Tagging Ingush – Language Technology For Low-Resource Languages Using Resources From Linguistic Field Work

Jörg Tiedemann
Department of Modern Languages
University of Helsinki, Finland
firstname.lastname@helsinki.fi

Johanna Nichols and **Ronald Sprouse**
Department of Linguistics
University of California, Berkeley
firstname@berkeley.edu

Abstract

This paper presents on-going work on creating NLP tools for under-resourced languages from very sparse training data coming from linguistic field work. In this work, we focus on Ingush, a Nakh-Daghestanian language spoken by about 300,000 people in the Russian republics Ingushetia and Chechnya. We present work on morphosyntactic taggers trained on transcribed and linguistically analyzed recordings and dependency parsers using English glosses to project annotation for creating synthetic treebanks. Our preliminary results are promising, supporting the goal of bootstrapping efficient NLP tools with limited or no task-specific annotated data resources available.

1 Introduction

Linguistic diversity is not only a fascinating research area for general linguists but also represents one of the main challenges for language technology. Natural language processing (NLP) becomes essential for people in the digital age and support of low-resource languages is a natural task that needs to be emphasised in the development of tools and applications. Modern language technology relies to a large extent on data-driven techniques that focus on machine learning techniques based on annotated linguistic data and large quantities of raw text. However, such techniques are usually not applicable for low-resource languages where sufficient training data is not available. On the other hand, a lot of effort is spent in field linguistics to record, transcribe and analyse minority languages – a resource that is under-explored in language technology.

In this paper, we take the example of Ingush, a Nakh-Daghestanian language, which has been thoroughly described by general linguists with data sets collected over several years. We use those manually created and curated collections of transcribed recording to bootstrap tools that can be used to process the language and to annotate utterances with interlinear glosses and syntactic dependencies. The paper is the first step in developing technology that could be used to support linguistic field work but also to develop NLP applications that can work with the Ingush language.

In the following, we first introduce Ingush and the linguistic resources developed for that language. Thereafter, we present our work on developing an interlinear gloss tagger and outline an approach for bootstrapping part-of-speech taggers and dependency parser by means of alignment and cross-lingual transfer models.

2 The Ingush Language

Ingush (glottolog code ingu1240) is a Nakh-Daghestanian language with some 300,000 speakers, traditionally spoken in the central Caucasus highlands in the Republic of Ingushetia and Chechen Republic of Russia. The language has extensive systems of both consonants and vowels, largely suffixing morphology, mostly dependent-marking morphosyntax, ergative alignment, and German-like AOV/V2 word order complete with detachable prefixes and proclitics in final position of V2 clauses, though with some

Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH), pages 148–155, Osaka, Japan, December 11-17 2016.

flexibility to use AOV order even in main clauses. Nouns belong to gender classes marked by root-initial agreement on about one-third of the verbs and one-fourth of the adjectives; there are four gender markers. Clause combining mostly uses converbs, some of which require S/O sharing and all of which use much null anaphora. Null arguments result from sharing (in some converb clauses and all infinitive and relative clauses) and unspecified reference; though not a pro-drop language there are many null arguments. The language is further described in Nichols (2011).

An active program to record the speech of the oldest generations of Ingush began in the mid 1990s and reached speed in 2002, with 50-100 hours of recorded speech added annually. Recordings are mostly made in the homes of speakers. There are also some legacy recordings made in the late 1980s and early 1990s. A number of the recorded speakers were born in the very early 20th century and at least one in the very late 19th century. Quality varies: legacy materials were made on inexpensive household cassette recorders; more recent recordings are often made on mobile telephones. There are also about 100 hours of video recordings.

The corpus for this project is the transcribed portion of those recordings, plus a number of transcribed published folklore and literature works which have no audio or have a recorded reading of the published text. The transcription is an all-lower-ASCII Latin system, slightly more abstract than phonemic. It is not ismorphic to the Cyrillic orthography, which is mostly phonemic for consonants but greatly underdifferentiates the vowels. Annotation is standard single-line interlinears with lexical gloss and morpheme/category identifications, plus a smooth translation.

Most of the annotated data was created with dedicated tools developed for the Berkeley Interlinear Text Corpus, BITC. They are designed for group collaborations and support semi-automatic interlinearization by building a lexicon of interlinears on the fly, which enables systematic annotations of large quantities of data. The system also provides powerful search capacities making it very useful as a dictionary. These tools and data sets have contributed greatly to the efficiency of grammatical analysis and grammar writing and in this work, we investigate their use in automatic language processing.

3 Data Preparation

As described above, the Berkeley Ingush Corpus includes a well organised collection of transcribed utterances together with interlinear glosses and translations into fluent English. Figure 1 shows an example taken from the corpus.

Ingush:	Cwaqqa hama dwajihwaajaacar, jihwaajarii?
Tokenized:	cwaqqa hama dwajihwaajaacar jihwaajarii
Interlinear glosses:	any thing DX-J.take away.PNW.NEG J.take away.PNW=Q
English:	Nothing had been taken away, right?

Figure 1: An example record from the Berkeley Ingush Corpus.

In this way, the data forms a manually curated parallel corpus with three dimensions that are all useful for our purposes. The interlinear glosses are the actual annotation that we try to produce from transcribed Ingush input and the English translations become interesting for transfer models we will discuss towards the end of the paper.

One of the main problems is that manually annotated data sets from many years of linguistic field work contain inconsistencies, which is unavoidable even with extensive efforts on normalizing their contents. The records include incomplete annotations and analyses, which we have to take care of when preparing data sets for training models that rely on the correctness of the examples. Therefore, we implemented a tool that extracts and converts data sets into a format that we can feed into our training procedures.

The first step is to convert the character encoding from mainly MacRoman to Unicode UTF8. The second step includes various heuristics to normalize the data and to exclude unreliable records. The following list summarizes the processing steps we have taken:

- Remove comments and separate alternative translations: Some records include alternatives for the translations into English. They are sometimes included in square brackets but most of the time

they are separated by multiple space characters. Square brackets otherwise contain comments and explanations, which we cannot safely use, and, therefore, we remove them. Possible translation alternatives that we extract by splitting on multiple space characters are then tested by comparing the lengths (in terms of tokens) to the lengths of the Ingush original string. If the length-ratio between the smallest potential translation alternative is smaller than the ratio between the raw input string and the concatenated English translation then we will discard splitting the translation string and consider it as one single translation. Otherwise, we keep multiple translation alternatives and duplicate the record with different translations in each of them.

- We replace spaces (that usually indicate multi-word-units) with underscores to make it easier to handle the data by subsequent processes. An example can be seen in Figure 1 (*take away*). Some morphosyntactic descriptions also include spaces and we replace them by dots.

- Some interlinears are incomplete and do not cover all tokens in the input. In many cases, this may only exclude some unimportant final tokens and we use the heuristics that we accept interlinears that are at most two items too short.

- We save all records that have non-empty entries for all three records, tokenized Ingush, interlinear glosses and English translations and convert the interlinear glosses into a factorized representation. Here, we separate lexical information from morphosyntactic descriptions to create a generalized tagset that we can use to train automatic annotation tools by means of standard sequence labeling models.

Figure 2 shows the result of our pre-processing on the previous example. Lexical information in the interlinear annotation is replaced by a placeholder 'xx' and the lexical information is included as a separate factor (before the slash). The *delexicalized* gloss terms are then also added to the original Ingush tokens to form the essential training data that we will use in our tagging experiments below. The task is, hence, to tag each Ingush token from arbitrary input with the complex but delexicalised tags given by our training data (line one in Figure 2).

Tagged Ingush:	cwaqqa/xx hama/xx dwajihwaajaacar/DX-J.xx.PNW.NEG jihwaajarii/J.xx.PNW=Q
Tagged Glosses:	any/xx thing/xx take_away/DX-J.xx.PNW.NEG take_away/J.xx.PNW=Q
English:	nothing had been taken away , right ?

Figure 2: Preprocessed data: Tagged source language, split glosses and tokenized English.

Note that we had to perform various additional pre-processing steps to increase consistency of the data. We had to normalize the use of dots (older versions used mid-dots for marking morpheme boundaries), we removed question marks, normalized the use of brackets, spaces, multiple dots, the use of conjunctive markers, normalized the specification of alternatives, foreign languages and multi-word units. We also made the inflectional markup for person and number more consistent and added a special tag for foreign words.

	Ingush		Tags		English	
	tokens	types	tokens	types	tokens	types
training	102,043	19,964	101,826	8,190	130,186	7,257
test	6,222	2,375	6,165	1,261	7,099	1,268

Table 1: Statistics of the data sets. *Tags* refers to the delexicalised interlinears. The test data contains 888 sentences and the training data includes 9,209 sentences.

The final data may still contain additional noise and inconsistencies but our main interest is now to see whether it is sufficient to train annotation tools that can produce interlinear-like tags from unrestricted Ingush input. For this we divided the corpus into training and test sets. Table 1 summarizes the statistics of

the data sets. We can clearly see the inflectional complexity of Ingush which is striking in the type/token ratio in comparison to English. The statistics also show the complexity of the tagset we are looking at even after delexicalization. Note that the glosses are certainly not a fixed tagset but a productive annotation scheme. Nevertheless, we will treat the task as a standard classification-based sequence-labeling process as we will explain below.

4 Tagging with Interlinear Glosses

One of the goals of our experiments is to automatically create glosses and interlinears. The first step to approach this goal is to be able to produce delexicalized interlinears for arbitrary utterances in transcribed Ingush. We model the task as a sequence labeling task in which we search for the optimal tag sequence given a sequence of input tokens and a model that is trained on annotated data. We use the tagged corpus described in the previous section and train a model based on conditional random fields (CRFs) (Lafferty et al., 2001), which is optimized for large tag sets that contain morphological features rather than single part-of-speech labels. We apply *marmot* (Mueller et al., 2013),[1] a popular tool for morphologically-rich languages and treat morpheme descriptions separated by dots as our inflectional features to be produced by the tagger. The model describes a structural prediction task and a standard CRF defines a globally normalized log-linear model of the conditional probability of a tag sequence $\vec{y} = y_1, y_2, .., y_n$ given a sentence $\vec{x} = x_1, x_2, ..., x_n$ of n tokens that can be used to guide the predictions:

$$p(\vec{y}|\vec{x}) = \frac{exp \sum_{t,i} \lambda_i \cdot \phi_i(\vec{y}, \vec{x}, t)}{Z(\vec{\lambda}, \vec{x})}$$

In this formulation, ϕ_i is a feature function and λ_i its associated weight, t is the token index and $Z(\vec{\lambda}, \vec{x})$ is a normalising constant. Marmot provides an efficient implementation of that model using pruned training and tag decomposition. The latter is especially important for tasks like ours where the tag set becomes extremely large. In training, we then learn the parameters of the model that maximise the prediction accuracy with respect to the training data and at test time we apply common inference procedures to search for the best tag sequence given some input data and the model we have created.

The main challenge is certainly the complexity of the interlinears we wish to produce. We simplify the task by ignoring the compositional way of interlinear glosses and assume that we can treat their components as individual morphological features. We, hence, train the tagger with standard settings using the morpheme-delimiter to split the interlinears into sub-tags and hope that the model can cope well enough with predicting even complex annotations. Let us first look at the overall outcome of this model before diving into a deeper analyses of the results and some simple ways of improving the performance. Table 2 summarizes the performance of the tagger when applied to our unseen test set.

(scores in %)	including xx	without xx
average precision	82.68	71.43
average recall	81.51	67.76
accuracy	65.50	54.72

Table 2: Tagging transcribed Ingush with delexicalized interlinears.

We present three metrics that demonstrate the performance of the model: Precision, recall and accuracy. The latter refers to the standard metric of comparing gold standard labels with the proposed ones. This means that the metric only accepts exact matches of the entire interlinear string per token. However, the interlinear glosses are typically composed of various elements and, therefore, we are interested in measuring the closeness of the tags created.[2] Hence, we divide the tags into parts again (separated by morpheme-boundary markers) and compare gold standard interlinears with proposed tags by means of string similarity measures. In particular, we compute the longest common subsequence (LCS) between

[1] Available from `http://cistern.cis.lmu.de/marmot/`.

[2] Note that dot-separated elements sometimes refer to categories that are fused into a single morpheme but in general the sequential order matters referring to the sequence of morphological elements of the corresponding word.

		including xx		without xx		
reference	predicted	P	R	P	R	token
xx.NW.D.NEG	xx.NW.D.NEG	100	100	100	100	xeattaadaac
DEM.PL.OBL	DEM.OBL	100	67	100	67	cy
xx.PL.DAT	xx.PL.DAT	100	100	100	100	bierazhta
D.PST=PTC	D.xx.PST=CUM	50	67	67	67	dar=q
DX-xx-J.xx.NW.J.NEG	DX-xx.AUX.NEG.PRS	25	20	25	25	dwachyjeannajaac
xx:NEG.PRS	xx.PRS.NEG	33	50	50	50	xaac
xx-J.xx.CVtemp	xx-D.xx.CVtemp	67	67	50	50	chyjiecha
J.xx.NEG.WP	J.AUX.NEG.WP	75	75	75	100	jaxandzar

Table 3: Examples of predicted interlinear glosses (precision (P) and recall (R) scores in %).

the two strings of morpheme descriptions (not characters) and use that subsequence to estimate precision and recall. Recall is then the ratio of correctly tagged morpheme descriptions (the sum of the lengths of each LCS) and the total number of morpheme descriptions included in the reference set. Precision is the ratio of the same number of correctly tagged morpheme descriptions divided by the number of proposed morpheme descriptions in the automatically tagged data. To illustrate this on an example, in the second row of Table 3, the longest common subsequence ('DEM.OBL') has length 2, which is the same as the length of the predicted tag (giving precision = 100%) whereas the reference annotation has length 3 which gives recall=67% in that example.

As we can see in the Table 2, the tagger reaches quite reasonable performances in terms of precision and recall. The scores go down for more complex tags as we can see when leaving out the placeholder tags for lexical information (xx). Table 3 shows some examples of predicted interlinears and their references from the test set. The examples show that the model is capable of predicting quite complex descriptions. In many cases, the errors are rather minor and in many cases acceptable or just an artifact of the manual annotation as manual inspections reveal.

	ambiguous			
(scores in %)	unambiguous	(train)	(test+train)	unknown
precision	95.06	83.64	49.19	72.13
recall	95.44	83.50	49.72	66.27
accuracy	90.38	70.74	4.24	34.39

Table 4: Tagger performance for ambiguous and unknown words. 1207 unambiguous cases, 1209 unknown words, 3457 ambiguous cases (train) and 165 tokens that are unambiguous in training but have a different tag in the test data (test+train).

An important question is how well the system handles ambiguous and unknown words. The latter is important, in particular, to show the ability of classifying unseen items that may appear in new material that is collected. Table 4 lists the scores for different categories with respect to tagging ambiguity and overlap with the training data. Not surprisingly, words that are unambiguous obtain high scores in all metrics (above 90%). Still, some of the words are mistagged, which is due to the contextual dependencies in the CRF model. Even though glosses are very much standardized, more than half of the test tokens refer to ambiguously analyzed words. There are two categories of ambiguous words, the ones that have multiple interlinears in the training data and the ones that are unambiguous (but probably very infrequent) in training but have a different interlinear in the test set compared to the one in training. Naturally, the latter case is particularly difficult for the tagger to handle correctly as the only information available in training is without variation whereas the model is expected to produce a different result at test time. The problem can especially be seen in the accuracy score (=4.24%). Fortunately, this is a very infrequent case. Finally, we have unknown words and we can say that the model is well capable of analyzing those words with reasonable precision. Considering the data size and the complexity of the task this is a very

encouraging result.

Looking at the result above, one also wonders if the performance can still be improved. Especially the errors among unambiguous words are unsatisfactory and a simple solution would be to force the labels to follow the markup from the training data. In our final tagging experiments we therefore study the impact of these entries on the overall results. Table 5 summarizes the outcome of those tests.

(scores in %)	precision	recall	accuracy
unambiguous only	21.32 (9.69)	21.30 (13.35)	19.81 (12.49)
tag all with 'xx'	73.26 (n/a)	51.66 (n/a)	36.32 (n/a)
most frequent tag	69.63 (55.34)	70.01 (60.34)	61.80 (53.25)
tagger	82.68 (71.43)	81.51 (67.76)	65.50 (54.72)
tagger + unambiguous	83.65 (73.15)	82.38 (69.38)	67.29 (56.99)
tagger + most frequent	84.07 (73.07)	83.39 (71.74)	68.77 (59.60)

Table 5: Tagging unambiguous words and known words (in brackets without considering xx).

First of all, we can see that only tagging words for which we have unambiguous tags from training does not work well, which is to be expected with the large number of ambiguous items in the data. Using a baseline of attaching the most frequent interlinear (including xx) to each known word performs much better but is still not very satisfactory (especially when looking at precision and recall). Surprisingly, even precision drops quite significantly compared to the sequence labeling approach of the statistical tagger. However, enforcing tags of known words according to the training data on top of the statistical tagger leads to visible improvements. Not only fixing unambiguous words to the analyses from the training data but also the baseline approach of using the most frequent interlinears to replace the predictions of the tagger improve the overall results (especially in recall and accuracy). The final precision and recall values of above 84% and 83% are quite satisfactory.

5 Alignment and Annotation Transfer

The next step we would like to explore is annotation transfer through the English glosses to bootstrap a syntactic dependency parser. Cross-lingual parsing has become increasingly popular (Hwa et al., 2005; Tiedemann, 2014; Xiao and Guo, 2014) and has been suggested for handling low-resource languages for which no explicit training data is available. Annotation projection typically requires parallel data sets and word alignment to transfer information from one language to another. As discussed earlier, the glossed data sets nicely form a parallel corpus with English translations of the collected and transcribed Ingush utterances. Furthermore, we also have the interlinears that give additional information that will be useful in the alignment of running texts. Below, we outline our approach of combining links between interlinear glosses and English with statistical word alignment between Ingush and running English, which will be the basis of our initial annotation projection experiments.

5.1 Alignment Through Interlinear Glosses

Interlinear glosses provide rich information about the recorded data sets. First of all, they are nicely aligned with the tokens in the original input. Furthermore, they include lexical information in English that can be used to find corresponding parts in the smooth English translations that we will use to transfer syntactic annotation from.

The first step in our process consist of parsing the English translations to obtain lemmatized and syntactically analyzed sentences with annotation that we would like to project in the end. Here, we apply UDPipe (Straka et al., 2016) with its pre-trained model for processing English trained on the universal dependency treebanks version 1.2.[3] UDPipe includes all necessary steps from part-of-speech tagging to morphological analyses and dependency parsing, which makes it a convenient tool to apply in our pilot study.

[3] http://universaldependencies.org

In our preprocessing steps, we already separated lexical information from morphosyntactic information in the interlinears. We now use the lexical information to match lemmas in the parsed English translations. We use several heuristics to increase the number of words that can be matched:

- We split multi-word units (for which we joined components using underscores) to match individual parts with existing lemmas and wordforms in the parsed data. We also check alternatives that are given in some of the interlinears.
- We expand placeholder tokens (1sg, 2sg, etc) with English pronouns they usually refer to.
- We implement a prefix match to test whether the lexical information from the interlinear is completely contained in a lemma or wordform of the parsed English glosses or vice versa.
- We always use the link closest to the relative token position in the sentence in cases where there are alternative matches.

Via the position of the interlinears, we can now align English words with corresponding Ingush words. All non-matching words remain unaligned and we will treat them using automatic word alignment as explained below. In total, we obtain 38,665 links in the way described above out of the 110,167 tokens in the parallel training data (which is around 35.1% of the data).

5.2 Adding Statistical Word Alignment

A large portion of the data cannot be aligned using the string matching techniques described above. Therefore, we also run automatic word alignment using techniques that have been proposed in the field of statistical machine translation (SMT). In particular, we apply fast_align, an efficient implementation and reformulation of the IBM model 2 (Dyer et al., 2013). We run the alignment in both directions and apply symmetrisation heuristics (grow-diag-final-and) as commonly used in the SMT community. Finally, we merge automatic word alignments with the links obtained by matching interlinear glosses.

The symmetrized word alignment provides 83,863 links and, therefore, a much larger coverage of the data set. However, we expect that statistical alignment is of lower quality than matching manually created interlinears especially with the small data sets we have available. Therefore, we give preference to the interlinear links and only add alignments for words that have not been aligned otherwise. Using this strategy, we obtain a total of 80,974 linked words, about 73.5% of the complete data. Note that the statistical word alignment emphasizes coverage and, therefore, includes a lot of many-to-many links whereas the merged alignment focuses more on precision and, hence, contains a slightly smaller number of links.

5.3 Dependency Parsing of Aligned Translations

Finally, we can now use the aligned training data to transfer annotations from the parsed English glosses to the tokenized Ingush input. We use simple heuristics mapping universal part-of-speech labels and dependency relations based on a direct correspondence assumption. Figure 3 shows two examples of projected dependency structure using the annotation projection approach. We ignore all unaligned words and create a synthetic treebank that we can use to train a statistical parser. Several options are possible for training such a parser model and we opt for the *mate-tools* (Bohnet and Kuhn, 2012) that have been shown to produce highly accurate parsing models for a variety of languages.

Unfortunately, at this point we cannot say much about the quality of this initial parser as we do not have any gold standard available. In future work, we will manually check transfered annotation and parsing results and iteratively bootstrap a usable parser based on some kind of active learning schema.

6 Conclusions

This paper presents on-going work on creating NLP tools for a low-resource language, Ingush, using data from extensive linguistic fieldwork. We discuss the challenges of converting data sets to useful training resources in data-driven language technology and outline the benefits of the rich annotation

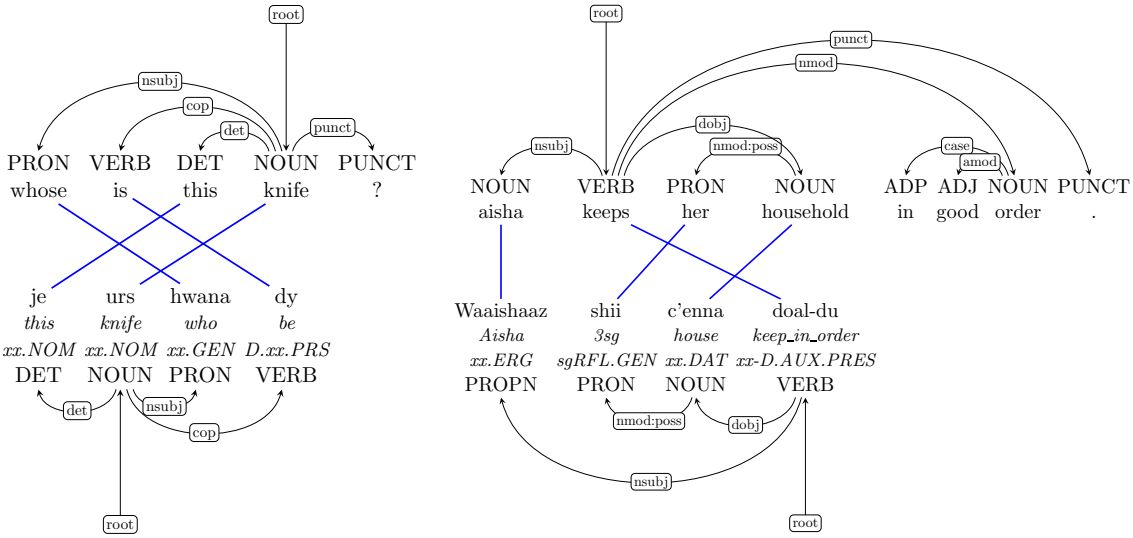

Figure 3: Aligned and projected dependency trees.

given by typological work. Interlinear glosses are very informative and capture important linguistic knowledge that can be facilitated in training automatic analyzers and syntactic parsers based on cross-lingual transfer models. We present an efficient tagger that can produce delexicalized interlinear glosses from raw transcriptions and we hope that these methods can develop into tools for linguistic field work and other applications for our selected target language or similar cases.

References

Bernd Bohnet and Jonas Kuhn. 2012. The Best of Both Worlds – A Graph-based Completion Model for Transition-based Parsers. In *Proceedings of EACL*, pages 77–87.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM Model 2. In *Proceedings of NAACL*, pages 644–648.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping Parsers via Syntactic Projection across Parallel Texts. *Natural Language Engineering*, 11(3):311–325.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic mod- els for segmenting and labeling sequence data. In *Proceedings of ICML*.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332, Seattle, Washington, USA, October. Association for Computational Linguistics.

Johanna Nichols. 2011. *Ingush Grammar*, volume 143 of *UC Publications in Linguistics*. Berkeley-Los Angeles: University of California Press.

Milan Straka, Jan Hajič, and Straková. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of LREC*.

Jörg Tiedemann. 2014. Rediscovering Annotation Projection for Cross-Lingual Parser Induction. In *Proceedings of COLING*, pages 1854–1864.

Min Xiao and Yuhong Guo. 2014. Distributed Word Representation Learning for Cross-Lingual Dependency Parsing. In *Proceedings of CoNLL*, pages 119–129.