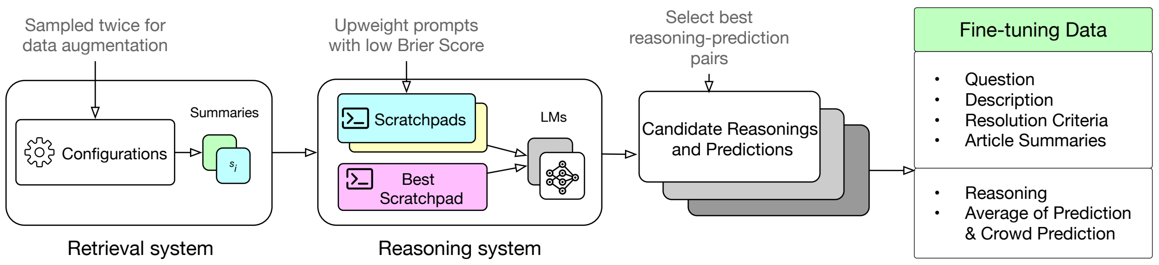

Sampled twice for data augmentation
Configurations
Summaries
s_i
Retrieval system
Upweight prompts with low Brier Score
Scratchpads
Best Scratchpad
LMs
Reasoning system
Select best reasoning-prediction pairs
Candidate Reasonings and Predictions
Fine-tuning Data
Question
Description
Resolution Criteria
Article Summaries
Reasoning
Average of Prediction & Crowd Prediction

Figure 2: **Our procedure of generating data for self-supervised training.** For each question, the method generates multiple candidate reasoning-prediction pairs and selects those that outperform human aggregates for fine-tuning.

- Forecasting requires a holistic consideration of the possibilities (Tetlock and Gardner, 2015). We next prompt the model to leverage the retrieved information and its pre-training knowledge to produce arguments for why the outcome may or may not occur.
- The model can potentially generate weak arguments. To avoid treating all considerations as equal, it is instructed to weigh them by importance and aggregate them accordingly into an initial forecast.
- Finally, to prevent potential bias and miscalibration, the model is asked to check if it is over- or under-confident and consider historical base rates (Tetlock and Gardner, 2015), prompting it to calibrate and amend the prediction accordingly.

Base model. We prompt GPT-4-1106-Preview with the best scratchpads (found via hyperparameter sweep), since it consistently gives the lowest Brier scores among the LMs we test (see Section 5.2 on reasoning).

Fine-tuned model. We also prompt a fine-tuned version of GPT-4 that we trained to generate reasonings with accurate predictions (Section 5.1). We prompt it with only the question's basic information (no scratchpad instructions) since our fine-tuned model is trained to reason without prescriptive instructions.

4.3 Ensembling

Since the aggregate of predictions is usually superior to individual forecasts (Tetlock and Gardner, 2015), we elicit multiple predictions from the base and fine-tuned models.

We prompt GPT-4-1106-Preview with the optimal scratchpad prompt (Figure 15), along with the 2 next best scratchpad prompts identified in Section 5.2. For our fine-tuned model, we set temperature $T = 0.5$ and prompt it 3 times to sample 3 additional forecasts. This gives us 6 forecasts in total: 3 from the base model, and 3 from the fine-tuned model. Given these forecasts, the system ensembles them into a final prediction by taking their trimmed mean, as this performs best on the validation set among the ensemble methods we implement (see Section 5.2 on ensembling).

We provide further details about our system in Appendix D, including hyperparameters and prompt designs.

5 Optimizing the System

We now describe the procedure to optimize our retrieval and reasoning system and the results obtained.

5.1 Fine-tuning a Reasoning Model

We fine-tune a LM to produce reasonings that lead to accurate forecasts. To generate the data for fine-tuning, we (1) collect a large set of forecasts on the train set, and then (2) select a subset where the model outperforms the human crowd.

Collecting fine-tuning data. To generate the preliminary data, we run our system at each retrieval date in the retrieval schedule and on each question in the train set, multiplied by 16 configurations described below.

First, as a form of data augmentation, we retrieve 2 sets of articles for each question by sampling 2 (distinct) retrieval configurations (Figure 2, left). Specifically, we sample the retrieval prompt, number of queries, and articles per query, twice (Section 4), with relevancy filtering and summarization following the process described in Section 4.1. This results in 2 inputs to the reasoning model per question, each with the same question but a different set of articles.

To increase the chance of attaining a prediction that outperforms the crowd, we generate 4 candidate outputs per input (8 total per question) by trying different scratchpad prompts. The first uses the optimal prompt found in Section 5.2 (Figure 15). We then sample 3 other scratchpad prompts, with probability inversely proportional to their Brier score on the validation set. We prompt both Claude-2.1 and GPT-4-Preview, since we find that Claude-2.1 is better on some questions. In total, this gives 16 candidate forecasts per question.

Selecting fine-tuning data. We seek to fine-tune our model on strong forecasts. To select the data, we only keep outputs that give a lower Brier score than the crowd's. However, this can inadvertently cause overconfidence in our fine-tuned model. To mitigate this, we discard pairs where the prediction deviates by more than 0.15 from the crowd prediction, and we also average our prediction with the crowd prediction when constructing the target output.

The resulting fine-tuning data has the following structure (Figure 2, right):

- The **input** to the model consists of the question, description, and resolution criteria, followed by summarized articles.
- The **target output** consists of a reasoning and a prediction.

Importantly, the fine-tuning input excludes the scratchpad instructions. By doing so, we directly teach the model which reasoning to apply in a given context.

In total, 73,632 reasonings are generated from which 13,253 meet the above desiderata. Finally, we fine-tune GPT-4-0613[2] on the 6,000 most recent points for 2 epochs, due to budget constraint (Figure 2, right).

5.2 Hyperparameter Sweep

Our hyperparameter sweep optimizes an (intermediate) metric over a discrete set of choices, such as prompts and the number of articles presented. We share the key findings below and more details in Appendix E.

Methodology. We divide the hyperparameters into groups of 1-2 and optimize them iteratively. For each group, we select the best configuration based on the average Brier score on the validation set, except for search query generation where we use proxy metrics for efficiency.

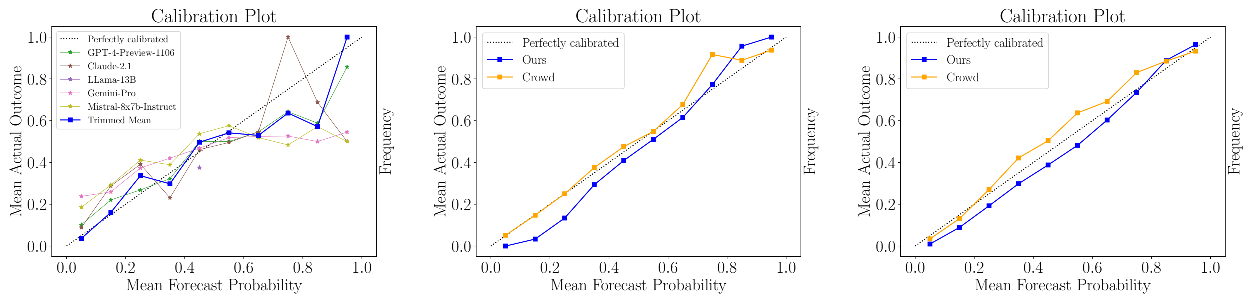
We optimize the groups sequentially, fixing the optimal configurations from previous groups while sweeping the current one. The hyperparameters yet to be swept are randomized independently for each input question.

Retrieval. Our retrieval uses LMs for search query generation, relevance rating, and summarization. We independently optimize the prompt choices for search query generation and summarization. The relevance rating prompt is fixed in our system (Figure 14).

For search query generation, we evaluate the prompts by retrieving articles with the generated queries and examining two metrics: (1) the average relevance score across all retrieved articles, and (2) the average relevance score of articles exceeding a relevance threshold of 4 on a 6-point scale. The 2 high-scoring prompts perform similarly under both metrics and generate queries with little overlap. As a result, we use both prompts (Figure 12) to generate queries and take the union.

For summarization, we run our system end-to-end and pick the top 1 prompt (Figure 13) with respect to the Brier score.

[2]While the more recent GPT-4-1106-Preview has 2 years of more recent knowledge, it was not available for fine-tuning.



(a) Calibration of Base Models on Test (b) Calibration of System on Validation (c) Calibration of System on Test

Figure 3: **Our system is naturally well calibrated** on both (b) validation and (c) test. The crowd is also well calibrated, consistent with Zou et al. (2022)'s findings. In contrast, the base models in the zero-shot setting (a) are less calibrated (Section 3.4).

Reasoning. The reasoning system takes a ranked list of article summaries and prompts LMs to make forecasts. We optimize: (1) the ordering criterion of the summaries (by relevance or recency); (2) the number k of article summaries presented to LMs; and (3) the choice of scratchpad instructions to elicit the forecasts.

For efficiency, we optimize them in 2 independent stages:

- In the first stage, we jointly optimize (1) and (2). Ranking by relevance and setting $k = 15$ achieve the lowest average Brier score.
- In the second stage, we optimize (3) the reasoning prompt. We identify the top 3 prompts out of 15 candidates to elicit 3 predictions from our base model in our system; see Figure 15 for the best one.

In optimizing the reasoning system, we test both Claude-2.1 and GPT-4-1106-Preview as candidate models for generating forecasts. GPT-4-1106-Preview consistently yields a 0.01-0.03 lower Brier score. Therefore, our final system elicits predictions from GPT-4-1106-Preview and the fine-tuned GPT-4-0613.

Ensembling. We implement 5 ensembling methods, including mean, median, geometric mean, trimmed mean, and a variant of universal self-consistency (USC; Chen et al. (2023)). Trimmed mean performs the best in our evaluation; see Section E.1 for details.

Calibration. Interestingly, our system is naturally well calibrated (Figure 3b), and we find that standard calibration methods such as binning or isotonic regression do not improve performance.

6 Evaluations

We evaluate our optimized system on the test set and find that it comes close to human crowd performance (Section 6.1). Next, we analyze its strengths and weaknesses (Section 6.2). Motivated by the observations, we introduce a relaxed setting, where the system may make forecasts selectively (given its identified strengths), and find that our system surpasses the crowd aggregate (Section 6.3). Finally, we demonstrate how our system can be used to complement aggregated human forecasts (Section 6.4).

6.1 System Nears Human Performance

We first evaluate the Brier score of our end-to-end system on the test set. Note that all hyperparameters were chosen based on the validation set and all test set questions appear temporally after the validation questions, mirroring the setting of a real-time forecasting competition. In addition to the Brier score, we also report accuracy to compare with past work (Zou et al., 2022; Yan et al., 2024).

As the main result, our averaged Brier score is .179, while the crowd achieves .149, resulting in a difference of .03. Our accuracy on the test set is 71.5%, whereas the community scores 77.0%, resulting in a difference of