# CornellNLP at CheckThat! 2025: Hybrid Llama–GPT-4 Ensembles with Confidence Filtering for Numerical Claim Verification*

Notebook for the CheckThat! Lab at CLEF 2025

Lea Duesterwald[1,*], Adi Arora[1] and Claire Cardie[1]

[1]Cornell University Ann S. Bowers College of Computing and Information Science

**Abstract**

Claim verification is an important challenge and remains difficult, particularly in the space of numerical claims, both for large language models (LLMs) and human annotators. This paper presents our participation in the CheckThat! Lab Task 3: Fact-Checking Numerical Claims shared task which aims to advance fact checking systems on claims that contain numerical and temporal details. We show that large LLMs like GPT-4 are able to rate their confidence in their predictions such that confidence ratings are indicative of model performance. We use this finding to develop an ensembling method, combining the predictions of GPT-4 with few-shot prompting and a fine-tuned Llama 2 7B model via a voting mechanism based on model self-confidence ratings. While our final ensemble achieved moderate performance overall, our results demonstrate a novel and effective use of LLM self-confidence as a lightweight way to integrate large LLMs with other models to improve claim verification.

**Keywords**

Numerical Claim Verification, Large Language Model, GPT-4, Llama 2, Self-Confidence Estimation, Ensemble Methods

## 1. Introduction

Fact checking systems are critical for stopping the spread of misinformation and automating fact checking has been an important area of research, especially recently [1]. Evaluating numerical claims is of particular importance [2]. However, numerical claims have been shown to be particularly difficult to verify, both for annotators [3] and for large language models (LLMs) [4].

The CheckThat! Task 3 Challenge [5, 6, 7], Fact-Checking Numerical Claims, focuses on evaluating claims that contain numerical and temporal details. The task provides training, development, and test sets developed from the QuanTemp dataset, a diverse, multi-domain dataset of numerical claims with detailed metadata and evidence collections [4]. The aim of the task is to improve the ability of NLP systems to process and understand numerical claims and to verify these claims. In this paper we describe our participation in Task 3 of the CheckThat! Lab.

With LLMs becoming more powerful, they now show promise to read and understand numerical claims, reason about the claim, and verify the correctness of the claim. However, these LLMs are not perfect as they can often make mistakes or hallucinate [8]. To further complicate their application to claim verification, LLMs can also hallucinate confidently, producing incorrect responses with full confidence and without the ability to determine their own lack of knowledge [9].

We sought to take a closer look at an LLM's ability to determine its level of confidence in the specific context of numerical claim verification. More specifically, we sought to prompt GPT-4 to not only predict a label for each claim, but also produce a self-confidence rating that should be indicative of correctness. If LLMs do possess this ability, we can leverage this to improve claim verification, ultimately combining a large LLM like GPT-4 with other models to selectively replace low-confidence predictions from GPT, providing an efficient way to increase performance without adding significant complexity.

*CLEF 2025 Working Notes, 9 – 12 September 2025, Madrid, Spain*

*Corresponding author.

lkd46@cornell.edu (L. Duesterwald); aka64@cornell.edu (A. Arora); ctc9@cornell.edu (C. Cardie)

0009-0006-0220-9125 (L. Duesterwald); 0009-0005-9189-0591 (A. Arora); 0000-0002-2061-6094 (C. Cardie)

In our work, we developed an ensemble of GPT-4 and a fine-tuned Llama 2 7B model, employing a voting mechanism based on self-confidence ratings, with the goal of improving performance on numerical claim verification.

The remainder of the paper is organized as follows: Section 2 reviews related work. Section 3 discusses our methodology, detailing the ensemble method and each individual component. Section 4 contains our results. Discussion and conclusions are presented in Section 5.

## 2. Related Work

Automated fact checking typically includes a pipeline consisting of claim detection, evidence retrieval, and finally verdict prediction. This verdict prediction often involves natural language inference (NLI) [10, 11, 12] making automation of fact checking an important application of LLMS. Verifying numerical remains a continuing challenge, both for human annotators and models [3, 4]. Claims that contain numerical expressions or temporal aspects also present a unique challenge and traditional fact-checking datasets often do not cover numerical expressions and are thus not representative of numerical claim verification. As a dataset focused exclusively on numerical claims QuanTemp addresses this gap, providing multi-domain data containing various types of numerical claims [4].

Powerful LLMs with huge knowledge bases provide the capability for fact verification [13]. However, LLMs still struggle with hallucinations [8] that hinder their ability to be applied to this task of numerical claim verification. Further, LLMs also often exhibit overconfidence in their outputs, even in incorrect outputs and achieving accurate calibration remains a challenge in many areas [14].

Prompt engineering and designing the optimal prompt for a task has been shown to significantly improve LLM performance, especially for tasks that require factual accuracy [15]. In particular, few-shot prompting has been shown to significantly enhance model performance on classification tasks without task-specific fine-tuning [16]. Beyond general-purpose LLMs, adapting large models to specific tasks such as fact-checking can improve model performance [17]. Specifically, parameter-efficient fine tuning (PEFT) methods such as LoRA can reduce computation costs while still retaining strong performance on specific tasks [18, 19, 20]. Ensemble methods provide the opportunity to improve performance and robustness by combining multiple models. Combining different LLMs such as GPT-4 and Llama can lead to improved accuracy across different NLP tasks [21, 22].

Our work leverages few-shot prompting with GPT-4 and LoRA for efficient fine-tuning of Llama 2 7B to independently optimize each model individually. We then combine these models in an ensemble system. This approach allows us to capture the strengths of both models to improve overall performance through the ensemble.

## 3. Methodology

### 3.1. Ensemble Overview

To classify numerical claims, we used an ensemble method which employed a voting mechanism between two models: a fine-tuned Llama 2 7B model and a GPT-4 model with few-shot prompting. The voting mechanism employs self-confidence ratings from GPT-4 in its predictions to settle disagreements, selecting GPT's predicted label when it indicated a high confidence in its answer and Llama's predicted label otherwise. Both Llama 2 7B and GPT-4 were separately optimized to achieve the best performance on a development test set and then combined via the voting mechanism on the final test set. The datasets used for both fine-tuning Llama and developing the few shot prompts for GPT-4 were the provided training and validation datasets with the included top-k BM25 evidences. Both methods made initial predictions individually on the provided test set which included the claims, decomposed sub-questions, and top-k BM25 evidences.

### 3.2. Fine-Tuning Llama

We fine-tuned a 7B-parameter Llama 2 model (huggyLlama/llama-7b) using LoRA adapters and a 4-bit quantization (to increase the efficiency of training by reducing memory usage). The model was fine-tuned on the provided training set. The input was structured as a prompt with the form *[Claim]: .... [Evidences]: ... [Label]: ...* and the model was trained for a causal language modeling task. The fine-tuning was performed over 3 epochs with a batch size of 1, learning rate of 2e-4 and using the paged_adamw_32bit optimizer.

To evaluate the model during development, we extracted a hold-out test set, taking 10% of data from both the provided train and validation sets for new data sizes of 8,942 for the train set, 2,776 for the validation set, and 1301 for the test set. The model was fine-tuned on this new train and validation set and evaluated on the hold-out test set. However, as part of the final ensemble when making predictions on the actual test set, we used the model fine-tuned on all of the available train and validation data.

### 3.3. Few-shot prompting with GPT-4

We employed few-shot prompting, providing a few examples from each label class from the development set, to improve the predictions made by GPT-4. The baseline model we used was GPT-4 Turbo with default parameter values due to its reasoning abilities and effectiveness in few-shot settings. The baseline prompt was:

*You are a numerical fact-checking assistant. For the given numerical claim and evidence, please decide if the claim is True (2), False (0), or Half True/False (conflicting) (1)*
*For help analyzing the claim, please reference the following examples, separated by ∼s*
Note: The above part of the prompt does not include the sections of the prompt which provide the claim and evidences on which to predict or the instructions for formatting the response. These were omitted above for brevity as they are the same across all prompt variations used during prompt development.

In order to maximize the predictions of GPT-4 on the test set, we investigated augmenting the baseline prompts in several ways. The different prompt variations we evaluated were:

1. *10 Examples*: Including more examples (10 instead of 2 per class) in the few-shot examples.
2. *Focus Numerical*: Directing the model to focus on the numerical aspects (to determine whether the model was getting distracted by wording or unimportant details rather than focusing on the numerical aspects of the claim).
3. *Focus Claim Decomp*: Directing the model to decompose the claim into sub-claims on its own (to encourage claim-decomposition reasoning in the model).
4. *Annotated Examples*: Providing human-produced annotations for each of the few-shot examples explaining why it received the label it did.
5. *Baseline + Reasoning*: Baseline prompt with 2 examples per class using GPT-4o-mini instead of GPT-4 Turbo, reasoning_effort = medium (to leverage reasoning capabilities).
6. *10 Examples + Reasoning*: 10 examples per class, using GPT-4o-mini.

To test these prompts, we created a development test set of the first 100 examples from the validation set and calculated the F1 scores of predictions generated with each prompt variation on this test set. This smaller test set was created to reduce cost and the number of calls to the OpenAI API.

To produce the best GPT-4 predictions in the final ensemble, we selected the prompt variation that had the highest F1 score in this development test set. We found the best performance using GPT-4 Turbo with 10 examples per class in the few-shot examples. While we found in testing that increasing the number of examples provided improved performance, for the final predictions on the test set we used 3 examples per class rather than 10 to reduce the token size of the prompt and associated cost while still including an increased number of examples over our baseline of 2.

**Confidence Self-Rating Prompt**: In addition to the optimized few-shot prompt we developed for the final ensemble (baseline prompt + 3 examples per class), we further prompted the model to rank its confidence in its prediction on a scale of 1 to 5 where 1 indicated the model was unsure of its predicted

label and 5 meant the model was completely sure its prediction was correct. The complete prompt used to make predictions on the final test set was thus:

---

*You are a numerical fact-checking assistant. For the given numerical claim and evidence, please decide if the claim is True (2), False (0), or Half True/False (conflicting) (1). You will be provided with the claim, the claim broken down into its sub-questions, and the top 5 documents which provide evidence/data. Your task is to evaluate the claim based on this provided data.*
*CLAIM (to be evaluated by you): < < < [claim] > > >*
*SUB-QUESTIONS (that the claim has been broken down into): < < < [sub-questions] > > >*
*TOP 5 EVIDENCES (that you will be evaluating the claim on) < < < [first 5 top-k evidences] > > >*
*When you are answering, please also rank your confidence in your evaluation of the claim on a scale of 1-5 where 1 means there is some unclarity in the claim or evidences and you are not sure of your evaluation and 5 means you are 100% sure you are correct in your evaluation"*
*Please respond with either 0, 1, or 2 based on the labels where 0 means you are classifying the claim as false, 1 means you are classifying the claim as half true/false (i.e., conflicting), and 2 means you are classifying the claim as true*
*When formatting your answer, please provide your verdict (as a number 0-2 corresponding to false, conflicting, or true) and your confidence rating (number 1-5). i.e., your final answer should have the format (0, 5) which would correspond to a verdict of 0 (false) and a confidence rating of 5 (completely sure). Please ONLY answer with this tuple and do not include anything else beyond this tuple in your response*
*For help analyzing the claim, please reference the following examples, separated by ∼s*
*[few shot examples: 3 examples from every class]*

---

Thus all predictions made by GPT-4 on the final test set had a self-confidence rating associated with the predicted label.

### 3.4. Confidence Self-Rating Analysis

As part of our model ensemble development, we sought to investigate whether GPT-4 had a sense of its own correctness, i.e., whether the model can determine when it has clearly understood the claim and evidences and can make a correct prediction for the verification of this claim and thus return a high self-confidence rating. Conversely, we also examined whether the model can determine when it is unsure of the predicted label, perhaps even guessing, meaning it is much more likely to be incorrect and thus return a lower self-confidence rating.

To determine whether the self-confidence ratings are indicative of the correctness of the model's prediction on that given example, we evaluated how performance changed as the confidence ratings increased on the predictions made on the development test set by GPT-4o mini using the prompt with 10 examples per class label in the few-shot examples. We compared the performance on subsets of the test set for which the model indicated a confidence of at least a certain value (i.e., the performance on only examples for which the self-confidence rating was greater than or equal to 1, 2, 3, 4, and 5, respectively).

Additionally, we evaluated whether the confidence ratings could be used as a predictor of accuracy. Using the same set of predictions, actual values, and confidence self-ratings, we fitted a logistic regression model to predict correctness (1 or 0 corresponding to whether predicted label == actual label) from the confidence self-ratings on the development test set. From this model we extracted the coefficient on the confidence predictor to determine the strength of the relationship between confidence and accuracy and we calculated the p-value using a permutation test.

### 3.5. Voting Mechanism for Ensemble

To determine the final predicted label, we used the predictions from both the fine-tuned Llama model as well as the GPT-4 model, using a voting mechanism based on GPT self-confidence ratings to determine

the label when the model predictions disagreed. We established a confidence rating threshold, in which the final predicted label was set to the prediction from GPT-4 when the self-confidence rating was greater than or equal to the threshold and the prediction from Llama when the confidence rating was below the threshold. For the final voting mechanism, we set this threshold to 5 to ensure that ties were only decided in favor of GPT predictions when GPT was highly confident the corresponding prediction was correct. For improved efficiency, we only computed Llama predictions on examples for which GPT did not return a self-confidence rating of 5 (as the voting mechanism would always return GPT's predicted label for these, whether or not the prediction of Llama agrees).

# 4. Results

## 4.1. Model Results

**Fine-tuned Llama**: The macro average F1 score of only the fine-tuned Llama model (fine-tuned on data excluding the hold-out set we created) on the hold-out test set of 1301 examples was 0.594 and the overall accuracy was 0.685 [Table 1]. Class-specific F1 scores and accuracies are shown in Table 1 and Figure 1. F1 scores and accuracy were the highest in the *False* class (0.822, and 0.858 respectively) with performances in the *True* class (0.590, and 0.6382) and Conflicting class (0.371, and 0.310) being lower. These results suggest a bias toward negative predictions, likely caused by a class imbalance with *False* making up 58% of the dataset. Figure 1 reflects this same bias toward *False* predictions, showing that the model frequently misclassified both *Half* and *True* claims as *False*.

**Table 1**
The F1 scores and accuracy of the fine-tuned Llama model for each label class and overall accuracy. Accuracy was computed over all examples with a certain true label (shown in the first column).

| **Fine-tuned LlaMA model results per class** | | | |
|---|---|---|---|
| **F1** | | **Accuracy** | |
| **F1 Metric** | **F1 Score** | **Class** | **Accuracy** |
| Macro Avg F1 | 0.594 | All Data | 0.685 |
| True F1 | 0.590 | True | 0.6382 |
| False F1 | 0.822 | False | 0.858 |
| Conflicting F1 | 0.371 | Conflicting | 0.310 |

**GPT-4 with Few-Shot Prompting**: The macro average F1 scores and accuracies of different prompt variations on the development test set we created are shown in Table 2. The best results were achieved with the *10 examples* prompt (F1: 0.550, accuracy: 0.62). For both GPT-4 Turbo and GPT-4o mini, increasing the number of examples from 2 to 10 per class in the few-shot prompt led to an improvement in results (F1: 0.520 −> 0.550, accuracy: 0.56 −> 0.62 for GPT-4 Turbo and F1: 0.382 −> 0.394, accuracy: 0.52 −> 0.55 for GPT-4o mini). Interestingly, instructional prompts that encouraged structured reasoning or numerical focus did not consistently improve accuracy, and in some cases (e.g., *Focus Numerical*) reduced it. GPT-4 Turbo outperformed GPT-4o mini, likely due to Turbo being a larger model that was thus able to better leverage few-shot examples.

## 4.2. Confidence Rating with GPT-4

To understand whether state-of-the-art powerful LLMs with reasoning capabilities have an understanding of their own corectness, we evaluated how effective confidence self-ratings were for GPT-4, looking specifically at how results changed as confidence ratings increased. The macro average F1 scores on only samples for which the model gave a self-confidence ranking of at least X (where X went from 1 to 5) are shown in Figure 2.

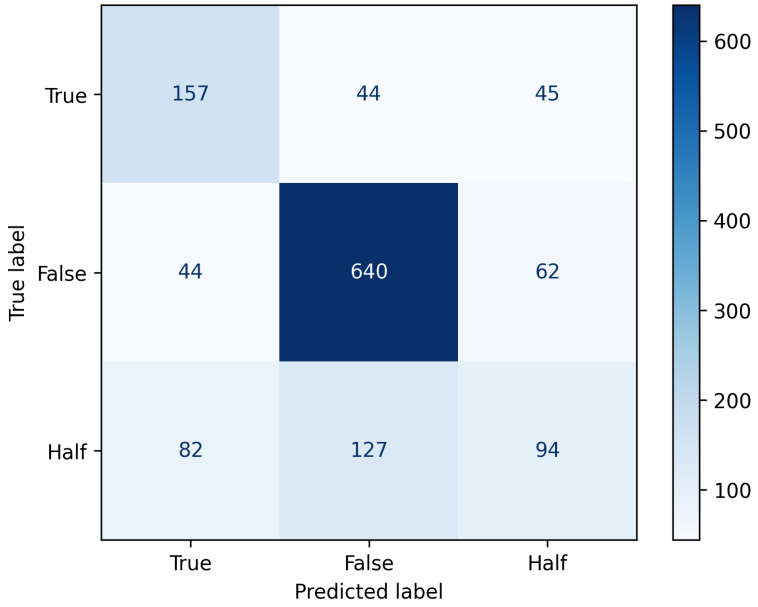


**Figure 1:** Confusion matrix of the fine-tuned Llama 2-7B model on the 1301-example hold-out test set.

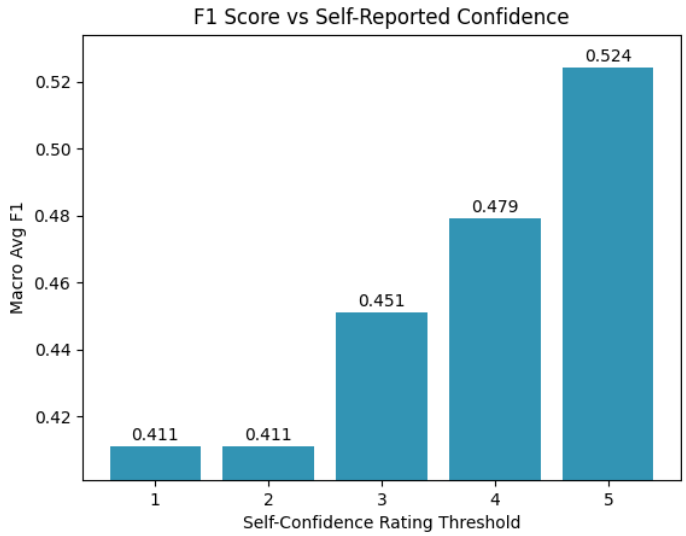


**Figure 2:** Macro average F1 score of GPT-4 on only samples for which the self confidence rating was greater than or equal to a threshold (ranging from 1-5). Threshold of 1 included all samples (GPT-4o-mini with prompt containing 10 examples per class in the few-shot examples).

These results show that except for the increase in confidence from 1 to 2, performance consistently increased as model confidence increased. Consistent increases in F1 score can be seen between the subset of the data for which the model gave a self-confidence ranking of 2 to 5. Overall, the F1 scores increased over 25%, going from 0.411 on the entire dataset (confidence rankings $\geq 1$) to 0.524 on only the examples for which the confidence ranking was 5 showing that model self-confidence is a strong indication of higher performance.

The results from the logistic regression model trained to predict accuracy from confidences are shown in [Table 3]. The coefficient on the confidence predictor was 0.3171, showing a moderately strong positive relationship indicating that as model confidence increases, the prediction is more likely to be correct. The p-value (0.0410) $< 0.05$, showing that the confidence does have statistically significant predictive power over the accuracy of predictions.

**Table 2**
Performance of GPT-4 on the development test set. Predictions were generated with different prompt variations and results of these predictions are reported. The prompt variation, prompt text, macro average F1, and accuracy are shown. For the prompt text, the modifications to the baseline prompt (defined in section 3.3) are shown. The actual prompt used to generate the predictions includes the claim and evidences as well as formatting instructions which are omitted from the table for brevity.

| **GPT-4 Macro Avg F1 Scores and Accuracy across Prompt Variations** | | | |
|---|---|---|---|
| **Prompt Variation** | **Prompt text** | **F1** | **Accuracy** |
| **Baseline Prompt** | *[baseline prompt] + [few shot examples: 2 examples from every class]* | **0.520** | **0.56** |
| **(1) *10 Examples*** | *[baseline prompt] + [few shot examples: 10 examples from every class]* | **0.550** | **0.62** |
| **(2) *Focus Numerical*** | *[baseline prompt] + When you are evaluating the claims please focus on the NUMERICAL aspects of the claim (i.e., are the numerical parts of the claim correct based on the evidence) rather than other aspects of the claim + [few shot examples: 2 examples from every class]* | **0.440** | **0.52** |
| **(3) *Focus Claim Decomp*** | *[baseline prompt] + When you are evaluating the claims, note that the claims often have several parts and at least one numerical part. Break down the claim into all its parts, evaluate each one and if there is a numerical part of the claim focus on the numbers in the claim and evidences + [few shot examples: 2 examples from every class]* | **0.521** | **0.59** |
| **(4) *Annotated Examples*** | *[baseline prompt] + [few shot examples: 2 examples from every class with each example including a human-created annotation explaining the reason it was classified as its label]* | **0.532** | **0.57** |
| **(5) *Baseline + Reasoning*** | *[baseline prompt] + [few shot examples: 2 examples from every class] (prediction generated with GPT-4o-mini)* | **0.382** | **0.52** |
| **(6) *10 Examples + Reasoning*** | *[baseline prompt] + [few shot examples: 10 examples from every class] (prediction generated with GPT-4o-mini)* | **0.394** | **0.55** |

**Table 3**
Results of fitting a logistic regression model to predict correctness from confidence self-ratings. The coefficient on the confidence predictor is shown, and p-values were calculated using a permutation test. (GPT-4o-mini with prompt containing 10 examples per class in the few-shot examples).

| **Logistic Regression Testing for Relationship Between Model Confidence and Prediction Accuracy** | |
|---|---|
| **Logistic Regression Coefficient** | **P-value** |
| 0.3171 | 0.0410 |

Overall, the results from the confidence self-ratings analysis show that confidence is indicative of performance and that on samples where the model reports a higher confidence, it is more likely to be correct. These results support the use of confidence self-ratings as a decider in a voting mechanism between GPT-4 and fine-tuned Llama 2 in our ensemble.

## 4.3. Ensemble Results

As described in section 3.1 and 3.5, the final ensemble combined the individually optimized models (fine-tuned Llama and GPT-4 with few-shot prompting) via a voting mechanism. The Llama model used in the final ensemble was fine-tuned on all available train and validation data. For GPT-4, the prompt variation with the highest performance was the baseline prompt with more examples per class included in the few-shot examples (run on GPT-4 Turbo) [Table 2]. Thus, we used this prompt variation for the GPT-4 model in the final ensemble. However, we did make one change to the prompt, namely

changing the number of examples from 10 to 3 to reduce token counts for API calls while still gaining the advantages of more examples per class in the prompt. The voting mechanism we used in the ensemble set the self-confidence rating threshold to 5 (i.e., disagreements were only decided in favor of GPT when the self-confidence rating was 5). Due to the large gap observed between performance with thresholds of 4 and 5 in testing [Figure 2], this ensured that GPT only acted as a decider when it had high confidence and thus was also the most likely to be correct.

**Task 3 Evaluation Results**: The results of our ensemble approach on the test set are shown in Table 4. The final Macro average F1 score of the ensemble was 0.4857. The ensemble performed the best on *False* examples with a False F1 score of 0.7820. The True F1 score was slightly lower (0.5090) and the models struggled the most with the *Conflicting* class, with the Conflicting F1 score being significantly lower, 0.1661. These results show that the ensemble observed the same effect as the fine-tuned Llama model alone with better performance on *False* examples than on *True* and *Conflicting*, likely reflecting the higher proportion of *False* examples in the datasets. While the overall performance was moderate, placing 7th on the leaderboard, the results suggest that the ensemble was a reasonably effective strategy, especially for handling clearer cases with high model confidence.

**Table 4**
The results of the final ensemble approach on the task 3 test set. The position in the leader board is shown in parentheses after the F1 score

| **Final Ensemble Evaluation Results** | | | |
|---|---|---|---|
| **Macro Avg F1** | **True F1** | **Conflicting F1** | **False F1** |
| 0.4857 (7) | 0.5090 (3) | 0.1661 (10) | 0.7820 (5) |

The performances, both F1 score (macro average F1) and accuracy (count correct / total count) of the ensemble method and each method individually are shown in Table 5. Note that the Llama results are only computed over the subset of total test set examples for which we produced Llama predictions on the test set (i.e., examples for which GPT did not give a self-confidence rating of 5) as described in section 3.5. Note that for 33 examples on which GPT-4 gave a self-confidence rating of < 5, the Llama model failed to produce an outputted label so the final prediction was set to the GPT prediction. These 33 examples are thus also not included in the calculation of Llama individual results.

These results show that the ensemble method and GPT-4 performed similarly, with the ensemble out-performing GPT-4 in terms of accuracy. Ensemble accuracy (0.6601) was higher than both GPT-4 accuracy (0.6556) and Llama 2 accuracy (0.4815) and the ensemble F1 score (0.4948) was slightly lower than the GPT-4 F1 score (0.5035) and greater than Llama 2 (0.4368). As the majority of the predictions in the test set had a confidence rating of 5, GPT-4 performance was close to ensemble performance. The overall ensemble still had a higher accuracy than both models individually, showing that ensemble based on self-confidence ratings can effectively leverage the complementary strengths of both models.

**Table 5**
Macro average F1 scores and accuracies of the ensemble and GPT-4 and Llama 2 individually on the test set. Results are computed over the 3510 available gold standard test claims. Llama results are computed only over the examples for which GPT-4 did not return a self-confidence rating of 5 (108 examples).

| **Ensemble and Individual Model Results** | | | | | |
|---|---|---|---|---|---|
| **F1 Score** | | | **Accuracy** | | |
| **Ensemble** | **GPT-4** | **Llama 2** | **Ensemble** | **GPT-4** | **Llama 2** |
| 0.4948 | 0.5035 | 0.4368 | 0.6601 | 0.6556 | 0.4815 |

## 5. Discussion and Conclusions

In this task we investigated using an ensemble approach to combine predictions of two LLMs: Llama 2-7B fine-tuned on training and validation data and GPT-4 with few-shot prompting. We developed a

voting mechanism based on self-confidence rankings reported by GPT. This confidence-based voting helped to mitigate errors without requiring additional models or complex calibration schemes.

The results showed that the ensemble model approach was moderately effective, achieving a Macro Average F1 score of 0.4857 on the test set in evaluation. We found that the ensemble performed particularly well on *False* examples and performed competitively on *True* examples, placing 3rd in the leader board for True F1. The model struggled the most with *Conflicting* data likely due to the less clear nature of these claims and its lower representation in training and validation datasets. This suggests that further approaches would be needed to improve the performance of both models, especially on the more confusing *Conflicting* claims. These results suggest that other approaches, particularly those incorporating explicit reasoning may be helpful to improve performance on claims that are ambiguous or confusing.

We also explored using a numerically-focused model trained on the FinQA dataset to improve performance on complex numerical claims. This approach involved experimenting with various input formats, including selectively incorporating *gold evidence*—annotated text spans from financial documents that ideally support the answer. We tested strategies like conditional inclusion of gold evidence only when the model previously answered incorrectly, aiming to guide learning without overfitting. Additionally, we attempted to ensemble this numerical model with a general-purpose model like Llama using majority voting. However, despite these efforts, accuracy improvements were minimal and did not meaningfully outperform the baseline.

We conducted an analysis on reported self-confidence for numerical claim verification and found that higher reported self-confidence did indicate higher likelihood of correctness and higher accuracy. Thus, we showed that for the task of numerical claim verification, state-of-the-art LLMs have the ability to judge their own predictions and report accurate confidences in these predictions.

Our evaluation also demonstrated that the ensemble method we developed had a higher accuracy than each constituent model individually on the test set. We show that integrating models via self-confidence rankings can lead to performance improvements. These results highlight the value of combining high-confidence predictions from large general-purpose models such as GPT-4 with specialized models like fine-tuned Llama 2 on examples where confidence is lower. This approach can be especially valuable in domains such as numerical claim verification where errors are common and can be difficult to detect.

Overall, we showed that in the context of numerical-claim verification, LLMs are capable of introspective reliability estimation, producing self-confidence ratings that are indicative of model performance. By leveraging these confidence ratings in combination with a fine-tuned Llama model, we show that LLM-based verification systems can be made more robust for factual reasoning tasks.

## Declaration on Generative AI

The authors have not employed any Generative AI tools.

## References


[1] Z. Guo, M. Schlichtkrull, A. Vlachos, A survey on automated fact-checking, Transactions of the Association for Computational Linguistics 10 (2022) 178–206.

[2] N. Sagara, Consumer understanding and use of numeric information in product claims, Ph.d. dissertation, University of Oregon, 2009.

[3] R. Aly, Z. Guo, M. Schlichtkrull, J. Thorne, A. Vlachos, C. Christodoulopoulos, O. Cocarascu, A. Mittal, Feverous: Fact extraction and verification over unstructured and structured information, arXiv preprint arXiv:2106.05707 (2021). arXiv:2106.05707.

[4] V. Venktesh, A. Anand, A. Anand, V. Setty, Quantemp: A real-world open-domain benchmark for fact-checking numerical claims, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery,

New York, NY, USA, 2024, pp. 650–660. URL: https://doi.org/10.1145/3626772.3657874. doi:10.1145/3626772.3657874.

[5] F. Alam, J. M. Struß, T. Chakraborty, S. Dietze, S. Hafid, K. Korre, A. Muti, P. Nakov, F. Ruggeri, S. Schellhammer, V. Setty, M. Sundriyal, K. Todorov, V. V., The clef-2025 checkthat! lab: Subjectivity, fact-checking, claim normalization, and retrieval, in: C. Hauff, C. Macdonald, D. Jannach, G. Kazai, F. M. Nardini, F. Pinelli, F. Silvestri, N. Tonellotto (Eds.), Advances in Information Retrieval, Springer Nature Switzerland, Cham, 2025, pp. 467–478.

[6] F. Alam, J. M. Struß, T. Chakraborty, S. Dietze, S. Hafid, K. Korre, A. Muti, P. Nakov, F. Ruggeri, S. Schellhammer, V. Setty, M. Sundriyal, K. Todorov, V. Venktesh, Overview of the CLEF-2025 CheckThat! Lab: Subjectivity, fact-checking, claim normalization, and retrieval, in: J. Carrillo-de Albornoz, J. Gonzalo, L. Plaza, A. García Seco de Herrera, J. Mothe, F. Piroi, P. Rosso, D. Spina, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Sixteenth International Conference of the CLEF Association (CLEF 2025), 2025.

[7] V. Venktesh, V. Setty, A. Anand, M. Hasanain, B. Bendou, H. Bouamor, F. Alam, G. Iturra-Bocaz, P. Galuščáková, Overview of the CLEF-2025 CheckThat! lab task 3 on fact-checking numerical claims, in: G. Faggioli, N. Ferro, P. Rosso, D. Spina (Eds.), Working Notes of CLEF 2025 - Conference and Labs of the Evaluation Forum, CLEF 2025, Madrid, Spain, 2025.

[8] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, et al., A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity, arXiv preprint arXiv:2302.04023 (2023).

[9] A. Simhi, I. Itzhak, F. Barez, G. Stanovsky, Y. Belinkov, Trust me, i'm wrong: High-certainty hallucinations in llms, arXiv preprint arXiv:2502.12964 (2025). URL: https://arxiv.org/abs/2502.12964.

[10] B. Botnevik, E. Sakariassen, V. Setty, Brenda: Browser extension for fake news detection, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 2117–2120. URL: https://doi.org/10.1145/3397271.3401396. doi:10.1145/3397271.3401396.

[11] Z. Guo, M. Schlichtkrull, A. Vlachos, A survey on automated fact-checking, Transactions of the Association for Computational Linguistics 10 (2022) 178–206. URL: https://doi.org/10.1162/tacl_a_00454. doi:10.1162/tacl_a_00454. arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00454/1987018/tacl_a

[12] X. Zeng, A. S. Abumansour, A. Zubiaga, Automated fact-checking: A survey, Language and Linguistics Compass 15 (2021) e12438.

[13] J. Guan, J. Dodge, D. Wadden, M. Huang, H. Peng, Language models hallucinate, but may excel at fact verification, arXiv preprint arXiv:2310.14564 (2023). URL: https://doi.org/10.48550/arXiv.2310.14564, accepted in NAACL 2024.

[14] Y.-J. Sun, S. Dey, D. Hakkani-Tür, G. Tür, Confidence estimation for llm-based dialogue state tracking, arXiv preprint arXiv:2409.09620 (2024). URL: https://arxiv.org/abs/2409.09620.

[15] B. Meskó, Prompt engineering as an important emerging skill for medical professionals: Tutorial, J Med Internet Res 25 (2023) e50638. URL: https://www.jmir.org/2023/1/e50638. doi:10.2196/50638.

[16] J. Wei, J. Wei, Y. Tay, D. Tran, A. Webson, Y. Lu, X. Chen, H. Liu, D. Huang, D. Zhou, T. Ma, Larger language models do in-context learning differently, 2023. URL: https://arxiv.org/abs/2303.03846. arXiv:2303.03846.

[17] S. Ott, K. Hebenstreit, V. Liévin, et al., Thoughtsource: A central hub for large language model reasoning data, Scientific Data 10 (2023) 528. URL: https://doi.org/10.1038/s41597-023-02433-3. doi:10.1038/s41597-023-02433-3.

[18] O. Razuvayevskaya, B. Wu, J. A. Leite, F. Heppell, I. Srba, C. Scarton, K. Bontcheva, X. Song, Comparison between parameter-efficient techniques and full fine-tuning: A case study on multilingual news article classification, PLOS ONE 19 (2024) e0301738. URL: https://doi.org/10.1371/journal.pone.0301738. doi:10.1371/journal.pone.0301738.

[19] N. Ding, Y. Qin, G. Yang, et al., Parameter-efficient fine-tuning of large-scale pre-trained lan-

guage models, Nature Machine Intelligence 5 (2023) 220–235. URL: https://doi.org/10.1038/s42256-023-00626-4. doi:10.1038/s42256-023-00626-4.

[20] N. Abou Baker, D. Rohrschneider, U. Handmann, Parameter-efficient fine-tuning of large pretrained models for instance segmentation tasks, Machine Learning and Knowledge Extraction 6 (2024) 2783–2807. URL: https://doi.org/10.3390/make6040133. doi:10.3390/make6040133.

[21] Y. Huang, X. Feng, B. Li, Y. Xiang, H. Wang, T. Liu, B. Qin, Ensemble learning for heterogeneous large language models with deep parallel collaboration, in: A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, C. Zhang (Eds.), Advances in Neural Information Processing Systems, volume 37, Curran Associates, Inc., 2024, pp. 119838–119860. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/d8a6eb79f8ccaacbe7198a5caf3a0323-Paper-Conference.pdf.

[22] H. Yang, M. Li, H. Zhou, Y. Xiao, Q. Fang, R. Zhang, One llm is not enough: Harnessing the power of ensemble learning for medical question answering, medRxiv (2023). URL: https://doi.org/10.1101/2023.12.21.23300380. doi:10.1101/2023.12.21.23300380, preprint.