

SINAI at CheckThat! 2024: Stealthy Character-Level Adversarial Attacks Using Homoglyphs and Iterative Search

José Valle-Aguilera^{*,†}, Alberto J. Gutiérrez-Megías[†], Salud María Jiménez-Zafra[†],
L. Alfonso Ureña-López[†] and Eugenio Martínez-Cámara[†]

Computer Science Department, SINAI, CEATIC Universidad de Jaén, Campus Las Lagunillas, 23071, Jaén, Spain.

Abstract

This paper describes the participation of the SINAI research group in Task 6 of CheckThat! Lab 2024 at CLEF. For this task, it is presented a language model adversarial attack scenario using the BODEGA framework. Given this scenario, the participants have to develop an Adversarial Attack Model Class, that is, a system that attacks a given text by modifying it, confusing the model, and making an incorrect prediction. We propose an adversarial attack experiment using search algorithms based on iterative search. Specifically, we propose a stealthy heuristic based on character level attack, modifying a letter (character) with its homoglyph. Our system obtained a 99% of preservation of meaning in the manual evaluation phase of this task, being the most stealthy proposal presented.

Keywords

Adversarial Attack, Homoglyph, Iterative Search, Stealth Attack

1. Introduction

With the rise of language models and their applications to different tasks and tools, new needs have arisen in the world of security, robustness and reliability of these models. In a scenario where language models are used to classify and label a given data, for example, spam detection, a malicious actor could modify the model's input data, which is known as an untargeted adversarial attack, or its internal architecture, which is called a targeted adversarial attack, to modify the behavior of the model, stealthily, for the user who is using it. These attacks reduce the trustworthiness of the model since its performance is made less significantly [1].

This paper presents the details and results of different types of adversarial attacks developed by the SINAI team for Task 6 of the CheckThat! lab [2] at CLEF 2024 [3][4]. The goal of this task is to develop an Adversarial Attack Model Class for attacking the input data in various language models trained for a binary classification task, changing the label of the input data via transformations. These transformations must be general attacks, and not focused on a model, because among the available models, there are three different models, BERT [5], BiLSTM [6], and RoBERTa-base [7], being the last one unknown for the participants. In addition, these Adversarial Attacks need to be stealthy [8], that is, they have to maintain a semantic and character similarity between the provided data and the result of the attack, called adversarial example. These attacks should be carried out with as few transformations as necessary while keeping the text as intact as possible. This is measured using the BODEGA framework [9], a system that provides metrics for semantic and character similarity. The data for this task consists of five datasets on different classification tasks: Propaganda Detection (PR), Fact-Checking (FC), Covid-19 (C19), Style-based News Bias Assessment (HN) and Rumour Detection (RD).

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France

*Corresponding author.

[†]These authors contributed equally.

✉ jvalle@ujaen.es (J. Valle-Aguilera); agmegias@ujaen.es (A. J. Gutiérrez-Megías); sjzafra@ujaen.es (S. M. Jiménez-Zafra); laurena@ujaen.es (L. A. Ureña-López); emcamara@ujaen.es (E. Martínez-Cámara)

ORCID 0000-0003-3274-8825 (S. M. Jiménez-Zafra); 0000-0001-7540-4059 (L. A. Ureña-López); 0000-0002-5279-8355 (E. Martínez-Cámara)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

For the development of our Adversarial Attack Class, we use different search algorithms along with character-based attacks, called heuristics, to determine which one (or which combination) offers better performance by reserving the label of the input data. We define character-based attacks as modifications, deletions, or additions of a character in a given string. Among them, we define the heuristics that modify a letter by its homoglyph and deleting simple characters such as punctuation marks or symbols. Additionally, to discover the tokens we need to attack to change the label, we have developed two search algorithms based on brute force. These search algorithms select each token iteratively to attack it with a specified heuristic, in our case, we select a homoglyph attack because it is the most stealthy to the human eye. Once we have the n most important tokens, we attack those strings with a search space, in this case, using a character-level modification homoglyphs. With this approach, we obtained a 99% of preservation of meaning in the manual evaluation phase of the competition, being the most stealthy proposal presented in this task.

The rest of the work is organized as follows: In Section 2 we describe the data used in this task. In Section 3 we define our solution, explaining all the contemplated proposals and the final attack used. We mention the results of our proposal in Section 4. We conduct an error analysis in Section 5. Finally, the conclusion can be found in Section 6.

2. Datasets

The datasets provided by the BODEGA framework are the ones used for this task. The BODEGA framework has a set represented by 5 datasets of different contexts:

- **Fact-Checking (FC):** It is the most advanced way in which human experts can verify the credibility of a given text, by assessing the veracity of the claims it includes concerning a knowledge base. It deals with Natural Language Inference (NLI) in the field of encyclopedic knowledge and newsworthy events [10].
- **Style-based news bias assessment (HN):** Assessing news credibility is usually based on three factors, writing style, veracity, and context. Training data includes statistical information to recognize sources with known credibility [11].
- **Propaganda detection: (PR):** Detecting propagandizing in a text does not necessarily imply that something is false, but also detecting the persuasion techniques used in this type of journalism. This corpus belongs to SemEval 2020 task 11 scored by practitioners on 371 articles [12].
- **Rumour detection (RD):** A rumour is information that is spread among people even if it does not originate from a reliable source. Although not all rumours have to be false, some can be confirmed later by legitimate sources [13].
- **Covid-19 (C19):** With the COVID-19 pandemic, the creation of pandemic conspiracy theories and disinformation became increasingly critical and dubious. Conspiracy theories about COVID-19 find more support among the general public than misinformation about treatment and transmissibility [14].

Table 1 shows the distribution of these datasets, which have been used to train the victim models of the competition. There is variability between the number of training data for each dataset, as we observe that datasets such as PR and RD are unbalanced.

3. Proposal description

The proposal presented for this task consists of different search algorithms, an attack heuristic and several constraints:

- **Search Algorithms:** Search systems based on finding tokens in a target text to attack it. These algorithms are based on brute force, differentiating two heuristics, one stores information to attack and the other does not.

Table 1

Distribution of the dataset: (Training) is the number of training data used to train the victim models, (Attack) is the number of instances of each dataset that are attacked and (Positive) is the percent of positive labels of the dataset.

Dataset	Training	Attack	Positive %
HN	60235	400	50.00%
PR	12675	416	29.42%
FC	172763	405	51.27%
RD	8694	415	32.68%
C19	1130	595	40.26%

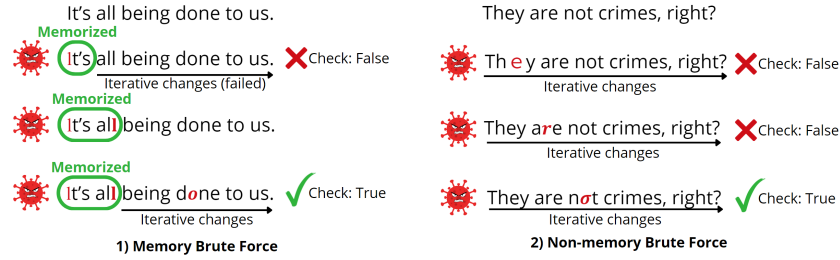


Figure 1: Types of brute force attacks implemented for this task.

- **Heuristics:** Attack algorithms that modify one or multiple elements of the input data. For this task, we have focused on homoglyph attacks.
- **Constraints:** Restrictions to take into account when attacking. These constraints can change or limit the attacks to make them less time-consuming or improve their performance.

Each of these approaches is explained in detail below and the proposal in which they are combined is presented at the end.

3.1. Search Algorithms

A search algorithm is defined as the method to select the tokens to attack which will change the predicted label. In our proposal, we base these search algorithms on brute force attacks to try every single token from the start of the text until it finishes. Also, two types of brute force attack are designed, one keeping “memory” of the previously attacked tokens, and the other without “memory”. Figure 1 shows how the search algorithm and attack heuristic work altogether.

In the case of the Non-Memory Search Algorithm, it selects the first token of the text and applies the attack heuristic. Then, the system predicts the label of the modified text, called adversarial example. If the label remains the same, the attack was unsuccessful and the search algorithm selects the next token to attack from the unmodified text until the label changes to its goal or we reach the end of the text. If the label changed to its goal, then the attack was successful and we may proceed with the subsequent provided data, starting from the beginning. If we reach the end of the text without correctly changing the label, we have failed to create an adversarial example and need to move on to the next one.

In the case of the Memory Search Algorithms, the process is similar to the previous one, with one single change. When an attack is unsuccessful, for the next iteration the search algorithm selects the next token from the previous iteration, memorizing the changes made in the text. With this approach, we “store” the changes made in the adversarial example and attack more tokens in every query we make to the model.

Now that the method for selecting a token to attack has been defined, it is time to establish the heuristics and attacks implemented in the Adversarial Attack class.

Earth is the fictional setting of much of British writer






- 
- Earth is the fictional se~~t~~ting of much of British writer
1) Homoglyph
- 
- Earth is the fictional <U+2003>setting of much of British writer
2) Invisible character
- 
- Earth is the fictional bbsetting of much of British writer
3) Character addition
- 
- Earth is the fictional sett~~i~~ng of much of British writer
4) Character deletion
- 
- Earth is the fictional setti~~o~~n of much of British writer
5) Character modification

Figure 2: Types of heuristics implemented for this task.

3.2. Heuristic

We select a character-level technique to choose the type of perturbation to use in the search space. This type of attack is more interesting, for the model and the human eye, as it makes it more difficult to identify these perturbations, with the use of imperceptible characters or small modifications [8].

The heuristic implemented for this task is a homoglyph based attack. It is a character-based modification of the input string without taking into account the meaning or structure of the objective tokens. For the development of our system, we focused on character-level attacks due to their stealthiness.

The words to be attacked are selected by brute force. Modifications are made by sweeping the target text, attacking each word individually, and checking the success of the modification. For each selected word, a letter is randomly chosen to be replaced by one of the available homoglyphs, also randomly selected, since the selection of the homoglyph does not change the final result. This heuristic has two variations explained in Section 3.4.

A homoglyph of a character is a symbol that resembles that character, usually a letter of the alphabet. This homoglyph confuses the language model, which has not been trained with these symbols, causing it to fail and change the label of the input data. In addition, since this symbol resembles an alphabet character, and has been changed to a real letter, it makes the attack very stealthy to the human eye. In order to obtain the homoglyphs of a letter, we use the Homoglyphs [15] Python library. With this library, we can obtain a list of homoglyphs of a given letter. For this approach, we randomly select a letter from the token, obtain the list of homoglyphs for this letter, and exchange it for a randomly selected homoglyph of the list.

Another character-based heuristic is the use of invisible characters. Similar to the homoglyphs, invisible characters are almost impossible to detect by the human eye and can reduce the model's precision. Finally, other character-based heuristics are deletion, modification or addition of arbitrary characters. Normally these heuristics follow a rule-based algorithm or rely simply on random modifications. However, these heuristics are easier to notice by the human eye than homoglyph attacks. Additionally, homoglyph attacks do not worsen the character and semantic score as much as other heuristics. Figure 2 shows examples of these attacks, with the homoglyph attack being the most stealthy to the human eye.

3.3. Constraints

We do small attack algorithms and restrictions that change the selection of tokens to attack or the heuristic to be used. Within the constraints, we find:

- Remove simple characters: When attacking a selected token, regardless of the attack heuristic applied, if the selected token has only one character, that token is removed from the text and it

is marked as attacked. On several occasions, simple characters are key to determining a label, such as the beginning and end of a quote (" "). Finding a homoglyph is more complicated in these cases, as it is not a letter. It is therefore considered necessary to remove this type of character.

- Exception management in attacks: When an exception occurs in an attack, the system tries to recover, normally selecting another letter or word to attack. In the case of a homoglyph attack, the letter “m” does not have any available homoglyph. In this case, the other letter is randomly selected until the attack finalizes correctly.

All these constraints are taken into account in the development of our system and can be easily activated, deactivated or configured in the Adversarial Attack class variables.

3.4. Final System

Now that each approach has been defined, we can describe our final proposal. One of the most important issues limiting our system is the execution time. Approaches such as using Memory Search Algorithms in large texts are too time-consuming. That is why we decided to limit and adapt some attack proposals, in order to sacrifice performance for more efficient approaches.

For sort texts (PR, C19 and FC datasets) we define our system as follows:

- Non-memory Search Algorithm: In the development phase, we detected that nearly 20% of the attacks were successful only modifying one token from the text. In order to make the attack less time-consuming, a first brute force attack without memory is made, that is, attack each token without maintaining the changes made in the text.
- Memory Search Algorithm: If the attack is unsuccessful, we start from the beginning of the , and initiate a brute force attack with memory, that is, maintaining the previous changes to the text until the attack succeeds or reaches the end of the text. In that case, the attack has been unsuccessful and we move on to the next text in the dataset.

For large texts (HN and RD), a different approach is taken into account, in order to reduce the time and resources consumed. In this case, only a Non-memory brute force attack is made.

4. Results

We perform the experiments with the test sets provided in the competition using our two search algorithms over RoBERTa-base as the victim model. Considering the results of attacking this new model with a character-level technique, we can conclude that we obtain a favorable result taking into account the semantic score and the Levenshtein distance (Character Score). This search space performs best on the PR2, FC and HN dataset. These datasets have a higher hit rate than the other datasets using a homoglyph-based search space.

Table 2

Final adversarial attack results using a homoglyph-based search space on the competition datasets. The victim model is RoBERTa-based, the model unknown to the participants in the test phase. PR2, FC and C19 are considered datasets of short texts, and HN and RD of long texts, which changes the search algorithm used in the attack of each one.

Dataset	Success Score	Semantic Score	Character Score	BODEGA Score
PR2	0.3052	0.8943	0.9660	0.2645
FC	0.9160	0.8648	0.9660	0.7689
C19	0.1848	0.8242	0.9961	0.1702
HN	0.4300	0.8848	0.9994	0.3802
RD	0.1012	0.8642	0.9970	0.0870

In general, the performance of the search space using homoglyphs works best with small text entries, although in the HN dataset has also performed well compared to the other long text datasets, this

may be due to the fact that datasets such as RD and PR2 being unbalanced are more difficult to attack. Due to the nature of long texts and the use of an iterative search that does not store information, it cannot compromise the decision of the model in this type of dataset. In short texts, in many cases, by removing punctuation marks or modifying a single word we are able to change the label of the victim model in a short time without the need to store information. However, in large texts, such strategies are not so successful, and using greedy techniques such as iterative search to store the information of these changes is computationally time-consuming, which in a real detection environment would be impractical.

Our final result of the competition is the average BODEGA score of each attack performed on a dataset and using a different victim model for each one. This means that the final score is the average of 15 executions, being our result 0.3507 avg. BODEGA score. This result is low considering, as shown in Table 2, that we have very disparate results, especially in the long text datasets compared to the short ones.

On the other hand, besides the automatic evaluation, a manual evaluation of a subset of the data has been carried out. Each of the attacked texts has been evaluated anonymously by two annotators, obtaining an average pairwise annotator agreement of 0.59 in Cohen’s Kappa. The results are represented in percentage which means how much they have kept the meaning of the text (N/100)%. Our work has achieved a 99% meaning score by the annotators, being the team in the competition with the highest score. We performed attacks based mostly on stealth. Therefore we chose to use character-level attacks. Among these attacks we selected homoglyphs, as they are very small modifications in the text, they are almost imperceptible to humans, so they do not change the meaning of the phrase. The results obtained in the manual evaluation highlights that, in case of obtaining only short texts or with the use of other more sophisticated techniques, the attack on character level, especially homoglyphs, can be a difficult attack to detect in a real environment.

5. Error analysis

After analyzing the results, we found that perturbing a single word in the dataset can change the outcome of the model, but in specific cases, such as longer input texts, it may be less successful. There is also a success ratio in the results if the dataset is balanced, as is the case for FC and HN. Iterative search in long texts has not been as successful as expected, so using advanced explainability methods or employing word or phrase-based techniques may be more effective in these cases. Many of the successful modifications made in short texts are due to the removal of punctuation marks.

We found that the use of search spaces based on brute-force techniques, without storing changes, can be fast but not as successful. Therefore, it is necessary to identify tokens of importance to the victim model and attack these targets using attacks such as homoglyphs, as they prove to be sufficiently stealthy for both the model and humans, and powerful enough to perform label switching.

One of the most relevant problems is the processing of long texts using these heuristics because of large processing time. One of the possible solutions could be to split the texts to be attacked in order to have a better performance using the methodology that uses memory. Another possible solution would be to identify which words are more important to modify in these longer texts, in order to increase efficiency.

6. Conclusion

In this paper an Adversarial Attack Model Class is proposed, using iterative search algorithms with an homoglyph attack. It is also worth noting the stealthiness of this attack and the difficulty of detecting them by an external agent, in addition to the fact that no advanced techniques are needed to attack widely used language models. It should be noted the stealthiness of our proposal, obtaining a 99% of meaning preservation among all datasets, being the most silent Attack Model presented in this task. Finally, future work could involve using explainability, as it allows computers to understand

human language by understanding the grammatical structure of the text and meaning of individual words. Methods could be used on long texts, splitting these long texts into smaller ones and applying different heuristics, either the iterative ones mentioned above or a mixture of greedy iterative search and explainability-based search space. Additionally, we can further refine these attacks to make them less time-consuming on large texts and try to apply explicability without penalty in time and resource consumption.

Acknowledgements

This paper has been partially supported by projects CONSENSO (PID2021-122263OB-C21), MODERATES (TED2021-130145B-I00), SocialTOX (PDC2022-133146-C21) and FedDAP (PID2020-116118GA-I00) funded by MCIN/AEI/ 10.13039/501100011033 and by the “European Union NextGenerationEU/PRTR”. The research work conducted by Salud María Jiménez-Zafra has been supported by Action 7 from Universidad de Jaén under the Operational Plan for Research Support 2023-2024.

References

- [1] N. Papernot, P. McDaniel, A. Sinha, M. P. Wellman, Sok: Security and privacy in machine learning, in: 2018 IEEE European Symposium on Security and Privacy (EuroSP), 2018, pp. 399–414. doi:10.1109/EuroSP.2018.00035.
- [2] P. Przybyła, B. Wu, A. Shvets, Y. Mu, K. C. Sheang, X. Song, H. Saggion, Overview of the CLEF-2024 CheckThat! lab task 6 on robustness of credibility assessment with adversarial examples (incrediblae), 2024.
- [3] A. Barrón-Cedeño, F. Alam, T. Chakraborty, T. Elsayed, P. Nakov, P. Przybyła, J. M. Struß, F. Haouari, M. Hasanain, F. Ruggeri, X. Song, R. Suwaileh, The clef-2024 checkthat! lab: Check-worthiness, subjectivity, persuasion, roles, authorities, and adversarial robustness, in: N. Goharian, N. Tonello, Y. He, A. Lipani, G. McDonald, C. Macdonald, I. Ounis (Eds.), Advances in Information Retrieval, Springer Nature Switzerland, Cham, 2024, pp. 449–458.
- [4] A. Barrón-Cedeño, F. Alam, J. M. Struß, P. Nakov, T. Chakraborty, T. Elsayed, P. Przybyła, T. Caselli, G. Da San Martino, F. Haouari, C. Li, J. Piskorski, F. Ruggeri, X. Song, R. Suwaileh, Overview of the CLEF-2024 CheckThat! Lab: Check-worthiness, subjectivity, persuasion, roles, authorities and adversarial robustness, in: L. Goeuriot, P. Mulhem, G. Quénot, D. Schwab, L. Soulier, G. M. Di Nunzio, P. Galušćáková, A. García Seco de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024), 2024.
- [5] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: <https://aclanthology.org/N19-1423>. doi:10.18653/v1/N19-1423.
- [6] Z. Huang, W. Xu, K. Yu, Bidirectional lstm-crf models for sequence tagging, arXiv preprint arXiv:1508.01991 (2015).
- [7] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized BERT pretraining approach, CoRR abs/1907.11692 (2019). URL: <http://arxiv.org/abs/1907.11692>. arXiv:1907.11692.
- [8] N. Boucher, I. Shumailov, R. Anderson, N. Papernot, Bad characters: Imperceptible nlp attacks, in: 2022 IEEE Symposium on Security and Privacy (SP), IEEE, 2022, pp. 1987–2004.
- [9] P. Przybyła, A. Shvets, H. Saggion, Bodega: Benchmark for adversarial example generation in credibility assessment, arXiv preprint arXiv:2303.08032 (2023).

- [10] J. Thorne, A. Vlachos, C. Christodoulopoulos, A. Mittal, FEVER: a large-scale dataset for fact extraction and VERification, in: NAACL-HLT, 2018.
- [11] J. Kiesel, M. Mestre, R. Shukla, E. Vincent, D. Corney, P. Adineh, B. Stein, M. Potthast, Data for pan at semeval 2019 task 4: Hyperpartisan news detection, November. type: dataset (2018).
- [12] G. Da San Martino, A. Barrón-Cedeño, H. Wachsmuth, R. Petrov, P. Nakov, Semeval-2020 task 11: Detection of propaganda techniques in news articles, in: Proceedings of the Fourteenth Workshop on Semantic Evaluation, 2020, pp. 1377–1414.
- [13] S. Han, J. Gao, F. Ciravegna, Augmented dataset of rumours and non-rumours for rumour detection, 2019. doi:10.5281/zenodo.3249977.
- [14] A. M. Enders, J. E. Uscinski, C. Klostad, J. Stoler, The different forms of covid-19 misinformation and their consequences, The Harvard Kennedy School Misinformation Review (2020).
- [15] Homoglyphs: get similar letters, convert to ascii, detect possible languages and utf-8 group., <https://pypi.org/project/homoglyphs/#description>, 2022. [Accessed 31-05-2024].