Uncertainty Calibration for LLMs on Multiple Choice Tasks

Akhil Kasturi, Vamsi Deeduvanu, Zach Lafeer

1 Introduction

Given the prevalence of modern LLMs for question-answering, it becomes increasingly crucial to determine when their outputs are to be trusted. In the academic setting in particular, students often query an LLM to produce answers for study materials in a multiple choice question format. Incorrect answers are then both harmful to learning and reduce confidence in the model. In this project, we aim to produce calibrated uncertainty estimates for LLM responses to multiple choice questions; that is, the overall model will output both an answer and a confidence level between 0 and 1 that can be directly interpreted as a probability of correctness. Specifically, we evaluate the usefulness of 1) hidden state information and 2) self-reported LLM uncertainty estimates for producing reliable overall uncertainty estimates. Our experiments focus on estimating uncertainty measures particularly for small language models and assessing their generalization on unseen topics.

2 Related Work

2.1 Supervised Approach for LLM Uncertainty Estimation

(Liu, Pan, Li, & Chen, 2024) describes a supervised approach for LLM uncertainty estimation using features derived from the model's hidden activations. The approach can be built on top of a white-box, gray-box, or black-box LLM, where either full hidden activation layer information, only entropy/probability outputs, or only the final output is available to be extracted from the model. For our project, we focus on the white-box case, where all features from a gray-box LLM are also available.

The supervised uncertainty estimation task is setup as follows. First, a prompt $\boldsymbol{x}$ is passed to the LLM which generates a response $\boldsymbol{y}$. Then, an auxiliary supervised learning model $\hat{g}$ is trained to produce the uncertainty estimates,

$$g(\boldsymbol{x}, \boldsymbol{y}) \approx \mathbb{E}[s(\boldsymbol{y}, \boldsymbol{y}_{\text{true}})|\boldsymbol{x}, \boldsymbol{y}]$$

for some score function s. In the multiple choice task, we choose,

$$s(\boldsymbol{y}, \boldsymbol{y}_{\text{true}}) = \mathbb{1}(\boldsymbol{y} = \boldsymbol{y}_{\text{true}})$$

The model $\hat{g}$ is trained on the dataset,

$$\mathcal{D}_{sl} = \{\boldsymbol{v}_i, s(\boldsymbol{y}_i, \boldsymbol{y}_{i,\text{true}})\}_{i=1}^{n}$$

where $\boldsymbol{v}_i$ is a set of features extracted from the LLM hidden activations. At test time, the original LLM and the learned uncertainty estimator $\hat{g}$ are used as a pipeline to produce new answers and corresponding uncertainty estimates.

2.2 Uncertainty Calibration Evaluation

Reliability diagrams (Guo, Pleiss, Sun, & Weinberger, 2017): Visual tool used to assess the calibration of a model by plotting the actual accuracy of predictions against their predicted confidence levels. To create a reliability diagram, the predictions are divided in M equally spaced confidence intervals or bins, denoted as

$$I_m = \left(\frac{m-1}{M}, \frac{m}{M}\right]$$

For each bin B_m, the empirical accuracy is computed as:

$$\text{acc}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}(\hat{y_i} = y_i)$$

where $\hat{y_i}$ is the predicted label and y_i is the true label. The average confidence interval for the bin is calculated as:

$$\text{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p_i}$$

where $\hat{p_i}$ is the model's predicted probability. A perfectly calibrated model has the property that for all bins m, $\text{acc}(B_m)$ is equal to $\text{conf}(B_m)$, producing a diagonal line when plotted. Any deviation from this diagonal indicated miscalibration, where the model is either over-confident or under-confident.

Expected Calibration Error (ECE) (Guo et al., 2017): ECE provides a scalar metric summarizing the model's calibration by measuring the average difference between confidence and accuracy across all bins. Formally, it is computed as:

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

where $|B_m|$ is the number of samples in bin B_m and n is the total number of samples. ECE effectively quantifies the degree to which the model's predicted confidence aligns with actual performance, with lower values indicating better calibration. However, the result is sensitive to the number of bins chosen and may not capture extreme miscalibrations in any specific bin.

2.3 Uncertainty Recalibration Techniques

Isotonic Regression (for calibration)(Guo et al., 2017): A non-parametric calibration method that learns a monotonic piecewise constant function mapping uncalibrated predicted probabilities to calibrated probabilities. By doing so, we aim to minimize the squared error between the calibrated predictions and the true labels while enforcing monotonicity. The optimization objective is:

$$\min \sum_{m=1}^{M} \sum_{i=1}^{m} \mathbb{1}(a_m \leq \hat{p_i} < a_{m+1})(\theta_m - y_i)^2$$

where θ_m is the learned constant prediction for bin B_m, and $a_1, \ldots, a_{M+1}$ are the bin boundaries satisfying the condition $a_1 = 0 \leq a_2 \leq \ldots \leq a_{M+1} = 1$. Additionally, the constraints $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_M$ are used to ensure that the fitted function is non-decreasing. Isotonic regression is flexible and capable of modeling complex calibration curves but may overfit when applied to small datasets due to its high expressiveness.

Platt Scaling (Guo et al., 2017): A parametric calibration method that fits a logistic regression model to map uncalibrated model outputs (logits) into calibrated probabilities. In the case of binary labels, the calibrated probability is computed as:

$$\hat{q}_i = \sigma(az_i + b) = \frac{1}{1 + e^{-(az_i+b)}}$$

where z_i is the logit or score for instance i, σ is the sigmoid function, and a and b are scalar parameters learned from a validation set. The parameters are optimized by minimizing the negative log-likelihood, defined as:

$$\min_{a,b} -\sum_{i=1}^{n}[y_i \log(\hat{q}_i) + (1 - y_i)\log(1 - \hat{q}_i)]$$

Platt scaling is simple and effective for binary tasks; while it can be used for multi-class problems the it is beyond the focus of this project.

Temperature Scaling (Guo et al., 2017): An extension of Platt scaling designed for multi-class situations. It introduces a single scalar parameters, known as the temperature T, which scales the logits before applying the softmax function. The calibrated confidence is computed as:

$$\hat{q}_i = \max_k \sigma_{SM}\left(\frac{z_i}{T}\right)^k = \max_k \frac{\exp(z_i^{(k)}/T)}{\sum_{j=1}^{k}\exp(z_i^{(j)}/T)}$$

where $z_i^{(k)}$ is the logit score for class k and σ_{SM} is the softmax function. The optimal temperature T is learned by minimizing the negative log-likelihood over the validation set, formulated as:

$$\min_T -\sum_{i=1}^{n}\log\left(\sigma_{SM}\left(\frac{z_i}{T}\right)^{(y_i)}\right)$$

Temperature scaling is highly efficient as it adjusts the sharpness of the softmax distribution without changing the predicted class labels. It effectively reduces the overconfidence by softening the output probabilities.

3 Methods

3.1 Approach

We build upon the methodology described in Section 2.1, introducing a novel extension. *In addition to the existing feature set, we propose incorporating the model's self-reported uncertainty estimate—obtained via prompting—as an additional feature.* To evaluate the effectiveness of this approach, we compare the performance of three models for the uncertainty estimator $\hat{g}$: a Support Vector Classifier (SVC), Gaussian Naive Bayes (GaussianNB), and an ExtraTrees classifier. For all three models, we apply Platt scaling for re-calibration.

3.2 Prompt Formats

For each question-answer pair, we query the LLM to produce a single token response, indicating which multiple-choice option it thinks is correct. Afterwards, we separately prompt the LLM to self-report a confidence level to the previous query, represented as a percentage between 0 and 100, inclusive. An example pair of prompts is provided in Figures 1 and 2.

Prompt 1: You are a helpful assistant, very good at answering multiple choice questions in the field of astronomy. You reply to each question with only the letter A, B, C, or D, indicating the option you think is correct.
Why do we look for water-ice in craters at Mercury's pole?
(A) Actually water-ice is all over Mercury and not just at the poles.
(B) The pole is the only place fortunate enough to have had comet impacts
(C) Radar from the earth can only see Mercury's poles.
(D) These craters contain the only permanently shadowed regions on Mercury

Figure 1: First prompt to the LLM to get a prediction.

Prompt 2:
How confident are you in your answer? Respond with a number from 0 to 100, where 0 means not at all confident and 100 means very confident.

Figure 2: Second prompt to the LLM to get self-reported uncertainty estimates.

3.3 Feature Extraction

As described by (Liu et al., 2024), we calculate the probability distribution of the 4 options, A, B, C, and D, and its entropy using the logits. Then, the hidden states from the last token from the middle layer of the transformer architecture are extracted. These raw features, and the label representing if the LLM was correct or not, are used to train a Lasso model to select K features with highest absolute coefficient values. Then another K features are extracted with highest absolute mutual information with the same label. Lastly, K more features are selected with highest Pearson correlation coefficient with the label. Feature extraction training is performed during the training phase of our overall algorithm. The indices of the features selected from the hidden states are saved to be used for inference on unseen data (without using the true labels).

3.4 Experiments

To examine uncertainty calibration for LLMs on multiple choice tasks, we use the model Gemma-3 (1 billion parameter, instruction-tuned) on HuggingFace as our base LLM. Then, we train an uncertainty estimator with the LLM predictions and ground truth labels from a training subset of STEM topics from the MMLU dataset (Hendrycks et al., 2021). Next, we evaluate the performance and calibration of the model on a held-out test subset of MMLU, also sampled from the same group of STEM topics. We call this evaluation setting the in-distribution setting with respect to the uncertainty estimator. Additionally, we evaluate the model on a subset of non-STEM topics from

MMLU, which we call the out-of-distribution setting. Each is tested with and without self-reported (SR) uncertainty estimates. Overall, the following settings are tested,

- In-distribution evaluation without SR,
- In-distribution with SR,
- Out-of-distribution without SR
- Out-of-distribution with SR.

For each setting, we evaluate, 1) ECE for the uncertainty estimator 2) a calibration plot for qualitative analysis and 3) a histogram of predicted uncertainties, to better assess the sharpness of the model.

4 Results

The performance of the base LLM on the two datasets is shown in Table 1. For conciseness, calibration and sharpness plots for the last test setting is shown below in Figure 3. The remaining such plots are provided in Appendix Figures 4, 5, and 6. Lastly, ECE results across all settings and models are shown in Table 2.

Dataset	LLM Accuracy
MMLU STEM (In-distribution)	0.269
MMLU Non-STEM (Out-of-distribution)	0.410

Table 1: LLM performance on each dataset.

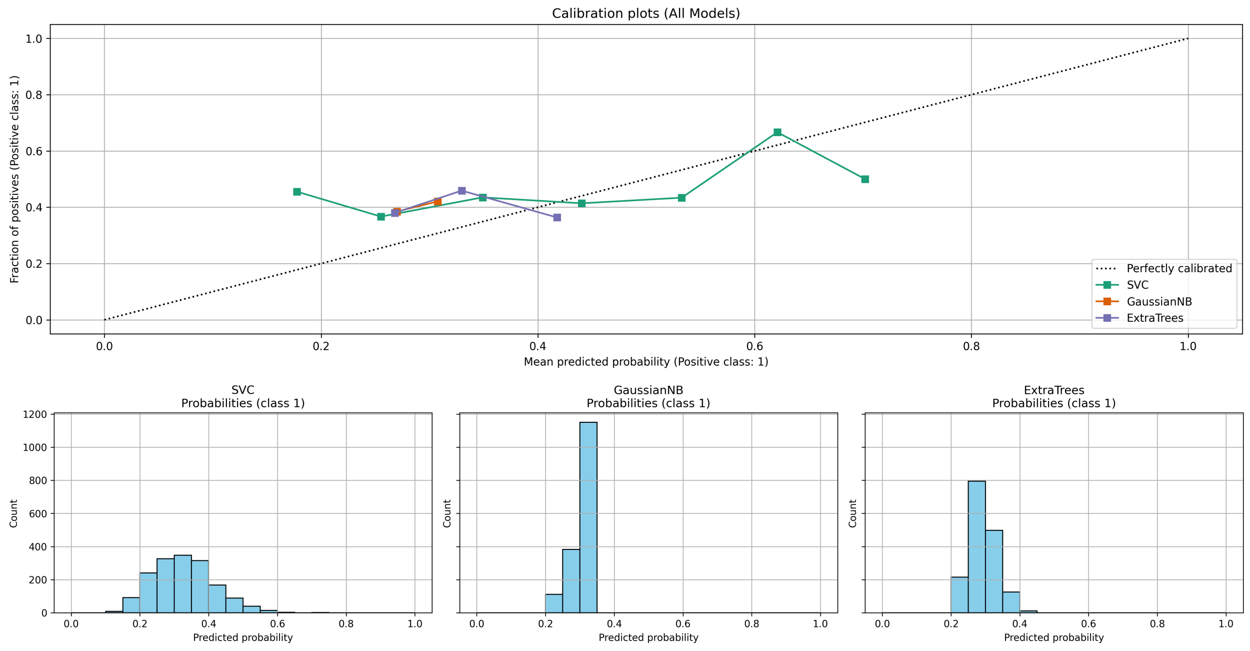

Figure 3: Out-of-distribution w/ SR: calibration and sharpness plots

	SVC	GaussianNB	ExtraTrees
In-distribution w/o SR	0.047	0.023	0.045
In-distribution w/ SR	0.047	0.023	0.021
Out-of-distribution w/o SR	0.098	0.115	0.121
In-distribution w/ SR	0.098	0.114	0.118

Table 2: ECE results for each model on the four evaluation settings.

5 Analysis

As shown in Table 2, adding self-reported uncertainty estimates does not appear to have any significant impact on the quality of the uncertainty estimator, regardless of the choice of particular model or evaluation dataset. However, the self-reported uncertainties from the 1 billion parameter base LLM had very low variance on this task. Specifically, most self-reported values were 95%, with very little deviation across all examples. Given this observation, it appears that the small LLM is not able to reliably assess its own uncertainty on this task. In conjunction with the low accuracies shown in Table 1, this suggests severe over-confidence from the base model.
Despite the low performance, an ideal uncertainty estimator should be able to produce reliable estimates of the probability that the LLM is correct. In Figure 3, we see that on the region [0.3,0.5], each of the three uncertainty estimators produce decently well-calibrated predictions. The sharpness diagrams further show that the SVC estimator produces somewhat well-distributed predictions, rather than concentrating mass on the overall LLM accuracy proportion. In comparison, GaussianNB achieves similar ECE, but does so by almost always predicting an uncertainty near the overall LLM accuracy of 0.269, which is not very useful. Despite this, the limited range of predictions for all trained uncertainty estimators limits the range of interpretability for the calibration plots.
Overall, the uncertainty estimator demonstrates similar performance when applied to the out-of-distribution task. Of course, there remain large similarities between the two datasets as both are subsets of MMLU and appear in the same question format. Nevertheless, the underlying language and concepts between the STEM and non-STEM subsets is notable when considering that the features are extracted from the middle layer of the LLM.

6 Future Work and Conclusion

This work leaves numerous opportunities for future work, especially with higher compute. First, testing with larger models would help show at what point models develop the capability to predict their own uncertainty (or at least, produce an estimate sufficiently correlated with the true uncertainty). Second, it is possible to try more sophisticated methods of uncertainty calibration, such as evidential methods, that are more suitable to larger datasets. Last, it remains to be seen what the effect of fine-tuning is on this model pipeline. In particular, it may be useful to instruction tune the model to improve both its accuracy and self-reported uncertainty estimates.

References

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). *On calibration of modern neural networks.* Retrieved from `https://arxiv.org/abs/1706.04599`

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., & Steinhardt, J. (2021). *Measuring massive multitask language understanding.* Retrieved from `https://arxiv.org/abs/2009.03300`

Liu, L., Pan, Y., Li, X., & Chen, G. (2024). *Uncertainty estimation and quantification for llms: A simple supervised approach.* Retrieved from `https://arxiv.org/abs/2404.15993`

Appendix

Below are additional figures showing calibration and sharpness in the remaining three evaluation settings. These are removed from the results section for the sake of limiting the page count.

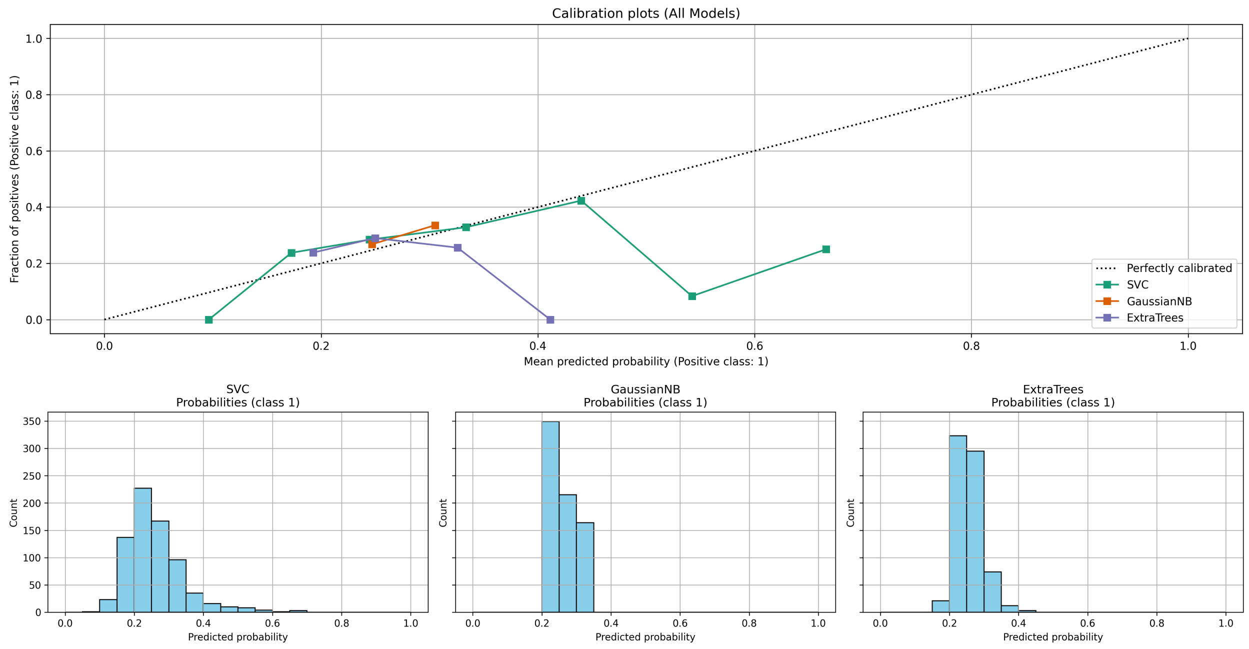

Figure 4: In-distribution w/o SR: calibration and sharpness plots

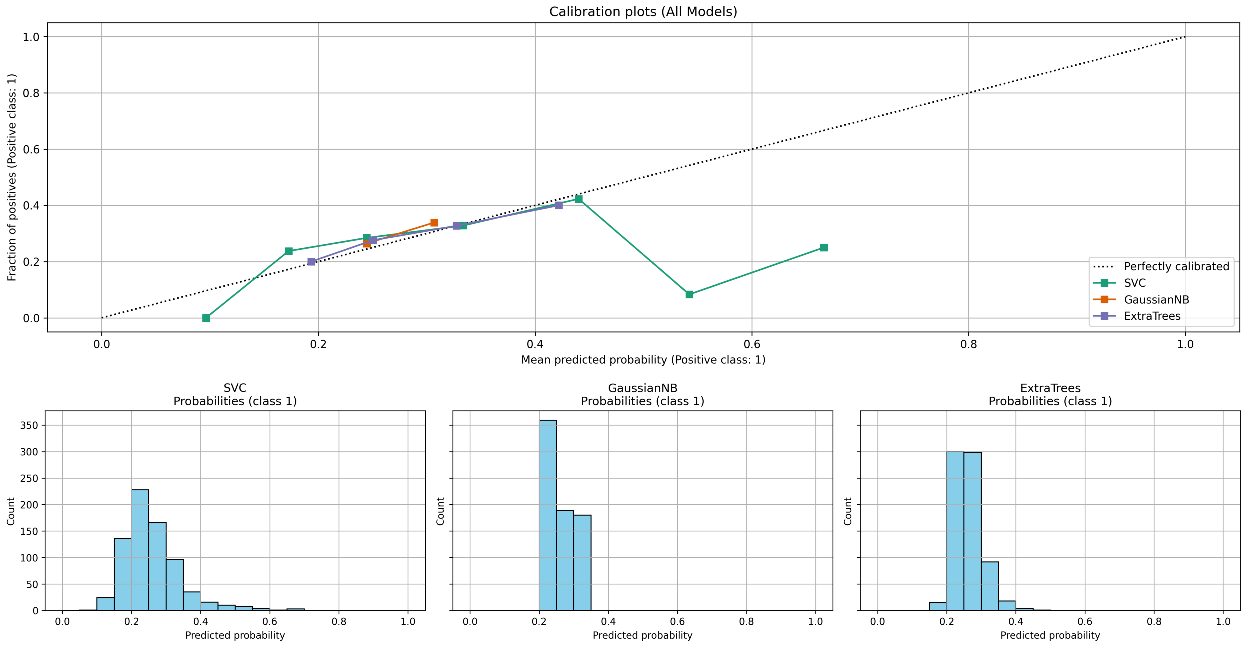

Figure 5: In-distribution w/ SR: calibration and sharpness plots

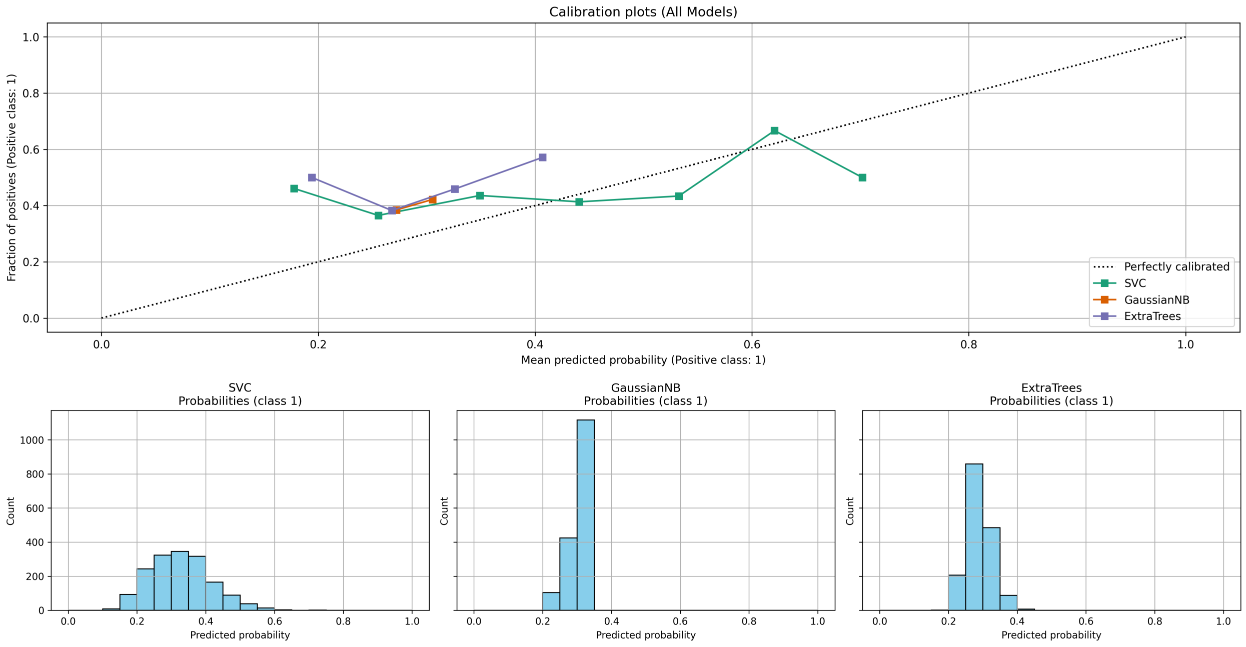

Figure 6: Out-of-distribution w/o SR: calibration and sharpness plots