TartuNLP @ SIGTYP 2024 Shared Task: Adapting XLM-RoBERTa for Ancient and Historical Languages

Aleksei Dorkin
aleksei.dorkin@ut.ee

TARTU NLP

UNIVERSITY OF TARTU · UNIVERSITAS TARTUENSIS · 1632

Kairit Sirts
kairit.sirts@ut.ee

Unconstrained subtask

1. Part of speech tagging
2. Full morphological annotation
3. Lemmatization
4. Word-level gap filling
5. Character-level gap-filling

Language	Code	Train Sentences	Valid Sentences	Test Sentences
Ancient Greek	grc	24,800	3,100	3,101
Ancient Hebrew	hbo	1,263	158	158
Classical Chinese	lzh	68,991	8,624	8,624
Coptic	cop	1,730	216	217
Gothic	got	4,320	540	541
Medieval Icelandic	isl	21,820	2,728	2,728
Classical and Late Latin	lat	16,769	2,096	2,097
Medieval Latin	latm	30,176	3,772	3,773
Old Church Slavonic	chu	18,102	2,263	2,263
Old East Slavic	orv	24,788	3,098	3,099
Old French	fro	3,113	389	390
Vedic Sanskrit	san	3,197	400	400
Old Hungarian	ohu	21,346	2,668	2,669
Old Irish	sga	8,748	1,093	1,094
Middle Irish	mga	14,308	1,789	1,789
Early Modern Irish	ghc	24,440	3,055	3,056

Table 1: Language statistics in the Shared Task.

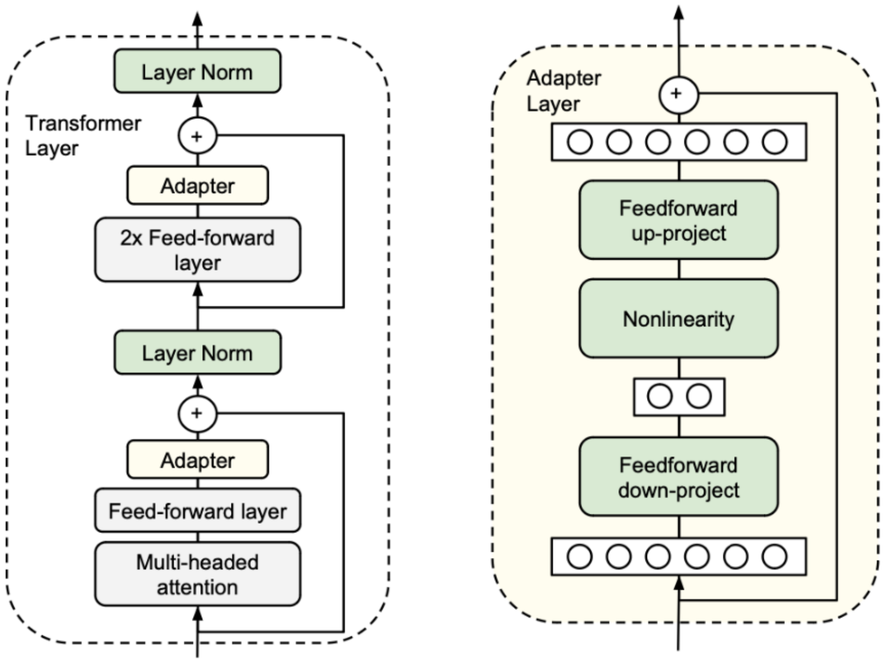
What if we make use of information that the multilingual models already posses?

The problem with full fine-tuning

- Small amount of training data
- Full fine-tuning may result in catastrophic forgetting and/or overfitting
- Tuning and maintaining a separate model for each language/task combination is computationally expensive
- Not all the weights in the multilingual model are relevant for our tasks/ languages. Why fine-tune everything?

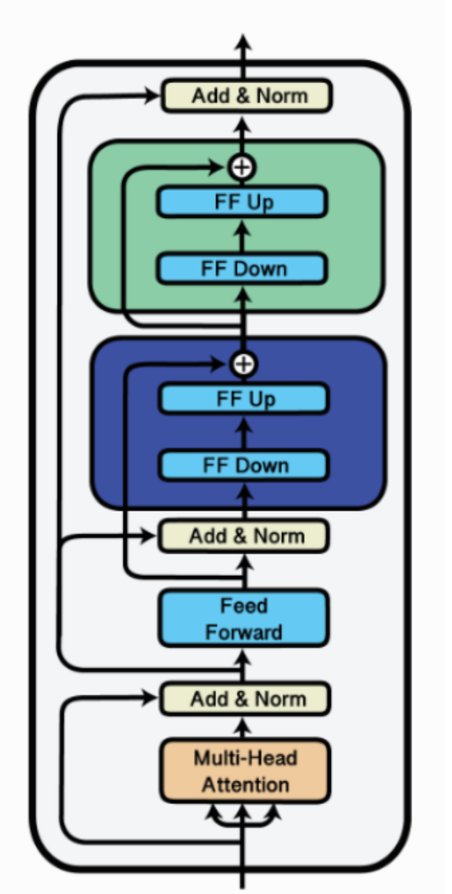
Introducing adapters

- Additional trainable layers injected into the network
- The rest of the weights remains frozen
- The injected layers amount to 2-4% of additional parameters
- Faster to train
- No danger of catastrophic forgetting, since the weights of the underlying model remain unaffected



Language adaptation with adapters

- For each language in the dataset we train a special language adapter
- Language adapters are trained on the umasked part of the word-level gap filling dataset for each language
- The training objective is Masked Language Modeling which itself is similar to the word-level gap filling task
- Some languages are not covered by XLM-RoBERTa's tokenizer, so we train a new tokenizer and an embedding layer for these languages
- Dual purpose: each language adapter is used on its own for word-level gap filling, as well as the base for the other tasks



Part of speech tagging & full morphological annotation

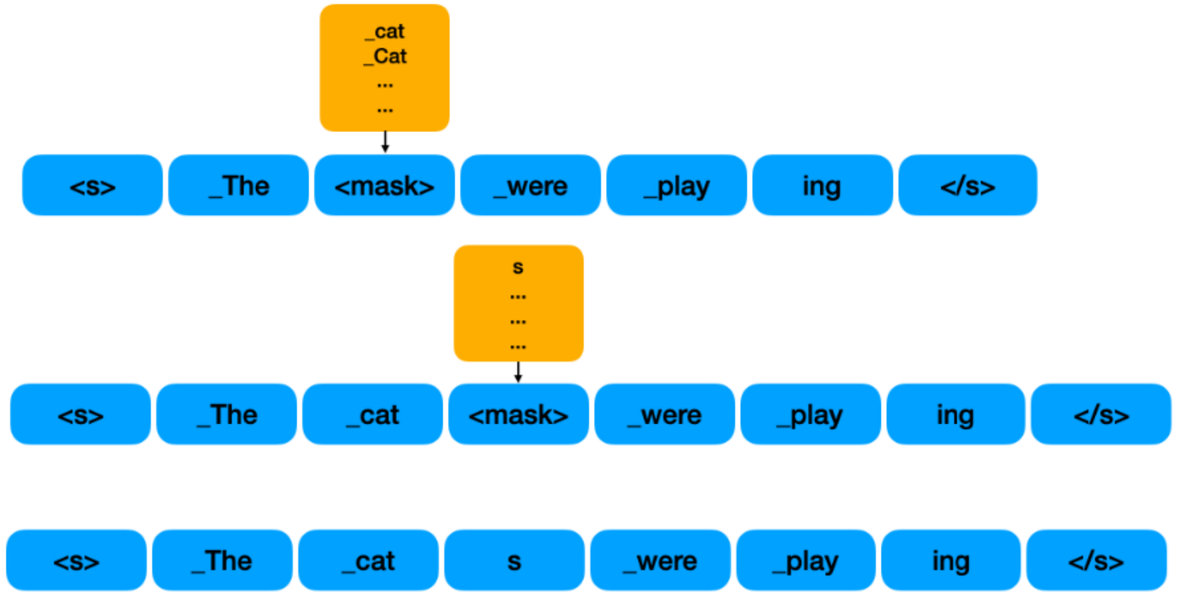
- Both task are simultaneously framed as a token classification task
- A single classification label corresponds to a combination of a part of speech and morphological features
- Every token in the data is assigned such a label
- During the inference we take the part of speech tag from the label for POS-tagging, while the rest of the features go to full morphological annotation

Lemmatization

- Similarly, framed as a token classification task
- For each form/lemma pair in the training data we generate a transformation rule which is used to transform the word form into its lemma
- The transformation rules are represented as individual strings
- We consider transformation rules as classification labels
- Accordingly, the inference involves 2 steps:
  - For each token the most likely transformation rule is predicted
  - The predicted rule is applied to the form to produce the lemma

Word-level gap filling

- Exploits the fact that the encoder transformer models' training is already very similar to the task
- We use the language modeling prediction head (adapted for the language at hand) to fill the gaps
- The caveat is that XLM-RoBERTa is based on subword tokenization, and that means that we do not always know whether the predicted token is a full word or not



Character-level gap filling

- A purely algorithmic approach
- A vocabulary is built based on the training data
- Regular expressions are used to compare masked words with the words of the same length in the vocabulary
- Surprisingly strong results
- However, additional improvements are possible to attain by making use of the language modeling head's prediction to rerank feasible candidates

Results

Team	Rank	Ancient Greek	Ancient Hebrew	Classical Chinese	Coptic	Gothic	Medieval Icelandic	Classical and Late Latin	Medieval Latin	Old Church Slavonic	Old East Slavic	Old French	Vedic Sanskrit	Old Hungarian	Old Irish	Middle Irish	Early Modern Irish
Team 1	1	0.7184	0.6705	0.5920	0.5890	0.7307	0.7198	0.7238	0.7479	0.7115	0.6915	0.7110	0.6961	0.7061	0.3059	0.2871	0.3069
Our submission	2	0.7025	0.6094	0.5654	0.5152	0.6974	0.7288	0.7315	0.7615	0.4985	0.5566	0.6893	0.6551	0.5198	0.1903	0.2174	0.2794
Team 2	3	0.0087	0.0054	0.0020	0.0089	0.0172	0.0153	0.0193	0.0182	0.0098	0.0093	0.0168	0.0152	0.0158	0.0392	0.0417	0.0437

Conclusion

- Competitive results
- Relatively easy to generalize to other languages which makes it a suitable solution for ancient and historical language in general, especially when applied to language models trained on typologically related languages
- However, the adaptation for languages with less represented scripts needs additional investigation

See the source code on GitHub