The Lies Characters Tell: Utilizing Large Language Models to Normalize Adversarial Unicode Perturbations

Portia Cooper, Eduardo Blanco, Mihai Surdeanu
University of Arizona, Tucson, AZ, USA
portiacooper@arizona.edu

Abstract

Homoglyphs, Unicode characters that are visually homogeneous to Latin letters, are widely used to mask offensive content. Dynamic strategies are needed to combat homoglyphs as the Unicode library is ever-expanding and new substitution possibilities for Latin letters continuously emerge. The present study investigated two novel mitigation approaches that do not rely on strict mappings but instead harness the power of large language models to neutralize both known and unknown homoglyphs: (1) indirectly normalizing homoglyphs by replacing non-Latin characters with a delimiter and prompting large language models to "fill in the blanks" and (2) directly normalizing homoglyphs by using large language models to determine which characters should be replaced with Latin letters. We found that GPT-4o-mini constructed normalized text with an average cosine similarity score of 0.91 to the original tweets when applying our indirect method and 0.96 to the original tweets when applying our direct method. This study indicates that large language model-based normalization techniques can effectively unmask offensive content concealed by homoglyphs. Code and data are available in our GitHub repository.[1]

1 Introduction

A significant subset of Unicode characters are visually similar to Latin letters but possess disjointed symbolic and linguistic meanings. For example, the Latin "a" and the Cyrillic "а" appear visually homogeneous, but their underlying Unicode code points, U+0061 and U+0430, are not equivalent.

These look-alike characters are referred to as homoglyphs and present significant challenges for modern content detection models. Hate speech infused with homoglyphs lowered the F1 scores of transformer-based content detection models by 56% on the Offensive Tweets with Homoglyphs (OTH) dataset (Cooper et al., 2023). Additionally, our analysis of the OTH dataset indicated that the set of characters from which progenitors of online offensive text draw homoglyphs grew each year from 2009 to 2022 (Figure 1). While several existing normalization tools and methods exist, the use of large language models for the mitigation of homoglyphs has not been explored in detail. We present two novel methods to normalize homoglyphs. Specifically, the key contributions of our work include:

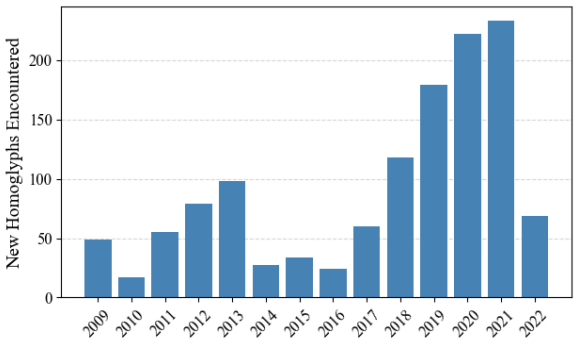

Figure 1: Number of new (previously unseen) homoglyphs detected in the Offensive Tweets with Homoglyphs dataset by year. We observed new homoglyphs yearly from 2009 to 2022.

- Creating a dataset of 29,846 tweets augmented with homoglyph patterns derived from real-world data.[1]
- Performing indirect and direct homoglyph normalization through proprietary and open-source large language models. To our knowledge, we are the first to utilize large language models to normalize homoglyphs.
- Comparing the robustness of transformer-based hate speech detection models on homoglyph-infused and normalized text.

[1] https://github.com/pcoopercoder/The-Lies-Characters-Tell

Findings of the Association for Computational Linguistics: ACL 2025, pages 18932–18944

2 Related Work

Previous work has demonstrated the effectiveness of homoglyph obfuscation. Boucher et al. (2021) used homoglyphs and other perturbations to expose vulnerabilities in models published by Facebook, IBM, and Hugging Face. Additionally, Roadhouse et al. (2024) and Valle Aguilera et al. (2024) explored the use of homoglyphs inserted in misinformation to avoid detection. And Creo and Pudasaini (2024) demonstrated that AI-generated text infused with homoglyphs avoided detection by content filters. These prior studies tested the effectiveness of homoglyphs as an adversarial strategy. Conversely, the present study investigated solutions to mitigate the malicious effects of homoglyphs.

The detection of homoglyphs has been studied in the context of domain spoofing and cyber security attacks. Almuhaideb et al. (2022) created a tool for detecting homoglyph phishing attempts by using a hash function and a machine learning model. Gupta et al. (2023) and Woodbridge et al. (2018) explored the use of convolutional neural networks to combat the usage of homoglyph characters in web domains. In addition, Ginsberg and Yu (2018) utilized hit-zone maps for predicting homoglyph characters. Unlike the previous studies, we focused on normalizing adversarial homoglyphs in offensive text.

Cooper et al. (2023) trained and evaluated transformer hate speech detection models on real-world offensive tweets with homoglyphs. And Le et al. (2022) created the ANTHRO dataset which captured over 600,000 examples of real world text perturbations and trained transformer models to detect toxic content containing homoglyphs. Kurita et al. (2019) experimented with training models on data with synthetically generated perturbations for content detection and proposed a contextual denoising autoencoder method robust against homoglyphs. Unlike prior work, the present study proposed normalization as a pre-processing strategy and evaluated both normalization and downstream task quality.

3 Approach

To ensure access to fully normalized variants when deriving our normalization methods, we augmented an existing corpus of hate and non-hate speech tweets with real-world homoglyph patterns. The tweets from the original data were used as a homoglyph-free baseline.

3.1 Formulation of the Homoglyph-Augmented Dataset

The real-world homoglyph patterns utilized in the present study were identified using a randomly selected subset ($N = 1,000$) of the OTH dataset (Cooper et al., 2023). All tokens within the subset that contained ≥ 1 homoglyph character were assigned a manually generated, normalized variant (e.g., аpple → apple (homoglyphs in red)). This process produced a total of 2,433 mappings for 982 unique words.

The pre-existing Hugging Face tweets_hate_speech_detection dataset ($N = 31,962$) by Sharma (2019) was selected for *homoglyph augmentation* as it is composed of real-world twitter data—the same corpus from which the OTH dataset was constructed (licensing information for both datasets is provided in Appendix A, subsection A.1).

The generated mappings were applied to the tweets_hate_speech_detection dataset, where any word that possessed a known homoglyph-infused mapping was replaced. If multiple mappings existed, a single one was randomly selected. Tweets with no possible word mappings were dropped. This process yielded 29,846 homoglyph-augmented tweets. This dataset was used to evaluate the effectiveness of several language models for homoglyph normalization.

3.2 Homoglyph Normalization

Two normalization strategies were applied using two large language models: GPT-4o-mini (OpenAI et al., 2024) and Llama 3.1 8B (Llama Team, 2024).

1. **Indirect normalization:** This method involved first detecting all non-Latin characters in a given text and replacing them with a selected delimiter character, such as _. Then, models were prompted to insert the missing characters into the text based on a set of in-context examples. We reasoned that indirect normalization could be a robust technique, as it does not rely on awareness of specific homoglyph characters but rather uses the surrounding Latin letters to replace homoglyphs with characters that are contextually logical.

2. **Direct normalization:** This method involved prompting models to replace all non-Latin characters with their normalized counterparts based on a set of in-context examples. By

allowing models the discretion to determine which characters to normalize, we reasoned that potential model-internal relations could be leveraged, as it is likely that both GPT-4o-mini and Llama 3.1 8B encountered homoglyphs during their initial training.

To select an optimal set of in-context examples for each model and normalization technique, we experimented with 10 random batches of $k \in \{1, 3, 5, 10\}$ in-context examples provided to both GPT-4o-mini and Llama 3.1 8B. For each k in-context examples from each batch, we applied both indirect and direct normalization on 1,000 homoglyph-augmented tweets using GPT and Llama and calculated average cosine similarity between the original (non-homoglyphed) tweets and the generated, normalized variants (Appendix A, Figure 2). Optimal prompts (reported in Appendix A, subsection A.2) were derived from the configurations with the highest scores. A discussion of prompt sensitivity is included in Appendix A, subsection A.3.

We evaluated the quality of our normalizations at scale by implementing each technique and corresponding optimal prompt on the remaining 28,746 homoglyph-augmented tweets. Data used to construct ($n = 100$) and tune ($n = 1,000$) the optimal normalization prompts were excluded. Average cosine similarity and Levenshtein distances between the normalizations and the original tweets were calculated using the Python Natural Language Toolkit (Bird and Loper, 2004). To build evidence towards the validity of our proposed normalization techniques over traditional normalization styles, we compare our methods against four tools (1) the `Unidecode` Python package,[2] (2) the `cyrtranslit` Python package (Labrèche, 2023),[3] (3) Unicode Normalization Form Canonical Composition (NFC), and (4) Unicode Normalization Form Compatibly Decomposition (NFKD). These normalization tools were chosen for comparison as they are standard and publicly available, comparably our method requires little implementation overhead. Existing machine learning methods were explored for additional comparison but were untenable due to domain and codebase issues. The average cosine similarity and Levenshtein distances between the original tweets and their homoglyph-augmented variants were also reported.

[2] https://pypi.org/project/Unidecode/

[3] https://pypi.org/project/cyrtranslit/

Normalization Approach		Average Cosine Similarity Score
Proposed Methods	GPT-4o-mini Indirect	0.91
	GPT-4o-mini Direct	**0.96**
	Llama 3.1 8B Indirect	0.81
	Llama 3.1 8B Direct	0.89
Existing Methods	Unidecode	0.78
	cyrtranslit	0.77
	NFC Form	0.64
	NFKD Form	0.62
None		0.64

Table 1: Average cosine similarity between original tweets ($N = 28,746$) and the four proposed normalization methods, four existing normalization methods, and the unnormalized homoglyph-augmented tweets.

Normalization Approach		Average Cosine Similarity Score
Proposed Methods	GPT-4o-mini Indirect	0.75
	GPT-4o-mini Direct	**0.89**
	Llama 3.1 8B Indirect	0.57
	Llama 3.1 8B Direct	0.65
Existing Methods	Unidecode	0.69
	cyrtranslit	0.69
	NFC Form	0.52
	NFKD Form	0.53
None		0.53

Table 2: Average cosine similarity between the human-normalized tweets from the Offensive Tweets with Homoglyphs evaluation data ($N = 700$) and four proposed normalization methods, four existing normalization methods, and the unnormalized real-world tweets.

3.3 Hate Speech Detection

We selected the two highest and two lowest performing transformer-based hate speech detection models from the zero-shot analysis reported by Cooper et al. (2023) for evaluation on the human-annotated sample ($N = 700$) of the OTH dataset (the models analyzed are listed in Appendix A, subsection A.4). GPT and Llama direct and indirect normalization and the four existing normalization tools were applied to the real-world homoglyphed tweets. Average cosine similarity and Levenshtein distances were calculated, and the four hate speech detection models were evaluated under both a zero-shot and five-fold cross-validation setting for each normalization method.

Normalization Approach	Model Number	A	Δ	P	Δ	R	Δ	F1	Δ
Human Normalized	1	0.70	-	0.59	-	0.83	-	0.69	-
	2	0.72	-	0.61	-	0.84	-	0.71	-
	3	0.69	-	0.65	-	0.51	-	0.57	-
	4	0.69	-	0.58	-	0.78	-	0.67	-
GPT-4o-mini Indirect	1	0.65	-0.05	0.54	-0.05	0.72	-0.11	0.62	-0.07
	2	0.65	-0.07	0.55	-0.06	0.69	-0.15	0.61	-0.10
	3	0.66	-0.03	0.63	-0.02	0.40	-0.11	0.49	-0.08
	4	0.64	-0.05	0.54	-0.04	0.64	-0.14	0.59	-0.08
GPT-4o-mini Direct	1	0.69	-0.01	0.58	-0.01	0.83	0.00	**0.68**	-0.01
	2	0.70	-0.02	0.59	-0.02	0.84	0.00	**0.69**	-0.02
	3	0.70	0.01	0.67	0.02	0.49	-0.02	**0.56**	-0.01
	4	0.68	-0.01	0.58	0.00	0.75	-0.03	**0.66**	-0.01
Llama 3.1 8B Indirect	1	0.61	-0.09	0.52	-0.07	0.38	-0.45	0.44	-0.25
	2	0.61	-0.11	0.53	-0.08	0.41	-0.43	0.46	-0.25
	3	0.63	-0.06	0.63	-0.02	0.20	-0.31	0.31	-0.26
	4	0.59	-0.10	0.49	-0.09	0.45	-0.33	0.47	-0.20
Llama 3.1 8B Direct	1	0.68	-0.02	0.60	0.01	0.59	-0.24	0.60	-0.09
	2	0.66	-0.06	0.58	-0.03	0.57	-0.27	0.57	-0.14
	3	0.68	-0.01	0.78	0.13	0.30	-0.21	0.43	-0.14
	4	0.65	-0.04	0.57	-0.01	0.56	-0.22	0.56	-0.11
None	1	0.57	-0.13	0.47	-0.12	0.57	-0.26	0.52	-0.17
	2	0.63	-0.09	0.54	-0.07	0.44	-0.40	0.49	-0.22
	3	0.61	-0.08	0.52	-0.13	0.27	-0.24	0.35	-0.22
	4	0.60	-0.09	0.49	-0.09	0.16	-0.62	0.24	-0.43

Table 3: Zero-shot accuracy (A), precision (P), recall (R), and F1 score (F1) for four hate speech detection models on the following variants of the Offensive Tweets with Homoglyphs evaluation data ($N = 700$): human normalized, GPT-4o-mini indirect and direct normalization, Llama 3.1 8B indirect and direct normalization, and unnormalized. Deltas between the human normalized and other variants presented for each metric.

4 Results

4.1 Homoglyph Normalization Quality

As shown in Table 1, the average cosine similarity scores associated with all forms of LLM-based normalization were greater than those yielded by the traditional normalization methods on the homoglyph augmented dataset. Specifically, the GPT-4o-mini direct normalizations produced the highest recorded score of 0.96.

The average cosine similarity scores reported between the normalization methods and the human-annotated sample of the OTH dataset (presented in Table 2) aligned with the scores generated on the synthetic data–the GPT-4o-mini direct normalizations yielded the largest similarity score (0.89). An error analysis of the LLM normalization methods on the OTH data is included in Appendix A, subsection A.5.

The lowest average Levenshtein distances for both the homoglyhph augmented dataset and the OTH dataset sample (1.03 and 3.44, respectively) were produced by the GPT-4o-mini direct normalization method. The full Levenshtein distances are reported in Appendix A, Table 5 and Table 6.

4.2 Hate Speech Detection Evaluation

In Table 3, we present the results of the four hate speech detection models evaluated on the human-annotated sample of the OTH dataset in a zero-shot setting. Excluding the human normalizations, the highest F1 scores (0.68, 0.69, 0.56, 0.66 for models 1–4, respectively) were achieved on the GPT-4o-mini direct normalizations.

When five-fold cross-validation was performed (Table 4), the models trained on GPT-4o-mini direct normalizations achieved the micro-averaged F1 scores (0.92, 0.92, 0.93, 0.92 for models 1–4, respectively) closest to those generated on the human normalizations (0.91, 0.94, 0.93, 0.93 for models 1–4, respectively).

The results of the models trained and evaluated on normalizations produced by the four existing tools in a zero-shot and five-fold cross validation setting are reported in Appendix A, Table 7 and Table 8. The highest F1 score achieved by any model on the normalizations produced by an existing tool was 0.52 for the zero-shot setting and 0.90 when five-fold cross validation was performed.

Normalization Approach	Model Number	A	Δ	P	Δ	R	Δ	F1	Δ
Human Normalized	1	0.92	-	0.89	-	0.92	-	0.91	-
	2	0.95	-	0.94	-	0.93	-	0.94	-
	3	0.94	-	0.93	-	0.94	-	0.93	-
	4	0.95	-	0.94	-	0.93	-	0.93	-
GPT-4o-mini Indirect	1	0.87	-0.05	0.87	-0.02	0.81	-0.11	0.84	-0.07
	2	0.88	-0.07	0.86	-0.08	0.84	-0.09	0.85	-0.09
	3	0.87	-0.07	0.85	-0.08	0.83	-0.11	0.84	-0.09
	4	0.87	-0.08	0.87	-0.07	0.81	-0.12	0.84	-0.09
GPT-4o-mini Direct	1	0.94	0.02	0.94	0.05	0.90	-0.02	**0.92**	0.01
	2	0.94	-0.01	0.92	-0.02	0.93	-0.00	**0.92**	-0.02
	3	0.95	0.01	0.93	0.00	0.94	0.00	**0.93**	0.00
	4	0.93	-0.02	0.90	-0.04	0.93	0.00	**0.92**	-0.01
Llama 3.1 8B Indirect	1	0.68	-0.24	0.78	-0.11	0.30	-0.62	0.43	-0.49
	2	0.79	-0.16	0.79	-0.15	0.66	-0.27	0.72	-0.22
	3	0.76	-0.18	0.71	-0.22	0.69	-0.25	0.70	-0.23
	4	0.77	-0.18	0.75	-0.19	0.65	-0.28	0.70	-0.23
Llama 3.1 8B Direct	1	0.78	-0.14	0.82	-0.07	0.58	-0.34	0.68	-0.23
	2	0.84	-0.11	0.81	-0.13	0.78	-0.15	0.79	-0.15
	3	0.87	-0.07	0.88	-0.05	0.78	-0.16	0.83	-0.10
	4	0.85	-0.10	0.84	-0.10	0.78	-0.15	0.81	-0.12
None	1	0.80	-0.12	0.86	-0.03	0.59	-0.33	0.70	-0.21
	2	0.90	-0.05	0.89	-0.05	0.85	-0.08	0.87	-0.07
	3	0.90	-0.04	0.87	-0.06	0.89	-0.05	0.88	-0.05
	4	0.90	-0.05	0.87	-0.07	0.89	-0.04	0.88	-0.05

Table 4: Five-fold cross-validation accuracy (A), precision (P), recall (R), and F1 score (F1) for four hate speech detection models on the following variants of the Offensive Tweets with Homoglyphs evaluation data (n = 700): human normalized, GPT-4o-mini indirect and direct normalization, Llama 3.1 8B indirect and direct normalization, and unnormalized. Deltas between the human normalized and other variants presented for each metric.

5 Discussion

For both GPT and Llama, direct normalization, in which LLMs determined which characters should be replaced with Latin letters, produced the highest quality outputs. On the homoglyph-augmented tweets, GPT's average cosine similarity scores were 0.05 higher for the direct normalizations. Similarly, Llama had a delta of 0.08 between the direct and indirect normalizations. This result is notable as it indicates that the LLMs possessed both an understanding of *what* homoglyphs were and *which* Latin letters the homoglyphs had replaced. We reason that indirect normalization was a more difficult task for the models, as 172 of the identified mappings derived from the OTH dataset sample were fully composed of homoglyphs. While training on the unnormalized, homoglyphed data granted all four hate speech detection models a significant performance boost (this aligns with the findings of Cooper et al. (2023), the models' performance on GPT direct normalizations was nearly identical to that on the human normalizations for *both* zero-shot and five-fold cross-validation. This feature is useful when large amounts of domain-specific annotated homoglyphed training data do not exist since high-quality normalizations can be produced by LLMs with 10 input examples.

Future work is needed to determine the effectiveness of the proposed normalization methods on Unicode perturbations which extend past homoglyph substitutions for the Latin letters. Specifically, additional studies may consider the evaluation of LLM-based homoglyph normalization on datasets such as ANTHRO. Further, little research currently exists which focuses on the normalization of homoglyphs of punctuation characters. Future work may include the evaluation of LLM-based normalization on perturbations of this type.

6 Conclusion

The homoglyph normalization tools of the future should not rely on rigid mappings or statically learned data. As expected, hate speech detection models performed best on human normalized text, but normalization of this type is untenable for many applications. We show that by harnessing the power of large language models, robust indirect and direct techniques for combating homoglyphs are possible.

7 Limitations

The homoglyph patterns that were used in the present study were derived from the OTH dataset that predominantly focused on the capture of tweets containing homoglyphs from the Cyrillic character family. Accordingly, our homoglyph-augmented dataset may not contain the full range of possible homoglyph characters. Thus, the data used to supply context to the large language models for the direct normalization approach in the present study likely includes only a subset of the possible perturbations. However, this is a lesser issue for the indirect normalization approach, as it has a greater reliance on the context of the non-homoglyph characters than the specific homoglyphs themselves.

8 Ethical Considerations

While the homoglyph-augmented dataset and list of homoglyphed word mappings contain adversarial homoglyph usage, we believe that the release of our data is important to facilitate future work on the study of malicious Unicode perturbations. As such, the public GitHub repository provided includes the full release of our data. If we become aware of adversarial usage of our work, data will instead be distributed to researchers on a case-by-case basis.

References

Abdullah M. Almuhaideb, Nida Aslam, Almaha Alabdullatif, Sarah Altamimi, Shooq Alothman, Amnah Alhussain, Waad Aldosari, Shikah J. Alsunaidi, and Khalid A. Alissa. 2022. Homoglyph attack detection model using machine learning and hash function. *Journal of Sensor and Actuator Networks*, 11(3).

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. 2020. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Nicholas P. Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2021. Bad characters: Imperceptible nlp attacks. *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004.

Portia Cooper, Mihai Surdeanu, and Eduardo Blanco. 2023. Hiding in plain sight: Tweets with hate speech masked by homoglyphs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2922–2929, Singapore. Association for Computational Linguistics.

Aldan Creo and Shushanta Pudasaini. 2024. Evading ai-generated content detectors using homoglyphs. *Preprint*, arXiv:2406.11239.

Avi Ginsberg and Cui Yu. 2018. Rapid homoglyph prediction and detection. In *2018 1st International Conference on Data Intelligence and Security (ICDIS)*, pages 17–23.

Akshat Gupta, Laxman Singh Tomar, and Ridhima Garg. 2023. Glyphnet: Homoglyph domains dataset and detection using attention-based convolutional neural networks. *Preprint*, arXiv:2306.10392.

Keita Kurita, Anna Belova, and Antonios Anastasopoulos. 2019. Towards robust toxic content classification. *CoRR*, abs/1912.06872.

Georges Labrèche. 2023. Cyrtranslit. A Python package for bi-directional transliteration of Cyrillic script to Latin script and vice versa. Supports transliteration for Bulgarian, Montenegrin, Macedonian, Mongolian, Russian, Serbian, Tajik, and Ukrainian.

Thai Le, Jooyoung Lee, Kevin Yen, Yifan Hu, and Dongwon Lee. 2022. Perturbations in the wild: Leveraging human-written text perturbations for realistic adversarial attack and defense. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2953–2965, Dublin, Ireland. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Nikola Ljubešić, Darja Fišer, and Tomaž Erjavec. 2019. The frenk datasets of socially unacceptable discourse in slovene and english. In *International Conference on Text, Speech and Dialogue*.

AI @ Meta Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch,

Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Charlie Roadhouse, Matthew Shardlow, and Ashley Williams. 2024. MMU NLP at CheckThat! 2024: Homoglyphs are adversarial attacks.

Roshan Sharma. 2019. Twitter-sentiment-analysis.

José Valle Aguilera, Alberto J. Gutiérrez Megías, Salud María Jiménez Zafra, Luis Alfonso Ureña López, and Eugenio Martínez Cámara. 2024. SINAI at CheckThat! 2024: Stealthy character-level adversarial attacks using homoglyphs and search, iterative.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *ACL*.

Jonathan Woodbridge, Hyrum S. Anderson, Anjum Ahuja, and Daniel Grant. 2018. Detecting homoglyph attacks with a siamese neural network. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 22–28.

A Appendix

A.1 Licensing Information for Utilized Data

Both the tweets_hate_speech_detection dataset, which was augmented, and the OTH dataset, from which the real-world homoglyphs were derived, are open-source. Our use of both aligns with the intended purposes detailed in their specified licensing.

A.2 Optimal Normalization Prompts

Figure 3 depicts the optimal GPT-4o-mini indirect normalization prompt. Figure 4 depicts the optimal GPT-4o-mini direct normalization prompt. Figure 5 depicts the optimal Llama 3.1 8B indirect normalization prompt. Figure 6 depicts the optimal Llama 3.1 8B direct normalization prompt. To access the prompts formatted as text files, please reference the /prompts folder in our GitHub repository.

A.3 Prompt Sensitivity for Normalization Models

While the standard deviations calculated for the average cosine similarity scores between the GPT-produced normalizations and the original tweets

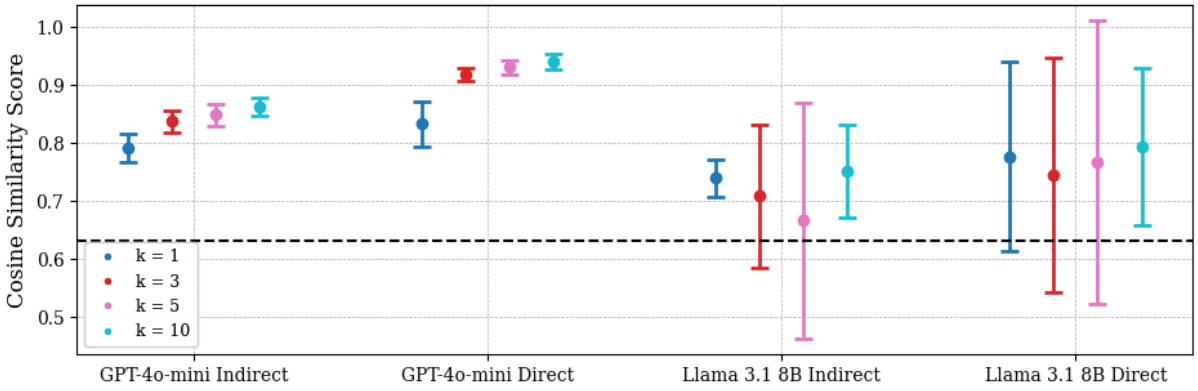

Figure 2: Results of optimal normalization prompt tuning. Average cosine similarity between original (non-homoglyphed) tweets and indirect and direct normalizations generated by GPT-4o-mini and Llama 3.1 8B using $k \in \{1, 3, 5, 10\}$ in-context examples on the associated homoglyph-augmented variants is reported. Experiment was repeated for 10 randomly selected batches of in-context examples (standard deviations reported as error bars). The black line corresponds with the average cosine similarity between original tweets and their unnormalized, homoglyph-augmented variants.

were low (range = [0.01, 0.04]) on the normalization tuning data ($n = 1,000$), Llama experienced variable performance depending on the given set of in-context examples. Across the 10 batches of $k \in \{1, 3, 5, 10\}$ in-context examples, the Llama 3.1 8B model refused to generate output on over 10.00% of the normalization tuning data, 12 times for indirect normalization and 8 times for the direct normalization. At worst, the configuration of $k = 5$ in-context examples pulled from the third randomly selected batch caused Llama to refuse output generation on 95.10% of the normalization tuning data. We observed that the high failure rates were typically associated with configurations containing ≥ 1 in-context example with inappropriate language—likely a result of the model guardrails imposed by Meta. Conversely, no failures were observed in the normalizations generated by GPT-4o-mini for any configuration across the normalization tuning data.

A.4 Hate Speech Detection Models

1. RoBERTa-base binary classification model trained on 58 million tweets. (Barbieri et al., 2020).

2. RoBERTa-base binary classification model by Liu et al. (2019) trained on the English subset of the FRENK Dataset (Ljubešić et al., 2019). *Note: This is an updated version of the model used in the OTH paper.*

3. RoBERTa-base binary classification model trained on 11 English hate speech datasets and

Normalization Approach		Average Levenshtein Distance
Proposed Methods	GPT-4o-mini Indirect	2.10
	GPT-4o-mini Direct	**1.03**
	Llama 3.1 8B Indirect	4.33
	Llama 3.1 8B Direct	2.73
Existing Methods	Unidecode	4.04
	cyrtranslit	3.93
	NFC Form	6.46
	NFKD Form	6.59
None		6.20

Table 5: Average Levenshtein distances between original tweets ($N = 28,746$) and the four proposed normalization methods, four existing normalization tools, and the unnormalized homoglyph-augmented tweets.

Normalization Approach		Average Levenshtein Distance
Proposed Methods	GPT-4o-mini Indirect	7.08
	GPT-4o-mini Direct	**3.44**
	Llama 3.1 8B Indirect	12.96
	Llama 3.1 8B Direct	10.59
Existing Methods	Unidecode	6.95
	cyrtranslit	6.95
	NFC Form	10.73
	NFKD Form	10.80
None		10.69

Table 6: Average Levenshtein distances between the human-normalized tweets from the Offensive Tweets with Homoglyphs evaluation data ($N = 700$) and the four proposed normalization tools, four existing normalization methods, and the unnormalized real-world tweets.

```
You are an assistant that inserts missing characters indicated by a _ in a given text.

For example you might receive text like:
'''uhhhh h_w l_ng __s _t ___n since __ graduated #missit'''
And you should update it to: uhhhh how long has it been since we graduated   #missit

Or you may receive text like: '''praying ƒ__ _ʟʟ affected i. ___ #orlando #attack last nite. _t matters n_t whether the_ w_re gay, wh_t matters í_ th_t _h__ w_re human.'''
And you should update it to: praying for all affected i. the #orlando #attack last nite. it matters not whether they were gay, what matters is that they were human.

Or you may receive text like: '''@user #eur/gbp recovers fr_m brexit poll led fall, _till below 0.7800 #blog #silver #gold #'''
And you should update it to:  @user #eur/gbp recovers from brexit poll led fall, still below 0.7800   #blog #silver #gold #

Or you may receive text like: '''#tfw _ c__e _o_ reads you_ snapchat ą_ď do_s n_t r_sp_nd _nd _t m_k_s y_u g___ #cancer'''
And you should update it to: #tfw a cute boy reads your snapchat and does not respond and it makes you   girl #cancer

Or you may receive text like: '''life's n_t ___ut h_w f_st __u run, it's a_o__ __w _igh ___ bounce! @user #forestpark #xtremeair5k #stl'''
And you should update it to: life's not about how fast you run, it's about how high you bounce! @user #forestpark #xtremeair5k #stl

Or you may receive text like: '''@user "today i l_st _ #thevoice sist_r senselessly - w_ sing f_r l_ve no_ d__th people." #stopthehate #ripchristina'''
And you should update it to:  @user "today i lost a #thevoice sister senselessly - we sing for love not death people." #stopthehate #ripchristina

Or you may receive text like: '''@user gr__t aicle o_ truth ___ut #trc needs. comments ba_k u_ w_y it's needed. #indignenous - via @user'''
And you should update it to: @user great aicle of truth about #trc needs. comments back up why it's needed.  #indignenous - via @user

Or you may receive text like: '''@user th_ squad summ_r camp _nd o_ week tri_ í_ ___ booked.'''
And you should update it to: @user the squad summer camp end of week trip is all booked.

Or you may receive text like: '''@user asians ne_d high_r __st scores? í_ _t harder f_r #asians _o g_t int_ college? t__ _nsw_r __ yes,'''
And you should update it to: @user asians need higher test scores? is it harder for #asians to get into college? the answer is yes,

Or you may receive text like: '''@user th_r_ ar_ s_ man_ #beautiful r__s_ns t_ ɑ_'''
And you should update it to: @user there are so many #beautiful reasons to be

Insert the missing characters indicated by _ in the following text delimited by triple single quotes.
'''{{input_str}}'''
```

Figure 3: Optimal GPT-4o-mini indirect normalization prompt.

```
You are an assistant that replaces homoglyphs characters with their Latin counterparts in a given text.

For example you might receive text like:
'''happy at work *conference: right mindset leads to culture-of-development organizations #work #mindset'''
And you should update it to: happy at work *conference: right mindset leads to culture-of-development organizations #work #mindset

Or you may recive text like: '''enjoy your life ..alex s. #day #business #relax #lifestyle #feliz #felicidad #life'''
And you should update it to: enjoy your life ..alex s. #day #business #relax #lifestyle #feliz #felicidad #life

Or you may receive text like: '''@user @user @user absolutely no! at least 3 other guys were more desserving. proves once again league's favoritism toward him'''
And you should update it to: @user @user @user absolutely no! at least 3 other guys were more desserving. proves once again league's favoritism toward him

Or you may receive text like: '''@user , shocked by your ignorance '''
And you should update it to: @user , shocked by your ignorance

Or you may receive text like: '''waterfight. the only way to keep cool thíss summer! #child #kids #family #smile #instakids'''
And you should update it to: waterfight. the only way to keep cool this summer! #child #kids   #family #smile #instakids

Or you may receive text like: '''(advanced value chain videos at ) #valuechain '''
And you should update it to: (advanced value chain videos at  ) #valuechain

Or you may receive text like: '''@user i'm an introve person, i talk little, and like to be alone.â-so ji-sub day cr:junnieuyendong @user htt'''
And you should update it to:  @user i'm an introve person, i talk little, and like to be alone.â-so ji-sub day cr:junnieuyendong @user htt

Or you may receive text like: '''guys! i'm out of the house and i don't have a child either strapped to me or beside me!! first time in almost three months'''
And you should update it to: guys! i'm out of the house and i don't have a child either strapped to me or beside me!! first time in almost three months

Or you may receive text like: '''checkout today's #trending #gif of the day! , cool, syfy, faceoff, really cool, this ís really cool via bit'''
And you should update it to: checkout today's #trending #gif of the day!  , cool, syfy, faceoff, really cool, this is really cool via bit

Or you may receive text like: '''#80 #thousand #care # workers in the #iuk are #immigrants and #most #elderly #people #say they cant even #understand #them …so #bloody'''
And you should update it to: #80 #thousand #care # workers in the #iuk are #immigrants and #most #elderly #people #say they cant even #understand #them so #bloody

Replace the homoglyphs in the following text delimited by triple single quotes.
'''{{input_str}}'''
```

Figure 4: Optimal GPT-4o-mini direct normalization prompt.

```
<SYS>
You are an assistant that inserts missing characters indicated by a _ in a given text.
Your only role is to insert the missing characters. Do NOT output anything besides the directly updated text.

Here are some examples to aid you.
You might recieve text like: '''ha___ _t work *conference: right mindset leads t_ culture-of-development organizations #work #mindset'''
And you should update it to: happy at work *conference: right mindset leads to culture-of-development organizations    #work #mindset

Or you may recive text like: '''_nj__ _our life ..alex s. #day #business #relax #lifestyle #feliz #felicidad #life '''
And you should update it to: enjoy your life ..alex s.     #day #business #relax #lifestyle #feliz #felicidad #life

Or you may recive text like: '''@user @user @user _bs_lut_l_ no! _t l_ast 3 _ther gu_s w_r_ _ore desserving. proves _n__ _gain league's favoritism toward him'''
And you should update it to: @user @user @user absolutely no! at least 3 other guys were more desserving. proves once again league's favoritism toward him

Or you may recive text like: '''@user , shocked __ y_ur ignorance '''
And you should update it to: @user , shocked by your ignorance

Or you may recive text like: '''waterfight. ___ _nl_ w__ _o k__p _o_l thí__ summer! #child #kids #family #smile #instakidsâ¦'''
And you should update it to: waterfight. the only way to keep cool this summer! #child #kids   #family #smile #instakidsâ¦

Or you may recive text like: '''(advanced value chain videos _t ) #valuechain '''
And you should update it to: (advanced value chain videos at  ) #valuechain

Or you may recive text like: '''@user i'm _n introve person, i t_lk little, _nd li__ __ ɑ_ alone.-so ji-sub day cr:junnieuyendong @user htt'''
And you should update it to:  @user i'm an introve person, i talk little, and like to be alone.-so ji-sub day cr:junnieuyendong @user htt

Or you may recive text like: '''guys! __ _ut _f ţ__ _o_se _nd i d_nt __v_ _ child either strapped _o m_ _r beside me!! fír_t tím_ _n alm_st thr__ months'''
And you should update it to: guys! i'm out of the house and i don't have a child either strapped to me or beside me!! first time in almost three months

Or you may recive text like: '''checkout today's #trending #gif _f ___ day! , cool, syfy, faceoff, r__lly cool, th__ í_ r__ll_ __ol via bit'''
And you should update it to: checkout today's #trending #gif of the day!  , cool, syfy, faceoff, really cool, this is really cool via bit

Or you may recive text like: '''#80 #thousand #care # workers _n th_ #iuk _re #immigrants _nd #most #elderly #people #say _h__ __nt ev_n #understand #them …s_ #bloody'''
And you should update it to: #80 #thousand #care # workers in the #iuk are #immigrants and #most #elderly #people #say they cant even #understand #them so #bloody

</SYS>
User: Insert the missing characters indicated by _ in the following text delimited by triple single quotes.
      '''{{input_str}}'''

You:
```

Figure 5: Optimal Llama 3.1 8B indirect normalization prompt.

```
<SYS>
You are an assistant that replaces homoglyphs characters with their Latin counterparts in a given text.
Your only role is to update the homoglyph characters. Do NOT output anything besides the directly updated text.

Here are some examples to aid you.
You might receive text like: '''@user ιм now en-route to тнe airpo as ιм off тo #cannes as ιм performing @user tomorrow. #canneslions'''
And you should update it to: @user i'm now en-route to the airpo as i'm off to #cannes as i'm performing @user tomorrow.   #canneslions

Or you may recive text like: '''dont мiss this: miriam herschlag's blog: haim yavin, мч ass via @user #journalism'''
And you should update it to: don't miss this: miriam herschlag's blog: haim yavin, my ass  via @user  #journalism

Or you may recive text like: '''just now watching season two of daredevil'''
And you should update it to: just now watching season two of daredevil

Or you may receive text like: '''тhey never show ţнe crowds. @user'''
And you should update it to: they never show the crowds. @user

Or you may receive text like: '''feliz tarde! #toptags days top.tags dач #smile #fun #instahappy #goodmood'''
And you should update it to: feliz tarde! #toptags days top.tags day #smile #fun #instahappy #goodmood

Or you may receive text like: '''finally theyre - #icecream wιll dθ that! #boysarefun'''
And you should update it to: finally they're - #icecream will do that! #boysarefun

Or you may receive text like: '''thrilled тo win. #inspired #berkshire #maidenhead #rbwm @user @user @user'''
And you should update it to: thrilled to win.   #inspired #berkshire #maidenhead #rbwm @user @user @user

Or you may receive text like: '''cuddling with мy baby....it's αℓℓ αвσut having fun!!!!! #love'''
And you should update it to: cuddling with my baby....it's all about having fun!!!!! #love

Or you may receive text like: '''@user have an amazing #weekend enjoy every moment #behappy #life #lovelife'''
And you should update it to: @user have an amazing #weekend enjoy every moment   #behappy #life #lovelife

Or you may receive text like: '''@user thanks ιм i нαvє #twitter тoo'''
And you should update it to: @user thanks i'm i have #twitter too
</SYS>

User: Replace the homoglyph characters in the following text delimited by triple single quotes.
      '''{{input_str}}'''

You:
```

Figure 6: Optimal Llama 3.1 8B direct normalization prompt.

Rounds 1 and 2 of the Dynamically Generated Hate Speech Dataset (Vidgen et al., 2021). *Note: This model was referred to as model 6 in the OTH paper.*

4. RoBERTa-base binary classification model trained on 11 English hate speech datasets and Rounds 1, 2, and 3 of the Dynamically Generated Hate Speech Dataset (Vidgen et al., 2021). *Note: This model was referred to as model 7 in the OTH paper.*

A.5 Error Analysis of Normalization Methods

Disclaimer: This section includes language that some readers might find offensive.

We analyzed a randomly selected sample (n=200) of the OTH dataset tweets to determine normalization error trends for both traditional and LLM-based normalization methods. Of the 200 GPT-4o-mini indirect normalizations, 15 contained at least one residual delimiter underscore character (_). Similarly, 20 of the indirect Llama 3.1 8B normalizations were found to have at least one residual underscore. We observed that the residual underscores occurred most frequently in tokens which started with a hashtag (#). It is likely that the models struggled with indirect normalization in these cases as underscores are typically used to represent spaces within hashtags on social media platforms.

We observed another pattern of both GPT and Llama censoring explicit words when performing direct normalization. For example, Llama 3.1 8B returned the direct normalization, "*Genie: "You have three wishes and that's it." Me: "I wish for 3 more Genies." Genie: "F*** you smart ass."*." In this case, the word which Llama replaced with "F***" initially contained only a single homoglyph (a lookalike character for the Latin letter "c")—unprompted, the model chose to censor the full word. We observed similar instances generated by GPT. For example, GPT opted to replace homoglyph characters with an asterisk instead of the letter "c" in each of the six occurrences of the F-word word, "*Fu*k the Fu*king Fu*k*ers before the Fu*king Fu*k*ers Fu*k you*." The censorship of expletives by both models was also observed for instances of the words "bitch" and "ass."

Normalization Approach	Model Number	A	Δ	P	Δ	R	Δ	F1	Δ
Human Normalized	1	0.70	-	0.59	-	0.83	-	0.69	-
	2	0.72	-	0.61	-	0.84	-	0.71	-
	3	0.69	-	0.65	-	0.51	-	0.57	-
	4	0.69	-	0.58	-	0.78	-	0.67	-
Unidecode	1	0.65	-0.05	0.81	0.22	0.16	-0.67	0.26	-0.43
	2	0.57	-0.15	0.46	-0.15	0.46	-0.38	0.46	-0.25
	3	0.54	-0.15	0.43	-0.22	0.45	-0.06	0.44	-0.13
	4	0.56	-0.13	0.45	-0.13	0.49	-0.29	0.47	-0.20
cyrtranslit	1	0.65	-0.05	0.82	0.23	0.15	-0.68	0.25	-0.44
	2	0.58	-0.14	0.48	-0.13	0.44	-0.40	0.46	-0.25
	3	0.56	-0.13	0.45	-0.20	0.47	-0.04	0.46	-0.11
	4	0.56	-0.13	0.46	-0.12	0.46	-0.32	0.46	-0.21
NFC Form	1	0.60	-0.10	0.02	-0.57	0.67	-0.16	0.04	-0.65
	2	0.60	-0.12	0.49	-0.12	0.16	-0.68	0.24	-0.47
	3	0.57	-0.12	0.47	-0.18	0.57	0.06	**0.52**	-0.05
	4	0.63	-0.06	0.54	-0.04	0.44	-0.34	**0.49**	-0.18
NFKD Form	1	0.60	-0.10	0.02	-0.57	0.67	-0.16	0.04	-0.65
	2	0.60	-0.12	0.49	-0.12	0.16	-0.68	0.24	-0.47
	3	0.57	-0.12	0.47	-0.18	0.59	0.08	**0.52**	-0.05
	4	0.63	-0.06	0.55	-0.03	0.44	-0.34	**0.49**	-0.18
None	1	0.57	-0.13	0.47	-0.12	0.57	-0.26	**0.52**	-0.17
	2	0.63	-0.09	0.54	-0.07	0.44	-0.40	**0.49**	-0.22
	3	0.61	-0.08	0.52	-0.13	0.27	-0.24	0.35	-0.22
	4	0.60	-0.09	0.49	-0.09	0.16	-0.62	0.24	-0.43

Table 7: Zero-shot accuracy (A), precision (P), recall (R), and F1 score (F1) for four hate speech detection models on the following variants of the Offensive Tweets with Homoglyphs evaluation data ($N = 700$): human normalized, Unidecode normalized, cyrtranslit normalized, NFC form normalized, NFKD form normalized, and unnormalized. Deltas between the human normalized and other variants presented for each metric.

Normalization Approach	Model Number	A	Δ	P	Δ	R	Δ	F1	Δ
Human Normalized	1	0.92	-	0.89	-	0.92	-	0.91	-
	2	0.95	-	0.94	-	0.93	-	0.94	-
	3	0.94	-	0.93	-	0.94	-	0.93	-
	4	0.95	-	0.94	-	0.93	-	0.93	-
Unidecode	1	0.63	-0.29	0.91	0.02	0.88	-0.04	**0.89**	-0.02
	2	0.85	-0.10	0.90	-0.04	0.70	-0.23	0.79	-0.15
	3	0.92	-0.02	0.89	-0.04	0.92	-0.02	**0.90**	-0.03
	4	0.92	-0.03	0.89	-0.05	0.91	-0.02	**0.90**	-0.03
cyrtranslit	1	0.92	0.00	0.90	0.01	0.89	-0.04	**0.89**	-0.02
	2	0.88	-0.07	0.87	-0.07	0.82	-0.11	0.85	-0.09
	3	0.92	-0.02	0.88	-0.05	0.93	-0.01	**0.90**	-0.03
	4	0.91	-0.04	0.88	-0.06	0.90	-0.03	0.89	-0.04
NFC Form	1	0.89	-0.03	0.88	-0.01	0.86	-0.06	0.87	-0.04
	2	0.80	-0.15	0.86	-0.08	0.59	-0.34	0.70	-0.24
	3	0.90	-0.04	0.87	-0.06	0.89	-0.05	0.88	-0.05
	4	0.90	-0.05	0.87	-0.07	0.89	-0.04	0.88	-0.05
NFKD Form	1	0.89	-0.03	0.87	-0.02	0.85	-0.07	0.86	-0.05
	2	0.78	-0.17	0.82	-0.12	0.57	-0.36	0.67	-0.27
	3	0.89	-0.05	0.89	-0.04	0.83	-0.11	0.86	-0.07
	4	0.89	-0.06	0.87	-0.07	0.86	-0.07	0.86	-0.07
None	1	0.80	-0.12	0.86	-0.03	0.59	-0.33	0.70	-0.21
	2	0.90	-0.05	0.89	-0.05	0.85	-0.08	**0.87**	-0.07
	3	0.90	-0.04	0.87	-0.06	0.89	-0.05	0.88	-0.05
	4	0.90	-0.05	0.87	-0.07	0.89	-0.04	0.88	-0.05

Table 8: Five-fold cross-validation accuracy (A), precision (P), recall (R), and F1 score (F1) for four hate speech detection models on the following variants of the Offensive Tweets with Homoglyphs evaluation data (n = 700): human normalized, Unidecode normalized, cyrtranslit normalized, NFC form normalized, NFKD form normalized, and unnormalized. Deltas between the human normalized and other variants presented for each metric.