Report on Solution Creation

Leila Khaertdinova, BS21 DS-02

November 2023

Baseline: Delete

In this report, I will discuss and outline the process steps I followed to create a solution for the task of text detoxification.

This baseline approach that firstly comes to my mind draws inspiration from how offensive language is handled in TV programs and articles, involves substituting swear words with asterisks (*). Following a similar principle, goal of this baseline is to just delete offensive words from the source text with a high toxicity level. To implement this approach, a dictionary was created, consisting of the bad words obtained from the obscenity list from the Github repository mentioned in the references directory.

This approach does not involve training loop, so I just used it on a test set containing 5000 text sentences.

Example:

Source sentence	**Detoxified sentence**
"What a stupid joke."	"What a joke."
"Fucking damn joke!"	"joke!"

Table 1: Baseline results

As can be seen from the table above, the bad words from the source sentences were removed. However, this approach is not the perfect one as it may result in the loss of contextual information within the original toxic sentence.

Hypothesis 1: Pytorch transfromer

As an alternative to the baseline delete approach, I explored the use of a Seq2Seq models for solving machine translation tasks. However, they also can be applied for text detoxification.

The hypothesis behind this approach is that by leveraging the power of a transformer model, we can replace offensive language style with the same meaning but in a neutral manner. In this step, I trained Transformer from

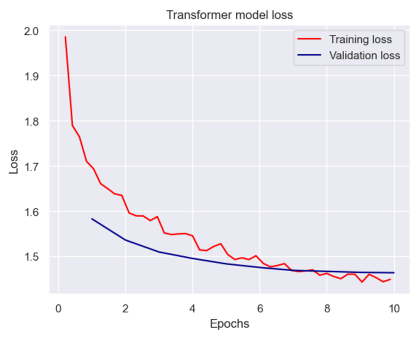


Figure 1: Pytorch transformer losses during training

scratch using PyTorch with batch size 32 during 10 epochs. The training and validation losses are shown in Figure 1.

Example:

Source sentence	**Detoxified sentence**
"What a stupid joke."	"what a joke."
"Fucking damn joke!"	"you got ta be kidding me."

Table 2: Pytorch transformer results

Consequently, the Pytorch transformer provides quite good results in a non-toxic manner, but some information from the original example sentences is missing. As shown in the table, the detoxified sentences are less expressive than the original ones. Therefore, further improvements are needed to make the detoxified sentences more informative and natural-sounding.

Hypothesis 2: Fine-tuning T5 model

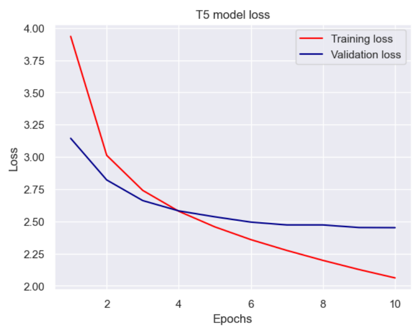
For this step, I explored the fine-tuning process for the given task. I used a pre-trained T5 model with T5-Base checkpoint from the HuggingFace and fine-tune it with batch size 16 for 10 training epochs (for bigger size of batch the GPU limit appeared). See Figures 2 and 3 to check the training logs.

Example:

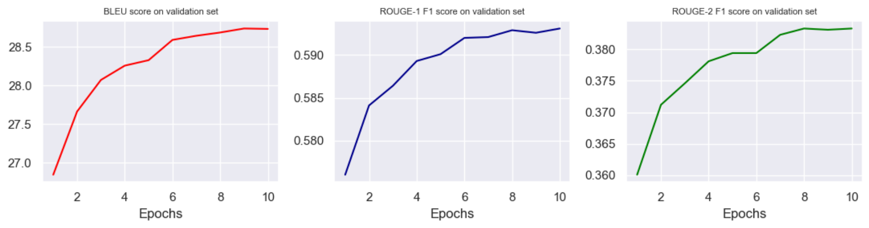
Source sentence	**Detoxified sentence**
"What a stupid joke."	"what a bad joke."
"Fucking damn joke!"	"it's a terrible joke!"

Table 3: T5 fine-tuned results

Based on the these examples, it can be concluded that the model can detoxify toxic sentences and provide non-toxic alternatives in a natural way.



(a) Losses during training



(b) Metrics on validation set

Figure 2: T5 model training results

Results

Here is a table presenting the performance comparison of different solutions created to address the text detoxification problem. The metrics were evaluated on a test set consisting of 5000 examples.

	BLEU-2	BLEU-4	ROUGE-1	ROUGE-2	Toxicity ratio
Baseline	0.3129	0.1148	0.5578	0.3063	0.1182
Pytorch transformer	0.3443	0.1499	0.5656	0.3469	0.089
T5 fine-tuned	**0.4236**	**0.2119**	**0.5982**	**0.3921**	**0.0592**

Table 4: Metrics on test set

As observed, the last approach, which involves fine-tuning the t5 model, demonstrates the highest metrics. For example, for t5 the toxicity ratio metric is nearly 0.059 while the target detoxified sentences metric stands at 0.062. This indicates that the approach delivers promising results.