ZEPHYRUS: AN AGENTIC FRAMEWORK FOR WEATHER SCIENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Foundation models for weather science are pre-trained on vast amounts of structured numerical data and outperform traditional weather forecasting systems. However, these models lack language-based reasoning capabilities, limiting their utility in interactive scientific workflows. Large language models (LLMs) excel at understanding and generating text but cannot reason about high-dimensional meteorological datasets. We bridge this gap by building a novel agentic framework for weather science. Our framework includes a Python code-based environment for agents (ZEPHYRUSWORLD) to interact with weather data, featuring tools like an interface to WeatherBench 2 dataset, geoquerying for geographical masks from natural language, weather forecasting, and climate simulation capabilities. We design ZEPHYRUS, a multi-turn LLM-based weather agent that iteratively analyzes weather datasets, observes results, and refines its approach through conversational feedback loops. We accompany the agent with a new benchmark, ZEPHYRUSBENCH, with a scalable data generation pipeline that constructs diverse question-answer pairs across weather-related tasks, from basic lookups to advanced forecasting, extreme event detection, and counterfactual reasoning. Experiments on this benchmark demonstrate the strong performance of ZEPHYRUS agents over text-only baselines, outperforming them by up to 35 percentage points in correctness. However, on harder tasks, ZEPHYRUS performs similarly to text-only baselines, highlighting the challenging nature of our benchmark and suggesting promising directions for future work.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities across diverse scientific domains (Birhane et al., 2023), revolutionizing fields from drug discovery (Zheng et al., 2024; Wu et al., 2024b) and materials science (Lei et al., 2024; Jablonka et al., 2023) to network biology (Theodoris et al., 2023). These models excel at processing textual content such as scientific literature, source code (Jiang et al., 2024), and structured data tables (Zhang et al., 2024). However, their application to domains requiring reasoning over high-dimensional numerical data remains limited (Wang et al., 2024).

Meteorology offers a compelling yet challenging case study, as combining natural language reasoning with complex atmospheric data has the potential to greatly advance weather research. Weather prediction is a critical scientific challenge, with profound implications spanning agriculture, disaster preparedness, transportation, and energy management (Alley et al., 2019). The field has witnessed remarkable progress through machine learning approaches, with foundation models (Nguyen et al., 2023; Kurth et al., 2023; Lam et al., 2023; Bi et al., 2023; Nguyen et al., 2024) now achieving state-of-the-art performance in medium-range forecasting, often surpassing traditional physics-based numerical simulations (Molteni et al., 1996; Bauer et al., 2015). However, current weather models operate exclusively on structured numerical datasets such as reanalysis data, cannot incorporate valuable alternative modalities like textual weather bulletins or field station reports, and crucially, lack interactive natural language interfaces for querying or reasoning.

Weather science workflows require substantial technical expertise to orchestrate complex ecosystems of tools, datasets, and models. Researchers must navigate disparate data sources, integrate outputs from multiple forecasting systems, combine observational datasets with model predictions, and

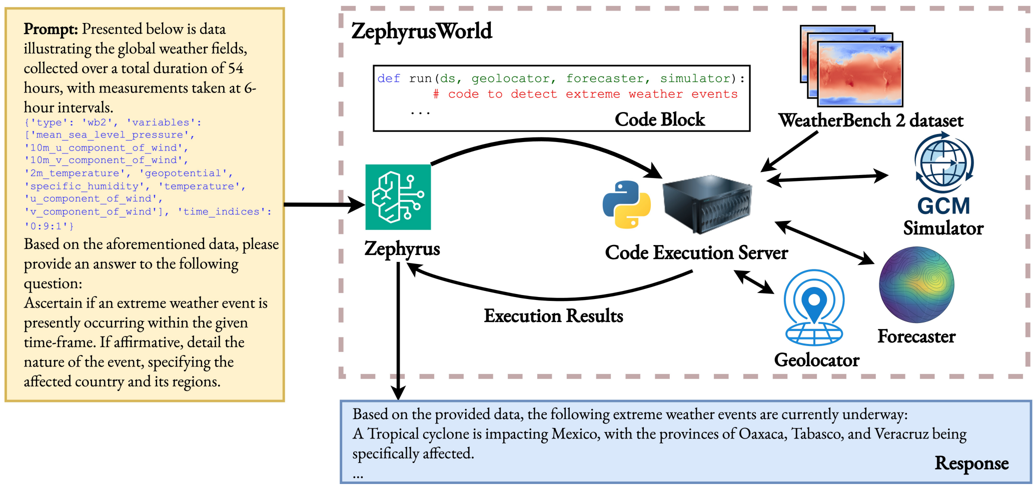

Figure 1: **Overview**: We develop ZEPHYRUS, an agentic framework for weather science. Given a query, the LLM-based agent ZEPHYRUS writes a code block which is sent to the code execution server. The server orchestrates several tools to execute the code block and returns the execution results to the agent. The agent either decides to execute more code to refine its output or respond back to the user. Refer to Appendix A.8 for the full prompt.

coordinate between different computational environments and APIs. This dependency on extensive technical knowledge creates barriers for domain experts, limiting broader participation in weather science. Traditional meteorological workflows therefore require expert interpretation to translate computational outputs into actionable insights, increasing costs and limiting their utility in human-in-the-loop decision-support systems.

Multimodal LLMs can handle data from diverse modalities and offer a potential pathway to address these challenges. Models capable of jointly processing text with images (Wang et al., 2022; Alayrac et al., 2022; Li et al., 2022; Liu et al., 2023c), video (Zhao et al., 2022; Zhang et al., 2023; Cheng et al., 2024; Lin et al., 2024; Zhang et al., 2025), and audio (Chu et al., 2023; Défossez et al., 2024; Wu et al., 2024a; 2025; Doh et al., 2025; Ghosh et al., 2025) have shown impressive cross-modal reasoning abilities. Yet atmospheric data poses unique challenges: its spatiotemporal, multi-channel structure is fundamentally different from conventional modalities, requiring specialized approaches for effective integration with language models. Initial attempts to bridge this gap have shown promise but remain limited in scope. Early vision-language approaches to meteorology (Chen et al., 2024a; Li et al., 2024; Ma et al., 2024) have focused on narrow applications like extreme weather prediction using restricted variable subsets, falling short of general-purpose meteorological reasoning. More recent multimodal weather-language models (Varambally et al., 2025) demonstrate the potential of this direction but still fail to match established baselines across many important meteorological tasks. This persistent gap highlights a fundamental challenge: despite significant progress in both weather foundation models and LLMs, no existing system successfully unifies meteorological data with natural language reasoning for broad, interactive scientific applications.

We address this challenge by first introducing an agentic environment that enables LLMs to interact programmatically with meteorological data and models. We setup ZEPHYRUSWORLD, a comprehensive execution environment that exposes weather-focused capabilities through easy-to-use Python APIs. The system includes interfaces to the WeatherBench 2 dataset (Rasp et al., 2024), geo-query functionality for translating between coordinates and named locations, state-of-the-art forecasting models (Nguyen et al., 2024), and physics-based simulators. A FastAPI backend parallelizes code execution from LLM-generated queries.

We then develop two code-generating systems of increasing sophistication within this agentic framework. ZEPHYRUS-DIRECT generates Python code in a single step to solve weather problems directly (Gao et al., 2023). ZEPHYRUS-REFLECTIVE employs an iterative execution–refinement (Yao et al., 2023b): it executes code to manipulate weather data, analyzes the results, and refines both code and output before providing a final answer. Both approaches can automatically detect and correct errors produced during code execution. Figure 1 gives an overview of our entire agentic pipeline.

To systematically evaluate these approaches, we construct ZEPHYRUSBENCH, a comprehensive benchmark built on ERA5 reanalysis data (Hersbach et al., 2020) from WeatherBench 2 (Rasp et al., 2024). The benchmark combines human-authored and semi-synthetic tasks spanning 2158 question–answer pairs across 46 distinct tasks. Tasks range from basic data lookups and forecasting to challenging research problems involving extreme event detection, forecast report generation, and prediction and counterfactual analysis. We also implement robust evaluation schemes to assess the scientific accuracy of all generated answers across diverse meteorological reasoning tasks. We summarize our key contributions below.

- We develop ZEPHYRUSWORLD, an agentic environment providing unified Python APIs for meteorological data, forecasting models, and climate simulation tools.
- We introduce two code-generating systems that leverage ZEPHYRUSWORLD: ZEPHYRUS-DIRECT for single-step code generation and ZEPHYRUS-REFLECTIVE for iterative execution-refinement workflows to solve open-ended meteorological problems.
- We curate ZEPHYRUSBENCH, a challenging weather reasoning benchmark with 2062 question-answer pairs across 46 meteorological task types.
- Our evaluation shows that LLM agents achieve encouraging results on the benchmark, suggesting that they can be effective assistants to weather scientists.

2 RELATED WORK

Weather Foundation Models. Neural network-based weather forecasting systems (Lam et al., 2023; Price et al., 2025; Bi et al., 2023; Pathak et al., 2022; Nguyen et al., 2023; Bodnar et al., 2024; Nguyen et al., 2024) have revolutionized meteorological prediction by demonstrating superior performance compared to conventional physics-based approaches (Molteni et al., 1996) while being significantly more computationally efficient. Nevertheless, these architectures are predominantly trained for forecasting. In particular, they do not support conversational interfaces or cross-domain reasoning capabilities.

Agentic frameworks for scientific discovery Agentic frameworks implement the perceive–reason–plan–act loop by pairing LLMs with tools, memory, and feedback to pursue long-horizon goals. Core patterns include interleaving reasoning with tool calls (ReAct (Yao et al., 2023a)), self-critique with episodic memory (Reflexion Shinn et al. (2023)), and self-supervised learning of API use (Toolformer (Schick et al., 2023)). General-purpose libraries such as AutoGen provide a standard interface for multi-agent conversation and tool invocation, making these patterns reusable across tasks (Wu et al., 2024c).

In many scientific applications, these frameworks appear as domain agents and self-driving labs. In chemistry, ChemCrow couples an LLM controller with a curated set of expert tools for synthesis and analysis (Bran et al., 2024), while Coscientist integrates retrieval, code execution, and laboratory APIs to plan and run experiments end-to-end (Boiko et al., 2023). Biomedical agents extend the approach across literature, databases, and analysis workflows (e.g., Biomni (Huang et al., 2025)). Despite these advances across multiple scientific domains, weather science remains largely unexplored territory for agentic approaches.

General-Purpose Vision-Language Models. Multi-modal vision language models (Li et al., 2021; Alayrac et al., 2022; Li et al., 2022; 2023; Liu et al., 2023c;b;a; 2024) demonstrate strong visual reasoning capabilities on general-purpose evaluation benchmarks. However, adapting these models for applications in weather science presents considerable difficulties. Standard VLM architectures assume RGB visual inputs and exhibit weaknesses in quantitative analytical tasks. Meteorological data presents fundamentally different challenges through high-dimensional, structured atmospheric measurements requiring specialized integration approaches for language model compatibility. While weather-language hybrid models (Varambally et al., 2025) seem promising, they underperform relative to domain-specific baselines across critical meteorological applications.

Multimodal Weather Datasets. Recent research has developed several multimodal frameworks that combine weather observations with textual information. These include the Terra collection (Chen et al., 2024b), which integrates geographical imagery with descriptive text for general earth observation, and ClimateIQA (Chen et al., 2024a), which focuses on extreme weather detection through wind measurement analysis. Similarly, WeatherQA (Ma et al., 2024) specializes in severe weather interpretation using remote sensing data and expert commentary, while CLLMate (Li et al.,

2024) connects media reports with ERA5 observations for weather event classification. Despite these valuable contributions, existing frameworks are narrow in scope. They concentrate on narrow applications or utilize only small subsets of atmospheric variables. This approach overlooks a fundamental characteristic of atmospheric dynamics: weather systems involve complex multi-scale interactions across numerous meteorological parameters. To address these limitations, our benchmark incorporates diverse weather reasoning tasks, both human-implemented and semi-synthetically generated, that span across most WeatherBench2 data channels.

3 ZEPHYRUS: AN AGENTIC FRAMEWORK FOR WEATHER SCIENCE

3.1 ZEPHYRUSWORLD: AN AGENTIC ENVIRONMENT FOR WEATHER SCIENCE

The fragmented nature of weather science tools makes it challenging for LLMs to effectively leverage them for scientific tasks. To address this, we introduce ZEPHYRUSWORLD, a comprehensive agentic environment that unifies weather science capabilities from diverse tools through a clean Pythonic interface. Given a question, we leverage LLMs' ability (Gao et al., 2023; Jimenez et al.) to generate Python code and execute it in a sandboxed environment. The output is then fed back to the model, along with any execution errors. We design high-level APIs for the tools for ease of use, and include documentation extracted from the docstrings in the models context at inference time.

The environment encompasses several essential weather science tools:

1. **WeatherBench 2 Data Indexer.** The environment provides the model access to the data through the `xarray` dataset interface.
2. **Geolocator.** This tool provides comprehensive geospatial functionality for weather data analysis. It handles forward geocoding (place names to coordinates) and reverse geocoding (coordinates to location names) using the Natural Earth dataset (Natural Earth, 2024). Key operations include finding geographic features at specific coordinates, retrieving boolean masks and area-weighted maps for regions, listing sublocations, and calculating geodistances. Built using `geopandas` and `shapely`, it maintains precomputed spatial caches for fast lookups.
3. **Forecaster.** We incorporate the Stormer model (Nguyen et al., 2024), a transformer-based neural weather prediction system trained on WeatherBench 2. We chose it for its strong performance at short to medium range forecasts while being orders of magnitude more efficient than traditional numerical models. Our implementation abstracts checkpoint loading and preprocessing, providing a simple interface to run forecasts from arbitrary atmospheric initial conditions and return outputs as `xarray` datasets.
4. **Simulator.** Our JAX-GCM simulator is an intermediate complexity atmospheric model built on NeuralGCM's dynamical core (Kochkov et al., 2024). It incorporates physical parameterizations from the SPEEDY Fortran model (Molteni et al., 1996), including radiation, moist physics (clouds and convection), and vertical and horizontal diffusion. We use the default T32 configuration (approximately $3.5°$ resolution) with 8 vertical layers. Built on JAX, we can run 5-day simulations in only $\approx 25s$ on an A100 GPU.

Code Execution Server. ZEPHYRUSWORLD requires a system capable of handling multiple weather analysis tasks simultaneously without resource conflicts. We implement a FastAPI-based server-client architecture that processes multiple weather analysis requests in parallel using resource pools to prevent conflicts between simultaneous executions. Each execution follows a strict protocol of acquiring resources, loading datasets, injecting tools, and executing code with timeout protection before returning outputs and errors to the client. More details are presented in Appendix A.1

3.2 THE ZEPHYRUS FAMILY OF WEATHER AGENTS

We design agentic systems that leverage ZEPHYRUSWORLD to solve complex meteorological tasks. Our approach constructs prompts containing comprehensive documentation of ZEPHYRUSWORLDtools, variable descriptions, units, and coordinate systems. The models generate Python functions using these tools to solve the given questions, which execute on ZEPHYRUSWORLD's code execution server. Any execution errors or timeouts are returned to the models, which regenerate code until the error is resolved. We implement two distinct systems that differ in their execution strategy and refinement approach. Both systems intentionally maintain simple designs to isolate and measure the agentic capabilities of LLMs for solving weather science problems.

ZEPHYRUS-DIRECT generates a complete Python solution in one attempt and reports the execution output as the final answer. This model runs the error-correction loop for a maximum of 5 times.

ZEPHYRUS-REFLECTIVE implements a multi-turn workflow that alternates between code generation and execution phases. The agent executes individual code blocks and receives the output as observations. The execution results are fed back to the LLM, which analyzes the observations and decides on the next step. This iterative process enables the model to assess the scientific plausibility of outputs, identify anomalies or mistakes in results, and refine subsequent code blocks to address logical errors. We run the interaction loop for a maximum of 20 times per question.

The complete prompts for both systems are presented in Appendix A.8.

4 ZEPHYRUSBENCH: A COMPREHENSIVE WEATHER BENCHMARK

Weather science problems require analyzing complex atmospheric patterns, modeling trends, and combining data from multiple sources. We introduce ZEPHYRUSBENCH, a comprehensive benchmark that evaluates how effectively LLMs can assist in real-world meteorological workflows. The benchmark comprises 46 distinct meteorological tasks with answers derived from curated weather reports and human-generated or verified code.

4.1 DATASET CURATION

We base our tasks around the ERA5 reanalysis dataset (Hersbach et al., 2020), specifically from WeatherBench 2 (Rasp et al., 2024). The dataset provides global atmospheric data from 1979 to 2022. We use 1.5° spatial resolution with 6-hourly temporal resolution.

The capabilities measured by our curated tasks range from basic data lookups and computations to more advanced problems involving forecasting, challenging research problems including extreme event detection, forecast report generation, prediction analysis, and counterfactual reasoning. We design tasks with increasing difficulty levels (Easy, Medium, Hard) based on the complexity of tool usage required to answer them, from simple single-step data queries to multi-step analytical workflows. Table A.2 provide an overview of the task types we implement as part of our benchmark.

For each task-type, we define natural language templates with placeholders such as location, variable, and time window. To create task-specific examples, these placeholders are filled by randomly sampling inputs, and the corresponding ground truth is computed deterministically using human-written or human-verified synthetic code applied to the raw ERA5 data. Figure 7 shows an example template, and a sample generated from it.

Using our framework, we construct a benchmark dataset comprising 2158 test samples spread across 46 tasks. For a detailed breakdown of dataset statistics, please refer to Appendix A.2. We provide more details about how the tasks are implemented in the subsequent sections.

4.1.1 HUMAN-GENERATED TASKS

The human-generated tasks span across the Easy and Hard difficulty levels and represent realistic meteorological queries curated in conjunction with a domain expert. For each task, a graduate student created a question template and wrote Python code to answer the query. Easy tasks focus on basic data retrieval operations like finding extrema, querying specific values, and identifying locations with particular weather conditions. Medium-difficulty tasks introduce forecasting elements, asking for future weather predictions at specific locations and times, and/or implementing complex data analysis pipelines. Hard tasks incorporate more complex analytical concepts such as anomaly detection relative to baselines and counterfactual scenario analysis. They demand comprehensive meteorological expertise and mirror real-world operational workflows. These include extreme weather event detection, comprehensive weather assessments, and generation of detailed forecast discussions that span regional to global scales. For instance, ENSO outlook reports require synthesizing complex interactions between multiple atmospheric and oceanic variables to produce coherent, scientifically grounded forecasts. We source the expert-generated weather discussion reports from several online

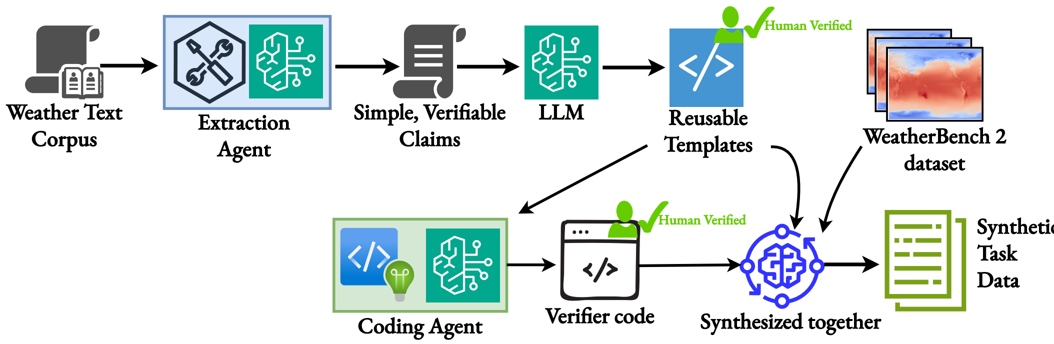

Figure 2: **Semi-synthetic task generation pipeline**: Weather-related texts are processed by a claim extraction agent to identify scientifically meaningful observational claims. Verified claims are transformed into reusable templates and manually reviewed. Code is generated by an LLM and verified by humans to validate each sample from a template against ERA5 meteorological data. We combine the verifier code with the templates and WeatherBench data to produce novel samples.

sources, such as the NOAA website[1] and IRI Seasonal Climate Forecasts/Outlooks[2]. For extreme weather event tasks, we use records from the EM-DAT international disaster database (Delforge et al., 2025), matching event entries by date and location to the ERA5 data.

4.1.2 Semi-synthetic task generation

To increase task diversity, we implement a semi-synthetic pipeline that transforms unstructured weather-related text into verifiable benchmark tasks. Figure 2 provides an overview of the procedure. The process begins with a claim extraction agent that analyzes weather texts from various sources, using an LLM to identify scientifically meaningful observational claims about weather phenomena. The agent focuses on quantifiable changes, trends, extremes, and relationships between variables.

These claims are then converted into question templates where we can substitute different locations, time periods, and weather variables to generate multiple benchmark examples from each original claim. For each template, an LLM writes a verification code block that can validate any instance generated from that template against the ERA5 data. This verification step ensures that the generated questions are not only linguistically coherent but also scientifically accurate when tested against actual meteorological observations. We generate multiple candidate instances from each template through this approach. Finally, we manually review them for scientific interest and code correctness. In this way, we generate 30 distinct human-validated synthetic task types.

We also include a semi-synthetic meteorological claim verification task to test whether models are capable of validating claims extracted from meteorological reports against the weather data. From pre-processed NOAA meteorological reports, we select individual claims and pair them with the 24-hour slice of WeatherBench2 data corresponding to the report's date. Negative instances are generated by systematically negating claims using an LLM. All examples are human-verified to ensure clarity, verifiability, and correctness of negation. More details on this task are included in Appendix A.2.2.

Difficulty	Human-Gen. Tasks	Human-Gen. Samples	Synthetic Tasks	Synthetic Samples	Total Samples
Easy	5	699	0	0	699
Medium	5	574	30	290	864
Hard	5	595	1	0	595
Total	**15**	**1,868**	**31**	**290**	**2,158**

Table 1: **ZephyrusBench Statistics**: Number of unique tasks and samples, grouped by difficulty and generation form.

[1] https://www.wpc.ncep.noaa.gov/discussions/hpcdiscussions.php?disc=pmdepd

[2] https://iri.columbia.edu/our-expertise/climate/forecasts/seasonal-climate-forecasts/

4.2 Evaluation Metrics

Since all our tasks are designed around weather tasks with objectively correct answers, we design an evaluation pipeline that can assess the scientific correctness of the answers produced by the models. The model answers fall into five primary categories: **numeric**, **temporal**, **spatial (location-based)** and **descriptive**. Given that model outputs are in natural language, we evaluate them through a multi-stage process:

1. **Verification:** Determine whether the models response contains a relevant and valid answer. At this stage, we merely assess whether or not the response has an appropriate answer to the given question, and not its correctness. We use `gpt-4.1-mini` for this purpose.
2. **Extraction:** Extract the specific answer from the model response using another LLM prompt.
3. **Scoring:** Apply scoring methods specific to the type of question, which are detailed below.

Numerical Answers. For numerical responses, we record the Standarized Median Absolute Error between the predicted and reference values. In addition, we also report the $25\%, 75\%$ and 99% quantiles of the standarized absolute error to provide a more complete picture of the error distribution. We use quantiles rather than means because large outliers can significantly skew mean values, obscuring typical model performance patterns. To compare across variables with different scales and units, we divide the absolute error by the standard deviation of the corresponding variable in the dataset.

Time-based Answers. We evaluate tasks with time values as responses using Median Absolute Error. We omit the standarization step, since all the answers are in the same units (that is, hours). Like the numerical answers case, we also report the $25\%, 75\%$ and 99% quantiles.

Location-based Answers. For questions whose answers are geographic locations, we first match the extracted location name to one of the expected entries from the NaturalEarth dataset (e.g., mapping "USA" to "United States of America"). For countries, we use the `country_converter` library (Stadler, 2017). For other geographic entities such as continents and water bodies, we apply fuzzy string matching (Bachmann et al., 2023), accepting matches above a predefined similarity threshold.

To quantitatively assess the geographic deviation between predicted and reference locations, we employ the Earth Mover's Distance (EMD) (Monge, 1781) as a primary evaluation metric. We begin by generating surface area-weighted masks over a latitude–longitude grid for both the predicted and reference locations. These masks are normalized to form probability distributions. To account for the curvature of the Earth, we compute pairwise distances between grid points using geodesic distance. The EMD is then calculated using the `POT` library (Flamary et al., 2021). As a complementary metric, we also report Location Accuracy, which simply measures whether the predicted and reference location strings are an exact match.

Descriptive Answers. To evaluate descriptive answers, we extract individual discussion points from both the model's response and the reference answer. We then classify each extracted claim from the model's response as either `SUPPORTED`, `REFUTED`, or `NEUTRAL` against the reference answer, obtaining logit scores from the language model and applying softmax normalization. Similarly, we perform the same procedure for claims from the reference text compared against the model response.

We then define two complementary metrics: precision measures the validity of the model's claims by computing the proportion that are supported rather than refuted by the reference answer, excluding neutral classifications: Precision $= \frac{\sum_{i \in S} P_{\text{model}\to\text{ref}}(\text{Supported}_i)}{\sum_{i \in S} P_{\text{model}\to\text{ref}}(\text{Supported}_i) + \sum_{i \in S} P_{\text{model}\to\text{ref}}(\text{Refuted}_i)}$ where $S = \{i : P_{\text{model}\to\text{ref}}(\text{Neutral}_i) < 0.5\}$ and $P_{\text{model}\to\text{ref}}(\text{Supported}_i)$ denotes the probability that model claim i is supported by the reference answer.

Recall measures coverage by evaluating how well the model response addresses the reference claims, computed as the average support probability across all reference points: Recall $= \frac{1}{N} \sum_{i=1}^{N} P_{\text{ref}\to\text{model}}(\text{Supported}_i)$ where N is the number of reference claims and $P_{\text{ref}\to\text{model}}(\text{Supported}_i)$ denotes the probability that reference claim i is supported by the model answer.

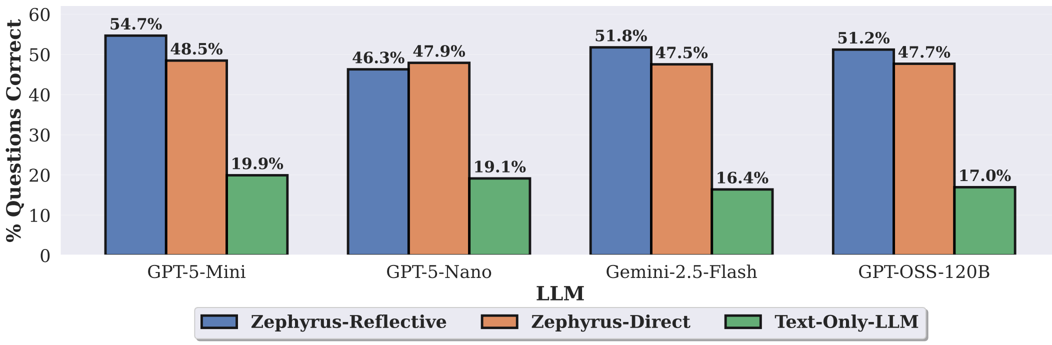

Figure 3: Percentage of questions in the complete dataset answered correctly by each LLM and model type. Definitions of correctness for each question type are detailed in Appendix A.4.

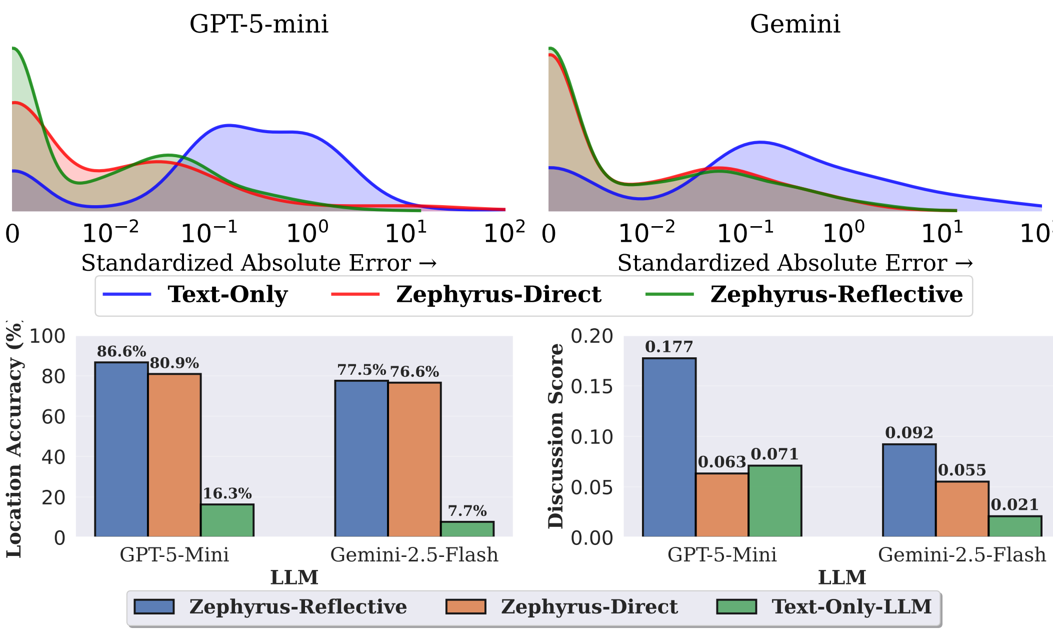

Figure 4: Plots showing (top) error distribution on numerical tasks (bottom-left) location accuracy (bottom-right) discussion scores for GPT-5-Mini and Gemini-2.5-Flash.

Finally, we define the **discussion score** as the F1 score $= \dfrac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$.

Extreme Weather Tasks. In order to evaluate the extreme-weather tasks, we report two metrics: (1) F1 score, which only assesses whether the model correctly predicts the *occurrence* of an extreme event anywhere in the world, without considering event type or exact location. (2) Earthmover's Distance, which measures the agreement between the reference and predicted list of countries.

Correctness. For ease of presentation, we define correctness criteria that vary depending on the task type. Rather than requiring exact matches, we consider an answer correct if it falls within an acceptable range of the target response. The precise criteria for determining correctness for each task type are detailed in Appendix A.4.

5 EXPERIMENTAL RESULTS

We evaluate model performance across all task types from Section 4 using four backend models: OpenAI GPT-5-Mini, GPT-5-Nano, Google Gemini 2.5 Flash, and OpenAI `gpt-oss-120b`. We compare three experimental settings: (1) a text-only baseline that attempts to answer weather reasoning questions using only natural language metadata without access to structured weather data or numerical inputs, (2) ZEPHYRUS-DIRECT, and (3) ZEPHYRUS-REFLECTIVE. The text-only baseline measures the extent to which models can utilize their prior meteorological knowledge.

The correctness results across all models and settings are presented in Figure 3. We observe that the ZEPHYRUS agents significantly outperform the text-only baseline across all models, demonstrating the agentic framework's ability to effectively ground answers by leveraging meteorological data from WeatherBench. For GPT-5-Mini, ZEPHYRUS-DIRECT and ZEPHYRUS-REFLECTIVE achieve 48.5% and 54.7% correctness respectively, compared to only 19.9% for the text-only baseline. This substantial improvement holds consistently across other LLMs, with ZEPHYRUS agents achieving 28.6-35.4% higher correctness than their text-only counterparts. The multi-turn execute-observe-solution framework implemented in ZEPHYRUS-REFLECTIVE enables it to outperform ZEPHYRUS-DIRECT by 3.5-6.2% across most models. The exception is GPT-5-Nano, where ZEPHYRUS-REFLECTIVE performs slightly worse than ZEPHYRUS-DIRECT, likely due to the smaller model's limited reasoning capabilities affecting the more complex multi-turn approach.

Figure 4 enables fine-grained analysis of error distributions for numerical tasks, location accuracy, and discussion scores for descriptive answers using GPT-5-Mini and Gemini 2.5 Flash as backend models. The agents particularly excel at numerical and location prediction tasks, achieving substantially lower Standarized Absolute Errors and higher location accuracies compared to text-only baselines. For location prediction, ZEPHYRUS-REFLECTIVE with GPT-5-Mini achieves strong performance with 86.6% accuracy. Once again, the reflective variant enjoys a small benefit in performance over the Direct approach. The difference between ZEPHYRUS-DIRECT and ZEPHYRUS-REFLECTIVE is pronounced on numerical tasks for GPT-5-Mini, while both variants perform similarly with Gemini 2.5 Flash.

However, all models struggle with the challenging task of generating textual weather reports. The best performing model (ZEPHYRUS-REFLECTIVE with GPT-5-Mini) only achieves a discussion score of 0.177. Nevertheless, ZEPHYRUS-REFLECTIVE demonstrates significant advantages over both ZEPHYRUS-DIRECT and text-only variants for these descriptive tasks. While the text-only variant lacks access to meteorological information, ZEPHYRUS-DIRECT produces rigid answers by directly outputting program results, making it ill-suited for nuanced textual generation. The execute-observe-solution framework in ZEPHYRUS-REFLECTIVE proves more effective.

Performance breakdown by difficulty level reveals interesting patterns (detailed results in Appendices A.5 and A.7). On easy tasks, which primarily involve data analysis questions, ZEPHYRUS agents perform well with 78.7-88.1% correctness. Medium difficulty tasks show moderate performance with 39.9-50.5% correctness. However, on hard tasks, all models struggle significantly, with ZEPHYRUS agents achieving similar performance to text-only baselines. This suggests that while current LLMs can effectively solve simple data analysis problems that pop up in meteorology, they do not yet possess the capability to reason about abstract weather phenomena even when provided with tools.

Task-wise analysis of "Hard" tasks reveals nuanced insights. For generating meteorological discussions and forecasts for the continental United States, models show promise with ZEPHYRUS-REFLECTIVE + GPT-5-Mini achieving an average discussion score of 0.31. This contrasts sharply with global climate forecasting tasks spanning three months, where all models fail completely, highlighting the current limitations in long-term, large-scale weather reasoning.

6 CONCLUSION

We tackled the challenging problem of enabling LLMs to reason over high-dimensional weather data by developing, to our knowledge, the first agentic model for meteorology. Our contributions include: (1) ZEPHYRUSWORLD, an agentic environment with comprehensive meteorological tools, (2) the ZEPHYRUS family of agents that leverage these tools, and (3) a scalable data pipeline producing a large, diverse benchmark dataset (ZEPHYRUSBENCH). Our empirical evaluation shows that the agentic framework enables effective reasoning about meteorological data, significantly outperforming text-only baselines. The agents excel at most tasks but struggle with complex challenges like forecast report generation. Beyond advancing weather science, our work provides a sandbox for developing more effective agentic workflows. Future work could explore using larger datasets to train agents that produce more scientifically accurate responses.

6.1 ETHICS STATEMENT

The authors have followed the ICLR Code of Ethics and have no conflicts of interest to declare.

6.2 Reproducibility Statement

The ZephyrusBench dataset will be made public for use as a benchmark dataset and further research on AI for weather science. All code used to create the dataset and the Zephyrus models will be open-sourced. Detailed results have been included in the paper and its appendices, along with the LLM models used to produce all results.

6.3 LLM Usage Statement

All LLM usage for the creation of ZephyrusBench and the evaluation of different models (Zephyrus-Reflective, Zephyrus-Direct) has been carefully described in the paper. We acknowledge the routine use of LLMs for coding assistance and refinement of writing. LLMs were not used for ideation, conceptual development, literature review, or other substantial contributions ot this work.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

Richard B Alley, Kerry A Emanuel, and Fuqing Zhang. Advances in weather prediction. *Science*, 363(6425):342–344, 2019.

Max Bachmann, layday, Georgia Kokkinou, Jeppe Fihl-Pearson, dj, Henry Schreiner, Moshe Sherman, Michał Górny, pekkarr, Delfini, Dan Hess, Guy Rosin, Hugo Le Moine, Kwuang Tang, Nicolas Renkamp, Trenton H, glynn, and odidev. rapidfuzz/rapidfuzz: Release 3.6.1, December 2023. URL `https://doi.org/10.5281/zenodo.10440201`.

Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

Abeba Birhane, Atoosa Kasirzadeh, David Leslie, and Sandra Wachter. Science in the age of large language models. *Nature Reviews Physics*, 5(5):277–280, 2023.

Cristian Bodnar, Wessel P Bruinsma, Ana Lucic, Megan Stanley, Johannes Brandstetter, Patrick Garvan, Maik Riechert, Jonathan Weyn, Haiyu Dong, Anna Vaughan, et al. Aurora: A foundation model of the atmosphere. *arXiv preprint arXiv:2405.13063*, 2024.

Daniil A Boiko, Robert MacKnight, Ben Kline, Gabe Gomes, et al. Autonomous chemical research with large language models. *Nature*, 624:570–578, 2023.

Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, Philippe Schwaller, et al. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6(5): 525–535, 2024.

Jian Chen, Peilin Zhou, Yining Hua, Dading Chong, Meng Cao, Yaowei Li, Zixuan Yuan, Bing Zhu, and Junwei Liang. Vision-language models meet meteorology: Developing models for extreme weather events detection with heatmaps. *arXiv preprint arXiv:2406.09838*, 2024a.

Wei Chen, Xixuan Hao, Yuankai Wu, and Yuxuan Liang. Terra: A multimodal spatio-temporal dataset spanning the earth. *Advances in Neural Information Processing Systems*, 37:66329–66356, 2024b.

Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*, 2024.

Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*, 2023.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*, 2024.

D. Delforge, V. Wathelet, R. Below, C. Lanfredi Sofia, M. Tonnelier, J. A. F. van Loenhout, and N. Speybroeck. EM-DAT: The Emergency Events Database. *International Journal of Disaster Risk Reduction*, pp. 105509, 2025. doi: 10.1016/j.ijdrr.2025.105509. URL https://doi.org/10.1016/j.ijdrr.2025.105509.

Seungheon Doh, Keunwoo Choi, and Juhan Nam. Talkplay: Multimodal music recommendation with large language models. *arXiv preprint arXiv:2502.13713*, 2025.

RÃ©mi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, AurÃ©lie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, LÃ©o Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78): 1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pp. 10764–10799. PMLR, 2023.

Sreyan Ghosh, Zhifeng Kong, Sonal Kumar, S Sakshi, Jaehyeon Kim, Wei Ping, Rafael Valle, Dinesh Manocha, and Bryan Catanzaro. Audio flamingo 2: An audio-language model with long-audio understanding and expert reasoning abilities. *arXiv preprint arXiv:2503.03983*, 2025.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly journal of the royal meteorological society*, 146(730):1999–2049, 2020.

Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, et al. Biomni: A general-purpose biomedical ai agent. *bioRxiv*, 2025. doi: 10.1101/2025.05.30.656746.

Kevin Maik Jablonka, Qianxiang Ai, Alexander Al-Feghali, Shruti Badhwar, Joshua D Bocarsly, Andres M Bran, Stefan Bringuier, L Catherine Brinson, Kamal Choudhary, Defne Circi, et al. 14 examples of how llms can transform materials science and chemistry: a reflection on a large language model hackathon. *Digital discovery*, 2(5):1233–1250, 2023.

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*, 2024.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.

Thorsten Kurth, Shashank Subramanian, Peter Harrington, Jaideep Pathak, Morteza Mardani, David Hall, Andrea Miele, Karthik Kashinath, and Anima Anandkumar. Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *Proceedings of the platform for advanced scientific computing conference*, pp. 1–11, 2023.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.

Ge Lei, Ronan Docherty, and Samuel J Cooper. Materials science in the era of large language models: a perspective. *Digital Discovery*, 3(7):1257–1272, 2024.

Haobo Li, Zhaowei Wang, Jiachen Wang, Alexis Kai Hon Lau, and Huamin Qu. Cllmate: A multimodal llm for weather and climate events forecasting. *arXiv preprint arXiv:2409.19058*, 2024.

Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5971–5984, 2024.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023b.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023c.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL `https://llava-vl.github.io/blog/2024-01-30-llava-next/`.

Chengqian Ma, Zhanxiang Hua, Alexandra Anderson-Frey, Vikram Iyer, Xin Liu, and Lianhui Qin. Weatherqa: Can multimodal language models reason about severe weather? *arXiv preprint arXiv:2406.11217*, 2024.

Franco Molteni, Roberto Buizza, Tim N Palmer, and Thomas Petroliagis. The ecmwf ensemble prediction system: Methodology and validation. *Quarterly journal of the royal meteorological society*, 122(529):73–119, 1996.

Gaspard Monge. Mémoire sur la théorie des déblais et des remblais. *Mem. Math. Phys. Acad. Royale Sci.*, pp. 666–704, 1781.

Natural Earth. Natural earth data. `https://www.naturalearthdata.com/`, 2024. Accessed: 2024-11-15.

Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. Climax: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

Tung Nguyen, Rohan Shah, Hritik Bansal, Troy Arcomano, Romit Maulik, Rao Kotamarthi, Ian Foster, Sandeep Madireddy, and Aditya Grover. Scaling transformer neural networks for skillful and reliable medium-range weather forecasting. *Advances in Neural Information Processing Systems*, 37:68740–68771, 2024.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044):84–90, 2025.

Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russell, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6):e2023MS004019, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. 36:8634–8652, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Konstantin Stadler. The country converter coco - a python package for converting country names between different classification schemes. *Journal of Open Source Software*, 2(16):332, 2017. doi: 10.21105/joss.00332. URL https://doi.org/10.21105/joss.00332.

Christina V Theodoris, Ling Xiao, Anant Chopra, Mark D Chaffin, Zeina R Al Sayed, Matthew C Hill, Helene Mantineo, Elizabeth M Brydon, Zexian Zeng, X Shirley Liu, et al. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, 2023.

Sumanth Varambally, Veeramakali Vignesh Manivannan, Yasaman Jafari, Luyu Han, Zachary Novack, Zhirui Xia, Salva Rühling Cachay, Srikar Eranky, Ruijia Niu, Taylor Berg-Kirkpatrick, Duncan Watson-Parris, Yian Ma, and Rose Yu. Aquilon: Towards building multimodal weather LLMs. In *ICML 2025 Workshop on Assessing World Models*, 2025. URL https://openreview.net/forum?id=KVxOwEYAF4.

Lei Wang, Shan Dong, Yuhui Xu, Hanze Dong, Yalu Wang, Amrita Saha, Ee-Peng Lim, Caiming Xiong, and Doyen Sahoo. Mathhay: An automated benchmark for long-context mathematical reasoning in llms. *arXiv preprint arXiv:2410.04698*, 2024.

Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. In *International Conference on Learning Representations*, 2022.

Junda Wu, Zachary Novack, Amit Namburi, Jiaheng Dai, Hao-Wen Dong, Zhouhang Xie, Carol Chen, and Julian McAuley. Futga: Towards fine-grained music understanding through temporally-enhanced generative augmentation. *arXiv preprint arXiv:2407.20445*, 2024a.

Junda Wu, Zachary Novack, Amit Namburi, Jiaheng Dai, Hao-Wen Dong, Zhouhang Xie, Carol Chen, and Julian McAuley. Futga-mir: Enhancing fine-grained and temporally-aware music understanding with music information retrieval. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2025.

Kehan Wu, Yingce Xia, Pan Deng, Renhe Liu, Yuan Zhang, Han Guo, Yumeng Cui, Qizhi Pei, Lijun Wu, Shufang Xie, et al. Tamgen: drug design with target-aware molecule generation through a chemical language model. *Nature Communications*, 15(1):9360, 2024b.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024c. URL https://openreview.net/forum?id=BAakY1hNKS.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023b.

Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025.

Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 543–553, 2023.

Xiaokang Zhang, Jing Zhang, Zeyao Ma, Yang Li, Bohan Zhang, Guanlin Li, Zijun Yao, Kangli Xu, Jinchang Zhou, Daniel Zhang-Li, et al. Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios. *arXiv preprint arXiv:2403.19318*, 2024.

Yue Zhao, Ishan Misra, Philipp Krähenbühl, and Rohit Girdhar. Learning video representations from large language models. In *arXiv preprint arXiv:2212.04501*, 2022.

Yizhen Zheng, Huan Yee Koh, Maddie Yang, Li Li, Lauren T May, Geoffrey I Webb, Shirui Pan, and George Church. Large language models in drug discovery and development: From disease mechanisms to clinical trials. *arXiv preprint arXiv:2409.04481*, 2024.

A APPENDIX

A.1 CODE EXECUTION SERVER

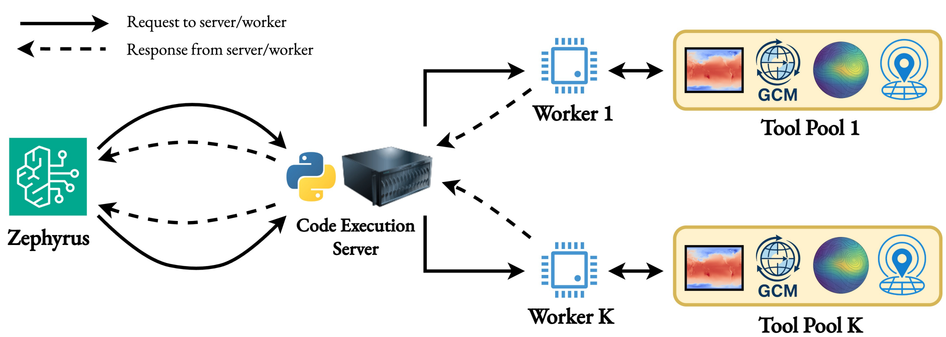

Figure 5: **Code Execution Server.** ZEPHYRUS sends parallel requests to the server, which distributes them to available workers. Each worker acquires resources from tool pools, loads datasets, injects tools into the execution environment, executes code, and returns results or errors to the agent.

In order to execute the code requests from the client, we implement a custom code-execution server program. More specifically, we implement a FastAPI-based server-client architecture where clients send code execution requests to a dedicated execution server that processes them in parallel. The system maintains resource pools for each tool component to prevent contention and enable true parallelism. Each pool contains one or more instances of the above tools. A resource manager implements acquire/release semantics to ensure each execution thread has exclusive access to a complete set of tools while preventing deadlocks.

Each execution follows a strict protocol: acquire resources from pools, load requested datasets, inject tool instances into the execution environment, and execute user code with timeout protection. The system captures all outputs and error information, which are sent back to the client for further processing by the agent. Figure 5 provides an overview of the server.

A.2 DATASET DETAILS

Table A.2 details all the tasks in ZEPHYRUSBENCH, and table 4.1.2 reports the number of samples generated grouped by difficulty and type.

For weather tasks, we leverage the ERA5 reanalysis dataset (Hersbach et al., 2020), specifically from WeatherBench 2 (Rasp et al., 2024), which provides global atmospheric data from 1979 to 2022. We use $1.5°$ spatial resolution with 6-hourly temporal resolution, and include 4 surface variables

ID	Natural Language Description	Answer Type	Difficulty	Type
1	Which geographic feature experienced the highest/lowest average value of a weather variable	Location	Easy	Human
2	What is the min/max/average/median value of a weather variable at a specific location	Numerical	Easy	Human
3	Which sublocation has the highest/lowest recorded variable value	Location	Easy	Human
4	How many hours from start did a location experience extremum	Temporal	Easy	Human
5	What is the weather variable value at a location at a specific time	Numerical	Easy	Human
6	What will the variable be at a location after time interval (forecast)	Numerical	Medium	Human
7	When will location experience its extremum in future period (forecast)	Temporal	Medium	Human
8	Identify extreme weather events that will occur in the next N hours (forecast)	List of locations	Hard	Human
9	Check if extreme weather events are currently happening	List of locations	Hard	Human
10	Which geographic features experienced unusual weather anomalies compared to baseline	List of locations	Medium	Human
11	Does maximum weather variable occur at same or adjacent grid point as another variable (forecast)	Yes/No	Medium	Synthetic
12	Does maximum weather variable in region remain lower than future maximum (forecast)	Yes/No	Medium	Synthetic
13	Does maximum weather variable occur at higher latitude than in another region (forecast)	Yes/No	Medium	Synthetic
14	Does mean weather variable in one region exceed another by specified amount (forecast)	Yes/No	Medium	Synthetic
15	Does mean weather variable exceed threshold while maximum of another stays below (forecast)	Yes/No	Medium	Synthetic
16	Does mean weather variable within region exceed specified threshold (forecast)	Yes/No	Medium	Synthetic
17	Does weather variable exceed threshold within any part of region (forecast)	Yes/No	Medium	Synthetic
18	Does weather variable exceed threshold in more grid points in one region than another (forecast)	Yes/No	Medium	Synthetic
19	Does area where weather variable exceeds threshold cover more than percentage of region (forecast)	Yes/No	Medium	Synthetic
20	Does area-averaged weather variable exceed threshold while another stays below (forecast)	Yes/No	Medium	Synthetic
21	Does maximum weather variable in one region exceed threshold while another stays below (forecast)	Yes/No	Medium	Synthetic
22	Does maximum weather variable within region exceed specified threshold (forecast)	Yes/No	Medium	Synthetic
23	Does maximum weather variable occur at latitude farther north than in another region (forecast)	Yes/No	Medium	Synthetic
24	Does maximum weather variable stay above threshold while another stays below (forecast)	Yes/No	Medium	Synthetic
25	Does maximum weather variable in one region exceed another by specified amount (forecast)	Yes/No	Medium	Synthetic
26	Does minimum weather variable within region remain above threshold (forecast)	Yes/No	Medium	Synthetic
27	What is the area where multiple weather variables exceed their percentile values (forecast)	Numerical	Medium	Synthetic
28	What is the area where weather variable exceeds its median value (forecast)	Numerical	Medium	Synthetic
29	What is the displacement between centroids of areas with top 10% values (forecast)	Numerical	Medium	Synthetic
30	What is the distance between centroids of maximum weather variable value areas (forecast)	Numerical	Medium	Synthetic
31	What is the maximum difference in weather variable between grid points within region (forecast)	Numerical	Medium	Synthetic
32	What is the minimum weather variable where another variable exceeds percentile (forecast)	Numerical	Medium	Synthetic
33	What is the difference between maximum weather variables in two regions (forecast)	Numerical	Medium	Synthetic
34	What is the difference in mean weather variable between two regions (forecast)	Numerical	Medium	Synthetic
35	What is the displacement of minimum weather variable location after time window (forecast)	Numerical	Medium	Synthetic
36	What is the latitude difference between centroids of high weather variable areas (forecast)	Numerical	Medium	Synthetic
37	What is the maximum weather variable difference between two regions (forecast)	Numerical	Medium	Synthetic
38	What is the mean weather variable where another variable exceeds percentile (forecast)	Numerical	Medium	Synthetic
39	What is the mean weather variable where another exceeds percentile threshold (forecast)	Numerical	Medium	Synthetic
40	What is the weather variable value where another variable reaches maximum (forecast)	Numerical	Medium	Synthetic
41	Generate comprehensive global climate forecast for temperature and precipitation for next 3 months (forecast)	Description	Medium	Human
42	Provide detailed meteorological discussion and forecast for continental United States (forecast)	Description	Hard	Human
43	Generate ENSO climate update and outlook based on atmospheric data (forecast)	Description	Hard	Human
44	How will weather variable change after specified time given an intervention (a specified change) in variable in the present (counterfactual)	Numerical	Medium	Human
45	What is the value of the input parameter of the simulator model that produces the simulation output	Numerical	Hard	Human
46	Check whether the given claim extracted from meterological report is supported by the data	Yes/No	Hard	Synthetic

Table 2: Complete set of Weather Tasks, grouped by difficulty.

and 5 atmospheric variables at 13 pressure levels. For each task-type, we define natural language templates with placeholders such as location, variable, and time window. For example, Task 1 is defined as 'Which {geofeature} experienced the {extremum_direction} average {variable}?'. To create task-specific examples, these placeholders are filled by randomly sampled inputs. Ground truths are derived through a deterministic procedure by applying human-written or human-verified synthetic code to the raw ERA5 data and other supplementary data.

A.2.1 Human-Generated tasks

Tasks 1 through 7 rely entirely on the raw ERA5 data; they include basic data lookups and computations, as well as more advanced forecasting. The Geolocator is introduced to enable these and other location-related tasks. It is a wrapper for the Natural Earth dataset (Natural Earth, 2024) that maps ERA5 grid points to natural language location names of countries, states, and water bodies.

For Task 8 and Task 9, which involve extreme event detection, we use records from the EM-DAT international disaster database (Delforge et al., 2025), matching event entries by date and location to ERA5 data.

Input for anomaly comparison (Task 10) comprises two components: recent global data and quantile statistics derived from a historical reference period. Ground truths are calculated by comparing the recent dataset against historical quantile thresholds. Locations where the recent values significantly exceed or fall below the reference quantile are flagged as anomalous. We then use the Geolocator to map flagged grid points to natural language region names.

Report generation tasks (ID 41, 42, 43) are designed to evaluate model climate forecasting and interpretation capabilities based on ERA5 atmospheric datasets. They all use global weather fields over the given time duration as context. Task 41 requires generating a comprehensive global climate forecast report for temperature and precipitation for a three month forward horizon. The task instructs the report to be structured into separate sections for precipitation and temperature, and to provide region-specific forecasts with probability-based language. Task 42 focuses on the continental United States, where the model must provide a detailed meteorological discussion and forecast, including current weather system positions and movements, temperature trends and expected changes over the coming days, precipitation patterns and likelihood of significant events, pressure system evolution and impacts, and notable atmospheric features such as fronts and jet stream positioning. Task 43 requires an ENSO (El Niño–Southern Oscillation) climate update and outlook. Models are tasked to analyze atmospheric variables to assess the current ENSO phase, evaluate strength and persistence indicators, forecast evolution over the next 3 – 6 months, and discuss global implications using probability-based language and standard ENSO terminology. Ground truth reports for these tasks are obtained from authoritative climate prediction and monitoring sources, which provide validated assessments of global and regional climate outlooks and ENSO conditions. The answer sources for these three tasks are NOAA Global Climate Reports, NOAA National Weather Service Area Forecast Discussions, and WMO ENSO Reports, respectively.

Task 44 and Task 45 both rely on the JAX-GCM simulator, an intermediate-complexity atmospheric model built on NeuralGCM's dynamical core (Kochkov et al., 2024). Task 44 is to assess the causal impact of localized perturbations on atmospheric states. To obtain each specific sample, a variable, location, and perturbation magnitude are first sampled, and a Gaussian mask is applied to induce the desired perturbation at the chosen location. The simulator is then run twice, once starting from the unperturbed initial state and once with the imposed perturbation. At the specified simulation end time, the target variable from both simulations is extracted and compared, with the difference quantifying the perturbation's impact.

Task 45 is a black-box optimization climate simulation task. The input consists of two components: (i) a segment of recent global data spanning a specified duration and interval, and (ii) simulated data generated by our JAX-GCM simulator. In the simulation, we vary one input parameter of the model by sampling its value randomly from the range [0,1], then save the resulting simulation output. The objective of the task is to estimate the original value of the underlying input parameter from observable simulation outputs. Since the climate simulator is presented as a black box, the model must infer the parameter solely from the input-output mapping, which can be highly nonlinear and sensitive to small parameter changes. By evaluating the model on novel simulator outputs, we benchmark its general handling of a domain optimization problem. Performance is assessed by comparing the predicted and ground-truth parameter values.

A.2.2 Meteorological Claim Verification

The Meteorological Claim Verification task (Task 46), We start with NOAA monthly climate report webpages (1988–2024). These are downloaded and subsequently scraped into structured text files. The scraped reports are then consolidated into CSV files, which contain raw textual summaries of meteorological conditions. To standardize this content, we prompt the LLM to remove author notes, editorial comments, data collection methods, map/color references, visual elements, and model disclaimers. We preserve forecast events, locations, timing, and patterns. From the resulting text-only, stand-alone weather reports, we extract individual claims to use for task examples.

The following data shows meteorological conditions over a 24-hour period:

{'type': 'wb2', 'variables': ['mean_sea_level_pressure', '10m_u_component_of_wind', '10m_v_component_of_wind', '2m_temperature', 'geopotential', 'specific_humidity', 'temperature', 'u_component_of_wind', 'v_component_of_wind'], 'time_indices': '47007:47011:1', 'start_idx': 47007}

Based on the provided data, answer the following question: Does this data support the provided meteorological claim? Answer with True or False.

Claim: A very strong jet in excess of 150 knots will be across the east central U.S. with a favorable left exit region of the jet over New England.

Positive Claim Type

Original Claim: A very strong jet in excess of 150 knots will be across the east central U.S. with a favorable left exit region of the jet over New England.

The following data shows meteorological conditions over a 24-hour period:

{'type': 'wb2', 'variables': ['mean_sea_level_pressure', '10m_u_component_of_wind', '10m_v_component_of_wind', '2m_temperature', 'geopotential', 'specific_humidity', 'temperature', 'u_component_of_wind', 'v_component_of_wind'], 'time_indices': '79179:79183:1', 'start_idx': 79179}

Based on the provided data, answer the following question: Does this data support the provided meteorological claim? Answer with True or False.

Claim: Dry conditions with minimal precipitation are expected across western Washington and into British Columbia.

Negative Claim Type

Original Claim: Heavy precipitation is expected across western Washington and into British Columbia.

Figure 6: (left) Positive claim and (right) negative example for meteorological claim verification

A.3 Example from the Dataset

The following data shows a snapshot of the global weather fields.

{data}

Based on the above data, answer the following question:

Which {geofeature} experienced the {extremum_direction} average {variable}?","Based on the provided data, {answer} experienced the {extremum_direction} average {variable} over the specified time-period, with an average {variable} of {answer_numeric}."

Example Template

The following data shows a snapshot of the global weather fields.

{'type': 'wb2', 'variables': ['mean_sea_level_pressure', '10m_u_component_of_wind', '10m_v_component_of_wind', '2m_temperature', 'geopotential', 'specific_humidity', 'temperature', 'u_component_of_wind', 'v_component_of_wind'], 'time_indices': '54746:54747:1'}

Based on the above data, answer the following question: Which continent experienced the highest average Surface temperature?

Based on the provided data, Africa experienced the highest average Surface temperature over the specified time-period, with an average Surface temperature of 303.5 K.

Generated Sample

Figure 7: (left) Example template from which samples are generated and (right) a sample generated using the template.

A.4 Definition of Task Correctness

Different task types in ZephyrusBenchare evaluated using relevant metrics. To create a unified definition of correctness, we employ the following requirements for each metric type:

- **Numerical:** Standardized difference $\frac{|\hat{y}-y|}{\sigma} < 0.05$, where σ is the standard deviation of the relevant task variable in the WeatherBench2 dataset.

- **Distance/Area/Coordinate/Simulation:** Relative error $\frac{|\hat{y}-y|}{|y|} < 0.05$. For true values of 0, we require $|\hat{y}| < 0.05$.
- **Location:** Exact locations string match, using fuzzy string matching logic.
- **Extreme Weather/Anomaly:** Earth Mover's Distance (EMD) score < 100 km. If both true and predicted values are empty lists, the answer is considered correct.
- **Boolean:** Exact match between model answer and ground truth boolean value.
- **Discussion:** Overall discussion score > 0.5.
- **Time:** Exact match required (absolute error $= 0.0$).

A.5 Performance by Difficulty Level

Below, we include a detailed breakdown of performance metrics by question difficulty level, as defined in Table A.2, for models GPT-5-Mini, GPT-5-Nano, Gemini 2.5 Flash, and `gpt-oss-120b`.

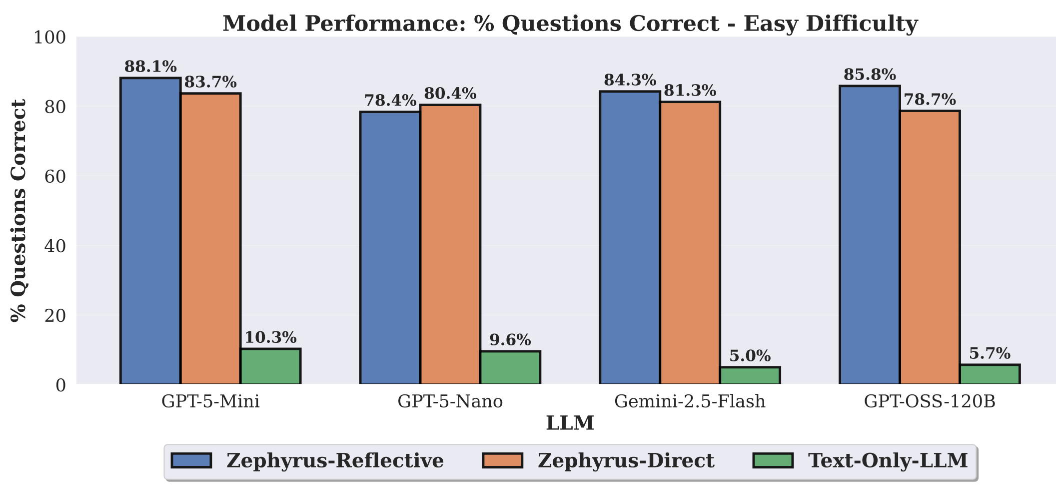

Figure 8: Questions correct by difficulty level: Easy.

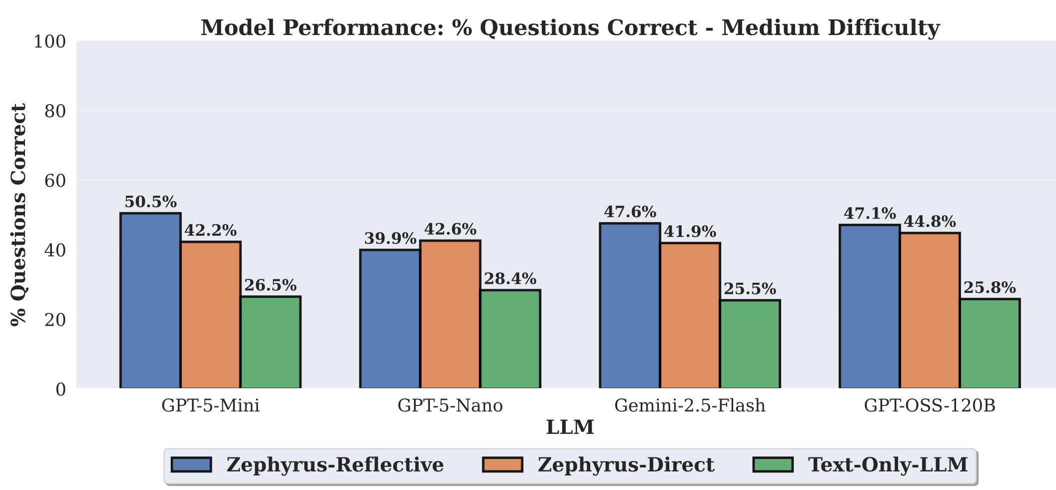

Figure 9: Questions correct by difficulty level: Medium.

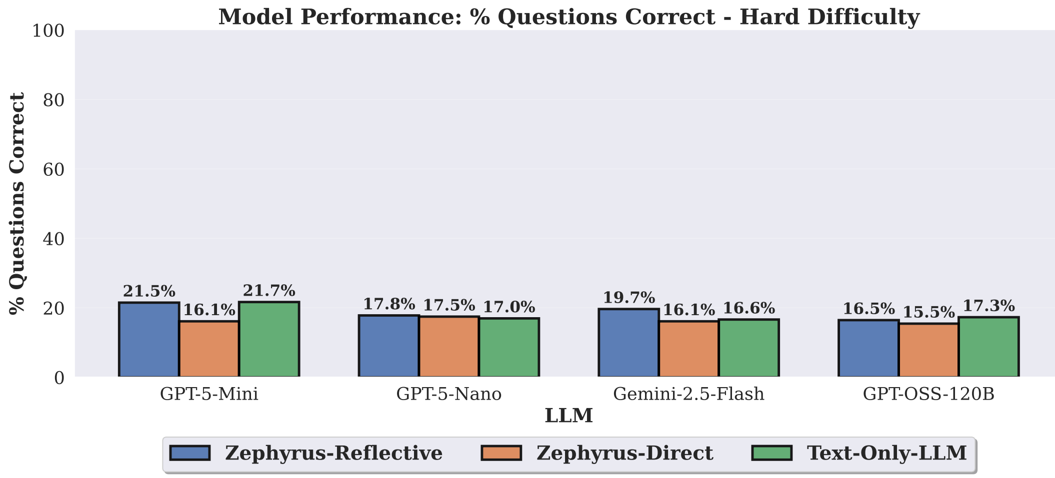

Figure 10: Questions correct by difficulty level: Hard.

Model	LLM	AE (Q25) (↓)	AE (Q50) (↓)	AE (Q75) (↓)	AE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**0.00**	**0.00**	**18.0**	**156.0**
ZEPHYRUS-DIRECT	`gpt-5-mini`	**0.00**	**0.00**	**18.0**	157.0
Text Only LLM	`gpt-5-mini`	12.0	30.0	66.0	168.0
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	**0.00**	**0.00**	**12.0**	158,000
ZEPHYRUS-DIRECT	`gpt-5-nano`	**0.00**	**0.00**	**12.0**	5.01e+18
Text Only LLM	`gpt-5-nano`	18.0	48.0	90.0	**186.0**
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**0.00**	**0.00**	**18.0**	**157.0**
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**0.00**	**0.00**	30.0	8.57e+18
Text Only LLM	`gemini-2.5-flash`	18.0	36.0	72.0	186.0
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**0.00**	**0.00**	**0.00**	**145.0**
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**0.00**	**0.00**	6.00	**145.0**
Text Only LLM	`gpt-oss-120b`	18.0	42.0	84.0	200.0

Table 4: Absolute error quantiles for time tasks, in units of hours.

Model	LLM	Location Accuracy (%)(↑)	EMD (km) (↓)	Extreme Weather F1 (↑)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**86.6**	**1,851**	**0.38**
ZEPHYRUS-DIRECT	`gpt-5-mini`	80.9	1,892	0.28
Text Only LLM	`gpt-5-mini`	16.3	5,783	**0.38**
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	68.9	2,568	**0.36**
ZEPHYRUS-DIRECT	`gpt-5-nano`	**73.7**	**2,126**	0.20
Text Only LLM	`gpt-5-nano`	15.3	5,070	0.00
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**77.5**	**2,021**	**0.38**
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	76.6	2,204	**0.38**
Text Only LLM	`gemini-2.5-flash`	7.66	2,303	0.03
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**77.0**	2,749	**0.49**
ZEPHYRUS-DIRECT	`gpt-oss-120b`	61.2	**2,435**	0.41
Text Only LLM	`gpt-oss-120b`	11.5	3,718	0.00

Table 5: Location metrics for location answer-based questions. EMD stands for Earth mover's Distance.

Model	LLM	% Valid Outputs (↑)	Discussion Score (↑)	Boolean F1 (↑)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**92.9**	**0.18**	0.51
ZEPHYRUS-DIRECT	`gpt-5-mini`	91.5	0.06	0.32
Text Only LLM	`gpt-5-mini`	91.9	0.07	**0.53**
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	88.8	**0.14**	**0.48**
ZEPHYRUS-DIRECT	`gpt-5-nano`	**91.3**	0.07	0.47
Text Only LLM	`gpt-5-nano`	88.9	0.07	0.37
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**91.0**	**0.09**	0.49
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	87.0	0.06	**0.52**
Text Only LLM	`gemini-2.5-flash`	71.9	0.02	0.16
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**90.5**	**0.09**	0.46
ZEPHYRUS-DIRECT	`gpt-oss-120b`	86.8	0.05	**0.47**
Text Only LLM	`gpt-oss-120b`	75.6	0.03	0.18

Table 6: Overall percentage of valid outputs, numerical score (0-1) for discussion questions, and F1 score for boolean questions.

A.6 DETAILED PERFORMANCE METRICS

We include detailed performance metrics from running several LLMs across all three modes on the entire ZEPHYRUSBENCHdataset.

A.7 PERFORMANCE METRICS BY TASK

Model	LLM	SAE (Q25) (↓)	SAE (Q50) (↓)	SAE (Q75) (↓)	SAE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**0.00**	**0.00**	**0.00**	**0.07**
ZEPHYRUS-DIRECT	`gpt-5-mini`	**0.00**	**0.00**	**0.00**	0.17
Text Only LLM	`gpt-5-mini`	0.23	0.80	1.53	8.93
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	**0.00**	**0.00**	0.04	**0.82**
ZEPHYRUS-DIRECT	`gpt-5-nano`	**0.00**	**0.00**	**0.00**	5.76
Text Only LLM	`gpt-5-nano`	0.21	0.62	1.65	586.7
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**0.00**	**0.00**	**0.00**	1.24
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**0.00**	**0.00**	**0.00**	**0.69**
Text Only LLM	`gemini-2.5-flash`	0.30	1.86	3.43	77.2
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**0.00**	**0.00**	**0.00**	9.23
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**0.00**	**0.00**	**0.00**	**0.69**
Text Only LLM	`gpt-oss-120b`	0.17	0.63	1.72	25.1

Table 7: Standardized Absolute Error (SAE) quantiles for Template ID 2: `What is the min/max/average/median value of a weather variable at a specific location`

Model	LLM	SAE (Q25) (↓)	SAE (Q50) (↓)	SAE (Q75) (↓)	SAE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**0.00**	**0.00**	**0.00**	0.28
ZEPHYRUS-DIRECT	`gpt-5-mini`	**0.00**	**0.00**	**0.00**	**0.27**
Text Only LLM	`gpt-5-mini`	0.20	0.54	1.18	35.2
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	**0.00**	**0.00**	0.01	1.37
ZEPHYRUS-DIRECT	`gpt-5-nano`	**0.00**	**0.00**	**0.00**	**0.96**
Text Only LLM	`gpt-5-nano`	0.20	0.53	1.46	320.2
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**0.00**	**0.00**	**0.00**	**0.44**
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**0.00**	**0.00**	**0.00**	**0.44**
Text Only LLM	`gemini-2.5-flash`	0.44	1.46	14.7	1.34e+08
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**0.00**	**0.00**	**0.00**	**0.56**
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**0.00**	**0.00**	0.01	0.75
Text Only LLM	`gpt-oss-120b`	0.26	0.91	2.29	3,360

Table 8: Standardized Absolute Error (SAE) quantiles for Template ID 5: `What is the weather variable value at a location at a specific time`

Model	LLM	SAE (Q25) (↓)	SAE (Q50) (↓)	SAE (Q75) (↓)	SAE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**0.02**	**0.06**	**0.11**	0.60
ZEPHYRUS-DIRECT	`gpt-5-mini`	**0.02**	**0.06**	**0.11**	**0.48**
Text Only LLM	`gpt-5-mini`	0.17	0.62	1.28	8.76
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	0.03	0.08	0.26	2.06
ZEPHYRUS-DIRECT	`gpt-5-nano`	**0.02**	**0.07**	**0.17**	**1.53**
Text Only LLM	`gpt-5-nano`	0.17	0.47	1.19	32,886
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**0.02**	**0.06**	0.13	1.12
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**0.02**	**0.06**	**0.10**	**0.57**
Text Only LLM	`gemini-2.5-flash`	0.39	0.93	2.30	56.2
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**0.02**	**0.06**	0.12	**0.60**
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**0.02**	**0.06**	**0.11**	0.78
Text Only LLM	`gpt-oss-120b`	0.17	0.89	2.34	1,021

Table 9: Standardized Absolute Error (SAE) quantiles for Template ID 6: `What will the variable be at a location after time interval (forecast)`

Model	LLM	SAE (Q25) (↓)	SAE (Q50) (↓)	SAE (Q75) (↓)	SAE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	gpt-5-mini	**0.00**	**0.00**	**0.02**	**0.11**
ZEPHYRUS-DIRECT	gpt-5-mini	**0.00**	**0.00**	0.03	0.13
Text Only LLM	gpt-5-mini	**0.00**	0.07	0.12	0.23
ZEPHYRUS-REFLECTIVE	gpt-5-nano	**0.00**	0.01	0.08	0.26
ZEPHYRUS-DIRECT	gpt-5-nano	**0.00**	**0.00**	**0.04**	0.61
Text Only LLM	gpt-5-nano	**0.00**	0.07	0.12	**0.23**
ZEPHYRUS-REFLECTIVE	gemini-2.5-flash	**0.00**	**0.00**	**0.00**	**0.10**
ZEPHYRUS-DIRECT	gemini-2.5-flash	**0.00**	**0.00**	0.01	0.13
Text Only LLM	gemini-2.5-flash	**0.00**	0.07	0.12	0.23
ZEPHYRUS-REFLECTIVE	gpt-oss-120b	**0.00**	**0.01**	**0.03**	0.33
ZEPHYRUS-DIRECT	gpt-oss-120b	**0.00**	**0.01**	0.04	0.84
Text Only LLM	gpt-oss-120b	**0.00**	0.07	0.12	**0.23**

Table 10: Standardized Absolute Error (SAE) quantiles for Template ID 44: `How will weather variable change after specified time with specified change in variable (counterfactual)`

Model	LLM	SAE (Q25) (↓)	SAE (Q50) (↓)	SAE (Q75) (↓)	SAE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	gpt-5-mini	**0.11**	0.37	0.67	0.99
ZEPHYRUS-DIRECT	gpt-5-mini	0.13	0.33	0.66	1.06
Text Only LLM	gpt-5-mini	0.16	**0.29**	**0.41**	**0.81**
ZEPHYRUS-REFLECTIVE	gpt-5-nano	0.18	0.47	0.68	0.94
ZEPHYRUS-DIRECT	gpt-5-nano	0.17	0.36	0.55	0.99
Text Only LLM	gpt-5-nano	**0.16**	**0.29**	**0.41**	**0.82**
ZEPHYRUS-REFLECTIVE	gemini-2.5-flash	**0.01**	**0.06**	**0.31**	**0.92**
ZEPHYRUS-DIRECT	gemini-2.5-flash	0.12	0.35	0.66	0.94
Text Only LLM	gemini-2.5-flash	0.16	0.29	0.40	0.94
ZEPHYRUS-REFLECTIVE	gpt-oss-120b	0.20	0.31	0.41	0.77
ZEPHYRUS-DIRECT	gpt-oss-120b	**0.16**	0.30	**0.40**	0.83
Text Only LLM	gpt-oss-120b	**0.16**	**0.29**	**0.40**	**0.68**

Table 11: Absolute Error (AE) quantiles for Template ID 45: `What is the value of the input parameter of the simulator model that produces the simulation output`

Model	LLM	AE (Q25) (↓)	AE (Q50) (↓)	AE (Q75) (↓)	AE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	gpt-5-mini	**0.00**	**0.00**	**0.00**	**101.0**
ZEPHYRUS-DIRECT	gpt-5-mini	**0.00**	**0.00**	6.00	122.6
Text Only LLM	gpt-5-mini	12.0	18.0	36.0	130.1
ZEPHYRUS-REFLECTIVE	gpt-5-nano	**0.00**	**0.00**	**0.00**	**60.0**
ZEPHYRUS-DIRECT	gpt-5-nano	**0.00**	**0.00**	**0.00**	6.87e+18
Text Only LLM	gpt-5-nano	12.0	30.0	72.0	144.0
ZEPHYRUS-REFLECTIVE	gemini-2.5-flash	**0.00**	**0.00**	**0.00**	**134.2**
ZEPHYRUS-DIRECT	gemini-2.5-flash	**0.00**	**0.00**	**0.00**	9.18e+18
Text Only LLM	gemini-2.5-flash	12.0	24.0	48.0	138.8
ZEPHYRUS-REFLECTIVE	gpt-oss-120b	**0.00**	**0.00**	**0.00**	**35.3**
ZEPHYRUS-DIRECT	gpt-oss-120b	**0.00**	**0.00**	**0.00**	136.6
Text Only LLM	gpt-oss-120b	12.0	30.0	66.0	162.0

Table 12: Absolute Error (AE) quantiles for Template ID 4: `How many hours from start did a location experience extremum`

Model	LLM	AE (Q25) (↓)	AE (Q50) (↓)	AE (Q75) (↓)	AE (Q99) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**0.00**	**18.0**	87.0	**168.9**
ZEPHYRUS-DIRECT	`gpt-5-mini`	**0.00**	24.0	**81.0**	7.75e+18
Text Only LLM	`gpt-5-mini`	48.0	72.0	126.0	187.7
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	6.00	**18.0**	120.0	263,293
ZEPHYRUS-DIRECT	`gpt-5-nano`	**1.50**	**18.0**	**46.5**	1,521
Text Only LLM	`gpt-5-nano`	54.0	84.0	126.0	**198.8**
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**0.00**	**18.0**	**64.5**	**186.9**
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**0.00**	21.0	84.0	1.52e+18
Text Only LLM	`gemini-2.5-flash`	43.5	69.0	118.5	192.9
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**0.00**	9.00	24.0	157.0
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**0.00**	**6.00**	**18.0**	**146.0**
Text Only LLM	`gpt-oss-120b`	24.0	66.0	126.0	341,254

Table 13: Absolute Error (AE) quantiles for Template ID 7: `When will location experience its extremum in future period (forecast)`

Model	LLM	Location Accuracy (%) (↑)	EMD Score (km) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**94.2**	**371.8**
ZEPHYRUS-DIRECT	`gpt-5-mini`	92.3	399.5
Text Only LLM	`gpt-5-mini`	14.6	8,213
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	75.0	1,720
ZEPHYRUS-DIRECT	`gpt-5-nano`	**81.7**	**1,263**
Text Only LLM	`gpt-5-nano`	12.4	8,368
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	**78.6**	**1,503**
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	77.9	1,634
Text Only LLM	`gemini-2.5-flash`	20.7	7,979
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	76.7	1,557
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**78.2**	**1,533**
Text Only LLM	`gpt-oss-120b`	22.7	8,987

Table 14: Location prediction metrics for Template ID 1: `Which geographic feature experienced the highest/lowest average value of a weather variable`

Model	LLM	Location Accuracy (%) (↑)	EMD Score (km) (↓)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	**81.4**	**992.4**
ZEPHYRUS-DIRECT	`gpt-5-mini`	79.3	1,674
Text Only LLM	`gpt-5-mini`	20.4	3,204
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	**75.0**	**1,733**
ZEPHYRUS-DIRECT	`gpt-5-nano`	**75.0**	1,980
Text Only LLM	`gpt-5-nano`	24.7	3,061
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	82.7	**849.0**
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**84.0**	1,119
Text Only LLM	`gemini-2.5-flash`	25.0	3,634
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	**87.2**	**624.2**
ZEPHYRUS-DIRECT	`gpt-oss-120b`	79.0	1,651
Text Only LLM	`gpt-oss-120b`	17.6	3,480

Table 15: Location prediction metrics for Template ID 3: `Which sublocation has the highest/lowest recorded variable value`

Model	LLM	Correctness (%) (↑)
Zephyrus-Reflective	gpt-5-mini	41.2
Zephyrus-Direct	gpt-5-mini	**45.6**
Text Only LLM	gpt-5-mini	36.8
Zephyrus-Reflective	gpt-5-nano	48.5
Zephyrus-Direct	gpt-5-nano	60.3
Text Only LLM	gpt-5-nano	**64.7**
Zephyrus-Reflective	gemini-2.5-flash	39.7
Zephyrus-Direct	gemini-2.5-flash	25.0
Text Only LLM	gemini-2.5-flash	**58.8**
Zephyrus-Reflective	gpt-oss-120b	33.8
Zephyrus-Direct	gpt-oss-120b	32.4
Text Only LLM	gpt-oss-120b	**64.7**

Table 16: Correctness metrics for Template ID 8: `Identify extreme weather events that will occur in the next` N `hours (forecast)`

Model	LLM	Correctness (%) (↑)
Zephyrus-Reflective	gpt-5-mini	14.6
Zephyrus-Direct	gpt-5-mini	43.8
Text Only LLM	gpt-5-mini	**57.3**
Zephyrus-Reflective	gpt-5-nano	44.8
Zephyrus-Direct	gpt-5-nano	53.1
Text Only LLM	gpt-5-nano	**70.8**
Zephyrus-Reflective	gemini-2.5-flash	33.3
Zephyrus-Direct	gemini-2.5-flash	27.1
Text Only LLM	gemini-2.5-flash	**64.6**
Zephyrus-Reflective	gpt-oss-120b	40.6
Zephyrus-Direct	gpt-oss-120b	28.1
Text Only LLM	gpt-oss-120b	**71.9**

Table 17: Correctness metrics for Template ID 9: `Check if extreme weather events are currently happening`

Model	LLM	Correctness (%) (↑)
Zephyrus-Reflective	gpt-5-mini	**5.10**
Zephyrus-Direct	gpt-5-mini	3.20
Text Only LLM	gpt-5-mini	3.80
Zephyrus-Reflective	gpt-5-nano	5.40
Zephyrus-Direct	gpt-5-nano	1.70
Text Only LLM	gpt-5-nano	**12.5**
Zephyrus-Reflective	gemini-2.5-flash	**13.1**
Zephyrus-Direct	gemini-2.5-flash	3.20
Text Only LLM	gemini-2.5-flash	5.00
Zephyrus-Reflective	gpt-oss-120b	3.10
Zephyrus-Direct	gpt-oss-120b	1.60
Text Only LLM	gpt-oss-120b	**11.1**

Table 18: Correctness metrics for Template ID 10: `Which geographic features experienced unusual weather anomalies compared to baseline`

Model	LLM	Discussion Score (Mean) (↑)	Discussion Score (Median) (↑)
ZEPHYRUS-REFLECTIVE	gpt-5-mini	**0.02**	**0.00**
ZEPHYRUS-DIRECT	gpt-5-mini	0.00	**0.00**
Text Only LLM	gpt-5-mini	0.00	**0.00**
ZEPHYRUS-REFLECTIVE	gpt-5-nano	**0.00**	**0.00**
ZEPHYRUS-DIRECT	gpt-5-nano	**0.00**	**0.00**
Text Only LLM	gpt-5-nano	**0.00**	**0.00**
ZEPHYRUS-REFLECTIVE	gemini-2.5-flash	**0.00**	**0.00**
ZEPHYRUS-DIRECT	gemini-2.5-flash	**0.00**	**0.00**
Text Only LLM	gemini-2.5-flash	**0.00**	**0.00**
ZEPHYRUS-REFLECTIVE	gpt-oss-120b	**0.00**	**0.00**
ZEPHYRUS-DIRECT	gpt-oss-120b	**0.00**	**0.00**
Text Only LLM	gpt-oss-120b	**0.00**	**0.00**

Table 19: Discussion score metrics for Template ID 41: `Generate comprehensive global climate forecast for temperature and precipitation for next 3 months (forecast)`

Model	LLM	Discussion Score (Mean) (↑)	Discussion Score (Median) (↑)
ZEPHYRUS-REFLECTIVE	gpt-5-mini	**0.31**	**0.32**
ZEPHYRUS-DIRECT	gpt-5-mini	0.09	0.05
Text Only LLM	gpt-5-mini	0.10	0.04
ZEPHYRUS-REFLECTIVE	gpt-5-nano	**0.23**	**0.21**
ZEPHYRUS-DIRECT	gpt-5-nano	0.10	0.06
Text Only LLM	gpt-5-nano	0.09	0.07
ZEPHYRUS-REFLECTIVE	gemini-2.5-flash	**0.15**	**0.11**
ZEPHYRUS-DIRECT	gemini-2.5-flash	0.10	0.07
Text Only LLM	gemini-2.5-flash	0.02	0.00
ZEPHYRUS-REFLECTIVE	gpt-oss-120b	**0.16**	**0.14**
ZEPHYRUS-DIRECT	gpt-oss-120b	0.10	0.04
Text Only LLM	gpt-oss-120b	0.02	0.00

Table 20: Discussion score metrics for Template ID 42: `Provide detailed meteorological discussion and forecast for continental United States (forecast)`

Model	LLM	Discussion Score (Mean) (↑)	Discussion Score (Median) (↑)
ZEPHYRUS-REFLECTIVE	gpt-5-mini	**0.26**	**0.22**
ZEPHYRUS-DIRECT	gpt-5-mini	0.15	0.14
Text Only LLM	gpt-5-mini	0.17	0.08
ZEPHYRUS-REFLECTIVE	gpt-5-nano	**0.23**	**0.19**
ZEPHYRUS-DIRECT	gpt-5-nano	0.18	0.16
Text Only LLM	gpt-5-nano	0.20	0.18
ZEPHYRUS-REFLECTIVE	gemini-2.5-flash	**0.18**	**0.16**
ZEPHYRUS-DIRECT	gemini-2.5-flash	0.09	0.00
Text Only LLM	gemini-2.5-flash	0.07	0.00
ZEPHYRUS-REFLECTIVE	gpt-oss-120b	**0.16**	**0.11**
ZEPHYRUS-DIRECT	gpt-oss-120b	0.08	0.05
Text Only LLM	gpt-oss-120b	0.14	0.03

Table 21: Discussion score metrics for Template ID 43: `Generate ENSO climate update and outlook based on atmospheric data (forecast)`

Model	LLM	F1 Score (↑)	Precision (%) (↑)	Recall (%) (↑)
ZEPHYRUS-REFLECTIVE	`gpt-5-mini`	0.51	**65.2**	41.5
ZEPHYRUS-DIRECT	`gpt-5-mini`	0.32	49.4	23.7
Text Only LLM	`gpt-5-mini`	**0.53**	63.0	**45.5**
ZEPHYRUS-REFLECTIVE	`gpt-5-nano`	**0.48**	63.2	**38.3**
ZEPHYRUS-DIRECT	`gpt-5-nano`	0.47	**63.7**	37.6
Text Only LLM	`gpt-5-nano`	0.37	59.5	26.7
ZEPHYRUS-REFLECTIVE	`gemini-2.5-flash`	0.49	62.3	40.8
ZEPHYRUS-DIRECT	`gemini-2.5-flash`	**0.52**	**66.0**	**42.2**
Text Only LLM	`gemini-2.5-flash`	0.16	47.1	9.47
ZEPHYRUS-REFLECTIVE	`gpt-oss-120b`	0.46	**63.0**	36.8
ZEPHYRUS-DIRECT	`gpt-oss-120b`	**0.47**	61.9	**37.4**
Text Only LLM	`gpt-oss-120b`	0.18	54.5	10.7

Table 22: Boolean score metrics for Template ID 46: `Check whether the given claim extracted from meterological report is supported by the data`

A.8 MODEL PROMPTS

We use the following core Instruction prompt for ZEPHYRUS-REFLECTIVE:

Zephyrus-Reflective Instruction Prompt

```
You are an AI weather expert agent. You will use an interactive coding environment with
↪ tool functions, data, and softwares to solve the user's task.

At each turn, you should first provide your thinking and reasoning given the
↪ conversation history (which might include output from executed code within
↪ <observation></observation>).
After that, you must do exactly one of the following:
1) Write code based on problem and/or observation. Your code should be enclosed using
↪ "<execute>" tag, for example: <execute> return "Hello World!" </execute>. IMPORTANT:
↪ You must end the code block with </execute> tag.
2) When you think you have a solution ready, directly provide a solution that adheres to
↪ the required format for the given task to the user.
Your solution should be enclosed using "<solution>" tag, for example: The answer is
↪ <solution> A </solution>. IMPORTANT: You must end the solution block with </solution>
↪ tag. When answering numerical questions, always use SI base unit (standard units of
↪ measurement) unless the problem specifically asks for a certain unit. For example,
↪ some questions may require you to answer in hours. Enclose ONLY the final answer to
↪ the question in these tags, do NOT include any other information.

In each response, you must include <execute> or <solution> tag. Not both at the same
↪ time. Do not generate code outside <execute>. Do not output answers outside
↪ <solution>. Do not respond with messages without any tags. No empty messages.

- Geolocator Documentation:

The detailed documentation for the Geolocator class, including its available methods, is
↪ provided below:

{geolocator_documentation}

------------------------------------------------------------------------
- Forecaster API Documentation:

{forecaster_documentation}

The Forecaster can reliably forecast at most 2 weeks into the future.
- IMPORTANT: If the question is about the future, you **will need to** use the Forecaster
↪ object to answer the question and solve the task.
             The input data **will not** contain the answer to questions about the
             ↪ future.
------------------------------------------------------------------------
- Simulator API Documentation:

{simulator_documentation}

- The Simulator provides atmospheric modeling and can be used for climate simulations,
↪ answering counterfactuals, sensitivity studies, or generating synthetic weather
↪ data.
- The Simulator can handle extended time periods (months to years) in a SINGLE call. DO
↪ NOT create loops or multiple simulator instances. Set total_time to the desired
↪ duration and call simulate() once.
------------------------------------------------------------------------

- Variable Descriptions:

A comprehensive description of every variable contained in the xarray datasets is given
↪ here:
{var_desc}

- Dataset Keys Explanation:

An explanation of what each key in the datasets represents is provided below:
{keys}

...(continued)
```

Zephyrus-Reflective Instruction Prompt (cont.)

```
(continued)...

- Units:

Always use the following SI units when reasoning and coding:
{units_desc}
Answer in SI units unless the problem specifically asks for a different unit. For
↪  example, some questions may require hours.

- Time Indices:
You should NOT slice the provided dataset according to the time indices. The datasets
↪  are already sliced to the correct time indices.
For any question that asks about the time offset, only provide the time indices relative
↪  to the provided dataset.
If the question asks for the time offset, return the answer in hours from the initial
↪  time index.
For example, if the question asks about a dataset with time interval 6 hours and time
↪  indices 12345:12351:1, and you think the answer is index 12350, you should return 30
↪  hours.
Do NOT return the time index as a timestamp or datetime object.

**Execution code requirements:**
- The code MUST all be defined with a function called `run`.
- The `run` function should accept four parameters:
  a. A list of one or more xarray datasets.
  b. A Geolocator object (which comes with a set of predefined helpful functions).
  c. A Forecaster object (which comes with a set of predefined helpful functions).
  c. A Simulator object (which comes with a set of predefined helpful functions).
- DO NOT write any code outside of the `run` function.

**IMPORTANT:**
- The Geolocator object is already constructed and passed in as `geolocator`.
- **Never open files, use `xr.open_dataset`, or import Geolocator.**
- If you are subsetting, make sure to subset carefully considering runtime. It is too
↪  slow to select the entire xarray dataset. If you are subsetting over multiple
↪  dimensions (e.g. spatially and temporally), make sure to apply the smaller subset
↪  operation first.
- By following these detailed instructions, your code should clearly use the provided
↪  datasets and tools to produce the correct result.

- Coordinate System:
The WeatherBench2 (WB2) dataset uses an equiangular grid with the following
↪  specifications:
- Latitude: 121 grid points ranging from -90° to +90° in 1.5° increments
- Longitude: 240 grid points ranging from 0° to 358.5° in 1.5° increments
- The latitude coordinates are: [-90, -88.5, -87, ..., 87, 88.5, 90]
- The longitude coordinates are: [0, 1.5, 3, ..., 355.5, 357, 358.5]


**Other Requirements:**
- Under NO circumstances should you loop over the grid points (i.e. you should NOT loop
↪  over latitudes and longitudes), but rather try to leverage vectorized operations,
↪  built-in functions or the Geolocator class as appropriate. This is a key requirement.
↪  DO NOT loop over the latitudes and longitudes ANYWHERE in your generated code.
- Ensure that you call and use the functions from the Geolocator object correctly as per
↪  its documentation.

**Question:**
{question}
```

For the reflective stage of ZEPHYRUS-REFLECTIVE, we use the following Observation prompt:

Zephyrus-Reflective Observation Prompt

```
The executed code produced the output above. Reason about your next step and either (1)
↪  output the final result based on this observation. Enclose your answer in
↪  <solution></solution> tags., or (2) generate another code block to execute. Enclose
↪  your code in <execute></execute> tags.
If you choose to give a solution, enclose ONLY the final answer to the question in these
↪  tags, do NOT include any other information.
You should execute code if you think you need more information before providing a final
↪  answer.
```

For ZEPHYRUS-DIRECT, we use the following direct Instruction prompt:

Zephyrus-Direct Instruction Prompt

```
Your objective is to write a Python function called 'run' that solves a specified
↪  problem using provided data and Toolset APIs. The function should be designed
↪  according to the following guidelines:

1. Function Definition:

- The function must be named run.
- It should accept four parameters:
  a. A list of one or more xarray datasets.
  b. A Geolocator object (which comes with a set of predefined helpful functions).
  c. A Forecaster object (which comes with a set of predefined helpful functions).
  c. A Simulator object (which comes with a set of predefined helpful functions).

2. Data Descriptions:

- Variable Descriptions:

A comprehensive description of every variable contained in the xarray datasets is given
↪  here:
{var_desc}

- Dataset Keys Explanation:

An explanation of what each key in the datasets represents is provided below:
{keys}

- Units:

Always use the following SI units when reasoning and coding:
{units_desc}

- Time Indices:

The datasets provided have been converted from using a time dimension to simple integer
↪  indices starting from 0. Each index step represents 6 hours of time in the original
↪  dataset.
You should NOT slice the provided dataset according to the provided indices. The
↪  datasets are already sliced to the correct indices.
For any question that asks about the time offset, only provide the time indices relative
↪  to the provided dataset.
If the question asks for the time offset, you should return the answer in hours from the
↪  initial time index.
For example, if the question asks about a dataset with time interval 6 hours and indices
↪  0:6:1, and you think the answer is index 5, you should return 30 hours.
Do NOT return the index directly.

3. Toolset APIs

You are given access to the following code tools. Please use them as needed inside your
↪  `run` function:

- Geolocator Documentation:

The detailed documentation for the Geolocator class, including its available methods, is
↪  provided below:

{geolocator_documentation}

------------------------------------------------------------------------
- Forecaster API Documentation:

{forecaster_documentation}
The Forecaster can reliably forecast at most 2 weeks into the future.
- IMPORTANT: If the question is about the future, you **will need to** use the Forecaster
↪  object to answer the question and solve the task.
             The input data **will not** contain the answer to questions about the
             ↪  future.
------------------------------------------------------------------------

...(continued)
```

Zephyrus-Direct Instruction Prompt (cont.)

```
(continued)...

- Simulator API Documentation:

{simulator_documentation}

- The Simulator provides atmospheric modeling and can be used for climate simulations,
↪  answering counterfactuals, sensitivity studies, or generating synthetic weather
↪  data.
- The Simulator can handle extended time periods (months to years) in a SINGLE call. DO
↪  NOT create loops or multiple simulator instances. Set total_time to the desired
↪  duration and call simulate() once.
------------------------------------------------------------------------

4. Task Details:

- The function should process the datasets using the pertinent variables as specified
↪  within the question.
- Under NO circumstances should you loop over the grid points (i.e. you should NOT loop
↪  over latitudes and longitudes), but rather try to leverage vectorized operations,
↪  built-in functions or the Geolocator class as appropriate. This is a key requirement.
↪  DO NOT loop over the latitudes and longitudes ANYWHERE in your generated code.
- Ensure that you call and use the functions from the Geolocator object correctly as per
↪  its documentation.

5. Returning the Answer:

- The final result should be returned by the function.
- Make sure to encapsulate your run function in triple backticks for clarity. For
↪  example:
'''
def run(...):
    return "Hello"
'''
- If the answer is a time value, make sure to return it in a unit of time rather than as
↪  a timestamp or datetime object. For example, return `5 hours` instead of `2022-01-01
↪  05:00:00`.
- Always return the answer in the same unit as the one used in the weatherbench dataset.
↪  Do not convert any units.

6. Problem Statement:

By following these detailed instructions, your code should clearly use the provided
↪  datasets and the Toolset APIs to produce the correct result.
The specific question that your function needs to answer is provided at the end of this
↪  prompt: {question}
```