

Morphological Data Generation from FLEx

Shengyu Liao
University of Florida

Beth Bryson
SIL International

Sarah Moeller
University of Florida

Abstract

Bootstrapping a neural morphological analyzer from artificial data is not often feasible for linguists who do not have specialized training. We show that grammatical information in FLEx, a software tool familiar to many linguists, can be used to generate word forms and their morphosyntactic descriptors. The results of a pilot experiment with Lezgi [lez] indicates that the grammatical information must be structured in FLEx with data generation in mind to generate usable training data. Adjustments with this goal in mind improved a neural morphological analyzer trained on the generated data, from .49 to .60 weighted average F_1 -score on the morphological descriptor labels.

1 Introduction

Significant data is needed to train a machine learning natural language processing (NLP) model but that training data is difficult to obtain for low-resource languages. Therefore, training data is sometimes synthesized from hand-crafted resources such as a finite state transducer (FST), using published grammatical descriptions. The additional synthetic data allows NLP models to generalize better to any data in the language.

Creating resources from which synthetic data can be generated usually requires special training. Linguists, who are most familiar with the grammar, may not possess the necessary skills. FSTs, for example, necessitate a rare combination of computer programming and linguistic knowledge. NLP training data for low-resource languages might be generated more equitably if such barriers were reduced. Ideally, linguists could provide the necessary grammatical information in a format that is both machine- and human-readable and they could enter the information in a familiar interface.

This paper describes a proof-of-concept for reducing barriers for non-computationally trained linguists. It will allow artificial training data to be

generated for low-resource languages with a tool already familiar to many linguists. As such, this paper asks two questions: 1) Can grammatical information already entered in *Fieldworks Language Explorer* (FLEx), a popular software for linguistic analysis and annotation, as part of a linguist’s workflow be used “as is” to generate training data for a morphological machine learning model? 2) Is the quality of data generated with the limited mechanisms of FLEx sufficient for training a usable morphological analyzer?

We show it is possible to synthesize inflected word forms and their morphosyntactic labels using the grammatical information provided in the *Grammar* and *Lexicon* tools of FLEx. We also show that to produce quality training data, the grammatical information must be entered with data generation in mind. Adjustments to a linguist’s approach to using FLEx can improve the quality of generated data.

2 Related Work

Morphology comprises word-building properties and accompanying morphosyntactic phenomena. Both NLP and general linguistics benefit from morphological analysis (Cotterell et al., 2015). Morphological analysis is foundational to deeper grammatical analysis and description. Parsing morphologically complex languages is important to NLP because the high number of inflected and compound words in morphologically complex languages compound the out-of-vocabulary (OOV) problem (Hammarström and Borin, 2011; Goldsmith et al., 2017). Addressing this problem affects downstream applications such as machine translation, voice recognition, and development of language learning apps.

When naturally occurring annotated language data is limited, generating synthesized or semi-synthesized word forms for training data has improved performance for tasks such as learning mor-

phology (Silfverberg et al., 2017) or word embeddings (Silva and Amarathunga, 2019). The data can be used by itself to train morphological analyzers (Moeller et al., 2018) or as an augmentation of natural data. For example, synthetic data augmentation has been used successfully in nearly all SIGMORPHON shared tasks (cf. Cotterell et al., 2016).

Synthesized data for morphology usually results in nonce words that have correctly inflected word-forms and accompanying morphosyntactic descriptive labels (e.g. affixes glosses). Synthesized morphological data can be generated from resources that incorporate the language’s morphological rules and grammatical information, such as found in spell checkers, dictionaries, Wiktionary tables, or Unimorph (Ahlberg et al., 2014; Bergmanis and Goldwater, 2019; Cotterell et al., 2015; Durrett and DeNero, 2013). FST-generated synthetic data has been used to train or augment neural morphological analyzers (Hämäläinen et al., 2021; Lane and Bird, 2019; Moeller et al., 2019; Schwartz et al., 2019; Silfverberg and Tyers, 2019). However, developing a robust FST model for a given language is time-consuming and requires in-depth knowledge of both the language and of finite state modeling tools (Maxwell, 2015a,b).

3 Morphology in FLEX Software

Field linguists use specially designed software to analyze data, annotate texts, build lexicons, and deduce the structure of a language. One such software is *Fieldworks Language Explorer* (FLEX)¹.

Users can specify morphological and lexical information in FLEX. The mechanisms in FLEX for describing the morphological patterns of a language include elements in the *Lexicon* area of its interface:

- Entries that specify the underlying form of a morpheme (roots/stems or affixes) (called *Lexeme Form*)
- Lists of each morpheme’s allomorphs
- Environments (either phonological or lexical) for allomorphs
- Morphological process rules (e.g., infixes, reduplication, metathesis)

¹<https://software.sil.org/fieldworks/>

mu-₁ N : ncl pfx 1	
Allomorphs	
Affix Allomorph	tkē mw
Morph Type	prefix
Environments	/_[V]
Affix Allomorph	tkē m
Morph Type	prefix
Environments	/_[N] /_[Lab]

Figure 1: A FLEX *Lexicon* entry showing the morpheme, its gloss, and its allomorphs with the phonological environments that constrain their realization.

- Specification of lexical category information for roots/stems and affixes

as well as items in the *Grammar*:

- Lists of phonemes, natural classes, inflection features, and environments
- Morphology position class charts specifying the relative position and complementary distribution of inflectional affixes

An example of a lexical entry with information about allomorphs and their environments in the FLEX *Lexicon* is shown in Figure 1. An example of a morphological template is shown in Figure 2.

Also, users can add morphological annotation to texts in the *Texts & Words* interface of FLEX. As shown in Figure 3, users may label words with their lexical category (part of speech or POS), segment words into morphemes, gloss the morphemes, and link those morphemes and glosses to the *Lexicon*.

3.1 Morphological Parser

In FLEX, a user may set up a rule- and lexicon-based morphological parser². The parser provides automated assistance by presenting all possible parses (segmentations and glosses) for words in texts based on the grammatical information specified in the *Lexicon* and *Grammar*. The parser cannot “learn” to parse unspecified morphological patterns, but can only process novel forms if all the word’s morphemes have an entry in the *Lexicon* and all relevant constraints are specified. The linguist can refine the lexical categories of morphemes and

²There are two automatic parsers in FLEX. For this experiment, the XAMPLE parser was used.

STEM	-(NUMBER)	-Oblique-erg	-SemCase	-(enclitic)
	-ap PL	-дц OBL	-э DAT	-ни and;FOCUS
			-н GEN	

Figure 2: An example FLEx morphological template for Lezgi [lez] nouns. The chart shows the type of affixes, their complementary distribution, and position in relation to the stem.

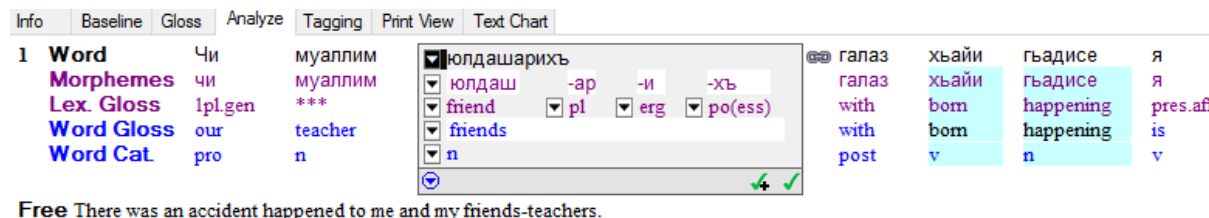


Figure 3: Interlinearized glossed texts for Lezgi [lez] in FLEx. Morpheme segment annotations are on the second line, morpheme glosses on the third, and word glosses on the fourth. POS tags are below the word glosses.

specify allomorph environments in the *Lexicon* or edit the position class morphology templates in the *Grammar*. These adjustments can constrain the parser to only offer valid suggestions.

3.2 Tools for Data Generation

With the help of a few additional tools, the grammatical information provided in FLEx to set up the morphological parser can also be used to generate synthetic morphological data. Currently, these tools produce synthesized word forms and their inflectional parses.

The additional tools needed for data generation are FlexTools, our custom Python modules, and STAMP (Weber et al., 1990). FLExTools³ is a software workbench that runs Python scripts that can directly read and manipulate the FLEx data store, thus achieving capability not built into FLEx itself. We enhanced some existing FlexTools modules⁴ to generate inflectional parses, using the information in FLEx. STAMP is a tool that can synthesize surface word forms from information produced by the parser in FLEx. Our package containing FlexTools, our custom modules, and STAMP is downloadable.⁵

4 Generating Synthetic Data from FLEx

The two modules we created for FlexTools are called *Generate all parses* and *Run STAMP*⁶. In

³<https://github.com/cdfarrow/flextools>

⁴We started with modules from the FLExTrans (Lockwood, 2015) package and customized them for our purposes.

⁵<https://github.com/shengyu-liao/Morphological-Data-Generation-from-FLEx>

⁶The *Run STAMP* module is based on one of the core modules of the FLExTrans package. It was modified only

FlexTools, the *Generate all parses* module accesses the FLEx data store and generates morphosyntactic parses for all combinations of affixes that the *Grammar* allows for all specified root morphemes in the *Lexicon*. These parses include the gloss of every affix taken from their lexical entries and the POS tag of the root morpheme:

$[baby] \langle N \rangle \{ PL \} \{ OBL \} \{ DAT \}$

In a later step, this parse output could be customized to include the information needed for various experiments. Our model is intended to analyze inflectional information, so the *Generate all parses* module was customized to also output a version with only the POS tag and affix glosses but no stem glosses, resulting in: $N; PL; OBL; DAT$.

The *Run STAMP* FLExTools module then builds the necessary input for synthesis, using the morphological information in FLEx and the morphosyntactic parses created by the *Generate all parses* module. It runs the background program STAMP (Weber et al., 1990) on these files, to generate inflected word forms. For instance, from $[baby] \langle N \rangle \{ PL \} \{ OBL \} \{ DAT \}$, STAMP generates the Lezgi word form: “балайриз” which corresponds to our customized inflectional tags: $N; PL; OBL; DAT$.

Currently, the files that are produced with the inflected words and morphosyntactic parses require additional processing.⁷ Several issues must be taken into consideration. First, obvious “ungram- slightly for this experiment. The *Generate all parses* module is also based on a module written by Lockwood that is not part of the FLExTrans package.

⁷Our post-processing code can be obtained in our GitHub repository linked in footnote 5.

Issue	Consequence	Solution
Template slot has no affixes	Generation script halted	Correct the template
Space in affix gloss	Gloss appears in word	Remove space in <i>Lexicon</i>
Stem/affix has no POS	Skipped in generation	Add POS in <i>Lexicon</i>
Space in POS <i>Abbreviation</i>	Gloss appears in word form	Remove space
Multiple morphemes in <i>Lexicon</i>	Both appear in word	Create separate entries in <i>Lexeme Form</i> field

Table 1: Issues that may be encountered when generating inflected word forms and parses from FLEx grammatical information, along with suggested solutions.

matical” words should be removed. These words are usually indicated by non-alphabetic characters. For example, when a synthetic surface word form fails some constraints in the synthesis process, the STAMP program inserts a “%” at the beginning of the generated string. When the stem has no gloss in the *Lexicon*, a “1.1” is added in the string. Other examples of non-alphabetic characters generated during the process and which should be eliminated include “;” (i.e. a comma was used in the *Lexicon* gloss) and “{” and “}” (i.e. the stem morpheme is missing a gloss in the *Lexicon*).

A second issue is that word forms must be “cleaned” by removing generated symbols that represent null affixes. This step cannot be generalized with a single Python script in the online package because the symbols may be specific to each database. For example, the symbol “*0” indicates null affixes in the Lezgi FLEx database. Others might use “∅” or “ø” (Unicode codepoint U+2205 and U+00F8, respectively).

A third issue that must be considered during the generation process and post-processing stage is whether uninflected words should be included. If all the affix slots in a template in the *Grammar* are optional, then the bare root will be generated as the word form but its parse will include only a POS tag. If these bare roots are not desired, the templates must be constructed so that at least one affix column is non-optional.

Finally, any FLEx database is likely to contain issues that need to be corrected or eliminated before the information can serve as quality training data. Some issues that we encountered, along with ideal solutions, are described in Table 1. Most can be avoided if users follow consistent conventions when entering information in the *Lexicon* and *Grammar*. For example, when multiple English words or grammatical tags are used to gloss a single morpheme, separating them by a period “.” in-

stead of a space (e.g. ‘1SG.PAST’ rather than ‘1SG PAST’) resolves the issue on the second line of the table.

5 Case Study: Lezgi

The Lezgi FLEx database that was used in our pilot experiment contained one morphological grammar template for nouns, a lexicon of root morphemes, and ten oral texts.⁸ The FLEx database was compiled and partially annotated as a basis for a master’s thesis in Linguistics.

Access to this data comes from linguists who gathered the data with informed consent for research purposes by the speaker community. One co-author (Moeller) acted as a consultant for Lezgi community members during a dictionary workshop and also trained a non-linguist native speaker to use FLEx and do some initial morphological segmentation and glossing.

The texts are transcribed in the language’s Cyrillic orthography. We adjusted the lexicon and grammar templates with data generation in mind. We also extracted word forms and glosses from the texts to serve as a test dataset for the neural morphological analyzer.

5.1 Lezgi Noun Morphology

Lezgi (Lezgian; ISO 639-3 code: lez; Glottolog code: lezg1247) is a Nakh-Daghestanian (North-east Caucasian) language spoken by over 600,000 speakers in Russia and Azerbaijan (Eberhard et al., 2022). The FLEx data is from the endangered Qusar dialect of Azerbaijan which differs from standard written Lezgi in a few notable ways. It borrowed a locative case morpheme from Azerbaijani which is used alongside the native inessive case morpheme with the same meaning. Also, an unusual stacking of the dative and directive cases

⁸The FLEx database will be deposited at SIL Language and Culture Archives.

occurs occasionally in the texts to indicate movement towards something.

Lezgi is a highly agglutinative language with overwhelmingly suffixing morphology. Fourteen noun cases are formed by case-stacking which is a unique characteristic of Nakh-Daghestanian languages. Instead of a unique morpheme for each case, case-stacking composes various case inflections by “stacking” sequences of case suffixes one after each other.

itim	SG.ABS ‘man’
itim-ar	PL.ABS ‘men’
itim-ar-di	PL-ERG ‘men’
itim-di-k	OBL-AD.ESS ‘near a man’
itim-di-k-di	OBL-AD-DIR ‘toward a man’
itim-ar-di-k-ay	PL-OBL-AD-EL ‘from men’

Table 2: A simplified example of Lezgi case-stacking on the noun root *itim* ‘man’.

A simplified example of Lezgi case-stacking is shown in Table 2. Absolutive (ABS) and essive (ESS) cases and singular number (SG) are marked by null morphemes. The plural suffix (PL) attaches directly to the noun stem. The ergative (ERG) and the oblique (OBL) suffixes attach after the number. The adessive case (AD.ESS) attaches to the oblique suffix. The elative (EL) and directive (DIR) cases are added in the fourth slot after the root. In the longest possible sequences (e.g. adelative and addirective cases), the final suffixes indicate directed-motion meaning. In the second-to-longest sequences (e.g. adessive case), locative meaning is indicated by one suffix after the oblique affix. The oblique affix is identical in form with the ergative morpheme but serves a purely grammatical function of forming the oblique stem onto which other cases may be “stacked”.

5.2 Test Data

To create a set of grammatically correct word forms and morphological descriptive labels for testing, 100 root morphemes were randomly chosen from the 541 noun roots in the *Lexicon*. The 441 roots were set aside to generate synthesized data. Using the FLEx concordance, words in the texts that were formed from these 100 roots were included in the test set. Any derived nouns in the texts that were not formed from either the 100 test noun roots or the 441 training noun roots were also included in the test set. See Table 3 for the final training and

test set sizes.

5.3 Modifying the Grammar and Lexicon

The original training data was generated from the grammatical information that had been entered by the linguist during their routine analysis and annotation work. Generating data from this “as-is” version produced over 300k training instances but presented two issues. First, allomorph environments had not been specified, resulting in a large proportion of invalid strings where root morphemes were combined with the first allomorph rather than the correct allomorph, as constrained by morphophonological rules. Second, the grammar template had been constructed as a comprehensive picture of Lezgi noun morphology and this resulted in many invalid morpheme co-occurrences.

In the *Lexicon*, constraints had to be added that specified the phonological environments of each allomorph listed in the lexical entries. Lexically determined allomorphs could sometimes be specified by creating a new lexical entry and ad hoc allomorphy rules. There are mechanisms in FLEx for constraining lexically determined allomorphy but the generation code we used did not yet implement those constraints⁹. During generation, allomorphs are tested in order, selecting the first one where the constraints are met. The “head” Lexeme Form functions as the “elsewhere” or default allomorph. Therefore, the allomorphs needed to be re-ordered to reflect a decreasing scope of specificity for the environments that trigger them.

The original morphology template lacked co-occurrence constraints needed to prevent overgenerated morpheme combinations. The linguist had rapidly set up the template for recognition of forms found in the text, not with the rigor needed to describe all and only the valid forms. The original single, comprehensive template was converted into multiple smaller templates that allow all possible morpheme combinations but no invalid combinations. For example, the original template had a column for the ergative and the oblique cases, followed by an optional column that included the adessive case. This generates an ungrammatical word form with both the ergative case and the adessive case. To correctly depict the correct forms in Table 2, two templates were needed. One template allows the ergative suffix but no additional

⁹This feature has now been implemented and is included in the modules available for download.

suffixes, and a second template has a non-optional column for the oblique affix followed by a non-optional column for the adessive case.

After adjusting the *Lezgi Lexicon* and *Grammar* with data generation in mind, the number of generated pairs of word forms and glosses was greatly reduced, as shown in Table 3, while the overall grammaticality of the data was increased. The adjusted dataset contains (a) a higher percentage of word forms that are phonologically valid and (b) fewer instances that have invalid co-occurrences of affixes.

6 Experiment

We tested the quality of data generation from FLE_x by training two neural morphological analyzers using only the word forms and morphosyntactic labels synthesized from the grammatical information specified in the FLE_x interfaces.

The morphological analysis task is treated as a problem of converting an input sequence of characters $\vec{x} = (x_1, \dots, x_n)$ to an output sequence of labels $\vec{y} = (y_1, \dots, y_n)$. The output sequence of labels are morphosyntactic descriptors indicating part-of-speech and inflectional information (i.e. affix glosses). An example input-output instance is shown below. We selected the Transformer (Vaswani et al., 2017), a stateless encoder-decoder model that uses additional attention layers to boost speed and performance, and implemented the *fairseq* (Ott et al., 2019) version with modifications that have been shown to be successful in low-resource character-level morphological tasks (Wu et al., 2021).¹⁰

INPUT (word): б а л а й р и з

OUTPUT (parse): N PL OBL DAT

We trained two models in order to compare the efficacy of adding morphophonological constraints to the *Lexicon* and adjusting the morphology templates in the FLE_x *Grammar*. One model is trained on the “as-is” grammatical information originally provided by the linguist. The second model is trained on the adjusted grammatical information. A 9/1 split of the generated data was used for training and development. The test data for both models is the same. The models were tested on naturally

¹⁰4 encoder-decoder layers, 4 self-attention heads, 256 embedding size, 1024 hidden size of feed-forward layer, layer normalization before self-attention, decoding left-to-right in a greedy fashion.

Version	Training	Test
original	216,000	2,125
adjusted	38,784	

Table 3: Total number of Lezgi training and test instances. Training data is generated data. Test data are naturally occurring words formed from roots that were not used to generate training data. The two rows show the training data generated before and after the information in the *Lexicon* and *Grammar* areas was adjusted with data generation in mind.

	Original	Adjusted
word acc.	6%	29%
weighted P	0.57	0.63
weighted R	0.49	0.63
weighted F ₁	0.49	0.60
macro P	0.13	0.34
macro R	0.11	0.46
macro F ₁	0.10	0.36

Table 4: Results comparing models trained only on original grammatical information and grammatical information adjusted for data generation. The accuracy reports the percentage of test words for which all glosses/labels were correctly predicted. The micro P(recision), R(ecall), and F₁ scores are the average scores weighted by the distribution of labels. Macro scores are unweighted.

occurring tokens that had been manually annotated with the morphosyntactic glosses/labels.

The performance of the two models is shown in Table 4. Performance is compared on two measures: per token and per label. The adjusted approach to preparing grammatical information in FLE_x improved all the results. The accuracy of per word analyses is a little less than 30%. The highest average F₁ score of .6 is weighted by the distribution of the gloss labels and indicates that the adjusted model did reasonably well recognizing inflectional affixes. The greatest improvement is seen in the macro scores which are not weighted by the number of instances of each label. The increase from .10 to .36 F₁ indicates that the biggest effect made by the adjustments and additional constraints in the *Grammar* and *Lexicon* was an improvement in recognizing the less common inflectional forms.

7 Discussion

The first question this paper put forth was whether grammatical information entered in FLE_x as part of a linguist’s workflow could be used “as is” to

generate synthetic training data for a morphological model. By any measure, the quality of our originally generated data is not sufficient to train a high performing neural morphological analyzer. However, the original grammatical information was provided by a linguist with no idea of applying NLP, but rather with the goal of implementing a morphological parser within FLEEx while expecting the parser to be constrained by forms found in naturally occurring data. In order to generate new wordforms, adjustments had to be made in the *Lexicon* and *Grammar* mechanisms that would constrain overgeneration. These adjustments improved the quality of the artificial data.

We found that building the morphology templates in the *Grammar* is not difficult for someone who is familiar with FLEEx, but it is important to understand, as the grammar and lexicon are built, how and why overgeneration should be constrained. We did discover, however, that knowing how and where to specify complicated allomorphy rules and lexically-determined constraints may require assistance even for linguists who use FLEEx on a regular basis. The FLEEx website has a long playlist of training videos but videos related to the parser setup are minimal. The parser is the main mechanism for generating synthetic training data.

The second question this paper put forth regards the quality of the generated data as measured by the performance level of the trained model. Once the information was adjusted, the resulting model has a weighted average F_1 score of .6. This is comparable to a threshold that “pre-annotation” by a morphological NLP model has been shown to improve human annotation (Felt, 2012). By that measure, FLEEx-generated data may be useful to train an NLP model useful for human assistance. For example, if a large amount of words need to be morphologically analyzed, it may save time and increase accuracy to pre-annotate the data with a model trained on FLEEx-generated data and then have humans correct the labels. This situation assumes that a small lexicon and a decent grammar sketch are available.

The main aim of this pilot experiment is to reduce barriers that discourage “paper-and-pen” linguists from being involved in NLP for low-resource languages. Specifying grammatical information in a familiar tool is one way linguists can help build resources needed to train NLP models. The results of this pilot project in FLEEx are promising but ad-

ditional work is needed. Further work is needed to develop the data generation mechanisms. We did quite a bit of programming to make any data synthesis possible, including modifying and enhancing existing FlexTools modules. There is a potential for independent programmers to contribute modules and improve the quality of the FLEEx-generated synthetic data.

Finally, there is an additional direct benefit to a linguist who attempts to generate synthetic data from FLEEx. The steps to create and improve FLEEx-generated training data also improve the software’s morphological parser. Even if the quality of the generated data for a given language is too poor to train a usable neural analyzer, users can benefit from an improved automatic parser that annotates new data in *Texts & Words*. The drawback to the rule-based parsers in FLEEx is that, unlike a machine learning model, they cannot “guess” the inflectional information of new forms. If a word’s root (or affix) morphemes are not already entered in the *Lexicon* by the user, the parser cannot do anything with that word form. The benefit of generating training data to improve a probabilistic model such as the one we tested, is that these models can provide reasonable guesses at inflectional analyses for word forms with previously unseen morphemes. Even these automated hypotheses may be beneficial when an annotator is faced with large numbers of new words.

8 Conclusion

Generating artificial training data based on knowledge of a language’s structure is one way to address the low-resource language challenge. In previous work, this has required special training for either NLP researchers or language experts. We have shown that it is possible to generate artificial data using FLEEx, a tool linguists are already familiar with. This means an NLP expert does not need to be a language domain expert and the linguist does not need specialized training. However, the quality of the synthetic data and, therefore, the usefulness of the trained model depends on how well the goal of data generation is understood by the FLEEx user. In the future, we would like to work more closely with the linguist or a native speaker to understand the patterns of errors in the generated data and determine what mechanisms either in FLEEx or the FlexTools modules could reduce them. Future work should expand the type of generated data

to include morpheme boundaries and train models that perform morpheme segmentation as well as labeling of inflectional information. Training morphological NLP models holds promise for language documentation and description to automate the interlinearization process and provide resources for developing language technology that could benefit the community such as language learning apps and machine translation.

Acknowledgements

We are grateful to the Lezgi linguist and speakers for sharing the Lezgi FLEx database, providing additional Lezgi annotations, and answering questions about the language. We are also grateful to other linguists who offered or shared databases in other languages: Aaron Broadwell, the Sena linguists, and especially Jeff Shrum. Thank you to Ron Lockwood for sharing FLExTrans modules and assisting with advice and testing. We would also like to thank the anonymous reviewers for their helpful comments and suggestions. The usual disclaimers apply.

References

- Malin Ahlberg, Markus Forsberg, and Mans Hulden. 2014. [Semi-supervised learning of morphological paradigms and lexicons](#). In *Proceedings of 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 569–578, Gothenburg, Sweden. Association for Computational Linguistics.
- Toms Bergmanis and Sharon Goldwater. 2019. [Data augmentation for context-sensitive neural lemmatization using inflection tables and raw text](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4119–4128, Minneapolis, MN. Association for Computational Linguistics.
- Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. [The SIGMORPHON 2016 shared task—Morphological reinflection](#). In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.
- Ryan Cotterell, Thomas Müller, Alexander Fraser, and Hinrich Schütze. 2015. [Labeled morphological segmentation with semi-Markov models](#). In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 164–174, Beijing, China. Association for Computational Linguistics.
- Greg Durrett and John DeNero. 2013. [Supervised learning of complete morphological paradigms](#). In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1185–1195, Atlanta, GA. Association for Computational Linguistics.
- David M. Eberhard, Gary F. Simons, and Charles D. Fennig, editors. 2022. *Ethnologue: Languages of the world*, 25 edition. SIL International, Dallas, TX.
- Paul Felt. 2012. [Improving the effectiveness of machine-assisted annotation](#). Master’s thesis, Brigham Young University, Provo, UT.
- John A. Goldsmith, Jackson L. Lee, and Aris Xanthos. 2017. [Computational learning of morphology](#). *Annual Review of Linguistics*, 3(1):85–106.
- Mika Härmäläinen, Niko Partanen, Jack Rueter, and Khalid Alnajjar. 2021. [Neural morphology dataset and models for multiple languages, from the large to the endangered](#). In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 166–177, Reykjavik, Iceland (Online). Linköping University Electronic Press, Linköping, Sweden.
- Harald Hammarström and Lars Borin. 2011. [Unsupervised learning of morphology](#). *Computational Linguistics*, 37(2):309–350.
- William Lane and Steven Bird. 2019. [Towards a robust morphological analyzer for Kunwinjku](#). In *Proceedings of the 17th Annual Workshop of the Australasian Language Technology Association*, pages 1–9, Sydney, Australia. Australasian Language Technology Association.
- Ronald M. Lockwood. 2015. [A linguist-friendly machine translation system for low-resource languages](#). Master’s thesis, University of Washington, Seattle, WA.
- Michael Maxwell. 2015a. [Accounting for allomorphy in finite-state transducers](#). In *Proceedings of the 12th International Conference on Finite-State Methods and Natural Language Processing 2015 (FSMNLP 2015 Düsseldorf)*. Association for Computational Linguistics.
- Michael Maxwell. 2015b. [Grammar debugging](#). In Cerstin Mahlow and Michael Piotrowski, editors, *Systems and frameworks for computational morphology. SFCM 2015.*, number 537 in Communications in Computer and Information Science, pages 166–183. Springer, Cham, Germany.
- Sarah Moeller, Ghazaleh Kazeminejad, Andrew Cowell, and Mans Hulden. 2018. [A neural morphological analyzer for Arapaho verbs learned from a finite](#)

- state transducer. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 12–20, Santa Fe, NM. Association for Computational Linguistics.
- Sarah Moeller, Ghazaleh Kazeminejad, Andrew Cowell, and Mans Hulden. 2019. [Improving low-resource morphological learning with intermediate forms from finite state transducers](#). In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, pages 81–86, Honolulu, HI. Association for Computational Linguistics.
- Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. [fairseq: A fast, extensible toolkit for sequence modeling](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, MN. Association for Computational Linguistics.
- Lane Schwartz, Emily Chen, Benjamin Hunt, and Sylvia L.R. Schreiner. 2019. [Bootstrapping a neural morphological analyzer for St. Lawrence Island Yupik from a finite-state transducer](#). In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, pages 87–96, Honolulu, HI. Association for Computational Linguistics.
- Miikka Silfverberg and Francis Tyers. 2019. [Data-driven morphological analysis for Uralic languages](#). In *Proceedings of the Fifth International Workshop on Computational Linguistics for Uralic Languages*, pages 1–14, Tartu, Estonia. Association for Computational Linguistics.
- Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. [Data augmentation for morphological reinflection](#). In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver, Canada. Association for Computational Linguistics.
- Amila Silva and Chathurika Amarathunga. 2019. [On learning word embeddings from linguistically augmented text corpora](#). In *Proceedings of the 13th International Conference on Computational Semantics - Short Papers*, pages 52–58, Gothenburg, Sweden. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS 2017)*, pages 5998–6008, Long Beach, CA. Curran Associates Inc., Red Hook, NY.
- David J. Weber, H. Andrew Black, Stephen R. McConnell, and Alan Buseman. 1990. *STAMP: A tool for dialect adaptation*. Number 15 in Occasional Publications in Academic Computing (OPAC). Summer Institute of Linguistics, Dallas, TX.
- Shijie Wu, Ryan Cotterell, and Mans Hulden. 2021. [Applying the transformer to character-level transduction](#). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1901–1907, Online. Association for Computational Linguistics.