Report. Solution steps.

1. Processing a raw dataset

Right after reading the assignment requirements, I downloaded and explored the dataset.
The first thing that I noticed was the presence of highly toxic sentences in both the reference and translation columns. Translation is not non-toxic as I first thought. This made me think that using reference as input and translation as approximate target, or vice versa, might not be a good idea.
After examining some specific sentences for unusual wording or abnormalities in text, I decided to conduct a literature review of existing methods for text detoxification.

2. Baseline search

One of the first articles I found was a paper by Tang et. al [1], which gave me an idea of a model architecture that could be used to solve the proposed problem.
Also, I found other papers proposing solutions to this problem [2, 3, 4].

3. Model architecture

In summary, I decided to use the following algorithm as the basis for the architecture:

3.1. Classification of a sentence as toxic or not.
3.2. For toxic sentences, searching for toxic words in it.
3.3. Masking the toxic words and solving the fill-mask problem.

4. Binary classification

I formulated a hypothesis that the raw dataset can be preprocessed and a binary classification model (toxic/non-toxic) can be trained based on it.

Hypothesis 1: Use the custom binary classification model trained on the preprocessed raw dataset.

This hypothesis was only partially confirmed, since it obtained "bad" results.
The poor results of the binary classifier led me to think that it was redundant at all.

Hypothesis 2: A custom binary classifier can reject toxic sentences by labeling them as non-toxic. In other words, it creates redundant bias for the final prediction.

However, on the other hand, it works as a filter for non-toxic sentences. So, it was decided to leave the model, but change the architecture.
For this reason, it was made an attempt to search pre-trained models for word classification. The model facebook/bart-large-mnli in zero-shot-classification mode from the transformers library was taken as a primary. This model showed much better results in the data classification process.

5. Model masker

To construct the masker model, it was necessary to find a dataset consisting of texts whose parts/words/tokens were mapped to one of two classes: toxic or non-toxic.
I found such a dataset in the repository of the user hexinz [5]. There were actually two datasets, one was a raw dataset and the other was a preprocessed dataset.

Hypothesis 3: The preprocessed dataset would be perfectly suitable for the problem.

However, unfortunately, I was not able to bring the preprocessed dataset to the form I needed, because during the processing the user hexinz made assumptions that were inappropriate for my task.
I took the raw dataset and reduced it to the following structure:
(sentence_id, token, class), where class is toxic/non-toxic.

Using my preprocessed dataset, I successfully trained an LSTM cell-based model for marking non-punctuated sentence tokens using toxic/non-toxic classes, achieving an accuracy > 0.98.

6. Final model

For the fill-mask problem, it was decided to use the pre-trained distilbert-base-uncased model to replace masked words with non-toxic expressions.

Hypothesis 4: The pre-trained distilbert-base-uncased model "doesn't know" rude, foul, toxic and "bad" words.

As it turned out, sometimes the model suggested replacing the masked word with a toxic word and in some cases even more rude than the original one).

This required an additional post-processor which checked the proposed word for toxicity. This post-processor was the same masker used previously, since the first step was to classify the tokens as toxic or not.
In addition, to achieve semantic similarity between the paraphrased sentence and the original sentence, 5 non-toxic words are proposed to replace each toxic word.

The word with which the embeddings of the two sentences (constructed using the distilbert-base-nli-mean-tokens model) were closest is selected.

As a result, the resulting model is quite complex in structure but simple in idea, and it was tested on the raw dataset sought. For 1000 examples, the model reduced the number of toxic sentences by 34%.

References

[1] Z. Tang, K. Zhou, P. Wang, Y. Ding, J. Li, and Minzhang, “Detoxify language model step-by-step,” *arXiv [cs.CL]*, 2023.

[2] S. Hallinan, A. Liu, Y. Choi, and M. Sap, “Detoxifying text with MaRCo: Controllable revision with experts and anti-experts,” in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2023.

[3] D. Dale *et al.*, “Text detoxification using large pre-trained neural models,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

[4] G. Floto *et al.*, “DiffuDetox: A mixed diffusion model for text detoxification,” in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023.

[5] Dataset: https://github.com/hexinz/SI630_final_project/tree/main/Data

[6] Fill-mask tutorial: https://huggingface.co/learn/nlp-course/chapter7/3?fw=pt

[7] Zero-shot tutorial: https://joeddav.github.io/blog/2020/05/29/ZSL.html