y_k \in \mathcal{Y}$. Then, we prompt the LLM, which we will call θ, with the word sequence given by $\mathbf{e}$ followed by $\mathbf{q}$ to get a score

$$s_k = P_\theta(w_k|\mathbf{q}, \mathbf{e}) \tag{1}$$

for the class y_k. Finally, the probability distribution $P(y|\mathbf{q}, \mathbf{e})$ is computed by normalizing this score to get a probability distribution over the classes:

$$P(y = y_k|\mathbf{q}, \mathbf{e}) = \frac{s_k}{\sum_{k'} s_{k'}} \tag{2}$$

Note that there may be cases in which the label name is represented with more than one token (see Table 1 for a complete list of datasets used and the label names of each one). In those cases the probability $P_\theta(w_k|\mathbf{q}, \mathbf{e})$ can be computed as

$$P_\theta(w_k|\mathbf{q}, \mathbf{e}) = \prod_{m=0}^{M_k-1} P_\theta(w_k^{m+1} \mid \mathbf{q}, \mathbf{e}, w_k^{1:m}) \tag{3}$$

where $w_k^{1:m} = [w_k^1, \ldots, w_k^m]$ and $w_k^{1:0}$ is an empty string. The posteriors on the right-hand side are obtained directly from the LLM.

Using the definition of conditional probability, the posterior $P(y|\mathbf{q}, \mathbf{e})$ can be written as:

$$P(y \mid \mathbf{q}, \mathbf{e}) = P(y \mid \mathbf{e}) \frac{P(\mathbf{q} \mid y, \mathbf{e})}{P(\mathbf{q} \mid \mathbf{e})} \tag{4}$$

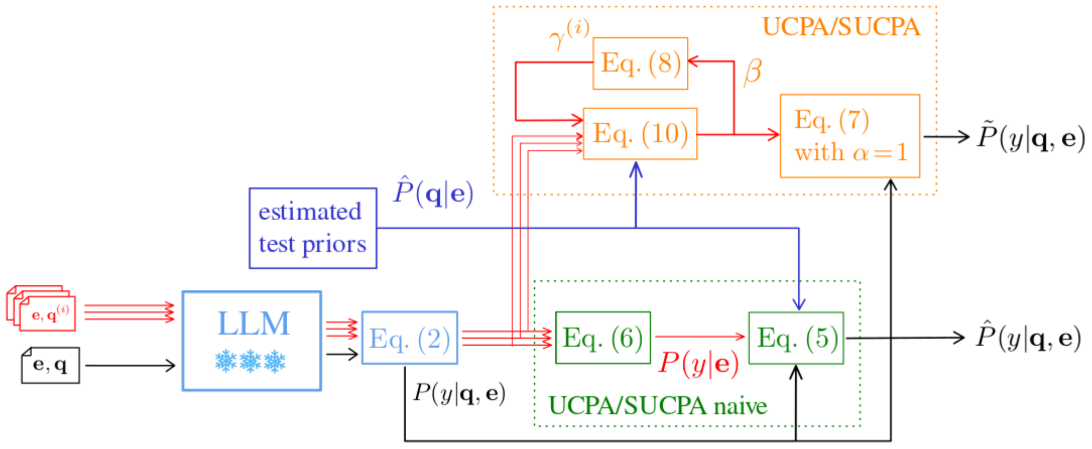

Figure 1: Schematic of the proposed approach. The test sample $(\mathbf{e}, \mathbf{q})$ is processed by the LLM and plugged to Equation (2) to produce the posterior $P(y \mid \mathbf{q}, \mathbf{e})$. In addition, a set of in-domain queries $\{\mathbf{q}^{(1)}, \ldots, \mathbf{q}^{(N)}\}$ is used to reestimate the priors in a "naive" (bottom) or iterative (up) way. Lastly, estimated test priors $P(\mathbf{q} \mid \mathbf{e})$ are used to produce adapted posteriors $\tilde{P}(y \mid \mathbf{q}, \mathbf{e})$ or $\hat{P}(y \mid \mathbf{q}, \mathbf{e})$. For the UCPA approach, $P(\mathbf{q} \mid \mathbf{e})$ is assumed uniform, whereas for SUCPA, specific knowledge of the task is used to estimate that prior.

The factor $P(\mathbf{q} \mid y, \mathbf{e})/P(\mathbf{q} \mid \mathbf{e})$ is the ratio between the likelihood of the query given the preface and the class name, and the likelihood given only the preface. This likelihood ratio (LR) reflects the increase in the likelihood of the query obtained by adding the class name to the response.

The quantity $P(y|\mathbf{e})$ can be understood as a prior probability in the sense that it is not conditioned on the query: it is the probability of the class given only the preface. This prior depends strongly on the task of interest. While the LLM might have a tendency to predict a certain class given the preface, this may not be the most likely class for the task of interest. As we mentioned in Section 1, adapting the model to the application of interest is key for obtaining relevant responses from the LLM. While the preface $\mathbf{e}$ is, in fact, a way to adapt the LLMs outputs to the task of interest, it may not be sufficient to fully adapt the posteriors since not all the information about the task can be represented in a short text explanation.

In this work we propose to improve the posteriors computed from the LLM's scores by explicitly adjusting the priors to the task of interest. This is done by assuming the following expression for the in-domain posterior:

$$\hat{P}(y \mid \mathbf{q}, \mathbf{e}) = \delta \; P(y \mid \mathbf{q}, \mathbf{e}) \frac{\hat{P}(y \mid \mathbf{e})}{P(y \mid \mathbf{e})} \qquad (5)$$

which can be interpreted as taking away the effect of the mismatched prior $P(y|\mathbf{e})$ from the LLM-derived posterior $P(y|\mathbf{q}, \mathbf{e})$, replacing it with the in-domain prior $\hat{P}(y|\mathbf{e})$, and then rescaling by δ to make sure the resulting distribution adds up to one. To obtain $\hat{P}(y|\mathbf{q}, \mathbf{e})$ we need to compute the posterior, which is obtained directly from the LLM using Equation (2), and the two priors.

The prior $\hat{P}(y|\mathbf{e})$ is the prior we expect for our task of interest. We may or may not know this distribution. In this work we compare results under two assumptions: 1) that we do not know anything about the prior distribution in which case we simply assume a uniform distribution $\hat{P}(y_k|\mathbf{e}) = 1/K$ for all k, and 2) that, even though we do not have labelled data, we do have a good estimate of the frequencies of the classes we expect to see in practice. In our experiments, for this second scenario we compute $\hat{P}(y_k|\mathbf{e}) = N_k/N$, where N_k is the number of training samples of class k. In practice, though, these priors could be estimated from knowledge of the task rather than from in-domain labelled data. Arguably, this second scenario is no longer unsupervised, so we will call this method Semi-Unsupervised Calibration through Prior Adaptation (SUCPA), while the method that uses uniform priors will be called Unsupervised Calibration through Prior Adaptation (UCPA).

The prior $P(y \mid \mathbf{e})$ is the prior for y that is implicit in our LLM. It is the distribution of classes that the model would output for this task, across all possible relevant queries we may provide. Hence, we estimate this prior by simply running the LLM on (unlabelled) training data $Q^{\text{train}} = \{\mathbf{q}^{(1)}, \ldots, \mathbf{q}^{(N)}\}$ and averaging the resulting posteriors:

$$P(y|\mathbf{e}) \approx \frac{1}{N}\sum_{i=1}^{N} P(y|\mathbf{q}^{(i)},\mathbf{e}) \tag{6}$$

The method above is a heuristic that relies on an assumption (Equation (5)) that may or may not hold, as well as on the approximation of the prior above. When using this expression to obtain the prior we call the method "UCPA/SUCPA-naive". As we will see, this heuristic works quite well in our experiments. Further, as we explain in Section 4, we can also obtain the prior in a more principled way using an expression derived from linear logistic regression.

3.1 Content-free Prompts

The approach proposed by Zhao et al. (2021) can be seen as a special case of the UCPA-naive method. In that work, calibrated posteriors are computed using Equation (5), where the training set Q^{train} is composed of one or more "content-free" inputs, in the sense that they do not contain any relevant meaning, and they are created manually by the user. For example, the authors experiment with using "[MASK]", "N/A", and the empty string.

4 Supervised Affine Calibration and Semi-UCPA (SUCPA)

A standard way to adapt the posterior probabilities from a classifier to a certain domain of interest is to calibrate them using in-domain labelled data. Calibration refers to the process of transforming the scores of a system to optimize the quality of the scores as posteriors. This is usually done by choosing a certain parameterized form for the transform and training those parameters to minimize a proper scoring rule like the cross-entropy (Filho et al., 2021). One instance of this approach is linear logistic regression.

Linear logistic regression assumes that the logarithm of the calibrated posteriors are given by:

$$\log \tilde{P}(y_k|\mathbf{q},\mathbf{e}) = \gamma + \alpha_k \log P(y_k|\mathbf{q},\mathbf{e}) + \beta_k \tag{7}$$

where $P(y_k|\mathbf{q},\mathbf{e})$ is given by Equation (2), α and β are parameters, and the value of γ is determined so that $\sum_{k=1}^{K} \tilde{P}(y_k|\mathbf{q},\mathbf{e}) = 1$. That is,

$$\gamma = -\log \sum_{k'=1}^{K} P(y_{k'}|\mathbf{q},\mathbf{e})^{\alpha_k} e^{\beta_{k'}} \tag{8}$$

The α_k and β_k parameters are estimated by minimizing the cross-entropy on a training set $\mathcal{C}^{\text{train}} = \{(\mathbf{q}^{(1)}, y^{(1)}), \ldots, (\mathbf{q}^{(N)}, y^{(N)})\}$ where $\mathbf{q}^{(i)}$ and $y^{(i)}$ are the query and the class of sample i:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \log \tilde{P}(y^{(i)}|\mathbf{q}^{(i)},\mathbf{e}) \tag{9}$$

In this work we take α_k to be a scalar, independent of the class. This is what is usually done for calibration (Brummer and Van Leeuwen, 2006; Guo et al., 2017; Platt et al., 1999). In particular, temperature scaling (Guo et al., 2017), one of the most widely used calibration methods, corresponds to taking $\beta_k = 0$ for all k and α_k a single scalar.

If we restrict the calibration transformation to have $\alpha_k = 1$ and set the derivative of the cross-entropy to zero, we can derive the following expression for β_k:

$$\beta_k = \log\frac{N_k}{N} - \log\left[\frac{1}{N}\sum_{i=1}^{N} P(y_k|\mathbf{q}^{(i)},\mathbf{e}) e^{\gamma^{(i)}}\right] \tag{10}$$

where $\gamma^{(i)}$ is given by Equation (8) with $\mathbf{q} = \mathbf{q}^{(i)}$. Note that this is not a closed-form expression for β_k but rather a system of equations since the right-hand side contains all the β_k within the $\gamma^{(i)}$.

We can now compare Equation (5) and Equation (7). Taking the logarithm of Equation (5) for one specific class k we get:

$$\log \hat{P}(y_k|\mathbf{q},\mathbf{e}) = \gamma' + \log P(y_k|\mathbf{q},\mathbf{e}) + \beta'_k \tag{11}$$

where $\gamma' = \log \delta$ and

$$\beta'_k = \log \hat{P}(y_k|\mathbf{e}) - \log P(y_k|\mathbf{e}) \tag{12}$$

The form of this expression is identical to that of Equation (7) when taking $\alpha_k = 1$ for all k. Both γ and γ' are determined so that the posterior on the left-hand side adds to one. Hence, if $\beta_k = \beta'_k$, the two posteriors would be identical.

Comparing the expressions for β_k and β'_k we can see that they coincide if we take $\hat{P}(y_k|\mathbf{e}) = N_k/N$ as we assume in our experiments for the SUCPA approach, and if we take $\gamma^{(i)} = 0$, since in that case the second term in β_k coincides with Equation (6). Of course, $\gamma^{(i)}$ is not necessarily zero. Yet, as we will see in the experiments, making this assumption has little effect on the results. Nevertheless, we can also estimate the β_k that satisfies Equation (10) exactly using an iterative approach where we first set $\gamma^{(i)} = 0$ and compute

β_k for all k, plug those values into Equation (8) to get a new value for $\gamma^{(i)}$ and plug that back into Equation (10), repeating these steps until convergence. We find that this algorithm leads to identical results as running linear logistic regression with $\alpha_k = 1$. In the following, we will refer to the UCPA (and SUCPA) approach described in section 3 as "UPCA-naive" (and "SUCPA-naive"), whereas the iterative version of this method will be called simply UCPA (and SUCPA). Figure 1 summarizes both approaches for both the UCPA and SUCPA variants.

5 Experimental Set Up

We evaluate the proposed approach on the n-shot text classification task following a similar procedure as Zhao et al. (2021). We use four datasets for which the task is classification of a single text into known categories: binary sentiment analysis using SST-2 (Socher et al., 2013), 6-way question classification using TREC (Voorhees and Tice, 2000), 4-way news classification using AGNews (Zhang et al., 2015), and the 14-way ontology classification using DBPedia (Lehmann et al., 2015). Table 1 shows the number of test samples, class priors and prompt used for each dataset.

We used the standard train and test partitions for all datasets. For AGNews and DBPedia we selected 1000 random samples from the test set since their test splits were too large for computation in our infrastructure. This approach was similar to the one used by Zhao et al. (2021) where they selected 300 test samples (see their github repository[1]). Also following this work, we used GPT-2 XL which has 1.5B parameters and consists of a decoder-only transformer architecture (Vaswani et al., 2017). The checkpoint was downloaded from the `huggingface` website[2], and the code used to run experiments is available on github[3].

For each dataset, a set of n shots were selected by random sampling the train split and added to the preface (see Table 1). Then, the Q^{train} set was generated by random sampling 600 samples from Q^{train} after discarding the samples added to the preface. The $\mathcal{C}^{\text{train}}$ set to train the calibrator was the same as Q^{train} with the difference that $\mathcal{C}^{\text{train}}$ contains the labels and Q^{train} does not. For some of the experiments we further subset the training set to smaller sizes. When doing this, we use 10 different seeds to generate the subsets to assess the variation in results due to varying training sets. For each training set we obtain posteriors on the test set and generate 100 bootstrap samples (Tibshirani and Efron, 1993; Keller et al., 2005). The curves in the figures 2 and 3 show the mean performance on the pooled performance estimated from all training sets and test bootstraps and confidence intervals plotted one standard deviation away from the mean.

We show results in terms of error rate (1-accuracy), equivalently to Zhao et al. (2021). Further, in the final results, we also include the cross-entropy performance. Cross-entropy is a proper scoring rule which means that it evaluates the performance of the provided scores as posterior probabilities (Gneiting and Raftery, 2007). In the figures we show normalized cross-entropy, where the cross-entropy is divided by the cross-entropy of a naive system that always outputs the prior distribution, ignoring the input sample. A normalized cross-entropy larger than 1.0 indicates that the system is so badly calibrated that its performance is worse than that of a naive system (Brummer, 2010; Ferrer, 2023).

6 Effect of the Training Set Size

Figure 2 shows the error rate for all datasets as a function of the number of training samples used to perform domain adaptation for the 0-shot scenario (no examples added to the preface). This set is used either to train the calibration model (in which case the class labels are used), to compute Equation (6) for UCPA/SUCPA-naive, and to compute Equation (10) for UCPA/SUCPA. For SUCPA, the training labels are used to compute N_k/N which is used to obtain $\hat{P}(y|\text{e})$ and in Equation (10). For UCPA, the training labels are never used and N_k/N is assumed uniform over the classes. The figure also shows the results for the baseline system which takes the posteriors from Equation (2) without any prior adaptation. Finally, we show the performance of a naive baseline that always chooses the most likely class in the training data.

We first note that, as explained in Section 4, SUCPA and linear logistic calibration with $\alpha = 1$ gives identical performance. We found the same results for the case of 1, 4 and 8-shot learning, indicating that our iterative algorithm for estimating the β_k that satisfies Equation (10) is working as ex-

[1] `https://github.com/tonyzhaozh/few-shot-learning`
[2] `https://huggingface.co/gpt2-xl`
[3] `https://github.com/LautaroEst/efficient-reestimation`

Dataset	Class Priors	Test Samples	Prompt Template
TREC	0.28: Description 0.23: Number 0.19: Entity 0.16: Location 0.13: Person 0.02: Abbreviation	500	"*Classify the questions based on whether their answer type is a Number, Location, Person, Description, Entity, or Abbreviation.* *Question:* `[example 1]` *Answer Type:* `[label 1]` ... *Question:* `[example n]` *Answer Type:* `[label n]` *Question:* `[query]` *Answer Type:*"
SST-2	0.50: Negative 0.50: Positive	1821	"*Review:* `[example 1]` *Sentiment:* `[label 1]` ... *Review:* `[example n]` *Sentiment:* `[label n]` *Review:* `[query]` *Sentiment:*"
AGNews	0.27: Technology 0.26: Business 0.25: Sports 0.22: World	1000	"*Classify the news articles into the categories of World, Sports, Business, and Technology.* *Article:* `[example 1]` *Answer:* `[label 1]` ... *Article:* `[example n]` *Answer:* `[label n]` *Article:* `[query]` *Answer:*"
DBpedia	0.09: Artist 0.09: Nature 0.08: Athlete 0.08: Plant 0.08: Company 0.07: School 0.07: Village 0.07: Animal 0.07: Transportation 0.07: Politician 0.07: Album 0.06: Book 0.06: Building 0.05: Film	1000	"*Classify the documents based on whether they are about a Company, School, Artist, Athlete, Politician, Transportation, Building, Nature, Village, Animal, Plant, Album, Film, or Book.* *Article:* `[example 1]` *Answer:* `[label 1]` ... *Article:* `[example n]` *Answer:* `[label n]` *Article:* `[query]` *Answer:*"

Table 1: Number of test samples, class priors in the test set, and prompt used for each dataset in this work. The instruction text for each case is taken from (Zhao et al., 2021).

pected. We can also see for both UCPA and SUCPA (i.e., regardless of how the in-domain priors are estimated), the naive and the iterative approaches give similar performance, indicating that the average posterior in Equation (6) is a good approximation for the model's prior. Both proposed approaches show better performance than the original model for three of the four datasets even when very few (as low as 10) training samples are available to do the prior adaptation. In the case of DBPedia, however, we see that the SUCPA methods degrade performance compared to the baseline when the number of training samples is smaller than 80. This is due to the fact that the priors estimated as N_k/N cannot be robustly estimated on so few samples for this 14-class. Since the priors in this dataset are close to uniform (see Table 1), in this case it is better to assume them uniform than to estimate them from a very small dataset. A similar trend can be found for SST-2 and AGNews for which the priors are perfectly uniform so that assuming them is always better than estimating them from data. On the other hand, for the TREC dataset we can see that, given enough training samples, SUCPA works better than UCPA when the test priors are not uniform. Similar trends to those seen in this figure were found for a prompt containing 1, 4 and 8 shots.

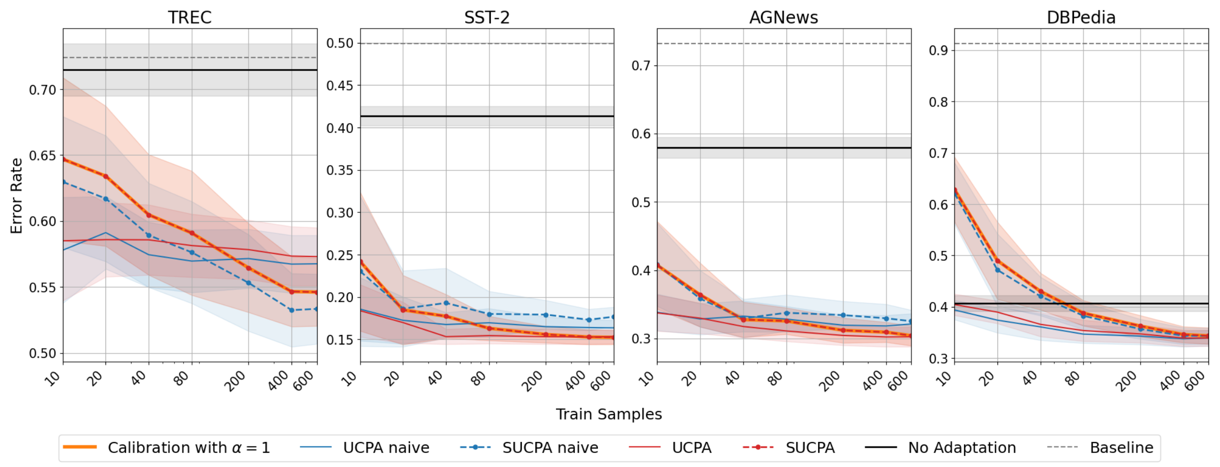

Figure 2: Error rate vs. the number of training samples used in the prior adaptation process in a zero-shot configuration. The red lines show the iterative approach for UCPA and SUCPA, whereas the blue lines show the naive version. The orange curve shows the calibration results for $\alpha = 1$. The black curve shows the results without adaptation and the grey dotted line represents the majority-class classifier (both are constant because they do not use training data).

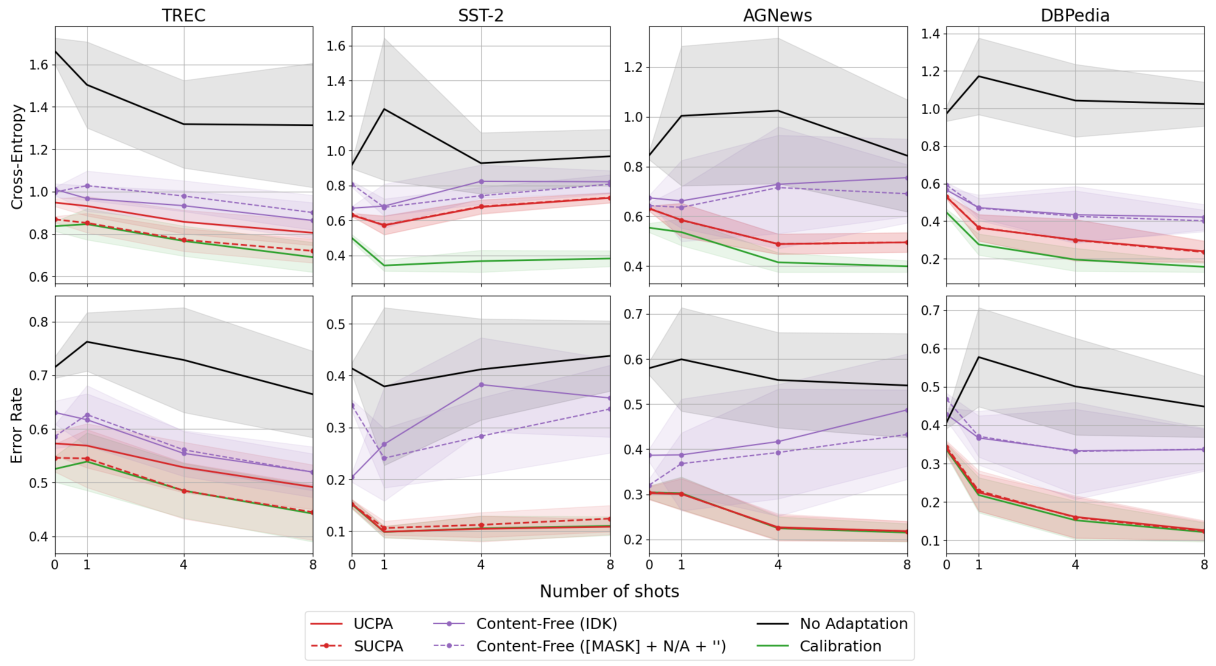

Figure 3: Cross-Entropy and Error Rate (1-Accuracy) vs. the number of examples (shots) contained in the prompt for 600 training samples. The red lines show the iterative approach for UCPA and SUCPA. The lines in purple show the results for content-free adaptation and the green line is the calibration using parameters α and β. As before, the black line shows the case for which no adaptation has been performed.

7 Effect of the Number of Shots

Figure 3 shows the effect of the number of examples (shots) added to the preface for each dataset in terms of error rate and cross-entropy when the number of training samples is 600. We compare our proposed methods with four systems: 1) the non-adapted posteriors (solid black line), 2) the affine calibration method (solid green line) in which parameters α and β are trained using the labelled training data, and 3) and 4) the two content-free calibration methods explained in Section 3.1 with the two sets of content-free inputs that were used in the authors' code (see footnote above), namely, {'IDK'} and {'[MASK]', 'N/A', ''}. We can see that, for 600 training samples, our proposed methods consistently improve upon the content-free baseline, as well as over the non-adapted posteriors. In most cases, the non-adapted model presented a mean cross-entropy close to or higher than 1 for all number of shots, which implies that the model is useless for this task and cannot learn from the prompt shots. They also tend to have small standard deviation and a more stable tendency across the number of shots. Of course, the content-free approaches have the advantage that they do not require training data at all. Yet, these results show that, if unlabelled training data is available, we can obtain significant gains from our proposed UCPA approach. Further, if an estimate of the class priors is available, the SUCPA approach can lead to additional gains in datasets with imbalanced priors, like TREC.

Figure 3 also shows that the affine calibration system performs consistently better than our proposed approaches, particularly in terms of cross-entropy which better highlights issues of miscalibration compared to accuracy. This is expected since the calibrator is taking advantage of the labelled training data. Note that affine calibration is one specific case of a downstream classifier, one with very few parameters which can be trained with a small number of samples. A more complex downstream classifier may give further improvements, but would require larger labelled training datasets.

With some exceptions (like in SST-2), increasing the number of examples added to the preface leads to gains in all methods that do some kind of adaptation. The original posteriors, on the other hand, have erratic behavior as a function of the number of shots with a very large standard deviation resulting from the specific selection of examples.

Additional experiments were performed when the number of training samples is set to 40 showing similar trends to those in Figure 3 with the exception that SUCPA works worse than UCPA in most cases due to a bad estimate of the class priors. In practice, the class priors would be estimated from knowledge of the task rather than from the training dataset, so that the SUCPA performance would in fact depend on how good that estimate is.

Overall, we can see that the proposed method leads to a large gain with respect to the non-adapted posteriors, even in a scenario where a relatively small amount of unlabelled data is available for training. In terms of computational requirements in comparison with the baseline method, the proposed approach requires the computation of the probabilities $P(y|\mathbf{e})$. The time needed to compute these probabilities depends linearly on the number of training samples. During evaluation, the proposed approach has a negligible overhead with respect to the baseline system since the only difference is that it needs to compute Equation (5) using the pre-computed $P(y|\mathbf{e})$ and target priors $\hat{P}(y|\mathbf{e})$.

8 Conclusion and Future Work

In this work we proposed a method for calibrating the posteriors generated by an LLM for a certain text-classification task. We assume that only a relatively small number of unlabelled in-domain samples are available for adaptation. We propose a simple method for calibrating the posteriors generated by the LLM by adapting the prior class distribution to the task of interest in an unsupervised manner. Optionally, the method allows the new priors to be set to the ones we expect to see during deployment, when known. When such priors are unknown, they can be assumed uniform. We show that, as long as the test priors can be estimated reasonably well, or that the uniform assumption is not too far off from the test distribution, the proposed approach works significantly better than the un-adapted posteriors even with a small amount of available adaptation samples. Further, we show that it works better than a previous approach where calibration is performed without using any adaptation data.

In our experiments, the proposed method was shown to work well for classification tasks where the number of classes was relatively small in number compared to the amount of training data. We hypothesize that the requirement on training data would increase linearly with the number of classes. Future work will include a comparison with finetuning techniques, as well as experiments with LLMs larger than GPT-2 XL. Further, we will explore the generalization of the proposed method to other NLP tasks like question-answering and summarization where the required output is not necessarily restricted to just a few tokens and to tasks outside of the NLP domain where prior mismatch may also be a common scenario.

References

Mark Braverman, Xinyi Chen, Sham M. Kakade, Karthik Narasimhan, Cyril Zhang, and Yi Zhang. 2019. Calibration, entropy rates, and memory in language models. *CoRR*, abs/1906.05664.

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. *CoRR*, abs/2005.14165.

Niko Brummer. 2010. *Measuring, refining and calibrating speaker and language information extracted from speech*. Ph.D. thesis, Stellenbosch: University of Stellenbosch.

Niko Brummer and David A. Van Leeuwen. 2006. On calibration of language recognition scores. In *2006 IEEE Odyssey - The Speaker and Language Recognition Workshop*, pages 1–8.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Luciana Ferrer. 2023. Analysis and comparison of classification metrics.

T. S. Filho, H. Song, M. Perello-Nieto, R. Santos-Rodriguez, M. Kull, and P. Flach. 2021. Classifier calibration: How to assess and improve predicted class probabilities: a survey. *arXiv preprint arXiv:2112.10327*.

Tilmann Gneiting and Adrian E Raftery. 2007. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. *CoRR*, abs/1706.04599.

Abhyuday Jagannatha and Hong Yu. 2020. Calibrating structured output predictors for natural language processing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2078–2092, Online. Association for Computational Linguistics.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Mikaela Keller, Samy Bengio, and Siew Wong. 2005. Benchmarking non-parametric statistical tests. In *Advances in Neural Information Processing Systems*, volume 18. MIT Press.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv:2212.14024*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, S. Auer, and Christian Bizer. 2015. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *CoRR*, abs/1808.08745.

Reham Omar, Omij Mangukiya, Panos Kalnis, and Essam Mansour. 2023. Chatgpt versus traditional question answering for knowledge graphs: Current status and future directions towards knowledge graph chatbots. *arXiv:2302.06466*.

OpenAI. 2023. Gpt-4 technical report.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35.

John Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Robert J Tibshirani and Bradley Efron. 1993. *An introduction to the bootstrap*, volume 57. Monographs on statistics and applied probability.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, page 200–207, New York, NY, USA. Association for Computing Machinery.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *CoRR*, abs/2109.01652.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*. PMLR.