ICL's Submission to the WMT21 Critical Error Detection Shared Task

Genze Jiang **Zhenhao Li** **Lucia Specia**
Language and Multimodal AI (LAMA) Lab, Imperial College London, UK
`{genze.jiang20, zhenhao.li18, l.specia}@imperial.ac.uk`

Abstract

This paper presents Imperial College London's submissions to the WMT21 Quality Estimation (QE) Shared Task 3: Critical Error Detection. Our approach builds on cross-lingual pre-trained representations in a sequence classification model. We improve the base classifier by (i) adding a weighted sampler to deal with imbalanced data and (ii) introducing feature engineering, where features related to toxicity, named-entities and sentiment, which are potentially indicative of critical errors, are extracted using existing tools and integrated to the model in different ways. We train models with one type of feature at a time and ensemble those models that improve over the base classifier on the development set. Our official submissions achieve very competitive results, ranking second for three out of four language pairs.

1 Introduction

Critical Error Detection (CED) is a new task which has been introduced in the WMT21 Quality Estimation (QE) Shared Task.[1] The purpose of CED is to address a challenging problem in Machine Translation (MT): translations produced by state-of-the-art MT systems can be grammatical and fluent but do not always retain the meaning of the source text. More importantly, incorrect translations can be misleading and even have catastrophic consequences such as health, safety, legal, or financial implications. However, these can be hard errors to capture by general QE architectures, which have been shown to be prone towards relying mainly on the translated sentence (Sun et al., 2020).

According to the Shared Task definition, a critical translation error is a type of error that occurs when the meaning of the translation deviates from source sentence in a critical way. The task data (Section 2.1) includes five categories of such errors: deviation in toxicity (TOX), in named entities (NAM), in sentiment polarity or negation (SEN), or in numbers (NUM), or introduction of health or safety risks (SAF).

The baseline model for this task utilises the XLM-RoBERTa (Conneau et al., 2020) for sequence classification model, following the MonoTransQuest architecture proposed by Ranasinghe et al. (2020). Inspired by the fact that these five critical error types refer to specific linguistic phenomena, we aim to bring additional information to the models on the presence of such phenomena. The intuition is that sentences containing certain types of linguistic features, such as named entities or dates, are more likely to lead to errors. Therefore, we first process the dataset to extract features reflecting the sentences' toxicity, sentiment and named entities, using off-the-shelf toolkits or APIs (Section 2.2). We then enhance the baseline architecture with this additional information.[2]

We experiment with two approaches to take the additional features into account, at token and hidden state levels. We build multiple models taking one type of feature at a time and finally ensemble "promising" models. Promising models are those that lead to improvements over the baseline on the dev set (Section 2.3).

Our results comparing different features show that some of the features are indeed useful, but there is no general pattern that applies to all language pairs (Section 3.1). The official submission, which uses an ensemble of the models that lead to improvements over the baseline on the dev set for each language shows that ensembling only models with promising features are better than ensembling models with all kinds of features (Section 3.2). Upon manual inspection, we observed that additional features indeed help the model to make predictions but this is subject to the accuracy of features (Section 3.3).

[1] http://statmt.org/wmt21/quality-estimation-task.html

[2] Our code and data are available from https://github.com/conanjgz/critical-error-detection-for-MT

2 Experiment Settings

2.1 Dataset

According to the description of WMT21 CED Shared Task, the dataset for this task was collected from Wikipedia comments (Wulczyn et al., 2017) in English with translations generated by the ML50 multilingual translation model (Tang et al., 2020), consisting of four language pairs: English-Czech (En-Cs), English-German (En-De), English-Japanese (En-Ja) and English-Chinese (En-Zh). The number of data samples in the training set differs for the four language pairs but is around 6500-8000. Each language pair has 1000 data samples in the dev set and 1000 data samples in the test set. For each sentence pair in the dataset, there are three labels given by three human annotators. The three labels are aggregated to the final label of the dataset using majority strategy. The final label is either `ERR` or `NOT`, where `ERR` means the translation has at least one critical error and `NOT` means the translation does not have a critical error.

Pair	Dataset	Count	Label	Count
En-Cs	Train set	7476	NOT	6188
			ERR	1288
	Dev set	1000	NOT	840
			ERR	160
	Test set	1000	–	–
En-De	Train set	7878	NOT	5674
			ERR	2204
	Dev set	1000	NOT	719
			ERR	281
	Test set	1000	–	–
En-Ja	Train set	7658	NOT	6939
			ERR	719
	Dev set	1000	NOT	904
			ERR	96
	Test set	1000	–	–
En-Zh	Train set	6859	NOT	5749
			ERR	1110
	Dev set	1000	NOT	859
			ERR	141
	Test set	1000	–	–

Table 1: Statistics of datasets for four language pairs. The distribution of labels for the test set is unknown as this is a blind evaluation task.

The dataset information for each language pair can be found in Table 1. As can be seen, the data is very imbalanced, with the En-Ja dataset suffering the most: the `ERR` label only accounts for 9.4% in the training set. The En-De dataset is the least imbalanced compared to other three language pairs, where the proportion of `ERR` label in En-De training set reaches 27.9%.

2.2 Features

We extract features reflecting sentences' toxicity score, sentiment and named entities. The expectation is that these features could be helpful in detecting critical errors since these errors stem from issues with the translation/introduction of these and other linguistic phenomena. Ideally we would have wanted to extract this information for both source and translated sentences to be able to perform some sort of comparison between the two, for example, presence of toxicity in the translation but not in the source sentence. However, we are limited by the availability of tools in the four language pairs, as we explain below.

For all features, our goal is to have a discrete representation which will allow us to easily incorporate them to the architecture, as will be explained in Section 2.3.2. Therefore, we need to threshold some of these features.

The toxicity score is produced by Perspective API,[3] which supports only English and German amongst our five languages. Based on some manual inspection of the predictions by Perspective, we consider that if the toxicity score of a sentence is greater than 0.5, the sentence will be regarded as toxic. We leave for future work experiments varying this threshold. Since this API does not support Czech, Japanese and Chinese, we were only able to extract a toxicity feature in the source sentences for En-Cs, En-Ja and En-Zh.

The sentiment score is produced by Google Cloud Natural Language API,[4] which supports English, German, Japanese and Chinese. Therefore, we can get the sentiment feature of both source sentence and translation for En-De, En-Ja and En-Zh. The score returned by this API is a float number ranged from -1 to 1. Empirically, we consider a sentence to be negative if the score is smaller than -0.2, and positive if the score is greater than 0.2, otherwise the sentence's sentiment is neutral. In our experiments, the sentiment feature is not applied to En-Cs because Czech is not supported by this API.

The information of named entities (NE) is extracted using spaCy,[5] which can recognise over 15 NE types. We count the number of named entities for each NE type and finally choose seven NE types with the highest counts as features. The description

[3]https://www.perspectiveapi.com/
[4]https://cloud.google.com/natural-language
[5]https://spacy.io/

of the seven NE types can be found in Table 2. We extract named entities in both source sentence and translation for En-De, En-Ja and En-Zh. However, Czech is not supported by spaCy, therefore we do not use NE features for En-Cs.

Type	Description	Abbr.
ORG	Organisation name	ORG
PERSON	Person name	PER
DATE	Year, month or day	DAT
CARDINAL	Numerals	CRD
ORDINAL	Ordinal numerals	ORD
NORP	Religious group, etc	NRP
GPE	Geographical name	GPE

Table 2: Descriptions of seven types of NE features and their abbreviations.

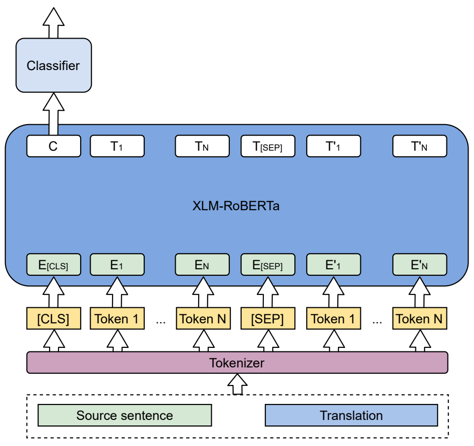

Figure 1: Architecture of baseline model. This is a MonoTransQuest model where we pass the output of the [CLS] token to a classifier.

2.3 Models

2.3.1 Baseline Model

The baseline model employs the MonoTransQuest framework (Ranasinghe et al., 2020), which is proposed for general quality estimation (QE) tasks and is shown in Figure 1. Essentially this is used to produce the baseline score of CED Shared Task. The model is based on a pre-trained XLM-RoBERTa transformer model (Conneau et al., 2020) and is used to perform sentence-level classification tasks. The model takes a sequence of tokens as input which starts with <s>, denoting [CLS] token, followed by tokens for source sentence and translation and ended with </s> token. The source sentence and its translation, separated by [SEP] token, are fed into one single transformer encoder at the same time. Then the output of the transformer encoder is fed into a classification head where cross-entropy is adopted as the loss function. We use pre-trained XLM-RoBERTa models released by HuggingFace's model repository (Wolf et al., 2020) for the implementation.

To alleviate the influence of imbalanced training data, a weighted sampler can be applied to the data loader during training. The weighted sampler is to make the label distribution in the training batch as balanced as possible. The weight of the sampler is computed as reciprocals of label proportions.

2.3.2 Model with Features

To utilise the features mentioned in Section 2.2, we proposed two different approaches.

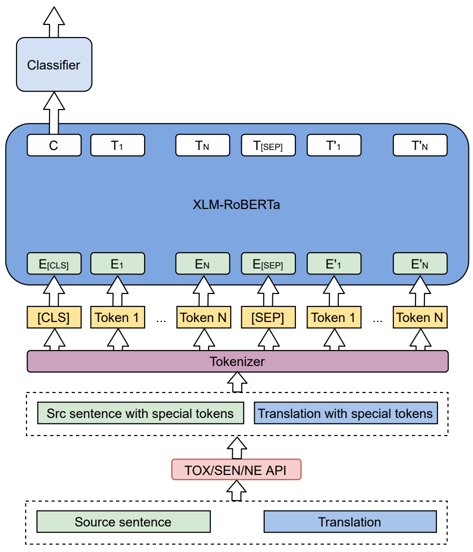

Figure 2: Architecture of the first approach (adding special tokens). We insert TOX/SEN/NE information to the source sentence and its translation as special tokens, and then feed sentences with special tokens to the baseline architecture.

The first approach (shown in Figure 2) is to **add special tokens**. Here the features (toxicity, sentiment, named entities) are directly inserted as special tokens to the input source sentence and, where available, its translation before getting tokenised. To correctly tokenise sentences with features, these special tokens are also added to the XLM-RoBERTa tokeniser. The remaining architecture is the same as the baseline model except for the dimension of model's word embeddings as the model's token embeddings should be resized when adding new tokens.

For the toxicity feature, a special token [TOX] is added to the beginning of the input token sequence if and only if the sentence is toxic. If the sentence is not toxic, the [TOX] token will not be added. For En-De, the [TOX] token is applied to both source sentence and translation. But for other three language pairs it is only applied to the source sentence (English), because the Perspective API does not support Czech, Japanese and Chinese.

For the sentiment feature, there are three special tokens, [SEN_POS], [SEN_NEG], [SEN_NEU], representing positive, negative and neutral sentiment respectively. Each time only one token denoting sentence's sentiment is added to the beginning of that sentence. All the sentences should have one sentiment token at the beginning. The sentiment token is applied to both source sentence and translation for En-De, En-Ja and En-Zh. We do not perform experiments on sentiment feature for En-Cs due to lack of support on Czech from sentiment analysis API.

For named-entities feature, there are seven special token pairs corresponding to seven types of named entities generated by SpaCy API, e.g. [ORG] and [/ORG], [DAT] and [/DAT], etc. We use special token pairs to encase the named entities with relevant type in sentences at word level. Similarly to sentiment feature, the tokens of named-entities feature are also applied to both source and translation for En-De, En-Ja and En-Zh. Czech is not supported by spaCy, hence we do not apply this feature to En-Cs.

By adding extra features to the texts, we expect to guide the model with the toxicity/named-entities/sentiment information on the source sentence or the discrepancy of such information between the source sentence and the translation, which might indicate the existence of critical translation errors.

The second approach (shown in Figure 3) is to **modify hidden states**, where the extracted features are presented as numerals and appended to the hidden states of [CLS] token. Due to limited time, we only experimented with NE features using this approach. Since some named entity types are similar, they can be grouped as one type. In this approach, except for DAT, which is an independent category, we group ORG and PER as a category, CRD and ORD as a category, NRP and GPE as a category so that finally we have four categories. The feature that is used here is the count of the four NE types in source and target sentences. It is presented as a vector of length 8, where the first 4 numbers represent the counts of these NE categories for the source sentence, and the last 4 numbers are for the translation. First we feed the source sentence and its translation into the XLM-RoBERTa encoder, then we append the vector of counts to the output of the [CLS] token. The modified hidden states is then fed to the classification head.

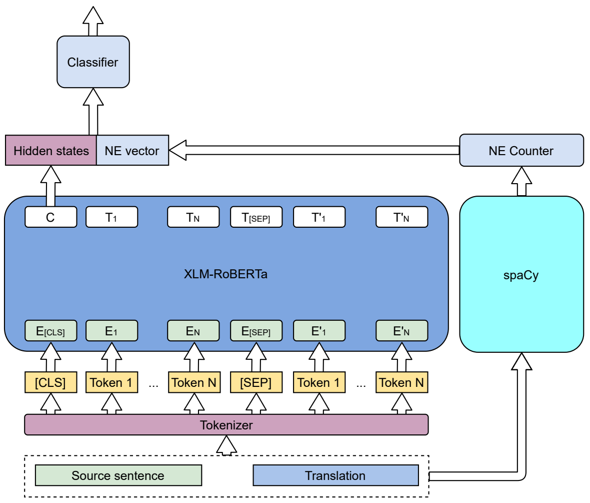

Figure 3: Architecture of the second approach (modifying hidden states). The sentence pair is fed into the XLM-RoBERTa encoder and into spaCy to generate NEs, resulting in the NE vector with the count of name entities of different types. We concatenate the output of [CLS] with the NE vector and send the modified hidden states to the classifier.

Our expectation is that the additional information (vector of counts) could guide the classifier to give more accurate predictions, because a deviation in named entity counts may be indicative of critical errors. For example, if the source sentence contains 3 named entities and the translation contains only 1 named entity, the translation may be missing some named entities.

2.3.3 Ensemble

To boost the performance, we ensemble several models to produce the final predictions. We experiment with two ensemble strategies. One strategy is label-level (late) ensemble. We first obtain the label predictions generated by different models using different features, then combine these predicted labels by performing majority vote to get a final label. The other strategy is logit-level ensemble, where we average the logits produced by different models and then produce the final label using the averaged logits.

3 Results

This section presents the evaluation results of the proposed methods. Except for the baseline score on

Method		En-Cs	En-De	En-Ja	En-Zh
Baseline		0.393	0.413	0.217	0.255
Baseline	best	0.397	0.422	0.231	0.276
(with sampler)	average	**0.396**	0.418	0.224	0.262
Adding TOX token	best	0.391	0.448	0.254	0.284
	average	0.385	**0.435**	0.220	0.261
Adding SEN token	best	–	0.429	0.228	0.296
	average	–	0.416	0.226	**0.276**
Adding NE (`ORG`) token	best	–	0.430	0.237	0.248
	average	–	0.422	0.211	0.239
Adding NE (`PER`) token	best	–	0.446	0.229	0.265
	average	–	0.415	0.224	0.225
Adding NE (`DAT`) token	best	–	0.439	0.220	0.259
	average	–	0.427	0.212	0.232
Adding NE (`CRD`) token	best	–	0.438	0.248	0.222
	average	–	0.426	0.195	0.217
Adding NE (`ORD`) token	best	–	0.442	0.236	0.262
	average	–	0.430	0.214	0.239
Adding NE (`NRP`) token	best	–	0.440	0.247	0.252
	average	–	0.420	0.193	0.247
Adding NE (`GPE`) token	best	–	0.429	0.220	0.251
	average	–	0.418	0.186	0.247
Modifying hidden states	best	–	0.455	0.257	0.280
	average	–	**0.435**	**0.238**	0.253

Table 3: Matthews's Correlation Coefficient (MCC) between predictions and gold labels using different methods on development set, trained on XLM-RoBERTa-base model. "Best" and "average" stand for the highest score and average score of three runs respectively. The bold numbers are the best result for the average of three runs in that language pair. For En-Cs, we only experiment on two cases due to lack of feature availability for Czech.

test set in Section 3.2 which is produced by MonoTransQuest using pretrained XLM-RoBERTa-base model with batch size of 128, learning rate of 2e-5, and a linear learning rate warm-up ratio of 10%, all the other scores (including baseline score on dev set in Section 3.1) are produced using following hyperparameters: 64 for batch size, 2e-5 for learning rate, 30% for the warm-up ratio.

3.1 Results on Dev Set

As described in Section 2.2, we explore nine feature types: source and target toxicity, source and target sentiment and 7 types of source and target named entities. We trained our model using the first approach (adding special token) for each of the nine feature types and the second approach (modifying hidden states) for named entities only. For each method or feature, we run the model for three times with different seeds and report average performance, as well as the performance of the best of the three models. The results on the development set are shown in Table 3.

The results follow our expectation that En-De could achieve the highest MCC score among the four language pairs as the training set of En-De is more balanced, compared to other three language pairs. Meanwhile, En-Ja has the lowest MCC score, as the dataset is the most imbalanced. The results also show that adding a weighted sampler to deal with imbalanced data improves the models' performance in most cases. As for the additional features, some of them are useful, but it depends on the language pair. For example, the toxicity feature can improve the score in En-De but cannot improve performance in En-Ja and En-Zh, while the sentiment token is helpful in En-Ja and En-Zh but not boost the score in En-De.

We note that the results may be affected by fluctuations because of different random seeds. Sometimes multiple runs of the same case will produce fairly different results. This is a general problem of neural models for QE as well as other tasks and requires further investigation. For example, the results of three runs of adding NE (`NRP`) feature in En-Ja vary a lot. The best score from the three runs is 0.247 which is over the baseline score, but the average score is 0.193 which is largely below the baseline.

3.2 Results on Test Set

We use ensembling to produce final results. The different models to ensemble are trained using different features, and hence focus on difference types of errors, thus potentially leading to different predictions. Not all these models lead to improvements over the base (no features) model; in fact, adding some features decreases the performance for some languages. Therefore, we tested ensembles of models with all feature and ensembles of only models with features which achieve higher score on the development set in our ablation experiments (Table 3). We found that ensembling all models leads to a lower score than ensembling the best few models.

In our experiments, ensembling models with better performance than the base model improves the results of all languages except En-De. For that language pair, the best result is achieved by only adding NE (`DAT`) token to the XLM-RoBERTa-large base architecture. The final results of submission for CED shared task and ranks are shown in Table 4.

Method	En-Cs	En-De	En-Ja	En-Zh
Baseline	0.388	0.397	0.214	0.187
Ours (*gjiang*)	0.473 (2nd) reached by ensembling following: • 4 × baseline models (base) • 1 × baseline model (large)	0.498 (2nd) reached by single model: • 1 × NE (`DAT`) model (large)	0.314 (2nd) reached by ensembling following: • 1 × baseline models (base) • 1 × NE (`CRD`) model (base) • 1 × TOX model (large)	0.305 (4th) reached by ensembling following: • 1 × baseline models (base) • 2 × SEN models (base) • 1 × TOX model (base) • 1 × baseline model (large) • 1 × NE (`DAT`) model (large) • 1 × TOX model (large)

Table 4: Final results and ranks of CED shared task in WMT 2021. These results can also be found on CodaLab result page where the name of our team is *gjiang*. The metric in the table is Matthews's Correlation Coefficient (MCC) between predictions and gold labels. Below our score for each language pair is the model(s) that we used to achieve this score. "(base)" and "(large)" denotes this model is trained using XLM-RoBERTa-base and XLM-RoBERTa-large respectively.

Source	YOU SUCK IT!!! AS YOU'RE USED TO SUCK PHALLUS, NAZI HINDUIST LIKE HITLER!!!
Translation	你吸了它, 就像你以前吸过帕卢斯一样, 纳西迷人就像希特勒一样!
Baseline label	`NOT`
Our label	`ERR`
True label	`ERR`
Analysis	The source sentence is toxic and has negative sentiment. But the sentiment of translation is positive. Therefore, there is a deviation in sentiment between source and translation and this is a critical error.
Source	Upon further research I have found irrefutable proof that he got the nickname for the masterful way he cleaves beavers with his massive member.
Translation	经过进一步的研究, 我发现了不可否认的证据, 那就是他用他巨大的成员把贝弗切开的巧妙方法获得了绰号。
Baseline label	`NOT`
Our label	`ERR`
True label	`ERR`
Analysis	The source sentence is not toxic and the sentiments of both sentences are neutral. However, the machine translator mistakenly regards "beavers" as a name and produce a name in Chinese, which is detected by spaCy. The translation introduces one named entity which does not exist in source sentence. Therefore, this is a critical error.
Source	REDIRECT Talk:Historical Archive of the City of Cologne
Translation	主题演讲: 科隆市历史档案
Baseline label	`NOT`
Our label	`ERR`
True label	`NOT`
Analysis	In this case, spaCy does not report "Cologne" as a named entity in source sentence, but in translation it reports the city name in Chinese as a named entity (`GPE`). Therefore, our model regards the translation introduces a new named entity. There is a deviation in named entities between source and translation and this is mistakenly classified as a critical error.
Source	Goanikontes is an oasis is hidden within the Goanikontes Region.
Translation	戈亚尼科恩特是戈亚尼科恩特地区内的一个绿洲。
Baseline label	`NOT`
Our label	`ERR`
True label	`NOT`
Analysis	Similarly to previous case, spaCy correctly detects "Goanikontes" as a location name, but in translation spaCy mistakenly reports the corresponding location name in Chinese as person's name. Hence, our model thinks there is a deviation in named entities and predicts this case to a critical error. The mistakes from APIs are likely to lead the model to give wrong predictions.

Table 5: Case study: comparison of baseline predictions and our ensembled predictions in En-Zh

3.3 Qualitative Analysis

We conducted manual inspection on En-Zh in an attempt to understand whether the additional features actually contribute to better performance. The choice of the language pair that we analysed was determined by the availability of understanding languages in both sides. We compared our final submitted predictions on test set with the baseline predictions. We found that, compared to the baseline result, our final model predicts more `ERR` labels. 82 out of 1000 samples' label in test set are flipped from `NOT` to `ERR`, among which 35 samples are correct change (from false negative to true positive), 47 are incorrect (from true negative to false positive). We give some examples in Table 5 to compare our predictions with the baseline results. These examples show that feature engineering actually pushes the model to predict more `ERR`s. Overall this improves the performance to some extent, but is subjected to the correctness of the feature extractor. Inaccurate results from APIs will give the model wrong information and limit the improvement of performance of our models.

4 Conclusions

This paper describes our submission to sentence-level CED task in WMT21. Our work extends the baseline MonoTransQuest architecture by exploring feature engineering and model ensembling, as well as weighted sampling to deal with imbalanced datasets. Potentially due to the skewed distribution of labels in the dataset, the model performance varies substantially over different runs. However, our results averaged over multiple random seeds show that our feature engineering and ensembling lead to large improvements over the baseline. Our official submission achieves the 2nd position in En-Cs, En-De, En-Ja, and the 4th postion in En-Zh.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2020. TransQuest: Translation quality estimation with cross-lingual transformers. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5070–5081, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shuo Sun, Francisco Guzmán, and Lucia Specia. 2020. Are we estimating or guesstimating translation quality? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6262–6267, Online. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Wikipedia Talk Corpus.