Lexical Normalization as a Machine Translation problem

Constantin Gabriel-Adrian (507)
Sociu Daniel (507)

Problem definition

Task definition

- Lexical Normalization:
  - Involves correcting the data to the usual canonical form
  - Correcting = transforming an abbreviation or mistaken words to the correct grammatical or dictionary form
- Example:
  - Original: new pix comming tomoroe
  - Corrected: new pictures coming tomorrow

What are we trying to achieve ?

- Multilingual model that can do lexical normalization on a combination of datasets
- Quality annotated data = a pre-processing method for various datasets like:
    - comments
    - posts from social media sources
- Typos are very common, therefore a fast and reliable model to solve them is important

Dataset

MultiLexNorm

The processed dataset turned out to have:

- 1200 training samples
- 1200 validation samples

Language	Data from	Original Source	Size (#words)
Croatian	Twitter	Ljubešić et al, 2017 [bib]	75,276
Danish	Twitter/Arto	Plank et al, 2020 [bib]	11,816
Dutch	Twitter/sms/forum	Schuur, 2020 [bib]	23,053
English	Twitter	Baldwin et al, 2015 [bib]	73,806
German	Twitter	Sidarenka et al, 2013 [bib]	25,157
Indonesian-English	Twitter	Barik et al, 2019 [bib]	23,124
Italian	Twitter	van der Goot et al, 2020 [bib]	14,641
Serbian	Twitter	Ljubešić et al, 2017 [bib]	91,738
Slovenian	Twitter	Erjavec et al, 2017 [bib]	75,276
Spanish	Twitter	Alegria et al, 2013 [bib]	13,827
Turkish	Twitter	Çolakoğlu et al, 2019 [bib]	7,949
Turkish-German	Twitter	van der Goot & Çetinoglu [bib]	16,546

```
if       If
i        i
have     have
a        a
head     headache
ache
tomorro  tomorrow
ima      i'm going to
be       be
pissed   pissed
```

Proposed solution

T5-model

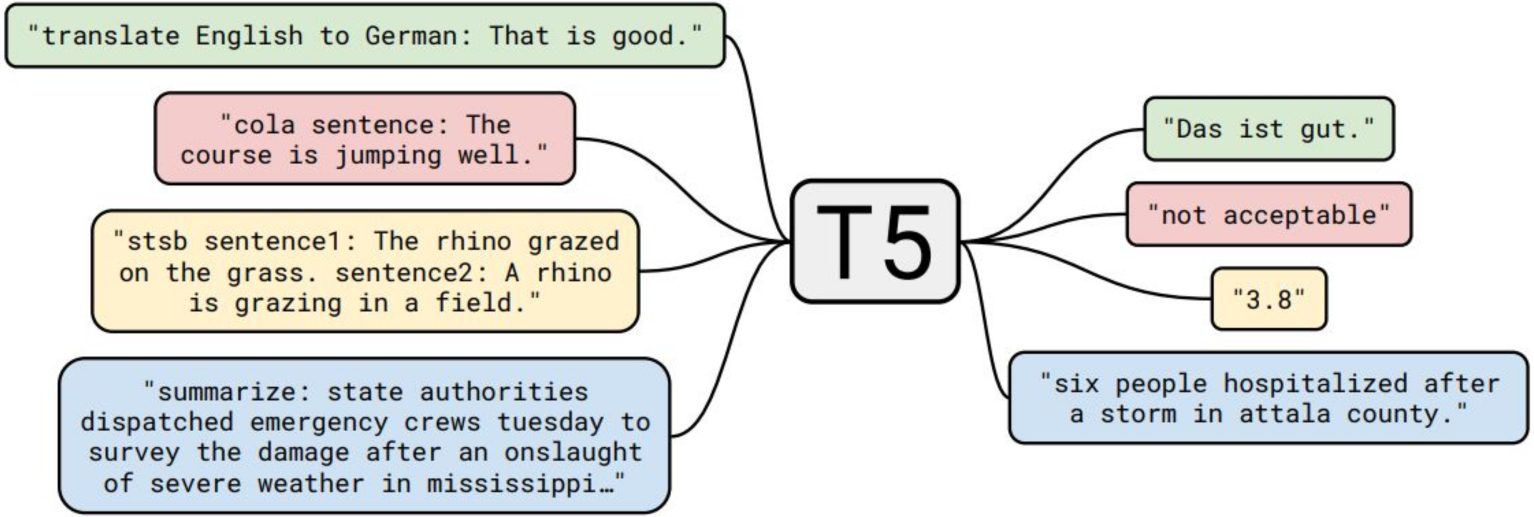
- Text-to-text transformer
- Follows the classic encoder-decoder approach
- Prepends a different prefix to the input corresponding to each task and uses it during training and inference
- Pretrained on text classification, question answering, text sumarisation and even machine translation

- Examples:
    - for translation: “translate English to German: …”
    - for summarization: “summarize: ….”

T5-model

- mT5 model is the multilingual version of the T5 model
  - it is trained on more languages
- The T5 models are pre-trained on both supervised and self-supervised tasks:
  - Supervised training is conducted on downstream tasks provided by the GLUE and SuperGLUE benchmarks
  - Self-supervised training uses corrupted tokens, by randomly removing 15% of the tokens and replacing them with individual sentinel tokens
- Architectures we used to train:
  - T5-small (60M params)
  - mT5-small (Multilingual T5-small) (60M params)

T5-model


"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Experimental Setup

nanoT5

- Specifically optimized to fine-tune big models efficiently
- Maintains close accuracy to the original model
- Implemented in PyTorch
- Uses Hydra for config handling and model parametrization
- Uses Accelerator for fast implementation of training pipeline

Training tricks & hyper-parameters

- AdamW as optimizer
- Dynamic learning rate changes
  - LambdaLR
  - ReduceLROnPlateau
- About 25 epochs of training, with Early Stopping
- Saved best model based on validation accuracy
  - Evaluated after each training epoch

Metrics used

- Accuracy
  - Number of correct samples divided by the number of total samples.
- Rouge-L
  - Relies on the longest common subsequence
  - Measures the number of matching n-grams between reference text and the generated output text
  - Used mainly for summarisation and machine translation

$$F1 = \frac{2 \times Precision \times Recall}{(Precision + Recall)}$$

$$R_{lcs} = \frac{LCS(X,Y)}{m}$$

$$P_{lcs} = \frac{LCS(X,Y)}{n}$$

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

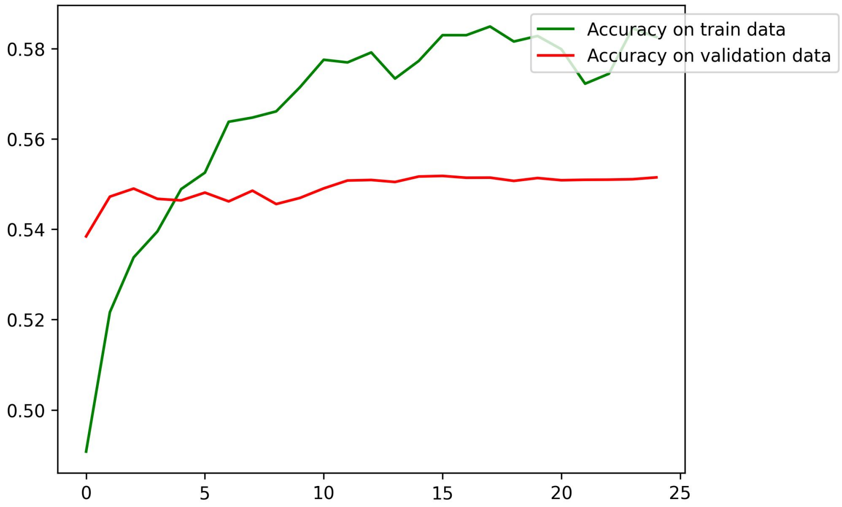
Results

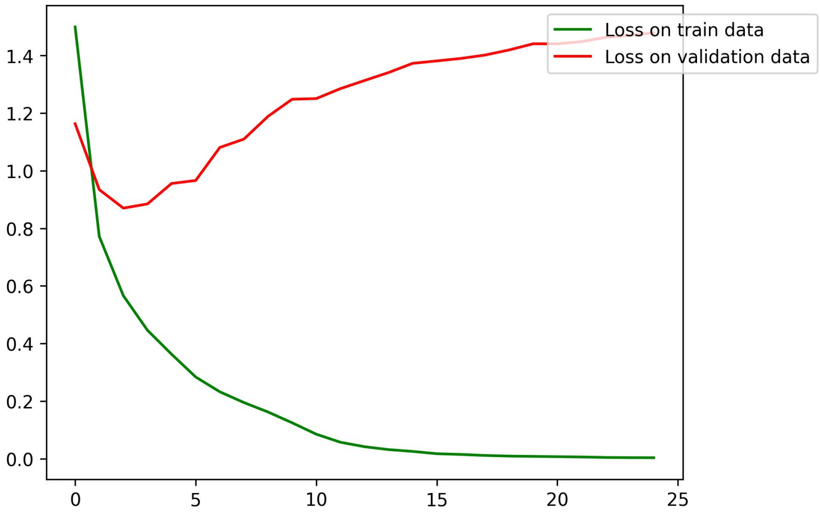
Results

Model	Dataset Lang.	Batch size	Scheduler	Train Acc.	Val Acc.	Rouge-L
T5-small	EN	16	LambdaLR	58.5	53.5	50.2
T5-small	Multilingual	16	LambdaLR	42.5	41.8	60.5
mT5-small	Multilingual	8	LambdaLR	48.0	45.9	70.8
T5-small	EN	4	ReduceLROnPlateau	*65.9*	*58.8*	*54.6*
T5-small	Multilingual	4	ReduceLROnPlateau	**59.7**	**55.1**	**71.3**
mT5-small	Multilingual	4	ReduceLROnPlateau	58.2	53.8	67.8

Table 1: Results of training on MultiLexNorm dataset using nanoT5. In **bold**, the best performance on multilingual dataset and in *italic* the best performance on English

Plots - best model





Conclusions and Future Work

Conclusions

- We demonstrated that a multi-lingual approach for lexical normalization is feasible using transformers.
- More computational resources needed for base or higher model testing, or different architectures such as Mixtral.
- More experimentation can be done with layers frozen at different depths.

Thank you!