Natural Language Engineering **1** (1): 1–22. Printed in the United Kingdom
© 2019 Cambridge University Press

Neural Morphosyntactic Tagging for Rusyn

Yves Scherrer, Achim Rabus

(*Received 8 April 2019*)

Abstract

The paper presents experiments on part-of-speech and full morphological tagging of the Slavic minority language Rusyn. The proposed approach relies on transfer learning and uses only annotated resources from related Slavic languages, namely Russian, Ukrainian, Slovak, Polish, and Czech. It does not require any annotated Rusyn training data, nor parallel data or bilingual dictionaries involving Rusyn. Compared to earlier work, we improve tagging performance by using a neural network tagger and larger training data from the neighboring Slavic languages. We experiment with various data preprocessing and sampling strategies and evaluate the impact of multi-task learning strategies and of pretrained word embeddings. Overall, while genre discrepancies between training and test data have a negative impact, we improve full morphological tagging by 9% absolute micro-averaged F1 as compared to previous research.

1 Introduction

Rusyn is a Slavic minority language predominantly spoken in the Carpathian area, most notably in Transcarpathian Ukraine, Eastern Slovakia, and Poland, where it is usually called Lemko. While Ukrainian is the Slavic standard language linguistically closest to the Rusyn varieties, their linguistic status is disputed. Some scholars claim that the Rusyn dialects are a part of the Ukrainian linguistic continuum (Skrypnyk, 2013), others argue in favor of a separate linguistic and cultural identity of the speakers of Rusyn (Plishkova, 2009; Magocsi, 2004). From a structural viewpoint, there are certain similarities with Ukrainian, e.g., with respect to common sound changes on the one hand, e.g., East Slavic pleophony such as in *korova* ‘cow’ or the rendering of Common Slavic *jat’* as /i/ such as in *lito* ‘summer’. On the other hand, certain properties make the Rusyn varieties similar to the adjacent West Slavic languages, i.e., Polish and Slovak. Examples include clitic pronouns or past tense formation using forms of the auxiliary ‘to be’ (Boudovskaia, 2017; Rabus, 2019). Currently, the Rusyn varieties are under strong sociolinguistic pressure from the neighboring umbrella languages Standard Ukrainian, Slovak, and Polish. Instances of contact-induced transfer can thus be observed that make the linguistic situation dynamic and complex (Rabus, 2015). The current dynamic tendencies can be researched using the Corpus of Spoken Rusyn[1] comprising Rusyn

[1] Available at `http://www.russinisch.uni-freiburg.de/corpus`.

data from Zakarpattia Ukraine, Eastern Slovakia, Southeastern Poland, and Hungary.

From the viewpoint of NLP, Rusyn has to be classified as low-resource language. No large or medium-sized manually annotated corpora that could serve as training data exist, which means that one has to resort to transfer learning approaches, taking advantage of better-resourced similar Slavic languages. Still, even for Standard Ukrainian, the language closest to the Rusyn varieties, the situation is far from ideal, with the respective Universal Dependencies (version 2.2) treebank containing only 100 000 annotated tokens.

The approach proposed in this paper relies on resources from several neighboring Slavic languages in order to create part-of-speech tagging and full morphological tagging models for Rusyn. The practical goal of these experiments is to provide reliable automatic annotation for the Corpus of Spoken Rusyn in order to facilitate linguistic research. This paper extends our previous work (Scherrer & Rabus, 2017) in several respects: we take advantage of expanded training corpora made available through the Universal Dependencies project, replace the traditional CRF-based tagging model by a state-of-the-art neural network tagging architecture, and evaluate the usefulness of various strategies recently proposed in the context of transfer learning, such as multi-task learning (using language prediction as the auxiliary task) and cross-lingual word embeddings.

This paper is structured as follows. We start by reviewing the previous works on Rusyn NLP and on transfer learning in general. In Section 3, we present the data that are at our disposal and discuss issues related to the evaluation of tagging performance in our transfer learning setup. Section 4 presents the proposed tagger architecture in detail and compares its performance to traditional taggers. Section 5 deals with multi-task learning experiments, and Section 6 explores the impact of pretrained word embeddings. Finally, we give an outlook on future perspectives.

2 Previous work

To our knowledge, besides our own experiments, varieties of Rusyn have not been the target of NLP experiments yet.[2] In our previous work on Rusyn morphosyntactic tagging, we followed a twofold approach. First, we built a morphosyntactic dictionary of Rusyn, applying bilingual lexicon induction techniques on corpora from the East Slavic languages Russian and Ukrainian and on cyrillicized corpora from the West Slavic languages Slovak and Polish (Rabus & Scherrer, 2017). Word forms were matched by vowel-sensitive Levenshtein distance, manually written transformation rules and combinations of both. Second, we trained a tagger on the concatenation of annotated Russian, Ukrainian, Slovak, and Polish data taken from the Universal Dependencies repository (Nivre et al., 2016). Besides cyrillicizing the

[2] There have been plans on using machine translation tools to revitalize the Lemko variant of Rusyn by creating a Lemko-to-English translation system (`https://www.mail-archive.com/moses-support@mit.edu/msg15849.html`), but we are not aware of any published results in this area.

Slovak and Polish data and balancing the different sources, no further processing was applied to these data for the base model (Scherrer & Rabus, 2017). We devised three strategies for improving tagging accuracy: (1) adding the morphosyntactic dictionary induced in the first experiments, (2) including word clusters trained on the (yet unannotated) Corpus of Spoken Rusyn, and (3) "translating" some of the words in the training data to Rusyn to make the training data more similar to the test data. The underlying tagging model of all experiments was MarMoT (Müller, Schmid, & Schütze, 2013), a CRF-based tagger particularly adapted for full morphological tagging with large tagsets. Our best results with this approach yielded 82.4% part-of-speech accuracy and 75.5% full morphological tagging accuracy. These results mark the baseline for the experiments that we report in this paper.

The task of creating taggers for languages lacking manually annotated training data has inspired a lot of research, which can be subsumed under the name *transfer learning*: a model (or the annotated data required to train it) is transferred from a *high-resource language* (HRL, or *source language*) to a *low-resource language* (LRL, or *target language*). Besides the *data transfer* approach, introduced by Yarowsky and Ngai (2001) and requiring large word-aligned parallel corpora,[3] the most popular approach nowadays is the *model transfer* approach.

One of the first model transfer proposals, now known under the term *relexicalization*, consists in training a tagger for a source language and "translating" the words inside the trained tagging model to the target language. The translation lexicons can be created with a hand-written morphological analyzer and a list of cognate word pairs (Feldman, Hana, & Brew, 2006), extracted from a parallel corpus (Zeman & Resnik, 2008), or induced from monolingual corpora (Scherrer, 2014). A variant of this approach, *delexicalization*, proposed initially for parsing (Zeman & Resnik, 2008; McDonald, Petrov, & Hall, 2011), consists in replacing the word forms of the source language by language-independent features, such that a tagger trained on such a transformed version of the training corpus can be applied without change to an identically transformed corpus of the target language. Täckström, McDonald, and Uszkoreit (2012) adapt this idea to morphosyntactic tagging by replacing the word forms by word class numbers; the latter are inferred using a clustering algorithm and synchronized across the two languages through a parallel corpus. Zhang, Gaddy, Barzilay, and Jaakkola (2016) use cross-lingual word embeddings (representing each word by a high-dimensional vector of real numbers, see below) instead of cross-lingual word clusters (representing each word by a single integer number).

[3] Recent work (Agić, Hovy, & Søgaard, 2015; Plank & Agić, 2018, for instance) has successfully applied the data transfer approach to large numbers of languages using massively parallel datasets like the Bible. However, parallel corpora are hard to find for minority languages like Rusyn since there is no large market for translations. We were unable to obtain any translations from high-resource languages into varieties of Rusyn in digital form. While a printed version of the Gospel in the Slovak Rusyn variety is available, we were merely able to locate an online version of some Psalms, which, due to them being in verses, would not be very helpful for transfer learning. It would therefore be a challenging, but worthwhile endeavor for both computational and corpus linguists to compile a Rusyn parallel corpus in close cooperation with the local Rusyn community.

Yu, Mareček, Žabokrtský, and Zeman (2016) replace the word forms by language-independent features such as word length and word frequency classes, forgoing the need for parallel data to synchronize features across languages.

In recent years, following a general trend in machine learning and NLP initiated by Collobert et al. (2011), tagger architectures based on neural networks have been introduced successfully. A representative example of a neural tagger architecture is described by Plank, Søgaard, and Goldberg (2016), which our own work relies on, and which will be described in more detail later on. In this context, multilingual models (e.g. Ammar, Mulcaire, Ballesteros, Dyer, and Smith (2016) for parsing) have become a popular option for leveraging abundant data of related languages. The general idea is that a single model is trained on the concatenation of training corpora from several languages. For this approach to work, the representations of the input data as well as the output labels must be unified across languages. The resulting model can then not only be used to annotate data from any of the training languages, but also for unseen languages whose word representations correspond to the ones used during training. In many cases, although not obligatorily, it has been found useful to inform the model about the language of each sentence it is processing, so that it can better learn language-specific parameters.

Our previous work on Rusyn (Scherrer & Rabus, 2017) follows this multilingual modeling approach, but relies on a traditional tagger architecture. Moreover, the model is not given any language information during training. Cotterell and Heigold (2017) present various experiments with multilingual neural network taggers. They show that models trained on multiple languages often outperform single-language models. However, all of their models include limited amounts (100 or 1000 tokens) of annotated target language data. Also, the model trained on Slavic languages – while performing considerably worse than the model trained on Romance languages – exhibits an alphabet effect: usually, languages written in Cyrillic benefit most from other languages written in Cyrillic script, and analogously for Latin script.

Cross-lingual word embeddings have become one of the most popular options for representing (language-specific) word forms by language-independent features. Word embeddings in general, trained on large raw text corpora, have been shown to provide strong signals for a number of NLP tasks. When training multilingual models, however, the word embeddings must be "synchronized" across languages, i.e. projected onto the same vector space. This condition is not naturally fulfilled if the embeddings are trained independently for each language. A large body of recent research is dedicated to the problem of projecting the word embeddings of one language onto the vector space of another one (for an overview, see e.g. Ruder, Vulić, and Søgaard (2018)). Most approaches start with a seed dictionary containing word pairs of both languages, but some variants merely rely on identical tokens such as punctuation signs, numerals, or named entities (Artetxe, Labaka, & Agirre, 2017). More relevant to our setting is the work by Sharoff (2018) on related languages: he starts by automatically extracting a seed dictionary from Wikipedia page titles and uses these entries to determine edit distance weights for the language pair in question, assuming that most word pairs are cognates. Weighted Levenshtein distance is then included as a factor in the word embedding projection algorithm.

Corpus ID	Language	Training data		Development/Test data	
		Sentences	Tokens	Sentences	Tokens
CS-PDT	Czech	68 495	1 173k	9 270	159k
PL-LFG	Polish	13 774	105k	1 745	13k
PL-SZ	Polish	6 100	63k	1 027	10k
RU-GSD	Russian	3 850	76k	579	12k
RU-SynTagRus	Russian	48 814	870k	6584	118k
SK-SNK	Slovak	8 483	81k	1 060	12k
UK-IU	Ukrainian	4 513	75k	577	10k
5k	Multilingual	35 000	489k	4039	59k
10k	Multilingual	59 950	805k		
Test	Rusyn	—	—	105	1 051

Table 1. *Sizes of the training and test corpora used in our experiments.*

In this way, Sharoff (2018) creates a "Panslavonic" word embedding space for seven Slavic languages, among which are Czech, Polish, Russian, Slovak and Ukrainian.

3 Data and evaluation

3.1 Training data selection and preprocessing

Following the approach outlined in Scherrer and Rabus (2017), our goal is to create a tagger for Rusyn by relying entirely on annotated training data from related Slavic languages. This goal is facilitated by the existence of large corpora annotated according to the same guidelines within the Universal Dependencies (UD) project[4]. The experiments reported here make use of the UD v2.2 editions of the corpora. The sizes of the relevant corpora are summarized in the upper part of Table 1.[5]

It can be seen that the amount of training data varies a lot across languages. Crucially, the largest training corpora stem from the languages least closely related to Rusyn, namely Czech and Russian. In order to alleviate possible negative effects of this skewed distribution, we create two balanced subsets for training (see central part of Table 1):

1. The *5k* subset contains the first 5 000 sentences of each training corpus.[6] For the smaller RU-GSD and UK-IU corpora, sentences are repeated to obtain 5 000-sentence sets.
2. The *10k* subset contains the first 10 000 sentences of each training corpus, repeating sentences from SK-SNK and UK-IU, but not from PL-SZ and RU-GSD.

[4] `http://universaldependencies.org/`
[5] We only use the development sets of these languages, not the test sets.
[6] For the sake of replicability, we do not randomly sample the sentences.

CS	Nejzajímavější dotazy najdete zodpovězené pravidelně na této stránce .
	Нейзаїмавiйши дотазы найдете зодповiзене правиделнi на тето странце .
PL	Uświadomiła sobie , że kilkadziesiąt sekund to całe wieki .
	Усьвядомила собе , же килкадзесят секунд то цале веки .
SK	Ďalšie kroky švajčiarskej sudkyne povedú do Belehradu .
	Дялшие крокы швайчиарскей судкыне поведу до Белеграду .

Fig. 1. Examples of transliterations for Czech, Polish and Slovak.

As we do not have a dedicated Rusyn development set and do not want to further split the small test set (presented below) into two pieces, we create a balanced multilingual development set by concatenating the 577 first sentences of the seven development sets (see central part of Table 1). Note however that this development set is very different from the real Rusyn test set and therefore a poor indicator of the expected system performance on Rusyn.

Following the comparatively poor results obtained by Cotterell and Heigold (2017) for languages written in different scripts, we decided to convert the Czech, Polish and Slovak parts of the multilingual training and development corpora to Cyrillic script, as in our earlier work (see Figure 1). During this process, we applied certain transformation rules in order to "rusynify" our training data (e.g., transform Polish *ć* to Cyrillic *ть* or Polish *ą* to Cyrillic *у* or *я*, respectively, which is in line with well-known historical phonological processes). No preprocessing is applied to the (Ukrainian, Russian and Rusyn) corpora originally written in Cyrillic script.

The morphological annotation guidelines in UD are supposed to be applied equally across all languages. However, in practice, this goal is not quite achieved. We identified a number of inconsistencies in the corpora cited above. These inconsistencies partly stem from the presence or absence of optional features, from distinctions that were not present in the original datasets before their conversion to UD, and from proper linguistic differences between the languages in question. We reduced the most frequent and most regular inconsistencies through the application of the following harmonization rules:

- All participles are assigned the VERB part-of-speech regardless of their grammatical function.
- The Polish three-way animacy distinction of masculine nouns (*Hum/Nhum/Inan* or *Masc1/Masc2/Masc3*) is reduced to a two-way distinction (*Anim/Inan*).
- The distinction between *Ptan* (*pluralia tantum*) and *Plur* (plural) is only made in the Ukrainian corpus; we collapse both labels to *Plur*.
- The Ukrainian corpus distinguishes between semantic and grammatical animacy. We only keep the grammatical animacy labels, as this is done in the other corpora.
- A large number of features are only annotated in the Czech corpus, e.g. style,

named entity type, punctuation type, typographical mistakes, etc. We removed these features, as they do not relate to inflectional morphology.

3.2 Test data

The last row of Table 1 shows the size of the Rusyn test set that is used for evaluation of the different models presented here. It corresponds to a manually annotated Rusyn file consisting of a spoken narrative by a speaker of one Rusyn variety spoken in Transcarpathia, Ukraine, with 1 051 tokens.[7] This poses certain issues, the most problematic being the discrepancy between the genres of training and test data. While our concatenated UD training data consist of written text, the style and syntax of our test data significantly deviates from those of the training data. Naturally, this results in loss of accuracy that can be ascribed to genre discrepancy. One way to alleviate this issue would be to use test data more similar to the training data, i.e. data adhering to the written Rusyn norm as codified (Koporová, 2015). However, since our main practical goal is to develop tagging resources for the Corpus of Spoken Rusyn, we resolved to stick with the oral genre of our Rusyn test data.

3.3 Evaluation

The evaluation of a part-of-speech tagger is quite straightforward: for each word, one single part-of-speech is predicted, and this prediction is either correct or wrong, such that the accuracy measure is adequate for this purpose. For full morphological tagging however, the situation is more complicated, as each word is associated with a set of feature-value pairs whose size is determined by the part-of-speech tag predicted at the same time. A typical morphological tag according to UD conventions looks as follows:
POS=NOUN|Animacy=Inan|Case=Acc|Gender=Masc|Number=Plur

A sensible evaluation measure should give partial credit for partial matches and should take into account situations where the number of predicted features does not match the number of gold features. Accuracy measured on the full labels is too harsh for these cases. Therefore, most recent work reports averages over per-feature F1-scores, either macro-averages (Buys & Botha, 2016; Cotterell & Heigold, 2017) or micro-averages (Pinter, Guthrie, & Eisenstein, 2017). We report micro-averages here and include the part-of-speech tag predictions in the averages.

A further complication arises from the incomplete annotation of the Rusyn test corpus. For example, all training corpora annotate adjectives and adverbs with the *Degree* feature, but this feature is not annotated in the test set. We do not want to penalize a tagger for predicting a value of this feature, since it may very well be correct. This situation can be remedied in two ways, either by removing such

[7] We would like to thank Elena Boudovskaia for kindly sharing parts of her manually annotated data with us. We converted the original annotation to the UD conventions for use as a gold standard.

features from the training data, or by ignoring them during evaluation. After initial experiments, we chose the latter approach, as the additional predictions may still be useful in the context of the annotation of larger corpora. Concretely, we define the set of annotated features for each part-of-speech and mask all other features during evaluation.[8]

Cotterell and Heigold (2017) convincingly demonstrate that using related languages for transfer learning yields considerably higher results within the Romance family than within the Slavic family: Within the Romance family, the language pair performing worst is Catalan used for tagging Portuguese ($|D_t| = 1000$) with 89.0%, and the best is Portuguese and a combination of all Romance language used for tagging Spanish ($|D_t| = 1000$) with 94.2%. Within the Slavic language family, however, the language pair with worst performance is Russian to be used for tagging Ukrainian ($|D_t| = 1000$) with 72.7%, while the pair with best performance is Polish used for tagging Russian ($|D_t| = 1000$) with 84.2%. Obviously, the rich morphology and relatively free word order specific to the Slavic languages that results in a larger tagset yields an at least 10% lower performance as compared with the Romance language family even for language pairs where the alphabet effect does not apply. This has to be taken into account when assessing tagging performance. As mentioned above, the detrimental effect of the discrepancy between the genre of training and test data has to be taken into account as well.

4 Character-level LSTM taggers for Rusyn

4.1 Model architecture

Plank et al. (2016) describe a classic neural tagger architecture. In this architecture, each token is encoded as a multi-dimensional vector, and the token vectors are fed into a bidirectional recurrent neural network consisting of LSTM cells. The LSTM states of each word are then linked to an output layer, which predicts the part-of-speech tag of the word. All parts of this neural network are trained jointly, such that the recurrent hidden layer is able to capture the relations between adjacent words. Note that in this architecture, and in contrast to traditional HMM-based or CRF-based taggers, the tags are predicted greedily for each token; the choice of a tag therefore does not depend on the tag choices made for the surrounding words. This may in principle lead to inconsistent tagging within a sentence, but the rich contextual information represented in the LSTM prevents such inconsistencies to

[8] The following list enumerates, for each part-of-speech tag, the features that are included in the evaluation, all others being masked:

ADJ:	Animacy, Case, Gender, Number, Poss
AUX:	Aspect, Mood, Number, Person, Tense, VerbForm
DET:	Animacy, Case, Gender, Number, Person, Poss
NOUN:	Animacy, Case, Gender, Number
NUM:	Animacy, Case, Gender, NumType, Number
PRON:	Animacy, Case, Gender, Number, Person, PronType, Reflex
PROPN:	Animacy, Case, Number
VERB:	Aspect, Case, Gender, Mood, Number, Person, Tense, Variant, VerbForm, Voice

a certain extent. Neural network taggers including a CRF-based output layer have also been proposed (Lample, Ballesteros, Subramanian, Kawakami, & Dyer, 2016; Malaviya, Gormley, & Neubig, 2018).

There are essentially two ways of representing words as vectors. Intuitively, these vectors can either represent the phonological and morphological structure of the words (their character sequence) or the distributions of the words in sentential context. The latter type is typically referred to as *word embeddings*; we call the former *character embeddings* – somewhat abusively, as a more precise term would be *character-level word embeddings.* The term *character-level tagger* is used for tagging architectures that use character embeddings (exclusively or not).

Word embeddings consider each token as a distinct entity. They are naturally inferred on the input layer by the tagger during training, but they can also be trained separately using a large raw corpus. There are three potential advantages of pre-training: larger vocabulary coverage (i.e., lower out-of-vocabulary rates), more accurate representations (due to the larger training corpus), and faster training of the tagger. However, word embeddings do not help in the case of the remaining out-of-vocabulary words, which will just be associated with a vector of zeros.

In order to produce character embeddings, each word is split into its character sequence, which is then fed to a separate bi-directional LSTM layer one character at a time. The last states of this LSTM layer are then concatenated and serve as a representation of the word. With this option, words that are written in a similar manner obtain similar representations. Breaking down the words into characters enables the tagger to create meaningful representations for out-of-vocabulary words.[9]

Plank et al. (2016) show that character embeddings tend to be slightly more efficient for tagging than word embeddings, but that using both representation types (by concatenating the character embeddings and word embeddings) consistently improves the results. They also show that pretrained word embeddings lead to higher tagging accuracies compared to the embeddings trained within the tagger. Cotterell and Heigold (2017) only use character representations, arguing that they are better adapted to cross-lingual settings where out-of-vocabulary rates are high. We follow the same reasoning in this section and report experiments based on character embeddings alone. In Section 6, we discuss different ways of building cross-lingual word embeddings for Rusyn and their impact on tagging performance.

The work cited above considers each output tag as a distinct entity; consequently, tagging a word consists of choosing one element from the list of output tags. This approach works for part-of-speech tagging where the list of possible output tags is small, and elements are mutually exclusive. For full morphological tagging however, this approach may not be sufficient, for a number of reasons. First, due to the combinatorics of different features, the number of distinct output tags is very high (around 4 000 in our training data), making the choice considerably more difficult than for part-of-speech tagging (less than 20 distinct tags). Second, the tagger should be able to share information across tags that differ only partially, e.g. across

[9] This assumes that there are no out-of-vocabulary characters, i.e., that the whole character inventory has been seen at training time, which generally holds to a large degree.

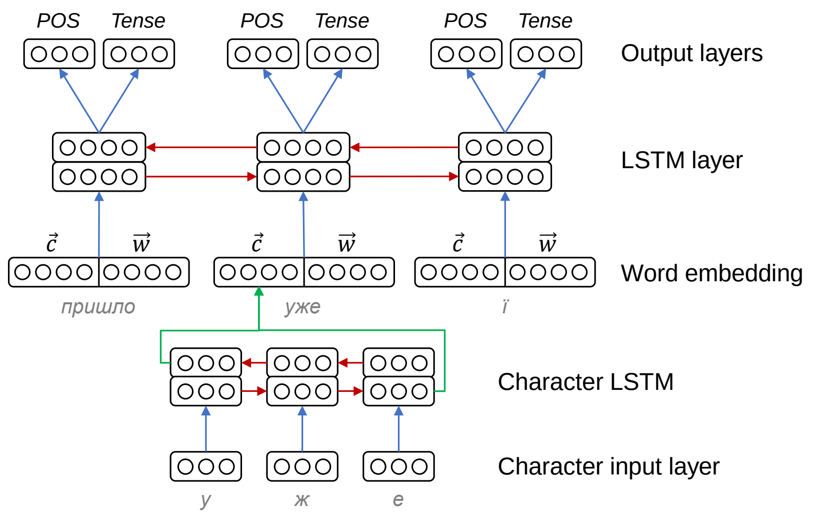

Fig. 2. Illustration of our tagging architecture. Blue arrows represent feed-forward connections, red arrows represent recurrent connections inside an LSTM layer, and green arrows represent mere copy operations. The embedding layer consists of a character embedding part $\vec{c}$ and an optional word embedding part $\vec{w}$. The number of output layers corresponds to the number of morphological features found in the data – we exemplify this with the two features *POS* and *Tense*.

the two following tags:
POS=NOUN|Animacy=Inan|Case=Dat|Gender=Masc|Number=Plur
POS=NOUN|Animacy=Anim|Case=Dat|Gender=Masc|Number=Plur
Third, the tagger should be able to predict combinations of values that it has never seen during training. For example, it should be (theoretically) able to compose a tag for a dative singular noun even if it only has ever seen dative plural and nominative singular nouns during training.

To alleviate these issues, we follow a proposal by Pinter et al. (2017) and create as many output layers as there are distinct features in the training data. Features that are not relevant are given a special value *N/A* that prevents them from being outputted. The part-of-speech tag has its own output layer, like any other feature.[10] Figure 2 illustrates the final neural tagger architecture. The $\vec{w}$ part is absent from the models presented in Sections 4 and 5, but present in the models of Sections 6.

As shown elsewhere for pre-modern Slavic including varieties of Church Slavonic (Scherrer, Mocken, & Rabus, 2018), character-level neural network taggers with per-feature output layers perform well on highly inflectional languages such as Slavic languages that require large tagsets, achieving well over 90% micro-averaged F1 for full morphological tagging. Using character embeddings instead of word embeddings made the tagger more robust towards orthographic variation. The pre-modern Slavic case is similar to the Rusyn case in that orthographic variability is a frequent

[10] A similar option has been proposed by Malaviya et al. (2018).

phenomenon leading to a comparatively large amount of OOV tokens. However, we cannot expect comparable performances for Rusyn, as we do not have any language-specific training data.

The architecture and training procedure of a neural network tagger are defined by a large number of hyperparameters. In the experiments presented here, we use a tagger with 2 LSTM layers of 256 dimensions each. The input to this tagger are character embeddings, which are produced by a character input layer of 128 dimensions and 2 additional LSTM layers with 256 dimensions each. We train all taggers during 20 epochs[11] using stochastic gradient descent with momentum; the initial learning rate is set to 0.01, with a decay of 0.1. We also apply dropout with a probability of 0.02. The tagger is implemented in Python and uses the DyNet toolkit (Neubig et al., 2017). We call this tagger CLSTM.[12]

4.2 Baseline systems

We compare our neural tagging experiments with two traditional tagger architectures: TnT (Brants, 2000) and MarMoT (Müller et al., 2013). TnT is based on a Hidden Markov model architecture and thus very fast to train, but is not particularly adapted for full morphological tagging, as it does not allow the decomposition of a full tag into a list of feature-value pairs. MarMoT uses a higher-order CRF architecture that is optimized for large tagsets such as those used in morphological tagging.

MarMoT has been used in our previous experiments (Scherrer & Rabus, 2017). We also present the results of these experiments using the current evaluation metrics.

4.3 Experiments and results

In a first set of experiments, we compare three tagger architectures, two different training sets (5k or 10k training sentences per source corpus) and two casing schemes (original case or all lowercase). Table 2 reports the results in terms of out-of-vocabulary rates and tagging F1 scores. As described above, reporting pure accuracy is too harsh, which is why we used micro-averaged per-feature F1 scores for evaluation. For our mixed-language development set, we report these scores on all features (*Full F1*) as well as on the reduced feature set (*Red. F1*) defined on the annotation of the Rusyn corpus. For the Rusyn test set, only the latter score is meaningful.

The two first lines of Table 2 show the results obtained with two systems presented in Scherrer and Rabus (2017); the *MS* system is most comparable to the new experiments, whereas the *LEX-L* system was the best-performing one using an

[11] We follow Plank et al. (2016) for this setting. Although we have found parameters to converge on the development set after about 20 epochs, we still observe considerable fluctuation in test set performance. In the absence of a proper Rusyn development set, we cannot devise a more principled way of determining convergence.

[12] The implementation is available under `https://github.com/yvesscherrer/lstmtagger`.

Table 2. *Comparison of tagging performance with different tagger architectures, training datasets and evaluation datasets.*

		Development data			Test data	
Tagger	Dataset	OOV rate	Full F1	Red. F1	OOV rate	Red. F1
MarMoT	MS				26.4%	70.5%
(S & R 2017)	LEX-L				1.1%	73.5%
TnT	5k	21.69%	87.64%	88.22%	23.9%	76.1%
	5k LC	20.05%	87.61%	88.17%	23.3%	**76.2%**
	10k	19.38%	88.51%	89.06%	23.0%	75.5%
	10k LC	17.82%	**88.54%**	**89.07%**	22.5%	75.3%
MarMoT	5k	21.69%	90.77%	91.44%	23.9%	**77.4%**
	5k LC	20.05%	90.58%	91.11%	23.3%	75.7%
	10k	19.38%	**91.50%**	**92.16%**	23.0%	75.1%
	10k LC	17.82%	91.28%	91.77%	22.5%	74.9%
CLSTM	5k	21.69%	93.05%	93.32%	23.9%	79.7%
	5k LC	20.05%	92.48%	92.55%	23.3%	78.3%
	10k	19.38%	**93.74%**	**93.98%**	23.0%	79.4%
	10k LC	17.82%	93.25%	93.31%	22.5%	**79.8%**

automatically induced morphological dictionary.[13] The comparison with the current MarMoT results indicates the performance boost obtained by the updated training resources (more and larger datasets, better annotation harmonization).

Regarding the three tagger architectures tested here, MarMoT performs around 3% better than TnT and CLSTM performs about 2% better than MarMoT on the multilingual development set. On the Rusyn test data, the ordering is confirmed, although the increase rates are lower (about +1% for MarMoT vs. TnT, about +2% for CLSTM vs MarMoT).

In terms of training data size, we expected the taggers trained on the larger 10k dataset to yield better performance. This behavior is confirmed on the development set, whereas the test set often obtained better performance with taggers trained on the smaller 5k dataset. The reason for this – somewhat surprising – outcome may be the proportionally higher amount of Ukrainian data in the 5k set, which is likely to be the most relevant source language for tagging Rusyn.

As it is often the case in transcription of oral corpora, the Rusyn test corpus is entirely lowercased. Therefore, we assumed that lowercasing the training data could make it more similar to the test data and hence lead to better tagging accuracies. While the TnT tagger is not very sensitive to casing, the more advanced taggers MarMoT and CLSTM usually perform significantly better with mixed case data, which makes them more suitable for real-world applications. On the test set, the

[13] The F1-scores shown here correspond to the accuracy scores of 72.0% and 75.5% respectively reported in Scherrer and Rabus (2017).

MarMoT results show the same preference, whereas the situation is less clear with CLSTM. However, it should be noted that the CLSTM F1 rates on the test set vary quite a lot from one training epoch to another.

Overall tagging performance on the development data using conventional taggers such as MarMoT and neural network taggers such as CLSTM yields results well beyond 90%, which is almost as high as the performance of monolingual systems on languages with rich morphology. In contrast, tagging performance on the Rusyn test data is considerably lower. Part of that loss may be due to the limited size of the test set. However, we believe that the main reason is the specifically oral syntax of the test data with omissions, repetitions, many discourse markers, etc. Because of that, our results should be compared to the performance of taggers trained on written data from languages with rich morphology applied to spoken data as test files. As is well known and has been demonstrated numerous times, the use of standard training data for tagging nonstandard, more specifically oral, test data inevitably leads to worse results as compared to standard test data (Ljubešić, Erjavec, & Fišer, 2017).

These experiments clearly confirm that neural taggers work successfully in difficult settings with little training data from a mix of different languages. In the light of these results, and since character-level neural taggers seems to be rather immune towards different casing and sampling strategies, we use a neural network architecture based on CLSTM for our further experiments.

5 Language identification as auxiliary task

A tagger trained on data from several languages may benefit from knowing the language of each sentence or word it is seeing. For example, the tagger may learn that a particular suffix corresponds to feature A in one language, but to feature B in another language. In the tagging experiments presented above, this information is not available – the distinction between different languages is even voluntarily blurred by converting all data to Cyrillic script.

Language information can be added in several ways:

1. Add an artificial token at the beginning of each sentence. This serves as a type of initialization of the tagging process. This approach has become popular in machine translation (Johnson et al., 2017).
2. Add a language label to each token, ideally encoded separately from the word form. This approach is used e.g. by Ammar et al. (2016).
3. Add a language label in the form of an additional morphological feature to the tag. This approach can be viewed as a form of multi-task learning, where the additional task is language identification (Cotterell & Heigold, 2017).

The first two approaches add the language information on the input side (the words to be tagged), whereas the third approach adds the information to the output side (the tags to be produced). This difference is crucial at test time: in the first two approaches, the language of the test data must be known, whereas in the third approach it is not given, but predicted as a byproduct of the tagging. In our case,

423	NUM	Animacy=Inan|Case=Acc|Gender=Masc|NumType=Card|Number=Plur|**Lang=pl**
разы	NOUN	Animacy=Inan|Case=Acc|Gender=Masc|Number=Plur|**Lang=pl**
споткалы	VERB	Animacy=Inan|Aspect=Perf|Gender=Masc|Mood=Ind|Number=Plur|Tense=Past|VerbForm=Fin|Voice=Act|**Lang=pl**
ся	PRON	PronType=Prs|Reflex=Yes|**Lang=pl**
чтеры	NUM	Animacy=Inan|Case=Nom|Gender=Masc|NumType=Card|Number=Plur|**Lang=pl**
рядове	ADJ	Animacy=Inan|Case=Nom|Degree=Pos|Gender=Masc|Number=Plur|**Lang=pl**
комитеты	NOUN	Animacy=Inan|Case=Nom|Gender=Masc|Number=Plur|**Lang=pl**
.	PUNCT	**Lang=pl**

Fig. 3. Example of the Polish sentence *423 razy spotkały się cztery rządowe komitety.*, cyrillized and annotated with language labels (in bold).

the test language (Rusyn) is different from all languages seen during training, so that the first two approaches cannot be used reasonably.

The second and third approaches add language information at the token level, whereas the first approach uses a single language label per sentence. Since our training data does not contain sentences mixing several languages, the first approach would be most reasonable. However, the Rusyn data may contain instances of code-switching or calques from different source languages, such that language annotation on token level may be useful.

For the reasons exposed above, we follow the third approach. In our neural tagger architecture, learning an additional task (language identification) is equivalent to learning an additional morphological feature. We therefore modify our training data by adding the feature *Lang* and the corresponding ISO code. An example of such a modified sentence is given in Figure 3.

This simple way of adding language labels has two drawbacks: Since all tokens of a sentence are given the same label, the tagger might just learn to predict the label for the least ambiguous word of the sentence and copy it to the remaining words, without associating the label prediction with the word forms, limiting its usefulness. Second, labeling tokens such as *423* or . as Polish does not make much sense, as they would look exactly the same in a different language. In order to address these potential problems, we devise a second way of assigning language labels. Tokens which occur in at least four languages, such as punctuation symbols or named entities as well as numbers are assigned the language label *gn* (generic). Moreover, we swap the labels of tokens which occur in more than one language with the same morphosyntactic annotation in order to break the "one sentence – one language" constraint. An example of this modified language labeling is shown in Figure 4.

Table 4 summarizes the results of the language labeling experiments using the

423	NUM	Animacy=Inan|Case=Acc|Gender=Masc|NumType=Card|Number=Plur|**Lang=gn**
разы	NOUN	Animacy=Inan|Case=Acc|Gender=Masc|Number=Plur|**Lang=ru**
споткалы	VERB	Animacy=Inan|Aspect=Perf|Gender=Masc|Mood=Ind|Number=Plur|Tense=Past|VerbForm=Fin|Voice=Act|**Lang=pl**
ся	PRON	PronType=Prs|Reflex=Yes|**Lang=pl**
чтеры	NUM	Animacy=Inan|Case=Nom|Gender=Masc|NumType=Card|Number=Plur|**Lang=pl**
рядове	ADJ	Animacy=Inan|Case=Nom|Degree=Pos|Gender=Masc|Number=Plur|**Lang=pl**
комитеты	NOUN	Animacy=Inan|Case=Nom|Gender=Masc|Number=Plur|**Lang=ru**
.	PUNCT	**Lang=gn**

Fig. 4. Example of the Polish sentence *423 razy spotkały się cztery rządowe komitety.*, cyrillized and annotated with swapped language labels. Note the generic *gn* label for numbers and punctuations, as well as the Russian label associated to the second and second-last token, based on the existence of the plural noun forms *разы* and *комитеты* in a Russian corpus.

originally cased 5k training dataset. Note that the correctness of language labels is evaluated separately and is not included in the F1 computations. For comparison, we repeat the results of the tagger without added language labels in the first row.

When evaluated on development data, the standard language labels do not lead to significant performance changes. Also, the accuracy of the language prediction is almost perfect, as the model learns to look at adjacent words for difficult cases. On the Rusyn test set however, the F1-score drops heavily, probably again due to the same effect: the model tries to adapt to a situation in which the whole sentence is written in a single language of the training set. Note that language prediction on the test data would always lead to 0% accuracy, since the model has not seen the correct label *rue* during training.

With the swapped language labels, language prediction accuracy drops significantly on the development data, because the swapping has made the problem more difficult and less deterministic. Consequently, tagging performance slightly drops as well. However, we expect this approach to work better on Rusyn, which is indeed the case when compared with the standard labels. However, both language labeling models are not able to match the performance of the simple model without language labels.

The predicted language labels of the test tokens may give some clues about the behavior of the two different labeling schemes. Table 3 shows the distribution of the labels across the 1 051 tokens of the test set. One can see that the effect of the standard language labels is to mark almost all Rusyn utterances as Ukrainian, which is indeed the most closely related training variety. With the swapped labels, this dominance is broken, partly due to the newly introduced generic label, but partly also to the higher percentage of Czech and Polish labels.

Table 3. *Comparison of language labeling schemes. The Lang Acc column shows the accuracy of the auxiliary language identification task.*

	Development data			Test data
CLSTM 5k Tagger	Full F1	Reduced F1	Lang Acc	Reduced F1
No language labels	93.05%	93.32%	—	79.7%
Standard language labels	93.09%	93.34%	97.82%	76.8%
Swapped language labels	92.92%	93.17%	92.17%	79.0%

Table 4. *Distribution of language labels in the Rusyn test set (absolute token numbers). The last column shows the proportion of uniquely labeled test set sentences (not taking into account the gn label).*

	gn	cs	pl	ru	sk	uk	Uniquely lab. sent.
Standard language labels	0	37	54	96	40	824	95/105
Swapped language labels	250	65	123	80	41	492	22/105

A more detailed analysis of the language labels may give us hints about regional variation in the Rusyn corpus. For example, labeled data can help discover the amount of lexical loans in each of the Rusyn varieties with Slavic umbrella languages and, thus, facilitate linguistic research.

6 Word embeddings

Many recent pieces of research include both character embeddings and (pretrained) word embeddings in tagging models. In this section, we investigate to what extent word embeddings can also help in a transfer learning setting.

As a starting point, we replace the $\vec{c}$ part of our LSTM tagger by the $\vec{w}$ part. These word embeddings are trained jointly with the rest of the neural network. As a result, each distinct word form of the training corpus is associated with an embedding vector that reflects the distributional properties of the word form. The word forms used to train the embeddings are lowercased cyrillicized forms, and the embeddings are set to 300 dimensions.[14] This model is not expected to yield good performance on Rusyn, as it essentially does not know what to do with out-of-vocabulary words.[15]

We also combine the $\vec{c}$ and $\vec{w}$ parts, as illustrated in Figure 2. This model is

[14] These casing and dimensionality choices were made to facilitate comparison with subsequent experiments.

[15] One option to reduce the number of out-of-vocabulary forms in morphologically rich languages is to use embeddings of lemmas instead of embeddings of inflected word forms. While we could train such a model using the available lemmas in the training data, our Rusyn data is not lemmatized. Solving the Rusyn lemmatization problem may actually be even harder than the tagging problem because there are no standardized spelling and lemmatization conventions.

Table 5. *Comparison of LSTM taggers trained on the 5k corpus with different types of word embeddings.*

	Development data			Test data	
Embeddings	OOV rate	Full F1	Red F1	OOV rate	Red F1
$\vec{c}$	21.69%	93.05%	93.32%	23.9%	79.7%
$\vec{w}$	20.03%	80.58%	80.50%	23.3%	67.1%
$\vec{c} + \vec{w}$	20.03%	93.11%	93.42%	23.3%	78.1%
$\vec{c} + \vec{w}$ Panslav5	6.28%	**93.59%**	**93.86%**	16.2%	77.7%
+ Rusyn OOV words	6.28%	93.39%	93.63%	0%	**81.0%**
$\vec{c} + \vec{w}$ Panslav6					
+ Rusyn OOV words	6.26%	93.39%	93.67%	0%	78.2%

expected to improve tagging accuracy on the development set thanks to the complementary information contained in the two embedding types. However, we do not expect significant improvements on the Rusyn test set, since the predictions for out-of-vocabulary words will have to back off to the character embeddings. The first three rows of Table 5 confirm our expectations. The test data F1 for $\vec{c} + \vec{w}$ decreases considerably compared to $\vec{c}$.[16]

Instead of training the word embeddings on the rather small tagging dataset, we can take advantage of word embeddings that have been pretrained on large raw corpora of the source languages. Sharoff (2018) takes the *Fasttext* (Bojanowski, Grave, Joulin, & Mikolov, 2017) embeddings as a starting point for his work; these embeddings have been independently trained on lowercased Wikipedia data and contain 300 dimensions. Sharoff (2018) converts the independently trained embeddings of all Slavonic languages into a shared vector space using known conversion methods and automatically induced bilingual dictionaries.

For our purposes, the entries of the five source language embedding files need to be cyrillicized and merged into a single file for inclusion in the tagger. When doing so, embeddings may overlap. For example, the Slovak word *budú* is cyrillicized to *буду*, which is also an Ukrainian and Russian word. Since double entries are not allowed in the word embedding files, we merge the vectors of identical word forms by taking their average. In our example, the vector of *буду* would therefore be computed as the average of Slovak *budú*, Ukrainian *буду* and Russian *буду*. We assume that a large proportion of these overlapping word forms correspond to those found by Sharoff (2018)'s lexicon induction technique, so that their respective vectors are already quite close to each other. Table 6 shows the statistics of the *Fasttext* embeddings; the cyrillicized merged embedding file is called *Panslav5*.

The fifth row of Table 5 shows the corresponding tagging results. The OOV rate

[16] The slightly decreasing OOV rate can be explained by the casing differences between character embeddings (original case) and word embeddings (lower case), as mentioned above.

Table 6. *Multilingual Fasttext word embeddings.*

	cs	pl	ru	sk	uk	Panslav5	rue	Panslav6
Entries	200k	200k	200k	55.6k	52.8k	581.8k	10.4k	587.4k
+ Rusyn OOV words:						582.0k	10.7k	587.6k

goes down considerably for the development set thanks to the addition of the large-coverage pretrained embeddings. The OOV rate also decreases on the test set, as many Rusyn words correspond to identically spelled source language words of the embeddings. Tagging performance on the development set increases, confirming the effectiveness of pretrained embeddings already observed by (Plank et al., 2016). In contrast, tagging performance on the Rusyn test set decreases substantially despite the lower OOV rate.

In Rabus and Scherrer (2017), we have found that a large proportion of Rusyn word forms can be matched with source language word forms using hand-crafted correspondence rules or vowel-sensitive Levenshtein distance. Here, we use the latter approach to artificially create Rusyn word embeddings for the out-of-vocabulary words: each Rusyn word form of the test set that does not occur in the Panslav5 file is associated with the most similar forms in Panslav5, and its embedding vector is computed by averaging the vectors of the found similar forms. For example, the embedding for the Rusyn word чекают corresponds to the averaged vector of the most similar words чекат and чекаюты, both at normalized Levenshtein distance of 0.83. This only adds 129 entries to the Panslav5 file (see second row of Table 6).

The fourth row of Table 5 shows that this technique considerably increases Rusyn tagging performance, bringing down the Rusyn OOV rate to zero by design. Note that this vocabulary expansion strategy could also be carried out using a large Rusyn word list in order to make it completely independent of the test set. However, the proposed setup is realistic in our real-world scenario of annotating the Corpus of Spoken Rusyn, as the text to be tagged can be made available before tagger training.

6.1 Pretrained Rusyn embeddings

The *Fasttext* project actually provides pretrained word embeddings for Rusyn based on Wikipedia data,[17] albeit of much smaller size than for the other languages (see Table 6). Using this existing resource may thus be a valid alternative to the above-mentioned approach of “synthesizing” Rusyn embeddings from similar words.

The pretrained Rusyn embeddings still need to be projected into the Panslav5 embedding space. To this end, we use a simpler procedure than Sharoff (2018) and take the intersection of word forms of the two files as the seed lexicon (i.e., we assume that words spelled the same should occur in the same area of the vector

[17] `https://fasttext.cc/docs/en/pretrained-vectors.html`

space). Almost half of the forms present in the Rusyn Fasttext embeddings are covered by this lexicon. We call the resulting embeddings *Panslav6* (see Table 6).

Not all words occurring in the Rusyn test set are covered by the Panslav6 embeddings. In order to obtain comparable results to those shown above, we again create synthetic word embeddings for the OOV words of the test corpus. However, we can use a potentially more accurate approach here: the *Fasttext* embedding format enables us to create embedding vectors for unknown words based on subword information (Bojanowski et al., 2017). We generate these using the Rusyn Wikipedia embeddings and then merge them alongside the known ones into the Panslav6 embeddings (see second row of Table 6).

The tagging results obtained with the Panslav6 embeddings are shown in the last row of Table 5. The drop in development F1 is striking and may be due to the additional alignment step performed to synchronize the Rusyn embeddings with the Panslav5 embeddings. Further analysis will be required on this subject. Consequently, the F1 score on the test data is not competitive either and merely matches the one obtained with word embeddings trained within the tagger.

These results suggest that the usefulness of pretrained word embeddings is still open to debate in the context of transfer learning settings like the one presented here. While small but consistent improvements can be observed in the development data set, they translated into degradations on the test data in all but one cases; only the addition of Rusyn OOV words into a previously aligned five-language vector space allowed us to break the 80%-mark in F1 score.

7 Conclusions and future work

Our experiments have shown that neural taggers help in improving tagging performance for low-resource languages such as Rusyn in a transfer learning setting relying only on training data from related languages. This setting is more challenging than e.g. Cotterell and Heigold (2017), as we do not use any annotated Rusyn data. We have shown that updated and cleaned training data boost tagging performance by about 7% F1-score, whereas the use of a neural network tagger adds another 2% to the F1-score. Pretrained word embeddings automatically extended to full test set coverage add another 1%, resulting in a F1-score of 81.0%.

In this paper, we have only explored a few options – there are a multitude of algorithms to infer word embeddings and to align independently trained word embeddings into a common cross-lingual space. For example, Plank and Agić (2018) report consistently better performance when starting with Polyglot embeddings (Al-Rfou, Perozzi, & Skiena, 2013) instead of Fasttext embeddings. Sharoff (2018) mentions further improvements on named entity recognition when using the MUSE method (Lample, Conneau, Ranzato, Denoyer, & Jégou, 2018) to align cross-lingual embeddings rather than the one used here. Multilingual extensions (i.e. merging more than two languages at a time) for cross-lingual embeddings have also been proposed and could be relevant in our case. Further work will thus be needed on finding the optimal combinations of methods for the given task.

We have also experimented with multi-task learning, using language identification as an auxiliary task, but did not find improvements in tagging performance. Further experimentation is required in this area as well; for example, multi-task learning could be combined with the pretrained word embeddings.

Our results also made clear that transfer experiments using Slavic languages are especially challenging because of the rich morphology of the Slavic languages. Furthermore, spoken data as represented in both our test data and the corpus data we intend to practically use the tagger on pose certain difficulties. Therefore, in further studies, the issue of tagging oral genres of low-resource languages should be tackled specifically.

Acknowledgements

Funding sources for this paper include DFG (RA 2212/2-1 "Rusyn as a minority language crossing state borders – dynamic processes") and Humboldt Foundation (Research Group Linkage Program "Digipalslav").

References

Agić, Ž., Hovy, D., & Søgaard, A. (2015). If all you have is a bit of the Bible: Learning POS taggers for truly low-resource languages. In *Proceedings of ACL-IJCNLP 2015* (pp. 268–272). Beijing, China.

Al-Rfou, R., Perozzi, B., & Skiena, S. (2013). Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of CoNLL 2013* (pp. 183–192). Sofia, Bulgaria.

Ammar, W., Mulcaire, G., Ballesteros, M., Dyer, C., & Smith, N. (2016). Many languages, one parser. *TACL*, *4*, 431–444.

Artetxe, M., Labaka, G., & Agirre, E. (2017). Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of ACL 2017* (pp. 451–462). Vancouver, Canada.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *TACL*, *5*, 135–146.

Boudovskaia, E. (2017). Past tense in the Rusyn dialect of Novoselycja: auxiliary vs. subject pronoun as the first- and second-person subject. *Journal of Slavic Linguistics*, *25*(1), 3–62.

Brants, T. (2000). TnT - a statistical part-of-speech tagger. In *Proceedings of ANLP 2000* (pp. 224–231). Seattle, Washington, USA.

Buys, J., & Botha, J. A. (2016). Cross-lingual morphological tagging for low-resource languages. In *Proceedings of ACL 2016* (pp. 1954–1964). Berlin, Germany.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *JMLR*, *12*, 2493–2537.

Cotterell, R., & Heigold, G. (2017). Cross-lingual character-level neural morphological tagging. In *Proceedings of EMNLP 2017* (pp. 748–759). Copenhagen, Denmark.

Feldman, A., Hana, J., & Brew, C. (2006). A cross-language approach to rapid creation of new morpho-syntactically annotated resources. In *Proceedings of LREC 2006* (pp. 549–554). Genoa, Italy.

Johnson, M., Schuster, M., Le, Q., Krikun, M., Wu, Y., Chen, Z., ... Dean, J. (2017). Google's multilingual neural machine translation system: Enabling zero-shot translation. *TACL*, *5*, 339–351.

Koporová, K. (Ed.). (2015). *Rusyn'skŷj literaturnŷj jazŷk na Slovakiji: 20 rokiv kodifikaciji: Zbornyk referativ z iv. midžinarodnoho kongresu rusyn'skoho jazŷka – Prjašiv, 23.–25.09.2015.* Prjašiv: Prjašivska univerzita v Prjašovi – Inštitut rusyn'skoho jazŷka i kulturŷ.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT 2016* (pp. 260–270). San Diego, California.

Lample, G., Conneau, A., Ranzato, M., Denoyer, L., & Jégou, H. (2018). Word translation without parallel data. In *Proceedings of ICLR 2018.* Vancouver, Canada.

Ljubešić, N., Erjavec, T., & Fišer, D. (2017). Adapting a state-of-the-art tagger for South Slavic languages to non-standard text. In *Proceedings of the 6th workshop on Balto-Slavic natural language processing* (pp. 60–68). Valencia, Spain.

Magocsi, P. R. (Ed.). (2004). *Rusyn'skŷj jazŷk.* Opole: Uniw. Opolski Inst. Filologii Polskiej.

Malaviya, C., Gormley, M. R., & Neubig, G. (2018). Neural factor graph models for cross-lingual morphological tagging. In *Proceedings of ACL 2018* (pp. 2653–2663). Melbourne, Australia.

McDonald, R., Petrov, S., & Hall, K. (2011). Multi-source transfer of delexicalized dependency parsers. In *Proceedings of EMNLP 2011* (pp. 62–72). Edinburgh, Scotland, UK.

Müller, T., Schmid, H., & Schütze, H. (2013). Efficient higher-order CRFs for morphological tagging. In *Proceedings of EMNLP 2013* (pp. 322–332). Seattle, Washington, USA.

Neubig, G., Dyer, C., Goldberg, Y., Matthews, A., Ammar, W., Anastasopoulos, A., ... Yin, P. (2017). DyNet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., ... Zeman, D. (2016). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of LREC 2016* (pp. 1659–1666). Portorož, Slovenia.

Pinter, Y., Guthrie, R., & Eisenstein, J. (2017). Mimicking word embeddings using subword RNNs. In *Proceedings of EMNLP 2017* (pp. 102–112). Copenhagen, Denmark.

Plank, B., & Agić, Ž. (2018). Distant supervision from disparate sources for low-resource part-of-speech tagging. In *Proceedings of EMNLP 2018* (pp. 614–620). Brussels, Belgium.

Plank, B., Søgaard, A., & Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In

Proceedings of ACL 2016 (pp. 412–418). Berlin, Germany.

Plishkova, A. (2009). *Language and national identity: Rusyns south of Carpathians.* New York: Columbia University Press and East European Monographs.

Rabus, A. (2015). Current developments in Carpatho-Rusyn speech – preliminary observations. In P. A. Krafcik & V. I. Padjak (Eds.), *Juvilejnyj zbirnyk na čest' profesora Pavla-Roberta Magočija* (pp. 489–496). Užhorod.

Rabus, A. (2019). Vergangenheitsbildung in gesprochenen karpatorussinischen Varietäten: Quantitativ-statistische Perspektiven. *Die Welt der Slaven*, *LXIX*(1), 15–33.

Rabus, A., & Scherrer, Y. (2017). Lexicon induction for spoken Rusyn – challenges and results. In *Proceedings of the 6th workshop on Balto-Slavic natural language processing* (pp. 27–32). Valencia, Spain.

Ruder, S., Vulić, I., & Søgaard, A. (2018). A survey of cross-lingual word embedding models. *JAIR.*

Scherrer, Y. (2014). Unsupervised adaptation of supervised part-of-speech taggers for closely related languages. In *Proceedings of VarDial 2014* (pp. 30–38). Dublin, Ireland.

Scherrer, Y., Mocken, S., & Rabus, A. (2018). New developments in tagging premodern Orthodox Slavic texts. *Scripta & e-Scripta*, *18*, 9–33.

Scherrer, Y., & Rabus, A. (2017). Multi-source morphosyntactic tagging for spoken Rusyn. In *Proceedings of VarDial 2017* (pp. 84–92). Valencia, Spain.

Sharoff, S. (2018). Language adaptation experiments via cross-lingual embeddings for related languages. In *Proceedings of LREC 2018* (pp. 844–849). Miyazaki, Japan.

Skrypnyk, H. A. (Ed.). (2013). *Ukrajinci-rusyny: Etnolinhvistyčni ta etnokul'turni procesy v istoryčnomu rozvytku.* Kyjiv: Instytut mystectvoznavstva, fol'klorystyky ta etnolohiji im. M.T. Ryl's'koho.

Täckström, O., McDonald, R., & Uszkoreit, J. (2012). Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of NAACL-HLT 2012* (pp. 477–487). Montréal, Canada.

Yarowsky, D., & Ngai, G. (2001). Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of NAACL-LT 2001* (p. 200-207). Pittsburgh, PA, USA.

Yu, Z., Mareček, D., Žabokrtský, Z., & Zeman, D. (2016). If you even don't have a bit of Bible: Learning delexicalized POS taggers. In *Proceedings of LREC 2016* (pp. 96–103).

Zeman, D., & Resnik, P. (2008). Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP 2008 workshop on NLP for less privileged languages* (pp. 35–42). Hyderabad, India.

Zhang, Y., Gaddy, D., Barzilay, R., & Jaakkola, T. (2016). Ten pairs to tag – multilingual POS tagging via coarse mapping between embeddings. In *Proceedings of NAACL-HLT 2016* (pp. 1307–1317). San Diego, California.