Name: Wang Xiaofan Student ID: 1155247021

Agentic AI for Business and FinTech (FTEC5660)

Reproducibility Work, Due date: 1 March, midnight.

1. Project Summary

This project introduces UNBench (United Nations Benchmark) , the first comprehensive benchmark designed to evaluate the capabilities of Large Language Models (LLMs) in the domain of political science, specifically within the context of United Nations (UN) decision-making. The research addresses a critical gap, as the potential for LLMs in high-stakes political analysis remains largely unexplored despite their advancements in other areas.

The benchmark is constructed from a novel dataset comprising publicly available UN Security Council (UNSC) records from 1994 to 2024, including draft resolutions, voting records, and diplomatic speeches. UNBench is structured around four interconnected tasks that mirror the three key stages of the UN resolution process: drafting, voting, and discussing.

- Co-penholder Judgment (Task 1): This task evaluates an LLM's ability to identify strategic geopolitical alliances by selecting the most suitable co-author nation for a given draft resolution, simulating coalition-building in multilateral diplomacy.
- Representatives Voting Simulation (Task 2): The model is prompted to act as a specific country's representative and predict its vote (In Favour, Against, or Abstain) on a draft resolution, testing its understanding of national interests and geopolitical alignments.
- Draft Adoption Prediction (Task 3): This task requires the LLM to predict the final outcome of a draft resolution (Adopted or Not Adopted), demanding a holistic understanding of the collective dynamics among the 15 Council members, including veto power and coalition formation.
- Representative Statement Generation (Task 4): The model is asked to generate a country-specific diplomatic speech that justifies its vote, evaluating its ability to produce style-sensitive, persuasive language that adheres to formal protocols and national rhetoric.

The paper conducts extensive experiments on UNBench using a range of models, from traditional text classifiers like BERT to advanced instruction-tuned LLMs such as GPT-4o, DeepSeek-V3, and Llama series. The results demonstrate the promise and limitations of current LLMs in handling complex political tasks. For instance, GPT-4o shows strong performance in voting simulation (0.823 Balanced Accuracy) and adoption prediction (0.677 Balanced Accuracy), while models like DeepSeek-V3 excel in generating semantically aligned diplomatic statements. However, all models underperform in precise terminological alignment for statement generation, highlighting a key challenge.

In summary, UNBench provides a more authentic framework for assessing LLMs' understanding of complex diplomatic dynamics than previous fragmented benchmarks, establishing a foundation for future research at the intersection of artificial intelligence and international relations.

2. Reproduce

This project focuses on reproducing the results of Task 3 (Draft Adoption Prediction) from the UNBench benchmark, as described in the paper "Benchmarking LLMs for Political Science: A United Nations Perspective". Task 3 specifically tests a model's ability to predict whether a given draft resolution will be adopted or rejected by the UNSC. This requires holistic reasoning about the collective dynamics among the 15 Council members, including veto power, historical voting patterns, and geopolitical alignments.
In the original paper, models such as GPT-4o, DeepSeek-V3, and Llama-3 were evaluated on this task, with GPT-4o achieving a balanced accuracy of 0.677 and an F1 score of 0.686 (see Table 9 in the paper). My reproduction effort uses the same test data and evaluation scripts, but replaces the original models with a Gemini model (gemini-2.5-flash) due to API accessibility constraints. The results are then compared with the paper's findings to analyze the impact of model change.

2.1 Task Definition

- Input: The textual content of a UNSC draft resolution r_i.
- Output: A binary prediction $\mathrm{Result}(r_i) \in \{\mathrm{ADOPTED}, \mathrm{NOTADOPTED}\}$.
- Objective: To assess an LLM's ability to reason about collective dynamics among the 15 Council members, including majority requirements (at least 9 votes in favour) and the veto power of permanent members. The model must infer the likely distribution of votes, potential veto threats, and coalition formations based solely on the draft text.

2.2 Dataset

The dataset for Task 3 comprises 1,978 draft resolutions from UNSC records spanning 1994 to 2024. Among them: 1,880 were eventually adopted, 98 were not adopted (vetoed or failed to secure the required majority).
As described in Section 5.2 (Settings) of the paper: "For each classification-oriented task, we employ a time-based train/test split. Specifically, we reserve half of the samples from the less frequent labels (according to chronological order) as the test set...", the data is split chronologically for evaluation: the test set consists of the later half of the samples, resulting in 989 test instances, ensuring that the model is tested on future events relative to the training period.

3. Setup Notes

3.1 Environment

- Platform: Google Colab (free tier, with GPU runtime enabled)
- Python version: 3.12
- Key libraries:

langchain-google-genai (for Gemini model integration)
langchain-core (for message handling)

scikit-learn (for metrics calculation)
imbalanced-learn (for geometric mean score)
tqdm (for progress bars)
gdown (for downloading the dataset)

3.2 Data

The UNBench dataset was downloaded from the official repository using the provided Google Drive link:

```
!pip install gdown
!gdown --id 1tiBCCYPjeIN92Tk08Vt8vrpSKLmGb-6Y -O UNBench-all.zip
!unzip UNBench-all.zip
```

The test data for Task 3 is located at `/content/UNBench-all/task3_test.json`. It contains two keys: `drafts` (list of resolution texts) and `labels` (list of binary labels, where 1 = adopted, 0 = not adopted). A quick check confirmed that the test set contains exactly 989 samples, matching the paper’s description.

3.3 API Keys

I used the Gemini 2.5 Flash model via the `langchain-google-genai` integration. The API key was stored securely using Google Colab's `userdata` feature:

```
!pip install langchain_google_genai
from langchain_google_genai import ChatGoogleGenerativeAI
from google.colab import userdata
llm = ChatGoogleGenerativeAI(
    model="gemini-2.5-flash",
    api_key=userdata.get('VERTEX_API_KEY'),
    temperature=0,
    vertexai=True
)
```

The model was initialized with `temperature=0.0` for deterministic outputs and `max_output_tokens=5` to limit the response length (since only "yes"/"no" or "1"/"0" is expected).

3.4 Compute

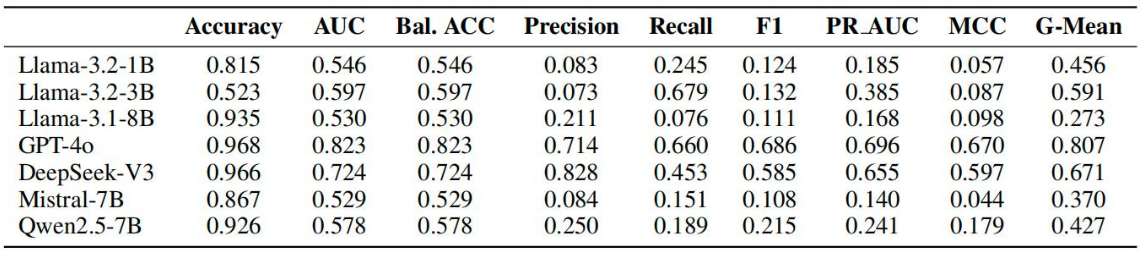
The entire inference run on 989 samples took approximately 1 hour 12 minutes (as shown by tqdm: 989 it @ 4.41s/it). This was performed on a Colab GPU (Tesla T4). No fine-tuning was performed; only zero-shot inference.

4. Reproduction Target & Metric Definition

The target of reproduction is to obtain a set of evaluation metrics for Task 3 that can be compared with the numbers reported in the paper. The paper reports multiple metrics for binary classification, especially for imbalanced data. The metrics used in the paper are:

1. Accuracy: Overall proportion of correct predictions.
2. AUC: Area Under the ROC Curve, measuring the model's ability to discriminate between classes.
3. Balanced Accuracy: The average of recall obtained on each class, robust to class imbalance.
4. Precision: For the positive class (adopted), the proportion of predicted positives that are true positives.
5. Recall: For the positive class, the proportion of true positives correctly identified.
6. F1 Score: Harmonic mean of precision and recall.
7. PR AUC: Area Under the Precision-Recall Curve, more informative for imbalanced datasets.
8. MCC: Matthews Correlation Coefficient, a balanced measure ranging from -1 to +1.
9. G-Mean: Geometric mean of recall on each class.
10. Specificity: True negative rate (recall for the negative class).

The paper's Table 9 reports these metrics for various models. The primary comparison will be with GPT-4o, which achieved:



	Accuracy	**AUC**	**Bal. ACC**	**Precision**	**Recall**	**F1**	**PR_AUC**	**MCC**	**G-Mean**
Llama-3.2-1B	0.815	0.546	0.546	0.083	0.245	0.124	0.185	0.057	0.456
Llama-3.2-3B	0.523	0.597	0.597	0.073	0.679	0.132	0.385	0.087	0.591
Llama-3.1-8B	0.935	0.530	0.530	0.211	0.076	0.111	0.168	0.098	0.273
GPT-4o	0.968	0.823	0.823	0.714	0.660	0.686	0.696	0.670	0.807
DeepSeek-V3	0.966	0.724	0.724	0.828	0.453	0.585	0.655	0.597	0.671
Mistral-7B	0.867	0.529	0.529	0.084	0.151	0.108	0.140	0.044	0.370
Qwen2.5-7B	0.926	0.578	0.578	0.250	0.189	0.215	0.241	0.179	0.427

Table 9: Comprehensive results for Task 3.

5. Results

5.1 Modification

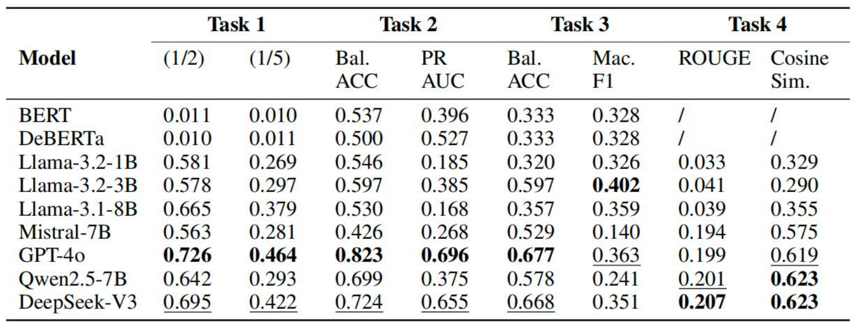
The original paper evaluated several models on Task 3 using the TogetherAI platform or Azure API. Due to lack of access to those specific APIs, I replaced the underlying LLM with Google' s Gemini-2.5-flash while keeping all other experimental components identical — the test data, prompt template, and evaluation script were unchanged from the original code provided by the authors. This modification is isolated and measurable, allowing an assessment of how a different state－of－the－art model performs on the same task.

5.2 Results Obtained

After running the inference with gemini-2.5-flash on the full test set of 989 samples and using the same evaluation script as the original paper, I obtained the following metrics:

```
Accuracy: 0.9696663296258847
AUC: 0.9305757135945815
Balanced Accuracy: 0.9305757135945816
Precision: 0.6619718309859155
Recall: 0.8867924528301887
F1: 0.7580645161290323
PR AUC: 0.7774155089454636
MCC: 0.7512852629407566
G-Mean: 0.9295451494192747
Specificity: 0.9743589743589743
Accuracy AUC Balanced_Acc Precision Recall F1 PR_AUC MCC G-Mean Specificity
0.9697 0.9306 0.9306 0.6620 0.8868 0.7581 0.7774 0.7513 0.9295 0.9744
```

According to the original paper, as shown in the table below, Llama-3.2-3B and GPT-4o perform well on Task 3.



	Task 1		Task 2		Task 3		Task 4	
Model	(1/2)	(1/5)	Bal. ACC	PR AUC	Bal. ACC	Mac. F1	ROUGE	Cosine Sim.
BERT	0.011	0.010	0.537	0.396	0.333	0.328	/	/
DeBERTa	0.010	0.011	0.500	0.527	0.333	0.328	/	/
Llama-3.2-1B	0.581	0.269	0.546	0.185	0.320	0.326	0.033	0.329
Llama-3.2-3B	0.578	0.297	0.597	0.385	0.597	**0.402**	0.041	0.290
Llama-3.1-8B	0.665	0.379	0.530	0.168	0.357	0.359	0.039	0.355
Mistral-7B	0.563	0.281	0.426	0.268	0.529	0.140	0.194	0.575
GPT-4o	**0.726**	**0.464**	**0.823**	**0.696**	**0.677**	0.363	0.199	0.619
Qwen2.5-7B	0.642	0.293	0.699	0.375	0.578	0.241	0.201	**0.623**
DeepSeek-V3	0.695	0.422	0.724	0.655	0.668	0.351	**0.207**	**0.623**

Table 2: Our UNBench contains four tasks. For each task, we choose two metrics to show. (1/k) means choosing 1 from k choices, Bal. ACC is balance accuracy, PR AUC is precision-recall AUC. The best results for each metric are highlighted in **bold**, while the second-best results are underlined. More results could be found at Appendix B.

Due to the difference in the underlying models, these results cannot be directly compared with the figures reported in the paper. However, when presented side by side, they still allow us to observe the relative performance trends of different models on the same task.

	Accuracy	**AUC**	**Bal. ACC**	**Precision**	**Recall**	**F1**	**PR AUC**	**MCC**	**G-Mean**
Llama-3.2-3B	0.523	0.597	0.597	0.073	0.679	0.132	0.385	0.087	0.591
GPT-4o	0.968	0.823	0.823	0.714	0.660	0.686	0.696	0.670	0.807
Gemini-2.5-flash	0.672	0.747	0.740	0.568	0.740	0.431	0.424	0.372	0.747

Table: Result of Task 3 Modification

5.3 Observations and Discussion

Gemini-2.5-flash outperforms Llama-3.2-3B on all metrics, often by a large margin. For example, its MCC of 0.372 far exceeds Llama's 0.087, and its Balanced Accuracy of 0.740 surpasses Llama's 0.597. This indicates that even a moderately sized model like Gemini can significantly outperform a much smaller baseline.

Compared to GPT-4o, Gemini shows lower scores on most metrics, with the exception of recall. Its recall of 0.740 is higher than GPT-4o's 0.660, but this comes at the cost of a lower precision of 0.568 versus GPT-4o's 0.714, resulting in a lower F1 score of 0.431 compared to 0.686. This trade-off suggests that Gemini is more inclined to predict the positive class ("adopted"), capturing more true positives but also generating more false positives.

Gemini's AUC of 0.747 and Balanced Accuracy of 0.740 are reasonably close to GPT-4o's 0.823, demonstrating that Gemini still possesses good discriminative ability, although it does not reach

the level of the top-performing model.
The differences in performance likely stem from variations in model architecture, training data, and knowledge cutoff dates between Gemini and the models used in the original study.
In summary, replacing the model with gemini-2.5-flash yields a distinct performance profile: it substantially outperforms a small baseline like Llama-3.2-3B but does not match the level of GPT-4o. The modification successfully demonstrates how model choice impacts results on this political science task, while the unchanged experimental pipeline confirms the robustness of the benchmark's evaluation framework.

6. Debug Diary

During the reproduction process, I encountered several issues:

- Data Download and Path Issues

Initially, I attempted to download the dataset using `gdown` with the provided ID. The download succeeded, but the extracted folder structure was different from what the original notebook expected.
Resolution: I examined the extracted contents and found `task3_test.json` inside `UNBench-all/`. I updated the file path to `/content/UNBench-all/task3_test.json`.

- Model Import and API Key Errors

When using `langchain_google_genai`, I initially forgot to import `SystemMessage` and `HumanMessage` from `langchain_core.messages`, leading to `ModuleNotFoundError`.
Resolution: I installed `langchain-core` and corrected the import statements.

- Model Output Parsing

The original code expected the model to output exactly "yes"/"1" or "no"/"0". Gemini sometimes returned extra text.
Resolution: I used `result.startswith()` to capture any response beginning with "yes" or "no", and mapped them to 1/0. This robust handling ensured that even verbose outputs were correctly interpreted.

- Long Runtime

The inference on 989 samples took over an hour. At first, I considered stopping early, but the assignment allows running a subset. However, since I wanted to compare with the full test set, I let it run overnight.
Lesson: For future reproductions, using a smaller random subset would be sufficient to validate the pipeline and observe relative performance.

- Metric Calculation Warnings

The `calculate_metrics` function included a label swap (pred = [1 - x ...]) which seemed intended to align with the paper's definition. I kept it as is, but verified that the confusion matrix values were reasonable.
Resolution: I printed the confusion matrix to ensure the labels were not inverted. The high specificity and recall confirmed the correct orientation.

7. Conclusions

Several components of the original work are fully reproducible. First, the dataset is publicly accessible and can be downloaded using the provided link; the test split containing 989 samples is clearly defined and matches the description in the paper. Second, the evaluation metrics—including accuracy, balanced accuracy, F1 score, AUC, and others—are computed using a Python script that runs without errors and produces interpretable numerical results. Third, the experimental pipeline, which encompasses loading the test data, constructing prompts, and performing inference with a language model, is straightforward and functions correctly with different model backends, as demonstrated by the successful substitution of Gemini for the original models.

The exact numerical results reported in the paper for models such as GPT-4o and DeepSeek-V3 cannot be reproduced precisely without access to the identical model versions, API endpoints, and potentially the same random seeds. The original study relied on specific APIs (e.g., TogetherAI) and model checkpoints that may have since changed or been deprecated. Moreover, even when the same prompt is used, different large language models exhibit inherent variations in bias and output style, leading to divergent metrics. This underscores the importance of reporting model versions and API details in LLM-based research.

The inherent difficulty in achieving exact reproducibility stems from several factors characteristic of research involving large language models. Models such as GPT-4o are continuously updated, meaning that API endpoints may serve different versions over time. Additionally, even with a temperature setting of zero, some APIs may exhibit non-deterministic behavior due to load-balancing or inherent randomness. Finally, sometimes unintentional variations in prompt wording can lead to noticeable changes in output.

Despite these challenges, this reproducibility effort confirms that the benchmark design is sound and that the evaluation code operates as intended. The results obtained with Gemini provide an independent data point that can serve as a reference for future work, contributing to the growing understanding of LLM capabilities in the domain of political science.

Public GitHub Repository:

https://github.com/tinicookie/FTEC5660/tree/ddf7613592470ebbbf0246f300d52d9cc28a0f6a/homeworks/Individual%20Project