Security Testing for Large Language Models Against Jailbreak Attacks in Low-Resource Languages

Dat Tran Hung [1] **Dung Ha Minh** [2] **Duc Luu Nguyen Chi** [3]

Abstract

In this project, we'll test the safety of large language models (LLMs), focusing on low-resource languages, where safety mechanisms are more susceptible to bypassing. We'll look at Google's Gemini models to see how well they handle attempts to bypass safety filters in these languages. By testing multiple low-resource languages, we hope to find areas where these models are most vulnerable. Our goal is to better understand where the safety gaps are, so that future models can be made safer for all languages.

1. Introduction

Large language models (LLMs) are widely used in applications that interact with users, such as chatbots and virtual assistants, and are designed to filter out harmful or inappropriate responses. However, studies have shown that these safety mechanisms are less effective in low-resource languages, making it easier to bypass them (Ngo et al., 2021). For example, while LLMs like GPT-4 perform well in popular languages, they struggle to maintain the same level of security in languages with limited data. This creates a significant risk, as a recent study found that low-resource languages have a 79% success rate in bypassing these safeguards (Yong et al., 2023).

Existing research on LLM security primarily focuses on high-resource languages or isolated models, leaving gaps in understanding the safety challenges across multiple languages and models (Perez et al., 2022). Furthermore, the proprietary nature of LLMs means that much of the information about their internal safeguards is not publicly available, limiting researchers' ability to identify vulnerabilities.

In this project, we focus on testing the Google Gemini Models to examine how it handles low-resource languages in terms of security. We will systematically evaluate the model's response to various jailbreak attempts across a range of low-resource languages to identify patterns and high-risk areas. By doing so, we hope to gain insights into the specific weaknesses that exist in multilingual safety.

Our project contributes to this area by expanding the scope of LLM security testing to include multiple languages and providing a focused examination of vulnerabilities in low-resource language support. The goal of our research is to highlight areas where LLMs can be improved, encouraging the development of more secure and inclusive language models for a diverse, global audience.

2. Related Work

Jailbreaking in the context of large language models (LLMs) refers to attempts by users to bypass safety mechanisms to make the model generate restricted or harmful responses. This adversarial approach is often done by crafting prompts designed to evade safety filters, resulting in responses that would normally be blocked. Prior research in AI safety has explored various methods of preventing such attacks, including reinforcement learning from human feedback (RLHF), where models are trained to prioritize safety-relevant data, and red-teaming, where teams actively attempt to break the model's safety protocols to improve safeguards (Bai et al., 2022).

While safety testing for high-resource languages like En-

[1]22dat.th@vinuni.edu.vn [2]22dung.hm@vinuni.edu.vn [3]22duc.lnc@vinuni.edu.vn. Correspondence to: Dat Tran Hung <22dat.th@vinuni.edu.vn>.

COMP3020 Machine Learning, CECS, VinUniversity.

glish has been well-explored, the vulnerability of LLMs in low-resource languages has received limited attention. Some studies have demonstrated that switching languages or using non-standard encoding, such as base64 encoding (Wei et al., 2023), leetspeak or even Morse code (Yuan et al., 2023), can be effective ways to bypass these safety filters. For example, code-switching approaches and encoded prompts have shown success in jailbreaking models by confusing the language model's safety protocols. However, most of these studies are limited to single languages or specific LLMs, with less focus on multilingual security testing across diverse models and languages.

Our work builds on previous research by focusing on low-resource languages. We examine how LLMs like Google Gemini handle these languages and whether their safety features work well across them. By finding weaknesses in multilingual safety, our project adds to research that calls for safety measures that include all languages. This work also shows that a wider approach to LLM security is needed to protect users in different languages as LLMs are used more worldwide.

3. Method

Problem Statement

The goal of this study is to assess the resilience of the Gemini series models against jailbreaking attacks, specifically when these models are presented with inputs in Low-Resource Languages (LRL).

Approach

We will focus on two models: Gemini 1.5 Flash and Gemini 1.5 Flash 8B, selected due to their high user engagement, substantial online traffic, and extensive user base, as reported by Business of Apps (Business of Apps, 2024). Although Gemini 1.5 Pro was initially considered, its high token consumption, cost, and time-intensive processing requirements were deemed beyond the original scope of this project. Thus, we opted to exclude this model from implementation to maintain feasibility and efficiency.

The primary aim is to provide Gemini API users with effective strategies to mitigate jailbreak vulnerabilities. Given that the API for these models is currently free to access, our approach is also more cost-effective compared to testing paid models like GPT.

We will use a structured prompt dataset, translating it into various low-resource languages to assess if the models adhere to safety instructions or generate potentially harmful responses.

4. Experiments

Datasets

The experiment will use both self-created prompts and the AdvBench Benchmark dataset (Zou, Wang, Kolter, & Fredrikson, 2023), which contains 520 harmful prompts designed to test model constraints. These prompts will be translated into low-resource languages for input into the Gemini models.

Metrics

We will measure the success rate of attacks across different languages and methods. Our goal is to determine if unsafe instructions can bypass the safety mechanisms of the Gemini models and produce harmful responses. Following Wei et al. (Wei, Haghtalab, & Steinhardt, 2023), we will use a classification scheme with three labels: BYPASS if the model engages with the prompt on-topic, REJECT if the model detects the harmful nature and refuses to respond, and UNCLEAR if the response is nonsensical or neither acknowledges nor refuses harmful content.

Baselines

We will compare the result of the Gemini models' performance on LRL, with that of GPT-4's known jailbreaking defenses to evaluate Gemini's relative robustness (Yong, Menghini, & Bach, 2023). Additionally, we will assess Gemini's resilience against alternative techniques, such as the PAIR (Chao, Robey, Dobriban, Hassani, Pappas, & Wong, 2023), ACE (Handa, Zhang, Saeidi, & Baral, 2024), and FFA (Zhou, Zou, Di Eugenio, & Zhang, 2024).

Implementation Details

- Our experiments will be conducted using the Gemini API across 12 languages, including low-, medium-, and high-resource languages based on their data availability (Joshi, Santy, Budhiraja, Bali, & Choudhury, 2020) as categorized by Gemini's support list (Google Cloud, 2024). The untranslated English prompt will serve as a baseline.

- Procedure: The dataset will be translated into various languages using Google Translate and then input into the Gemini models through the API. We will categorize each response as BYPASS, UNCLEAR, or REJECT based on the classification method from HW2, and document the results.

Results and Analysis

The results from the Flash and Flash-8b models demonstrate notable differences in their ability to resist jailbreak attacks across multiple low-resource languages (LRLs).

Flash Model

Language	BYPASS (%)	BYPASS (Count)	REJECT (%)	REJECT (Count)	UNCLEAR (%)
ar	25.8	134	73.5	382	0.8
bn	24.4	127	75.4	392	0.2
en	8.1	42	91.9	478	0.0
gd	44.2	230	55.2	287	0.6
gn	69.6	362	29.8	155	0.6
ire	10.2	53	89.8	467	0.0
hi	16.5	86	83.5	434	0.0
hmn	40.8	212	58.3	303	1.0
it	17.9	93	82.1	427	0.0
th	14.0	103	86.0	447	0.0
uk	19.8	103	80.2	417	0.0
zh-cn	29.0	151	71.0	369	0.0
zu	34.2	178	64.8	337	1.0

Figure 1. Classification results for Flash model across different languages

Performance Comparison

The Flash model generally exhibits higher rejection rates across most languages. For example, in English (en), the Flash model achieves a remarkable 91.9% rejection rate, significantly outperforming Flash-8b's 78.8% rejection rate.

Flash-8b Model

Language	BYPASS (%)	BYPASS (Count)	REJECT (%)	REJECT (Count)	UNCLEAR (%)
ar	32.9	171	66.9	348	0.2
bn	44.6	232	55.2	287	0.2
en	21.2	110	78.8	410	0.0
gd	69.0	359	28.7	149	2.3
gn	83.3	433	15.8	82	1.0
he	25.4	132	74.2	386	0.4
hi	32.1	167	67.9	353	0.0
hmn	57.3	298	41.2	214	1.5
it	24.0	125	76.0	395	0.0
th	27.1	141	72.9	379	0.0
uk	26.9	140	73.1	380	0.0
zh-cn	39.6	206	60.4	314	0.0
zu	41.2	214	58.5	304	0.4

Figure 2. Classification results for Flash-8b model across different languages

This trend suggests that Flash has stronger safety mechanisms. In other languages like Irish (ire), Hindi (hi), and Italian (it), Flash consistently maintains rejection rates above 80%, while Flash-8b demonstrates weaker defenses.

The Flash-8b model displays a higher susceptibility to jailbreak attacks, with bypassing rates reaching concerning levels in Guarani (gn) at 83.3% and Scottish Gaelic (gd) at 69.0%. In comparison, Flash achieves bypassing rates of 69.6% and 44.2%, respectively, indicating a relative improvement but still leaving vulnerabilities in these languages.

Notably, across both models, unclear responses are minimal (generally under 2%), suggesting that most outputs are confidently classified as either bypass or rejection.

High-Risk Languages

Certain languages demonstrate consistent vulnerabilities across both models:

- Guarani (gn): The highest bypass rate for both models, with Flash-8b at 83.3% and Flash at 69.6%, indicating a major gap in safeguards.

- Scottish Gaelic (gd): Both models show elevated bypass rates (69.0% for Flash-8b and 44.2% for Flash) compared to other languages.

COMPARISON WITH PREVIOUS WORK

In comparison to Yong et al. (**?**), who evaluated GPT-4's jailbreak vulnerabilities, our findings reveal several key differences:

LRL Vulnerabilities: Both Gemini models show heightened vulnerability to jailbreak attempts in LRLs, particularly in Guarani and Scots Gaelic, contrasting with GPT-4's more robust defenses. This disparity stems from:

- Limited training data leading to weaker model generalization
- Insufficient safety fine-tuning compared to GPT-4's extensive adversarial training

Scaling Challenges: The Flash-8b model's inferior performance despite larger capacity highlights two critical issues:

- Trade-off between model size and safety, where increased complexity may compromise security
- Tendency to overfit safety mechanisms to high-resource language patterns

Our findings suggest that while the Gemini architecture shows promise in high-resource languages, significant improvements are needed in multilingual safety mechanisms, particularly for low-resource languages.

5. Conclusion

Our comprehensive evaluation of Gemini Flash and Flash-8b models reveals significant disparities in multilingual safety mechanisms, particularly in low-resource languages. The study yields several important findings that have implications for the development and deployment of language models in a multilingual context.

The smaller Flash model demonstrated superior safety features compared to Flash-8b across most languages, achieving notably higher rejection rates (e.g., 91.9% vs 78.8% in English). However, both models showed concerning vulnerabilities in specific low-resource languages, particularly Guarani (gn) and Scots Gaelic (gd), where bypass rates reached as high as 83.3%. This suggests that current safety mechanisms are disproportionately effective in high-resource languages while leaving significant gaps in low-resource language protection.

Notably, our findings indicate that scaling up model size does not necessarily improve safety features. The larger Flash-8b model's inferior performance highlights a critical trade-off between model capacity and security, suggesting that current approaches to model scaling may inadvertently compromise safety mechanisms in low-resource languages.

These results underscore three critical areas for improvement in LLM development:

- The need for more comprehensive safety training across diverse languages, particularly focusing on low-resource languages
- The importance of developing language-specific safety mechanisms rather than relying on approaches primarily optimized for high-resource languages
- The necessity of reconsidering the relationship between model scaling and safety features to ensure that increased model capacity does not compromise security

Future work should focus on developing more robust multilingual safety mechanisms, expanding training data for low-resource languages, and investigating alternative architectures that can maintain safety features across all languages regardless of model size.

6. Acknowledgement

We would like to thank all the professors and researchers whose past work provided the foundation on which our project was built. Their insights on AI safety and multilingual natural language processing provided important guidance as we explored the security implications of large language models in under-resourced languages.

Each of the team members contributed greatly to the project as follows:

- Tran Hung Dat: Worked on data translation and preparation, automating the translation of the AdvBench

dataset and custom prompts into low-resource languages using a robust pipeline. This system utilized the googletrans library to ensure precise row-by-row translation of CSV files while maintaining the adversarial integrity of the prompts. The pipeline processed data, starting from the second row of input files, translating all content into a target language like Hindi, and saving the results in a CSV file with the same structure as the original input. To support consistency in analysis, Dat also applied the reverse translation of the model's output responses back into English, enabling the team to examine model behavior comprehensively across languages. Challenges, such as handling non-text data and API rate limits, were addressed by implementing safeguards like skipping invalid cells and retrying logic. The translation pipeline played a critical role in preparing high-quality multilingual datasets for safety testing, preserving data integrity while optimizing for scalability and accuracy.

- Ha Minh Dung completed the technical set-up and operation of the Google Gemini API. This involved the coding and setting up of the API so that it could receive translated input prompts, process them, and store the outputs from the model. He also handled data management and integration, ensuring effective and accurate processing of inputs through the API for all testing cases.

- Luu Nguyen Chi Duc: Designed and implemented a comprehensive text classification pipeline for analyzing the Gemini model's responses. This system automatically classifies each response into one of three categories: BYPASS (1), REJECT (0), or UNCLEAR (-1). Using machine learning techniques, the approach involved creating a robust workflow that includes dataset preparation, text preprocessing, handling class imbalances with SMOTE, and training multiple machine learning models (K-NN, Decision Tree, Logistic Regression, and MLPClassifier). By using TF-IDF vectorization with bi-gram features, the system transforms text data into numerical representations, enabling efficient classification. The best-performing model is selected based on weighted F1-scores and saved for future predictions. Automating this pipeline significantly enhanced the team's ability to analyze patterns in the model's behavior when responding to prompts in lowresource languages, facilitating an efficient and scalable evaluation process. Additionally, c ensured that results were rigorously validated and saved in reusable formats for further testing and deployment.

We worked collaboratively, using the strengths of each member to achieve our project goals and address these complexities of multilingual LLM safety testing. We would like to thank everyone for their commitment and contributions toward making this research possible.

6.1. Impact Statement

The findings of this study have significant implications for the future development and deployment of multilingual large language models (LLMs). By highlighting the vulnerabilities in low-resource languages (LRLs), this research emphasizes the need for more inclusive and equitable AI systems. Current disparities in safety mechanisms risk exacerbating inequalities in access to safe and reliable AI technologies, particularly in underrepresented linguistic communities.

From an ethical perspective, these vulnerabilities raise concerns about the potential misuse of LLMs in LRL contexts, where safety mechanisms are less effective. This could lead to the dissemination of harmful or inappropriate content in these languages, disproportionately affecting already marginalized groups. Addressing these issues requires prioritizing multilingual safety in LLM development, ensuring that all users, regardless of their language, can interact with AI systems securely and effectively.

The broader societal consequences of this work extend to fostering trust and adoption of AI technologies on a global scale. Robust and equitable safety mechanisms can empower users in diverse linguistic communities, enabling more widespread use of AI for education, communication, and social development. However, failure to address the identified gaps could undermine user trust and restrict the accessibility of these technologies.

In conclusion, this study serves as a call to action for AI researchers and developers to prioritize inclusivity and safety

in multilingual AI systems. By addressing the highlighted challenges, the field can move toward a future where LLMs are both secure and equitable, ensuring their benefits reach users in every corner of the world.

References

[1] Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., et al. (2022). Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. *arXiv.org*. Retrieved from https://arxiv.org/abs/2204.05862

[2] Business of Apps. (2024). Google Gemini statistics. *Business of Apps*. Retrieved from https://www.businessofapps.com/data/google-gemini-statistics/

[3] Chao, P., Robey, A., Dobriban, E., Hassani, H., Pappas, G. J., Wong, E. (2023). Investigating AI security vulnerabilities. *arXiv preprint arXiv:2310.08419*. Retrieved from https://arxiv.org/pdf/2310.08419

[4] Google Cloud. (2024). Gemini supported languages. Retrieved from https://cloud.google.com/gemini/docs/codeassist/supported-languages

[5] Handa, D., Zhang, Z., Saeidi, A., Baral, C. (2024). Securing AI models: Challenges and solutions. *arXiv preprint arXiv:2402.10601*. Retrieved from https://arxiv.org/pdf/2402.10601

[6] Joshi, P., Santy, S., Budhiraja, A., Bali, K., Choudhury, M. (2020). The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 560-570). https://doi.org/10.18653/v1/2020.acl-main.560

[7] Ngo, H., Raterink, C., Araújo, J. G. M., Zhang, I., Chen, C., Morisot, A., Frosst, N. (2021). Mitigating harm in language models with conditional-likelihood filtration. *arXiv.org*. Retrieved from https://arxiv.org/abs/2108.07790

[8] Perez, E., Huang, S., Song, F., Cai, T., Ring, R., Aslanides, J., Glaese, A., McAleese, N., Irving, G. (2022). Red Teaming Language Models with Language Models. *arXiv.org*. Retrieved from https://arxiv.org/abs/2202.03286

[9] Wei, A., Haghtalab, N., Steinhardt, J. (2023). Jailbroken: How does LLM safety training fail? *arXiv preprint arXiv:2307.02483*. Retrieved from https://arxiv.org/abs/2307.02483

[10] Yong, Z., Menghini, C., Bach, S. H. (2023). Low-Resource Languages Jailbreak GPT-4. *arXiv.org*. Retrieved from https://arxiv.org/abs/2310.02446

[11] Yuan, Y., Jiao, W., Wang, W., Huang, J., He, P., Shi, S., Tu, Z. (2023). GPT-4 Is Too Smart To Be Safe: Stealthy Chat with LLMs via Cipher. *arXiv.org*. Retrieved from https://arxiv.org/abs/2308.06463

[12] Zhou, Y., Zou, H. P., Di Eugenio, B., Zhang, Y. (2024). Preventing jailbreak attacks on AI systems. *arXiv preprint arXiv:2407.00869*. Retrieved from https://arxiv.org/pdf/2407.00869

[13] Zou, A., Wang, Z., Kolter, J. Z., Fredrikson, M. (2023). Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*. Retrieved from https://arxiv.org/abs/2307.15043