Composing Byte-Pair Encodings for Morphological Sequence Classification

Anonymous COLING submission

Abstract

This document contains the instructions for preparing a paper submitted to COLING-2020 or accepted for publication in its proceedings. The document itself conforms to its own specifications, and is therefore an example of what your manuscript should look like. These instructions should be used for both papers submitted for review and for final versions of accepted papers. Authors are asked to conform to all the directions reported in this document.

1 Introduction

Since its introduction [CITE:TODO], the transformer model (Vaswani et al., 2017) has emerged as the dominant architecture for statistical language model, gradually displacing recurrent neural networks, in particular the LSTM and its variants. The transformer owes its success to several factors, including the availability of pretrained models, which effectively yield rich contextual word embeddings. Such embeddings can be used as is (for so-called feature extraction), or the pre-trained models can be fine-tuned to specific applications.

At the same time as transformer models became popular, the tokenization of natural language texts have shifted away from methods explicitly oriented on words or morphemes. Rather, statistical approaches are favoured to splits strings of characters into units which are not necessarily meaningful linguistically, but rather have statistically balanced frequencies. For example, the word ”scientifically” may be composed of the tokens: ”scient”, ”ifical”, ”ly” — here the central token does not correspond to a morpheme. Rather than identifying complete words or morphemes, it aims to find sub-word units occurring significantly often. Typical approaches to composing tokens from sub-token units have focused on combining character n-grams (Bojanowski et al., 2017), while other approaches have looked at splitting words into *roots* and *morphemes* [CITATION], and then combining them. In this paper, we consider in particular Byte-Pair Encodings (BPE) (Sennrich et al., 2015), which take another approach. One does not specifically look for either character n-grams or morphs, but rather it aim to split a corpus $\mathcal{C}$ into N tokens, where N is user defined. [EXPAND]

One issue with statistical tokenization is that one is seldom interested in the encoding, but rather in the semantically meaningful units in the original texts. Thus the question of mapping data about the token back to the original text arises.

In this paper we explore how to combine Byte-Pair Encodings from a transformer model to perform sequence classification, which is used in particular in the popular BERT model (Devlin et al., 2018). For our purposes, the goal in sequence classification is to assign a label to every word in a sentence. When we are using byte-pair encoding (or similar sub-token representations) we must then find some way of combining the units that build the word, before it is eventually assigned a label. Coming back to our example, we must map the feature-set assigned (depending on the context) to ”scient”, ”ifical”, ”ly”. Then this combined feature set is mapped to a class for the whole word “scientifically”. [ADD NOTE ABOUT word/sentence-piece]

To our knownledge, this is a little-studied problem. [citet:TODO] have only brushed the surface by reporting that few differences are found between different methods. In this paper we wish to explore the problem in further detail and identify what effect different methods have on the final performance of a model.

Language	Typology	$\frac{BPE}{word}$	Tags	Train	Validation	Test
Basque-BDT	Agglutinative	1.79	919	97336	12206	11901
Finnish-TDT	Agglutinative	1.98	591	161791	19876	20541
Turkish-IMST	Agglutinative	1.73	1056	46417	5708	5734
Estonian-EDT	Agglutinative	1.86	512	346986	43434	43825
Arabic-PADT	Fusional	1.39	300	225494	28089	28801
Czech-CAC	Fusional	1.77	990	395043	50087	49253
Polish-LFG	Fusional	1.75	634	104730	13161	13076
Spanish-AnCora	Fusional	1.25	177	439925	55196	54449

Table 1: Treebank statistics showing the language typology, average number of BPE tokens per word, the number of morphological tags and the size of the datasets.

2 Task

In this paper we further focus on the task of morphological sequence classification. Morphological tagging involves identifying a set of morphological features that a word possesses, such as number, person, case, etc. In many languages, morphological features primarily depend on the affixes of words. However, conversely, the morphological class is not determined by the word affixes, nor the whole word. In many cases, the context of the sentence will affect which class should be assigned to a word.

3 Data

For the task we will use the Universal Dependencies dataset (Nivre et al., 2018) annotated with the UniMorph schema (McCarthy et al., 2018). We are interested both in the general effects of using different composition methods, and whether some methods favor languages with certain typologies. To explore this, we take a sample of eight languages from the Universal Dependencies dataset. Four of them use a *fusional* morphology, meaning that an affix may be indicative of one or more morphological features. Four of them use an *agglutinative* morphology, meaning that each affix is mapped to one and only one morphological feature.

The fusional languages that we consider are Arabic, Czech, Polish and Spanish, and the agglutinative languages that we consider are Finnish, Basque, Turkish and Estonian. We show the size, average number of BPE tokens per text-token and number of morphological tags for each treebank in Table 1.

The fusional languages where chosen such that two of them (Czech and Polish) have a higher BPE per token ratio than the other two (Arabic and Spanish). We make this choice because one factor that impacts the accuracy obtained by a composition method may be the BPE per token ratio. By having both fusional and agglutinative languages with similar BPE per token ratio we can take this variable into account properly in our analysis.

4 Method

(**JP:** This section should be revised, it's quite unclear what the model is. It would help to: 1. describe the model in computational order 2. use f consistently to denote the part which can vary in the model.)

In this section we present the model used for sequence classification, the methods that we use to compose BPE embeddings, and how we trained the model.

Transformer model For the task we use the XLMR (Conneau et al., 2019) model[1]. XLMR is a masked language model based on the transformer (specifically, RoBERTa (Liu et al., 2019)), and trained on data from 100 different languages, using a shared vocabulary. All languages we test are included in XLMR model. In this experiment we use XLMR_{base} model. It has 12 encoder layers, 12 attention heads and use 768 dimensions for its hidden size.

[1]We use the huggingface implementation [CITE/LINK]

4.1 Model

For morphological tagging we use the XLMR base model with a classification module on top. The classification module is an LSTM followed by a two layer linear transformation.(**JP:** Is the LSTM always there? I thought mean and sum were used as well?)

The model that we use to predict morphological features is as follows. For each sentence we extract n BPE embeddings x^0 to x^{n-1}. from $XLMR_{base}$, and then align them to words. We then feed all words which consist of more than one BPE embedding to a function f which combines the BPE embeddings. This produces one embedding per word, which we concatenate with character features generated by a character LSTM. We then pass the BPE features concatenated with the character features to an LSTM to extract contextual features.(**JP:** Again, is this active always?) We pass the LSTM outputs to a linear transformation layer that computes scores for each class in the output. We then use a softmax layer to assign probabilities, and compute the loss accordingly.

An outline of the model is presented in Figure 1, in the outline f represents the different methods we use to combine BPE embeddings.

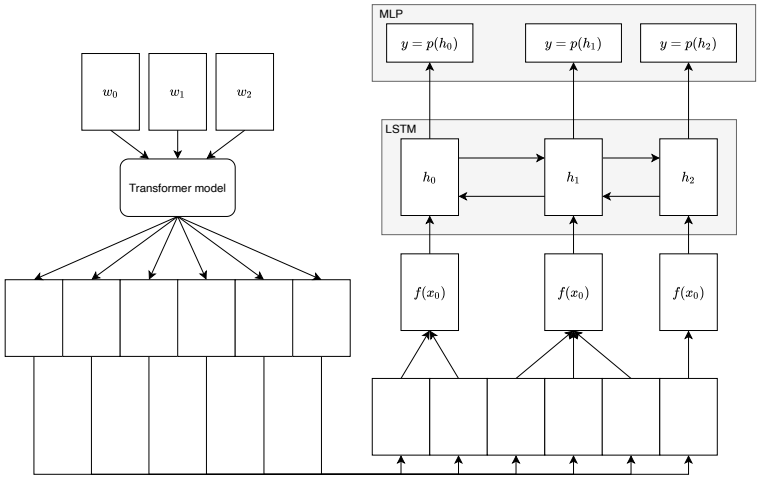

Figure 1: Procedure outline

4.1.1 BPE features

As mentioned previously, we look at methods for composing embeddings of BPE tokens into word embeddings. The XLMR model use 12 layers to compute an embedding for a BPE token, and it has been shown in previous research (Kondratyuk and Straka, 2019; Raganato et al., 2018) [Find 1 or 2 more citations] that the different layers of the transformer model encode different types of information. To take advantage of this we compute BPE embeddings as a weighted sum of layer representations (Kondratyuk and Straka, 2019). That is, we initialize a parameter from a standard normal distribution w of size l, where l is the number of laters in the transformer model. Where r_{ji} is the layer representation at layer j and token position i, we calculate the weighted sum as follows:

$$x_i = \sum_{1}^{J} softmax(w)_j r_{ji} \tag{1}$$

Where $softmax(w)_j$ indicate the softmax of the learned importance of layer j. After we have computed a weighted sum for each BPE token we proceed to combine them into the tokens as they appear in the data.

We look at three method in particular, summation and averaging along a dimension, and using an RNN. Summation and averaging have been used in previous work, but using an RNN have not been explored before to our knowledge.

Sum: For the sum method, we use an element-wise sum. That is, we take the sum for each dimension of the BPE embeddings separately. Thus, for token i we calculate a composite embedding by summing over dimensions $0, \ldots, N$:

$$f(x)_i = \sum_{j=1}^{n} x_i^j \tag{2}$$

Mean: In the mean method we calculate the sum and divide by the number of BPE embeddings in the word. Thus, for token i we calculate a composite embedding by averaging over dimensions $0, \ldots, N$:

$$f(x)_i = \frac{1}{n} \sum_{j=1}^{n} x_i^j \tag{3}$$

RNN: For this method we employ a bidirectional LSTM to compose the BPE embeddings. For each multi-BPE token, we pass the sequence of BPE embeddings through an LSTM and use the final output as the word representation.

4.1.2 Character features

In addition to layer attention we use a character LSTM to extract a word representation based on characters. The final representation that we pass to the word-LSTM is the concatenation of the word representation based on BPE compositions and characters, $w_i = \text{concat}(f(bpe_0, ..., bpe_K), f(c_0, ..., c_M))$

4.1.3 Label smoothing

Given that many of the languages have a large number of morphological tags, we want to prevent the model from growing overconfident for certain classes. To address this issue we introduce label smoothing (Szegedy et al., 2016), that is, instead of the incorrect classes having 0 probability and the correct class 100% probability we let each of the incorrect classes have a small probability.

Let α be our smoothing value, then given a one-hot encoded target vector t of size (N, C) where N is the number of examples and C the number of classes we calculate the smoothed probabilities as: (**JP:** This formula still looks incorrect; let's discuss it next time.)

$$t_{smooth} = (1 - \alpha)t + \frac{\alpha}{C} \tag{4}$$

The intuition is that we remove $1 - \alpha$ from the correct class then distribute α uniformly among all classes.

4.2 Training

In our experiments we consider two possible training regimes. In the first regime we finetune the XLMR models parameters, in the second regime we only extract weights for BPE tokens, that is, we use the model as a *feature extractor*.

When fine-tuning the model we freeze the XLMR parameters for the first epoch. When training the model we use cosine annealing learning rate with restarts every epoch, that is, the learning rate starts high then incrementally decrease to $1e-12$ during N steps, where N is the number of batches in an epoch. We use a learning rate of 0.001 for the word LSTM, classification layer and BPE combination module (when an RNN is used), for the transformer we use a lower learning rate of $1e-06$.

We use dropout thrughout the model, we apply dropout on the input to the transformer, replacing 0.2 of the tokens with `<UNK>`. After we have extracted layer features from the transformer we apply a dropout of 0.4, and before computing the weighted sum of layers we apply full dropout on a layers with a probability of 0.1. After the word LSTM have processed the sequence, before the final prediction we apply a dropout of 0.5.

Parameter	Value
Epochs	15
Batch size	4 / 32
Character representation size	128
Word LSTM size	1536
Linear transform size	1536
Optimizer	Adam
Learning rate	0.001
Learning rate$_{xlmr}$	1e-06
Weight decay	0.05
Label smoothing	0.03

Table 2: Hyperparameters used for training the model. Slashed indicate the value of a parameter when we finetune or extract features.

5 Results

We compare our results against the baseline reported in (McCarthy et al., 2019) to make sure our system works in general. Our results are not directly comparable to those of the shared task as many system employ treebank concatenation. The accuracy of our system in assigning morphological tags, using the three different methods, and two trainning regimes are presented in Table 3.

		Finetuning			Feature extraction		
Treebank	Baseline	Sum	Mean	RNN	Sum	Mean	RNN
Basque-BDT	.676	.905	.906	**.920**	.865	.865	**.888**
Finnish-TDT	.751	.965	.963	**.967**	.930	.931	**.942**
Turkish-IMST	.620	.898	.891	**.905**	.856	.849	**.866**
Estonian-EDT	.740	.960	.961	**.962**	.931	.934	**.939**
Spanish-AnCora	.842	.979	.979	**.980**	.968	.967	**.971**
Arabic-PADT	.770	.952	.953	**.954**	.941	.939	**.948**
Czech-CAC	.771	.976	.976	**.977**	.944	.944	**.952**
Polish-LFG	.657	.959	.956	**.960**	.907	.907	**.928**
Average	.728	.949	.948	**.953**	.918	.917	**.929**

Table 3: Accuracy for morphological tagging. We evaluate both when we finetune the XLMR model and when we only extract BPE embeddings.

As expected our system perform better than the baseline. As a general trend we can see that the RNN method tend to perform better than the summation or averaging methods. This is consistent across both languages and training regimes, showing that while the changes are small, they are there consistenty. Perhaps not surprisingly, we find that in general finetuning yields stronger performance than feature extraction. However, the difference is negligeble, increasing by 3.1% for summation and averaging and 2.4% for RNN.

When fineuning we see the largest changes in Basque and Turkish, with an increased performance of 1.45% and $.7\%$ when finetuning and using RNN over using mean or averaging. When only extracting features the see a larger increase when using an RNN of 2.3% and 1.35% respectively.

Table 3 reports average accuracy for every word, thus also including those which are only composed of a single BPE token. To highlight the strengths and weaknesses of each composition method, we also compute the accuracy for tokens which are composed of two BPE tokens or more. The results can be seen in Table 5. We see again that RNN seem to work better than summation or averaging BPE embeddings.

Given that the number of BPE tokens per natural-language token varies, we also look at the accuracy of the different methods given a different number of BPE tokens. We show per-language performance

Treebank	Baseline	Finetuning			Feature extraction		
		Sum	Mean	RNN	Sum	Mean	RNN
Basque-BDT		.884	.877	**.901**	0.789	0.780	**0.834**
Finnish-TDT		.958	.960	**.965**	0.856	0.847	**0.899**
Turkish-IMST		.859	.855	**.884**	0.741	0.735	**0.775**
Estonian-EDT		0	0	0	0.856	0.853	**0.901**
Spanish-AnCora		0	0	0	0.954	0.952	**0.962**
Arabic-PADT		0	0	0	0.923	0.902	**0.936**
Czech-CAC		0	0	0	0.887	0.881	**0.924**
Polish-LFG		0	0	0	0.844	0.840	**0.878**

Table 4: Without character LSTM, much more prominent changes in accuracy as expected.

Treebank	Finetune			Feature extraction		
	Sum	Mean	RNN	Sum	Mean	RNN
Basque-BDT	.838	.838	**.863**	.802	.803	**.840**
Finnish-TDT	.950	.947	**.954**	.893	.897	**.913**
Turkish-IMST	.825	.814	**.846**	.807	.796	**.816**
Estonian-EDT	.946	.957	**.950**	.904	.908	**.916**
Spanish-AnCora	.964	.963	**.965**	.952	.951	**.959**
Arabic-PADT	.906	.904	**.913**	.927	.925	**.935**
Czech-CAC	.958	.959	**.962**	.915	.916	**.930**
Polish-LFG	.924	.924	**.933**	.834	.833	**.876**

Table 5: Accuracy for morphological tagging on all tokens that are composed of 2 or more BPE tokens (with char-lstm).

with the differnt methods in Figure 2.

6 Discussion

Analogously to character embeddings, the RNN method seem to provide stronger results than using either Sum or Mean for combining BPE tokens. We conjecture that this discrepancy is can be attributed to the commutativity of the Sum and Mean operations. That is, the ordering of elements within a word does not affect the result of applying Sum or Mean.

(**JP:** In particular, the position of an affix is lost if the word which includes it is split in several tokens.)

7 Conclusions

As a general summary of our work, we highlight that language typology is not irrelevant when working with models that employ byte-pair encoding.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th*

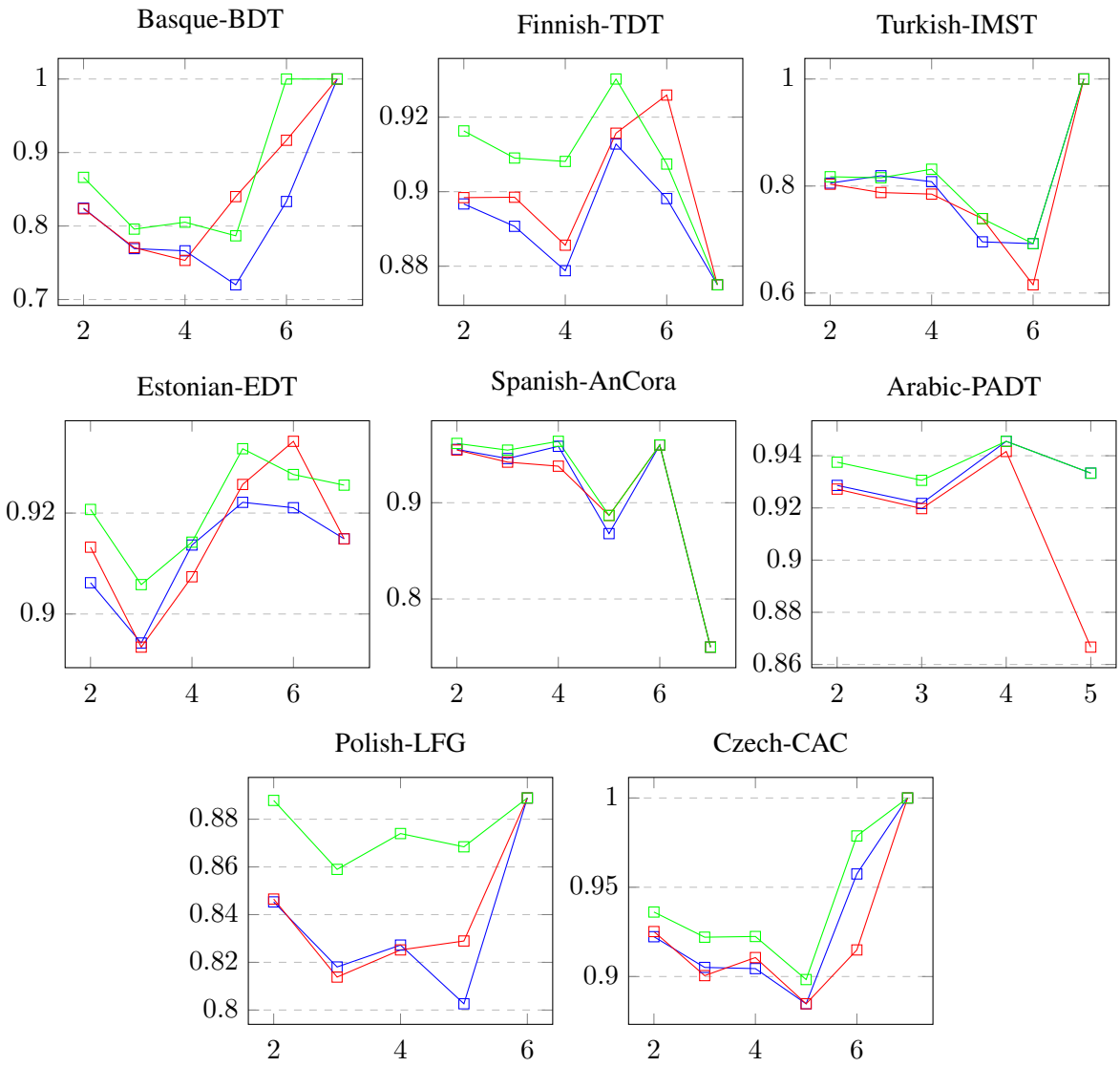

Figure 2: Per language accuracy of tokens composed of more than two BPE tokens. The x-axis indicate how many BPE tokens a word is composed of and the y-axis the accuracy. The different methods are distinguished by color, where Blue is the summation method, red the mean and green RNN.

International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Arya D McCarthy, Miikka Silfverberg, Ryan Cotterell, Mans Hulden, and David Yarowsky. 2018. Marrying universal dependencies and universal morphology. *arXiv preprint arXiv:1810.06743*.

Arya D McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian J Mielke, Jeffrey Heinz, et al. 2019. The sigmorphon 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. *arXiv preprint arXiv:1910.11493*.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Katya Aplonova, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Victoria Basmov, John Bauer, Sandra Bellato, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad

Bhat, Erica Biagetti, Eckhard Bick, Rogier Blokland, Victoria Bobicev, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Carly Dickerson, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Radu Ion, Elena Irimia, Ọlájídé Ishola, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Kamil Kopacewicz, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phng Lê H`ông, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Shinsuke Mori, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lng Nguy˜ên Thị, Huy`ên Nguy˜ên Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Adédayọ̀ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roca, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Yuta Takahashi, Takaaki Tanaka, Isabelle Tellier, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Jing Xian Wang, Jonathan North Washington, Seyi Williams, Mats Wirén, Tsegay Woldemariam, Tak-sum Wong, Chunxiao Yan, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2018. Universal dependencies 2.3. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Alessandro Raganato, Jörg Tiedemann, et al. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. The Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.