Real-World Semi-Supervised Learning of POS-Taggers for Low-Resource Languages

Dan Garrette[1] **Jason Mielens**[2] **Jason Baldridge**[2]

[1]Department of Computer Science
The University of Texas at Austin
dhg@cs.utexas.edu

[2]Department of Linguistics
The University of Texas at Austin
{jmielens,jbaldrid}@utexas.edu

Abstract

Developing natural language processing tools for low-resource languages often requires creating resources from scratch. While a variety of semi-supervised methods exist for training from incomplete data, there are open questions regarding what types of training data should be used and how much is necessary. We discuss a series of experiments designed to shed light on such questions in the context of part-of-speech tagging. We obtain timed annotations from linguists for the low-resource languages Kinyarwanda and Malagasy (as well as English) and evaluate how the amounts of various kinds of data affect performance of a trained POS-tagger. Our results show that annotation of word types is the most important, provided a sufficiently capable semi-supervised learning infrastructure is in place to project type information onto a raw corpus. We also show that finite-state morphological analyzers are effective sources of type information when few labeled examples are available.

1 Introduction

Low-resource languages present a particularly difficult challenge for natural language processing tasks. For example, supervised learning methods can provide high accuracy for part-of-speech (POS) tagging (Manning, 2011), but they perform poorly when little supervision is available. Good results in weakly-supervised tagging have been obtained by training sequence models such as hidden Markov models (HMM) using the Expectation-Maximization algorithm (EM), however most work in this area has still relied on relatively large amounts of data, both annotated and unannotated, as well as an assumption that the annotations are very clean (Kupiec, 1992; Merialdo, 1994).

The ability to learn taggers using very little data is enticing: only a tiny fraction of the world's languages have enough data for standard supervised models to work well. The collection or development of resources is a time-consuming and expensive process, creating a significant barrier for an under-studied language where there are few experts and little funding. It is thus important to develop approaches that achieve good accuracy based on the amount of data that can be reasonably obtained, for example, in just a few hours by a linguist doing fieldwork on a non-native language.

Previous work explored learning taggers from weak information, but the type, amount, quality, and sources of data raise questions about the applicability of those results to real-world low-resource scenarios (Toutanova and Johnson, 2008; Ravi and Knight, 2009; Hasan and Ng, 2009; Garrette and Baldridge, 2012). Most research simulated weak supervision with tag dictionaries extracted from existing large, expertly-annotated corpora. These resources have been developed over long periods of time by trained annotators who collaborate to produce high-quality analyses. They are also biased towards including only the most likely tag for each word type, resulting in a cleaner dictionary than one would find in a real scenario. As such, these experiments do not reflect real-world constraints.

One exception to this work is Goldberg et al. (2008): they use a manually-constructed lexicon for Hebrew in order to learn an HMM tagger. However, this lexicon was constructed by trained lexicographers over a long period of time and achieves very high coverage of the language with very good quality, much better than could be achieved by our non-expert linguistics graduate student annotators in just a few hours. Cucerzan and Yarowsky

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 583–592,
Sofia, Bulgaria, August 4-9 2013.

(2002) learn a POS-tagger from existing linguistic resources, namely a dictionary and a reference grammar, but these resources are not available, much less digitized, for most under-studied languages. Haghighi and Klein (2006) develop a model in which a POS-tagger is learned from a list of POS tags and just three "prototype" word types for each tag, but their approach requires a vector space to compute the distributional similarity between prototypes and other word types in the corpus. Such distributional models are not feasible for low-resource languages because they require immense amounts of raw text, much more than is available in these settings (Abney and Bird, 2010). Further, they extracted their prototype lists directly from a labeled corpus, something we are specifically avoiding. Täckström et al. (2013) evaluate the use of mixed type and token constraints generated by projecting information from a high-resource language to a low-resource language via a parallel corpus. However, large parallel corpora are not available for most low-resource languages. These are also expensive resources to create and would take considerably more effort to produce than the monolingual resources that our annotators were able to generate in a four-hour timeframe. Of course, if they are available, such parallel text links could be incorporated into our approach.

In our previous work, we developed a different strategy based on generalizing linguistic input with a computational model: linguists annotated either types or tokens for two hours, these annotations are projected onto a corpus of unlabeled tokens using label propagation and HMMs, and a final POS-tagger is trained on this larger auto-labeled corpus (Garrette and Baldridge, 2013). That approach uses much more realistic types and quantities of resources than previous work; nonetheless, it leaves many open questions regarding the effectiveness of incrementally more annotation, the role of unannotated data, and whether there is a good balance to be found using a *combination* of type- and token-supervision. We also did not consider morphological analyzers as a form of type supervision, as suggested by Merialdo (1994).

This paper addresses these questions via a series of experiments designed to quantify the effect on performance given by the amount of time spent finding or annotating training materials. We specifically look at the impact of four types of data collection:

1. Time annotating sentences (token supervision)
2. Time creating tag dictionary (type supervision)
3. Time constructing a finite state transducer (FST) to analyze word-type morphology
4. Amount of raw data available for training

We explore these strategies in the context of POS-tagging for Kinyarwanda and Malagasy. We also include experiments for English, pretending as though it is a low-resource language. The overwhelming take away from our results is that type supervision—when backed by an effective semi-supervised learning approach—is the most important source of linguistic information. Also, morphological analyzers help for morphologically rich languages when there are few labeled types or tokens (and, it never hurts to use them). Finally, performance improves with more raw data, though we see diminishing returns past 400,000 tokens. With just four hours of type annotation, our system obtains good accuracy across the three languages: 89.8% on English, 81.9% on Kinyarwanda, and 81.2% on Malagasy.

Our results compare favorably with previous work despite using considerably less supervision and a more difficult set of tags. For example, Li et al. (2012) use the entirety of English Wiktionary directly as a tag dictionary to obtain 87.1% accuracy on English, below our result. Täckström et al. (2013) average 88.8% across 8 major languages, but for Turkish, a morphologically rich language, they achieve only 65.2%, significantly below our 81.9% for morphologically-rich Kinyarwanda.

2 Data

Kinyarwanda (KIN) and Malagasy (MLG) are low-resource, KIN is morphologically rich, and English (ENG) is used for comparison. For each language, sentences were divided into four sets: training data to be labeled by annotators, raw training data, development data, and test data.

Data sources The KIN texts are transcripts of testimonies by survivors of the Rwandan genocide provided by the Kigali Genocide Memorial Center. The MLG texts are articles from the websites[1] *Lakroa* and *La Gazette* and Malagasy Global Voices.[2] Texts in both KIN and MLG were tok-

[1] www.lakroa.mg and www.lagazette-dgi.com

[2] mg.globalvoicesonline.org/

	KIN		MLG		ENG - Experienced		ENG - Novice	
time	type	token	type	token	type	token	type	token
1:00	801	559 (1093)	660	422 (899)	910	522 (1124)	210	308 (599)
2:00	1814	948 (2093)	1363	785 (1923)	2660	1036 (2375)	631	646 (1429)
3:00	2539	1324 (3176)	2043	1082 (3064)	4561	1314 (3222)	1350	953 (2178)
4:00	3682	1651 (4119)	2773	1378 (4227)	6598	1697 (4376)	2185	1220 (2933)

Table 1: Annotations for each language and annotator as time increases. Shows the number of tag dictionary entries from type annotation vs. token. (The count of labeled tokens is shown in parentheses). For brevity, the table only shows hourly progress.

enized and labeled with POS tags by two linguistics graduate students, each of which was studying one of the languages. The KIN and MLG data have 12 and 23 distinct POS tags, respectively.

The Penn Treebank (PTB) (Marcus et al., 1993) is used as ENG data. Section 01 was used for token-supervised annotation, sections 02-14 were used as raw data, 15-18 for development of the FST, 19-21 as a dev set and 22-24 as a test set. The PTB uses 45 distinct POS tags.

Collecting annotations Linguists with non-native knowledge of KIN and MLG produced annotations for four hours (in 30-minute intervals) for two tasks. In the first task, type-supervision, the annotator was given a list of the words in the target language (ranked from most to least frequent), and they annotated each word type with its potential POS tags. The word types and frequencies used for this task were taken from the raw training data and did not include the test sets. In the second task, token-supervision, full sentences were annotated with POS tags. The 30-minute intervals allow us to investigate the incremental benefit of additional annotation of each type as well as how both annotation types might be combined within a fixed annotation budget.

Baldridge and Palmer (2009) found that annotator expertise greatly influences effectiveness of active learning for morpheme glossing, a related task. To see how differences in annotator speed and quality impact our task, we obtained ENG data from an *experienced* annotator and a *novice* one.

Ngai and Yarowsky (2000) investigated the effectiveness of rule-writing versus annotation (using active learning) for chunking, and found the latter to be far more effective. While we do not explore a rule-writing approach to POS-tagging, we do consider the impact of rule-based morphological analyzers as a component in our semi-supervised POS-tagging system.

	ENG - Exp.		ENG - Nov.	
time	type	tok	type	tok
1:00	0.05	0.03	0.01	0.02
2:00	0.15	0.05	0.03	0.03
3:00	0.24	0.06	0.07	0.05
4:00	0.32	0.08	0.11	0.06

Table 2: Tag dictionary recall against the test set for ENG annotators on type and token annotations.

Annotations Table 1 gives statistics for all languages and annotators showing progress during the 4-hour tasks. With token-annotation, tag dictionary growth slows because high-frequency words are repeatedly annotated, producing only additional frequency and sequence information. In contrast, every type-annotation label is a new tag dictionary entry. For types, growth increases over time, reflecting the fact that high-frequency words (which are addressed first) tend to be more ambiguous and thus require more careful thought than later words. For ENG, we can compare the tagging speed of the experienced annotator with the novice: 50% more tokens and 3 times as many types. The token-tagging speed stayed fairly constant for the experienced annotator, but the novice increased his rate, showing the result of practice.

Checking the annotators' output against the gold tags in the PTB shows that both had good tagging accuracy on tokens: 94-95%. Comparing the tag dictionary entries versus the test data, precision starts in the high 80%s and falls to to the mid-70%s in all cases. However, the differences in recall, shown in Table 2, are more interesting. On types, the experienced annotator maxed out at 32%, but the novice only reaches 11%. Moreover, the maximum for token annotations is much lower due to high repeat-annotation. The discrepancies between experienced and novice, and between type and token recall explain a great deal of the performance disparity seen in the experiments.

3 Morphological Transducers

Finite-state transducers (FSTs) accept regular languages and can be constructed easily using regular expressions, which makes them quite useful for phonology, morphology and limited areas of syntax (Karttunen, 2001). Past work has used FSTs for direct POS-tagging (Roche and Schabes, 1995), but this requires tight coupling between the FST and target tagset. We use FSTs for morphological analysis: the FST accepts a word type and produces a set of morphological features. If there are multiple possible analyses for a given word type, the FST returns them all. For instance the Kinyarwanda verb *sibatarazuka* "he is not yet resurrected" is analyzed in several ways:

- *+NEG+CL2+1PL+V+arazuk+IMP*
- *+NEG+CL2+NOT.YET+PRES+zuk+IMP*
- *+NEG+CL2+NOT.YET+razuk+IMP*

FSTs are particularly valuable for their ability to analyze out-of-vocabulary items. By looking for known affixes, FSTs can guess the stem of a word and produce an analysis despite not having knowledge of that stem. For morphologically complex languages like KIN, this ability is especially useful. Other factors, such as a large number of morphologically-conditioned phonological changes (seen in MLG) make out-of-vocabulary guessing more challenging because of the large number of potential stems (high ambiguity).

Development of the FSTs for all three languages was done by iteratively adding rules and lexical items with the goal of increasing coverage on a raw dataset. To accomplish this on a fixed time budget, the most frequently occurring unanalyzed tokens were examined, and their stems plus any observable morphological or phonological patterns were added to the transducer. Additionally, developers searched for known morphological alternations to locate instances of phonological change for inclusion. Coverage was checked against a raw dataset which did not include the test data used for the POS experiments.

The KIN and MLG FSTs were created by English-speaking linguists who were familiar with their respective language. They also used dictionaries and grammars. Each FST was developed in 10 hours. To evaluate the benefits of more development time, a version of the English FST was saved every 30 minutes, as shown in Table 3.

elapsed	tokens		types	
time	count	pct	count	pct
2:00	130k	61%	2.1k	12%
4:00	159k	75%	4.1k	24%
6:00	170k	80%	6.7k	39%
8:00	182k	86%	7.7k	44%
10:00	192k	91%	10.7k	62%

Table 3: Coverage of the English morphological FST during development. For brevity, showing 2-hour increments instead of 30-minute segments.

	tokens		types	
	cov.	ambig.	cov.	ambig.
KIN	86%	2.62	82%	5.31
MLG	78%	2.98	37%	1.13
ENG	91%	1.19	62%	1.97

Table 4: Coverage and ambiguity of the final FST for each language.

4 Approach

Learning under low-resource conditions is more difficult than scenarios in most previous POS work because the vast majority of the word types in the training and test data are not covered by the annotations. When most words are unknown, learning algorithms such as EM struggle (Garrette and Baldridge, 2012). Recall that most work on learning POS-taggers from tag dictionaries used tag dictionaries culled from *test* sets (even when considering incomplete dictionaries). We thus build on our previous approach, which exploits extremely sparse, human-generated annotations that are produced without knowledge of which words appear in the test set (Garrette and Baldridge, 2013).

This approach generalizes a small initial tag dictionary to include unannotated word types appearing in raw data. It estimates word/tag pair and tag-transition frequency information using model-minimization, which also reduces noise introduced by automatic tag dictionary expansion. The approach exploits type annotations effectively to learn parameters for out-of-vocabulary words and infer missing frequency and sequence information. This pipeline is described in detail in the previous work, so we give only a brief overview and describe our additions.

The purpose of tag dictionary expansion is to estimate label distributions for tokens in a raw corpus, including words missing in the annotations. For this, a graph connecting annotated words to unannotated words via features is constructed and POS labels are pushed between these items using label propagation (LP) (Talukdar and Crammer, 2009). LP has been used successfully for extending POS labels from high-resource languages to low via parallel corpora (Das and Petrov, 2011; Täckström et al., 2013; Ding, 2011) or high- to low-resource domains (Subramanya et al., 2010), among other tasks. These works have typically used n-gram features (capturing basic syntax) and character affixes (basic morphology).

The character n-gram affix-as-morphology approach produces many features, but only a fraction of them represent actual morphemes. Incorrect features end up pushing noise around the graph, so affixes can lead to more false labels that drown out the true labels. While affixes may be sufficient for languages with limited morphology, their effectiveness diminishes for morphology-rich languages, which have much higher type-to-token ratios. More types means sparser word frequency statistics and more out-of-vocabulary items, and thus problems for EM. Here, we modify the LP graph by supplementing or replacing generic affix features with a focused set of morphological features produced by an FST. These targeted morphological features are effective during LP because words that share them are much more likely to actually share POS tags.

FSTs produce multiple analyses, which is actually advantageous for LP. Ambiguities need not be resolved since we just take the union of all morphological features for all analyses and use them as features in the graph. Note that each FST produces its own POS-tags as features, but these do *not* correspond to the target POS tagset used by the tagger. This is important because it decouples FST development and the final POS task. Thus, any FST for the language, regardless of its provenance, can be used with any target POS tagset.

Since the LP graph contains a node for each corpus token, and each node is labeled with a distribution over POS tags, the graph provides a corpus of sentences labeled with noisy tag *distributions* along with an expanded tag dictionary. This output is useful as input to EM because it contains labels for all seen word types as well as sequence and frequency information. There is a high degree of noise in the LP output, so we employ the model minimization strategy of Ravi et al. (2010), which finds a minimal set of tag bigrams needed to explain the sentences in the raw corpus. It outputs a corpus of tagged sentences, which are used as a good starting point for EM training of an HMM. The expanded tag dictionary constrains the EM search space by providing a limited tagset for each word type, steering EM towards a desirable result.

Because the HMM trained by EM will contain zero-probabilities for words that did not appear in the training corpus, we use the "auto-supervision" step from our previous work: a Maximum Entropy Markov Model tagger is trained on a corpus that is noisily labeled by the HMM (Garrette and Baldridge, 2012). While training an HMM before the MEMM is not strictly necessary, our tests have shown that this generative-then-discriminative combination generally results in around 3% accuracy improvement.

5 Experiments[3]

To better understand the effect that each type of supervision has on tagger accuracy, we perform a series of experiments, with KIN and MLG as true low-resource languages. English experiments, for which we had both experienced and novice annotators, allow for further exploration into issues concerning data collection and preparation.

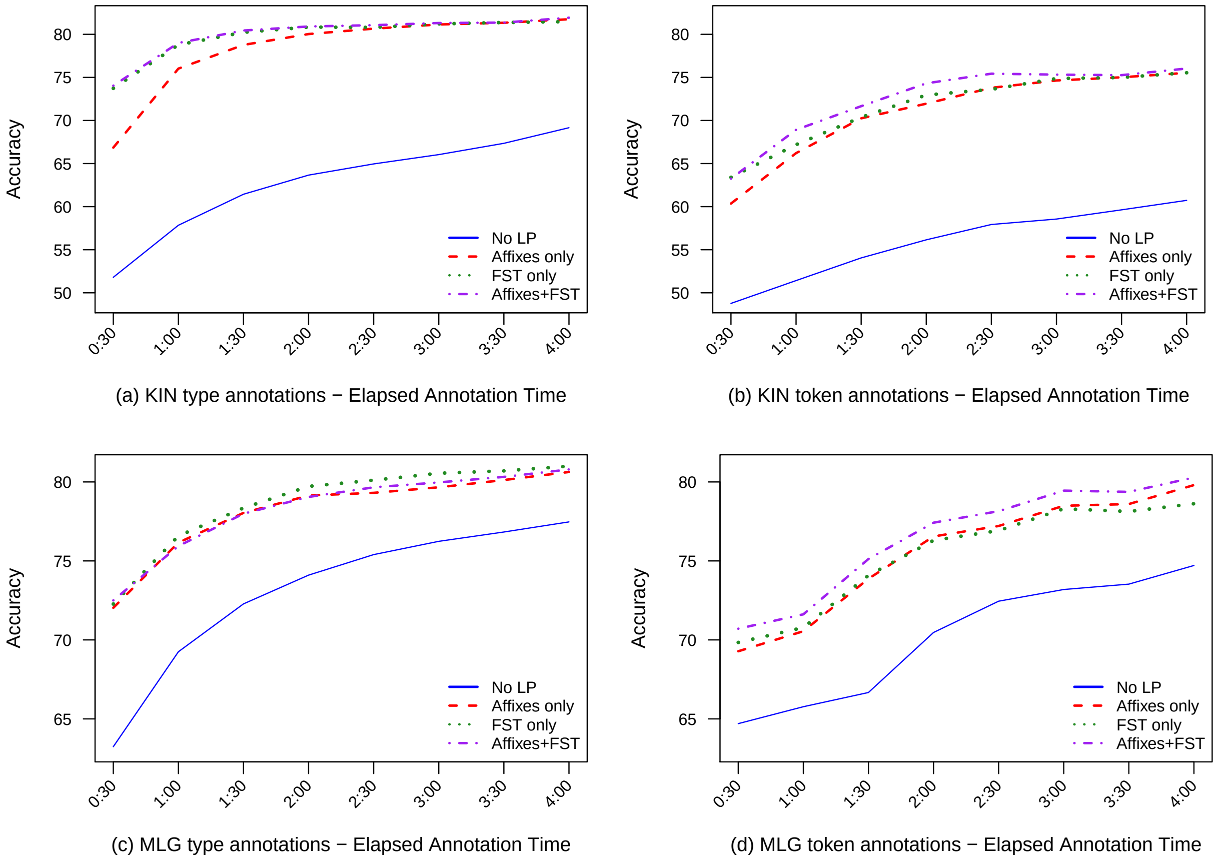
The overall best accuracies achieved by language are 81.9% for KIN using all types, 81.2% for MLG using half types and half tokens, and 89.8% for ENG using all types and the maximal amount of raw data. All of these best values were achieved using both FST and affix LP features.

All results described in this section are averaged over five folds of raw data.

5.1 Types versus tokens

Our primary question was the relationship between annotation type and time. Annotation must be done by someone familiar with the target language, linguistics, and the target POS tagset. For many low-resource languages, such people, and the time they have to spend, are likely to be in short supply. To make the best use of their time, we need to know which annotations are most use-

[3] Code and all MLG data available at `github.com/dhgarrette/low-resource-pos-tagging-2013` We are unable to provide the KIN or ENG data for download due to licensing restrictions. However, ENG data may be shared with those holding a license for the Penn Treebank and KIN data may be shared on a case-by-case basis.



(a) KIN type annotations − Elapsed Annotation Time

(b) KIN token annotations − Elapsed Annotation Time

(c) MLG type annotations − Elapsed Annotation Time

(d) MLG token annotations − Elapsed Annotation Time

Figure 1: Annotation time vs. tagger accuracy for type-only and token-only annotations.

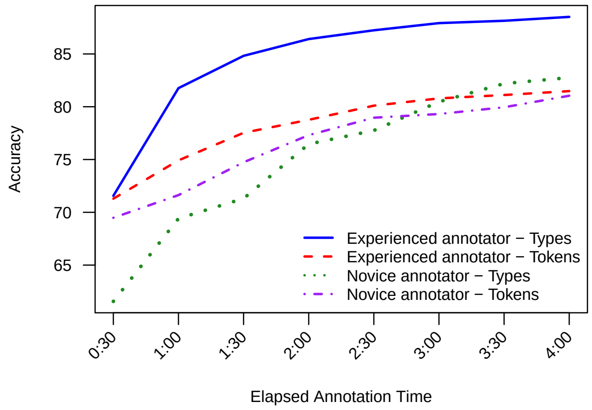


Figure 2: Annotation time vs. tagger accuracy for ENG type-only and token-only annotations with affix and FST LP features.

ful so that efforts can be concentrated there. Additionally, it is useful to identify when returns on annotation effort diminish so that annotators do not spend time doing work that is unlikely to add much value.

The annotators produced four hours each of type and token annotations, each in 30-minute increments. To assess the effects of annotation time, we trained taggers cumulatively on each increment and determine the value of each additional half-hour of effort. Results are shown for KIN and MLG in Figure 1 and ENG in Figure 2. In all scenarios, the use of LP (and model minimization) delivers huge performance gains. Additionally, the use of FST features, usually along with affixes, yielded better results than without. This indicates the LP procedure makes effective use of the morphological features produced by the FST and that the affix features are able to capture missing information without adding too much noise to the LP graph.

Furthermore, performance is considerably better when type annotations are used than only tokens. Type annotations plateau much faster, so a shorter amount of time must be spent annotating types than if token annotations are used. For KIN it takes approximately 1.5 hours to reach near-maximum accuracy for types, but 2.5 hours for tokens. This difference is due to the fact that the type annotations started with the most frequent words whereas the token annotations were on random sentences. Thus, type annotations quickly cover a significant portion of the language's tokens. With annotations directly on tokens, some of the highest

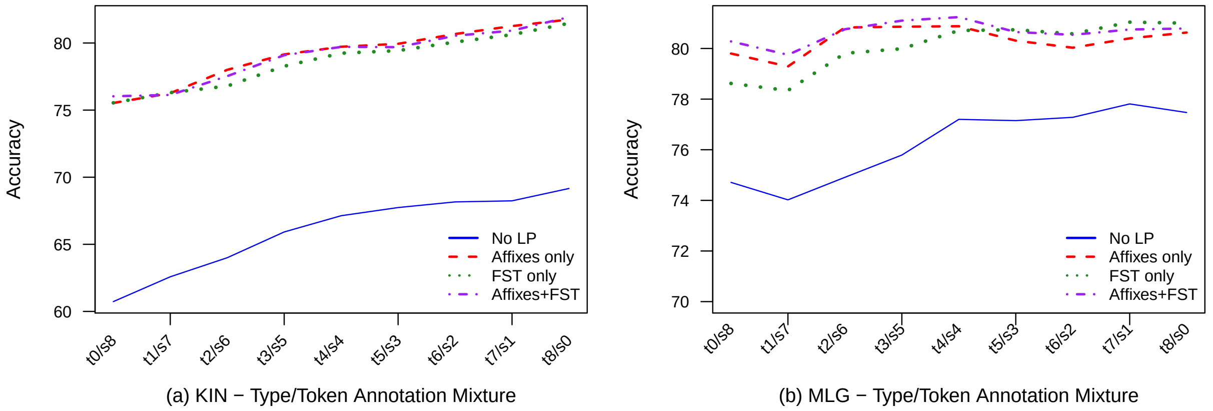


Figure 3: Annotation mixture vs. tagger accuracy. X-axis labels give annotation proportions, e.g. "t2/s6" indicates 2/8 of the time (1 hour) was spent annotating types and 6/8 (3 hours), full sentences.

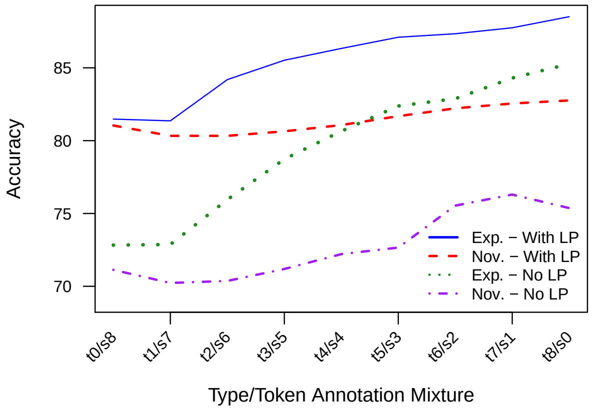


Figure 4: Annotation mixture vs. tagger accuracy on ENG using affix and FST LP features for experienced (Exp.) and novice (Nov.) annotators.

frequency types are covered, but annotation time is also ineffectively used on low-frequency types that happen to appear in those sentences.

Finally, the use of FST features yields the largest gains for KIN, but only when small amounts of annotation are available. This makes sense: KIN is a morphologically rich language, so sparsity is greater and crude affixes capture less actual morphology. With little annotated data, LP relies heavily on morphological features to make clean links between words. But, with more annotations, the gains of the FST over affix features alone diminishes: the affix features eventually capture enough of the morphology to make up the difference.

Figure 2 shows the dramatic differences between the experienced and novice ENG annotators.[4] For the former, results using types and tokens were similar after 30 minutes, but type annotations proved much more useful beyond that. In contrast, the novice annotated types much more slowly, so early on there were not enough annotated types for the training to be as effective. Even so, after three hours of annotation, type annotations still win with the novice, and even beat the experienced annotator labeling tokens.

5.2 Mixing type and token annotations

Because type and token annotations are each better at providing different information — a tag dictionary of high-frequency words vs. sequence and frequency information — it is reasonable to expect that a combination of the two might yield higher performance by each contributing different but complementary information during training. This matters in low-resource settings because type or token annotations will likely be produced by the same people, so there is a tradeoff between spending resources on one form of annotation over the other. Understanding the best mixture of annotations can inform us on how to maximize the benefit of a set annotation budget. To this end, we ran experiments fixing the annotation time to four hours while varying the mix of type and token annotations. Results are shown for KIN and MLG in Figure 3 and ENG in Figure 4.

For KIN and ENG, tagger accuracy increases as the proportion of type annotations increases for all LP feature configurations. For MLG, however, as the reliance on the FST increases, the optimal mixture shifts toward higher type proportions. When only affix features are used, the optimal mixture is 1 hour of types and 3 hours of tokens. When FST and affix features are used, the optimum is 2 hours

[4] The ENG graph omits "No LP" results since they followed patterns similar to KIN and MLG. Additionally, the results without FST features are not shown because they were nearly identical (though slightly lower) than with the FST.

each of types and tokens. When only FST features are used, it is best to use 3.5 hours of types and only 30 minutes of tokens. Because the FST operates on word types, it is effective at exploiting type annotations. Thus, when the LP focuses more on FST features, it becomes more desirable to have larger amounts of type annotations.

Types clearly win for ENG. The experienced annotator was much faster at annotating types and the speed difference was less pronounced for tokens, so accuracy is most similar when only token annotations are used. The performance disparity grows with increasing the type proportion.

Täckström et al. (2013) explore the use of mixed type and token annotations in which a tagger is learned by projecting information via parallel text. In their experiments, they—like us—found that type information is more valuable than token information. However, they were able to see gains through the complementary effects of mixing type and token annotations. It is likely that this difference in our results is due to the amount of annotated data used. It seems that the amount of type information collected in four hours is not sufficient to saturate the system, meaning that switching to annotating tokens tends to hurt performance.

5.3 FST development

The third set of experiments evaluate how the amount of time spent developing an FST affects the performance of trained tagger. To do this, we had our ENG FST developer save progress after each hour (for ten hours). The results show that, for ENG, the FST provided no value, regardless of how much time was spent on its development. Moreover, since large gains in accuracy can be achieved by spending a small amount of time just annotating word types with POS tags, we are led to conclude that time should be spent annotating types or tokens instead of developing an FST. While it is likely that FST development time would have a greater impact for morphologically rich languages, we suspect that greater gains can still be obtained by instead annotating types. Nonetheless, FSTs never seems to *hurt* performance, so if one is readily available, it should be used.

5.4 The effect of more raw data

In addition to annotations, semi-supervised tagger training requires a corpus of raw text. Raw data can be easier to acquire since it does not need the attention of a linguist. Even so, for many low-resource languages, the amount of digitized text, such as transcripts or websites, is very limited and may, in fact, require substantial effort to accumulate, even with assistance from computational tools (Bird, 2011). Therefore, the collection of raw data can be considered another time-sensitive task for which the tradeoffs with previously-discussed annotation efforts must contend.

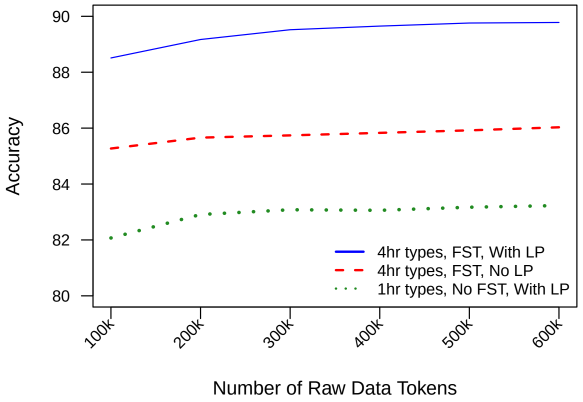


Figure 5: Amount of raw data vs. tagger accuracy for ENG using high vs. low amounts of annotation and using LP vs. no LP., for experienced annotator (novice results were similar).

It could be the case that more raw data for training could make up for additional annotation and FST development effort or make the LP procedure unnecessary. Figure 5 shows that that increased raw data does provide increasing gains, but they diminish after 200k tokens. The best performance is achieved by using more annotation and LP. Most importantly, however, removing either annotations or LP results in a significant decline in accuracy, such that even with 600k training tokens, we are unable to achieve the results of high annotation and LP using only 100k tokens.

5.5 Correcting existing annotations

For all of the ENG experiments, we also ran "oracle" experiments using gold tags for the same sentences or a tag dictionary containing the same number of type/tag entries as the annotator produced, but containing only the most frequent entries as determined by the gold-labeled corpus. Using this simulated "perfect annotator" data shows we lose accuracy due to annotator mistakes: for our experienced annotator and maximal FST, using 4 hours of types the oracle accuracy is 90.5 vs. 88.5 while using only tokens we see 83.9 vs.

81.5. This indicates that there are gains to be made by correcting mistakes in the annotations. This is true even after the point of diminishing returns on the learning curve, meaning that even when adding *more* annotations no longer improves performance, progress can still be made by correcting errors, so it may be reasonable to ask annotators to attempt to correct errors in their past annotations. Automated techniques for facilitating error identification can be employed for this (Dickinson and Meurers, 2003).

6 Conclusions and Future Work

Care must be taken when drawing conclusions from small-scale annotation studies such as those presented in this paper. Nonetheless, we have explored realistic annotation scenarios for POS-tagging for low-resource languages and found several consistent patterns. Most importantly, it is clear that type annotations are the most useful input one can obtain from a linguist—provided a semi-supervised algorithm for projecting that information reliably onto raw tokens is available. In a sense, this result validates the research trajectory of efforts of the past two decades put into learning taggers from tag dictionaries: papers have successively removed layers of unrealistic assumptions, and in doing so have produced pipelines for type-supervision that easily beat token-supervision prepared in comparable amounts of time.

The result of most immediate practical value is that we show it is possible to train effective POS-taggers on actual low-resource languages given only a relatively small amount of unlabeled text and a few hours of annotation by a non-native linguist. Instead of having annotators label full sentences as one might expect the natural choice would be, it is much more effective to simply extract a list of the most frequent word types in the language and concentrate efforts on annotating these types with their potential parts of speech. Furthermore, for languages with rich morphology, a morphological transducer can yield significant performance gains when large amounts of other annotated resources are unavailable. (And it never hurts performance.)

Finally, additional raw text does improve performance. However, using substantial amounts of raw text is unlikely to produce gains larger than only a few hours spent annotating types. Thus, when deciding whether to spend time locating larger volumes of digitized text or to spend time annotating types, choose types.

Despite the consistent superiority of type annotations in our experiments, it of course may be the case that techniques such as active learning may better select sentences for token annotation, so this should be explored in future work.

Acknowledgements

We thank Kyle Jerro, Vijay John, Jim Evans, Yoav Goldberg, Slav Petrov, and the reviewers for their assistance and feedback. This work was supported by the U.S. Department of Defense through the U.S. Army Research Office (grant number W911NF-10-1-0533) and through a National Defense Science and Engineering Graduate Fellowship for the first author. Experiments were run on the UTCS Mastodon Cluster, provided by NSF grant EIA-0303609.

References

Steven Abney and Steven Bird. 2010. The human language project: Building a universal corpus of the worlds languages. In *Proceedings of ACL.*

Jason Baldridge and Alexis Palmer. 2009. How well does active learning actually work? Time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of EMNLP*, Singapore.

Steven Bird. 2011. Bootstrapping the language archive: New prospects for natural language processing in preserving linguistic heritage. *Linguistic Issues in Language Technology*, 6.

Silviu Cucerzan and David Yarowsky. 2002. Bootstrapping a multilingual part-of-speech tagger in one person-day. In *Proceedings of CoNLL*, Taipei, Taiwan.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of ACL-HLT*, Portland, Oregon, USA.

Markus Dickinson and W. Detmar Meurers. 2003. Detecting errors in part-of-speech annotation. In *Proceedings of EACL.*

Weiwei Ding. 2011. Weakly supervised part-of-speech tagging for Chinese using label propagation. Master's thesis, University of Texas at Austin.

Dan Garrette and Jason Baldridge. 2012. Type-supervised hidden Markov models for part-of-speech tagging with incomplete tag dictionaries. In *Proceedings of EMNLP*, Jeju, Korea.

Dan Garrette and Jason Baldridge. 2013. Learning a part-of-speech tagger from two hours of annotation. In *Proceedings of NAACL*, Atlanta, Georgia.

Yoav Goldberg, Meni Adler, and Michael Elhadad. 2008. EM can find pretty good HMM POS-taggers (when given a good start). In *Proceedings ACL*.

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings NAACL*.

Kazi Saidul Hasan and Vincent Ng. 2009. Weakly supervised part-of-speech tagging for morphologically-rich, resource-scarce languages. In *Proceedings of EACL*, Athens, Greece.

Lauri Karttunen. 2001. Applications of finite-state transducers in natural language processing. *Lecture Notes in Computer Science*, 2088.

Julian Kupiec. 1992. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech & Language*, 6(3).

Shen Li, João Graça, and Ben Taskar. 2012. Wiki-ly supervised part-of-speech tagging. In *Proceedings of EMNLP*, Jeju Island, Korea.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of CICLing*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2).

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2).

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proceedings ACL*.

Sujith Ravi and Kevin Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *Proceedings of ACL-AFNLP*.

Sujith Ravi, Ashish Vaswani, Kevin Knight, and David Chiang. 2010. Fast, greedy model minimization for unsupervised tagging. In *Proceedings of COLING*.

Emmanuel Roche and Yves Schabes. 1995. Deterministic part-of-speech tagging with finite-state transducers. *Computational Linguistics*, 21(2).

Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings EMNLP*, Cambridge, MA.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. In *Transactions of the ACL*. Association for Computational Linguistics.

Partha Pratim Talukdar and Koby Crammer. 2009. New regularized algorithms for transductive learning. In *Proceedings of ECML-PKDD*, Bled, Slovenia.

Kristina Toutanova and Mark Johnson. 2008. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *Proceedings of NIPS*.