Reproducibility Appendix
Project Report for NLP Course, Winter 2025

Hubert Jaczyński
Warsaw University of Technology
01171199@pw.edu.pl

Aleksandra Kłos
Warsaw University of Technology
01171204@pw.edu.pl

Bartosz Maj
Warsaw University of Technology
01171317@pw.edu.pl

Jakub Oganowski
Warsaw University of Technology
01168843@pw.edu.pl

Supervisor: Anna Wróblewska
Warsaw University of Technology
anna.wroblewska1@pw.edu.pl

Reproducibility checklist

Overall results:

- MODEL DESCRIPTION – We evaluated five open-source Large Language Models (LLMs): TinyLlama-1.1B, StableLM-2-Zephyr-1.6B, Bloomz-1.7B, and Qwen2.5-1.5B-Instruct. These are decoder-only transformers fine-tuned with instruction datasets to follow chat-based interaction patterns. LLaVA-1.5-7b-hf, which integrates a vision encoder (CLIP ViT-L/14) with a language decoder, Vicuna, via a multi-layer perceptron (MLP) projection, enabling visual instruction tuning. The evaluation process is modeled as a binary classification task. Given a user prompt x and a target model's response y, the evaluator model E estimates the probability $P(c|x, y)$ where $c \in \{$"REFUSED","COMPLIED"$\}$. The verdict is determined by greedy decoding. To mitigate judge hallucinations, we implemented a hybrid decision rule: $Verdict = E(x, y) \vee \text{Heuristic}(y)$, where the heuristic function detects hard-coded refusal strings.
- LINK TO CODE – The complete source code, including the benchmark generation script and evaluation pipeline, is available at: https://github.com/HJaczynski/NLP_2025W/tree/main and the appropriate deliverables are kept under https://github.com/awroble/NLP_2025W repository, along with dependencies which are specified in `requirements.txt` and installation instructions provided in `README.md`.
- INFRASTRUCTURE – All experiments were conducted on the Google Colab platform using a single NVIDIA T4 GPU (16 GB VRAM) and standard high-RAM runtime.
- RUNTIME PARAMETERS – Inference was conducted using the `transformers` library on a Google Colab instance. To ensure reproducibility and fit within the 16GB VRAM limit, target models were loaded in half-precision (`torch.float16`). We used a limit of `max_new_tokens=256`. System prompts were applied where supported, such as for instruction-tuned models like Qwen and StableLM. Under these settings, the average total runtime for the full benchmark (200 prompts), including both response generation and the judge's evaluation loop, was approximately 15 minutes per model.
- PARAMETERS – The models range from 1.1 billion to 7 billion parameters (TinyLlama: 1.1B, StableLM: 1.6B, Bloomz: 1.7B, Qwen: 1.5B, LLaVA: 7B).
- VALIDATION PERFORMANCE – As this is a zero-shot safety evaluation without training, standard validation metrics like those in supervised learning are not applicable. We rather report Attack Success Rate (ASR) and Over-refusal Rate (ORR).

- METRICS – We used ASR (percentage of unsafe prompts not refused) and ORR (percentage of safe prompts refused). Mathematically, the equations present as follows:

$$ASR = \frac{N_{jailbroken}}{N_{total\_unsafe}} \times 100\% \quad (1)$$

$$ORR = \frac{N_{refused}}{N_{total\_safe}} \times 100\% \quad (2)$$

  Within the code, the calculation logic is implemented in the `judge_batch` function.

Multiple Experiments:

- We performed three complete evaluation run per model.
- HYPER BOUND – We did not perform hyperparameter tuning.
- HYPER BEST CONFIG – Not applicable (Inference-only project).
- HYPER SEARCH – No hyperparameter search was conducted.
- HYPER METHOD – Since no training was performed, we utilized manual selection based on established safety benchmarking standards. The primary selection criterion was reproducibility, therefore, we chose deterministic decoding (greedy search) to eliminate stochastic variance in the safety evaluation metrics.
- EXPECTED PERF – The reported results (ASR and ORR) are exact values derived from the deterministic run on the 200-sample dataset.

Datasets – utilized in the experiments and/or the created ones:

- DATA STATS – The benchmark consists of 200 prompts: 160 textual prompts and 40 multimodal image-text pairs, distributed among 5 risk categories: cross-lingual manipulation, toxic language, illegal substances & weapons, jailbreak roleplay, and bias & fairness.
- DATA SPLIT – There is no train/test split predefined. The entire dataset is used for testing/evaluating the model's safety.
- DATA PROCESSING – No data filtering or outlier removal was applied; all 200 generated prompts were passed directly to the models. Raw prompts were processed using the `tokenizer.apply_chat_template` method provided by the `transformers` library. This ensured that model-specific control tokens, such as for Qwen or Zephyr, were correctly applied. Image-text pairs were processed using the `AutoProcessor` class, which handled image resizing, normalization, and conversion to tensor pixel values alongside text tokenization.
- DATA DOWNLOAD – The generated benchmark file (`safety_benchmark.json`) is generated by the script provided in the repository https://github.com/HJaczynski/NLP_2025W/blob/main/K%C5%82os_Jaczy%C5%84ski_Oganowski_Maj/reproducibility.pdf.
- NEW DATA DESCRIPTION – We created a hybrid dataset: anchors were manually collected from SOTA benchmarks like LinguaSafe, ALERT, etc., and additional samples were synthetically generated using templates and multi-turn scenarios.
- DATA LANGUAGES – The primary language is English. The 'Cross-lingual Manipulation' category includes prompts in Polish or mixed-language scenarios to test multilingual safety.