# Evaluating Lemmatization Models for Machine-Assisted Corpus-Dictionary Linkage

Kevin Black, Eric K. Ringger, Paul Felt, Kevin Seppi, Kristian Heal\*, Deryle Lonsdale<sup>†</sup>

Dept. of Computer Science, \*Neal A. Maxwell Institute, <sup>†</sup>Dept. of Linguistics and English Language Brigham Young University, Provo, Utah 84602 USA kevinblack@byu.edu, ringger@cs.byu.edu, {paul\_felt, kseppi, kristian\_heal, lonz}@byu.edu

#### Abstract

The task of corpus-dictionary linkage (CDL) is to annotate each word in a corpus with a link to an appropriate dictionary entry that documents the sense and usage of the word. Corpus-dictionary linked resources include concordances, dictionaries with word usage examples, and corpora annotated with lemmas or word senses. Such CDL resources are essential for many tasks including assisting language learners, linguistic research, philology, and translation. Lemmatization is a common approximation to automating corpus-dictionary linkage, where lemmas stand in for the headwords of an actual dictionary. In our machine-assisted CDL system design, data-driven lemmatization models provide machine assistance to human annotators performing the actual CDL task. Assistance is provided in the form of pre-annotations that will reduce the costs of CDL annotation. In this work we adapt the discriminative string transducer DirecTL+ to perform lemmatization for classical Syriac, a low-resource language. We compare the accuracy of DirecTL+ with the Morfette discriminative lemmatizer. DirecTL+ achieves 96.92% overall accuracy, an improvement of 0.86% over Morfette but at the cost of a longer time to train the model. Error analysis on the models provides guidance on how to apply these models in a machine assistance setting for corpus-dictionary linkage.

Keywords: corpus-dictionary linkage, lemmatization, machine assistance

#### 1. Introduction

The task of corpus-dictionary linkage (CDL) is to annotate each token in a corpus with a link to an appropriate dictionary entry in order to document the token's sense and usage, and to gather related tokens under a representative headword. When there is no suitable dictionary entry for a given word token a new entry must be created. A lexicographer could follow this process to produce a dictionary (Kilgarriff, 2005); likewise, a concordance editor could employ this process to compile a concordance for a selected corpus. CDL resources include dictionaries with word usage examples, concordances, and corpora annotated with lemmas or word senses. Such CDL resources are essential for many tasks including assisting language learners, linguistic research, philology, and translation.

Manual construction of CDL resources is costly in terms of human annotator time and tedium which limits the size and scope of CDL annotated resources (Kiraz, 1994). The temporal costs of CDL annotation can be reduced by providing machine assistance to expert annotators in the form of automatic pre-annotations (Ringger et al., 2008; Carmen et al., 2010; Felt et al., 2013), which should additionally increase the feasible size and scope of CDL annotated resources and in turn provide more quality data with which to refine machine-learned models for automatic CDL annotation. For common languages, crowdsourcing might be a viable way to link corpus words to a dictionary; however, CDL for uncommon or old languages, such as classical Syriac, is not generally crowdsourceable because the pool of qualified workers is insufficient.<sup>1</sup> Furthermore, uncommon or old language corpora are primarily interesting to language experts who may be invested in providing quality annotations voluntarily.

The level of granularity of a dictionary determines the complexity of the CDL task for both humans and machines. Since most dictionaries organize dictionary entries around the lemmas (i.e., headwords, baseforms) of the language, linking tokens in the corpus to lemmas in such dictionaries requires the ability to determine the correct lemma of each word token. Linking tokens in a corpus to a dictionary at this finer granularity requires an understanding of the morphology of the language. Of course, beyond the level of lemmas, dictionaries can also organize the documentation of a single lemma as multiple entries, organized by the grammatical category (part-of-speech) and/or senses of the word. For this finer granularity, CDL further requires word-sense disambiguation.

In this work we focus on corpus-dictionary linkage as a lemmatization task, not focusing for the moment on disambiguating among multiple dictionary entries or sub-entries for a particular lemma. Whereas the morphological analysis task traditionally enumerates candidate lemmas (for our purposes, also "baseforms") for a given word token along with its morphological attributes, the task of discriminative lemmatization is to disambiguate and either choose from among the alternatives or rank them by some measure. Hence, discriminative lemmatization is a tagging process. For inflections and declensions of word types with regular morphology, a token is likely to resemble a substring of its baseform (e.g., "walked" contains its baseform "walk"). For irregular forms, such correspondence is less likely (e.g., "went" versus "go"). However, because of the substring correspondence of most word tokens and their baseforms, discriminative lemmatization can be implemented with general machine learning approaches to

<sup>&</sup>lt;sup>1</sup>We submitted a Syriac lemmatization task to CrowdFlower in the fall of 2013. None of the 3,051 workers that attempted the trial job performed at the required minimum accuracy of 75%.

string transduction trained from representative data.

We break the discriminative lemmatization process into two tasks: *token segmentation* (or simply *segmentation*), which separates prefix and stem and suffix morphemes, and *stem-baseform linkage*. Using sequence labeling methods, automatic segmentation can be performed with high accuracy. For example, in our work with classical Syriac, segmentation can be performed with 98.87% accuracy. For that reason, we focus exclusively on the harder task of stembaseform linkage. In this paper, we adapt the DirecTL+ discriminative string transducer (Jiampojamarn et al., 2008) to perform stem-baseform linkage for classical Syriac and compare its performance with the Morfette (Chrupała et al., 2008) approach. These results and the accompanying error analysis will inform and guide our ongoing work in machine assisted corpus-dictionary linkage.

## 2. Syriac Corpus-Dictionary Linkage

By text volume, Syriac is the largest dialect of Aramaic. Classical Syriac was the primary spoken and written language of the Christian Near East through the eighth century and is still used in the liturgy of the present Syriac Christian churches. Syriac has twenty-two letters (literals) like Hebrew; classical Syriac is usually written with very few diacritics (for vowels or other pronunciation marks). We use the Library of Congress' transliteration scheme for Syriac.<sup>2</sup> Syriac words are comprised of the following parts, with examples given in Table 1:

- stem: the main part a word token; the stem is a morphological inflection of a baseform
- **prefix**: clitic preceding the stem; some tokens do not have a prefix
- **suffix**: clitic following the stem; some tokens do not have a suffix
- **baseform**: (i.e., lemma) the basic word-form from which stems are inflected
- **root**: the word-form from which baseforms are derived; many Semitic roots consist of three literals

Word Token		Prefix Stem		Suffix	Baseform	
9	'mr		'mr		'mr	
10	lhwn		l	hwn	l	
11	pyltws		pylțws		pyltws	
12	lmlkkwn	1	mlk	kwn	mlk'	
13	'zqwp		'zqwp		zqp	

Table 1: Some Syriac (transliterated) words from John 19:15 in the Peshitta New Testament and their prefix, stem, and suffix parts with the associated baseforms.

Since a particular orthographic stem may be an inflection of two or more distinct baseforms, a discriminative lemmatizer must disambiguate among them. Syriac dictionaries are generally organized into entries for both baseforms and root forms. Jessie Payne Smith's (JPS) Compendious Syriac Dictionary (Payne Smith, 1903) contains more than 16,000 such entries. Furthermore, some dictionary entry headwords are homographs, introducing ambiguity into the collection (now a multi-set) of baseforms. Thus, a step beyond disambiguating among alternative baseforms is to disambiguate among homographic baseforms. By contrast, prefixes and suffixes of a given word token are on average much less ambiguous than stems and baseforms and are generally enumerated in a Syriac grammar.

While our ultimate goal is to link arbitrary Syriac corpora to entries in the JPS dictionary, this work focuses on stem-baseform linkage, the second and most challenging step of lemmatization, as a preliminary step to full corpusdictionary linkage. To appreciate the scale of the data to which our CDL models will ultimately be applied, consider the Syriac Electronic Corpus (SEC) project of the Center for the Preservation of Ancient Religious Texts in the Neal A. Maxwell Institute for Religious Scholarship at Brigham Young University. The aims of the project are to collect, digitize, annotate, and publish a large corpus of classical Syriac texts spanning multiple authors and centuries. To date the project has collected and digitized almost 6 million words of text, including a subcorpus consisting of all of the writings of Ephrem the Syrian (d. 373 A.D.). This subcorpus, subsequently referred to as EPHREM, is of particular historical interest. The next step in the project is to link each word in EPHREM to entries in the JPS dictionary and to extend the dictionary where entries are missing.<sup>3</sup>

EPHREM contains approximately 465,000 word tokens, more than four times as many as the Peshitta New Testament. By linearly interpolating from annotation timing data collected by Felt et al. (2013) in their user study of machine assistance for Syriac morphological annotation, we estimate it might take a single typical Syriac scholar more than two years to fully link EPHREM to the dictionary — assuming the annotator works 20 hours per week (48 weeks per year). The time to fully annotate is reduced to under one year if automatic pre-annotations are provided and automatically refined as the annotator corrects the suggested annotations.<sup>4</sup> Thus, machine assistance has the potential to play a critical role in this particular corpus-dictionary linkage project. Beyond EPHREM, machine assistance is even more vital for annotating the entire SEC.

We further estimate the number of unique stem and baseform types to be identified in the linkage from EPHREM to the JPS dictionary by linearly interpolating from corpus metrics on the PNT. Assuming EPHREM displays the same token-to-type ratio as the PNT, it will contain approximately 70,000 word types, 34,000 stem types, and 13,000 baseform types. Although the JPS dictionary contains over 16,000 distinct entries for roots and baseforms, it will undoubtedly need to be expanded in order to accommodate new types as EPHREM is linked.

Our experiments in this paper employ stem-baseform linkage data from the annotated Peshitta New Testament (PNT). The original morphological annotation of the PNT was done manually by The Way International Foundation

<sup>&</sup>lt;sup>3</sup>http://cpart.maxwellinstitute.byu.edu/home/sec/about

<sup>&</sup>lt;sup>4</sup>These time estimates are lower bounds since EPHREM contains more poetic and metaphorical language than the New Testament apocrypha text studied by Felt et al. (2013).

<sup>&</sup>lt;sup>2</sup>http://www.loc.gov/catdir/cpso/romanization/syriac.pdf

over the course of 15 years. Kiraz (1994) digitized and refined those morphological annotations into the Syriac Electronic Data Retrieval Archive III (SEDRA3). McClanahan et al. (2010) subsequently adapted the SEDRA3 annotations to include suffix annotations. The current annotated PNT used in this work has 109,640 word tokens, 16,439 word types, 7970 stem types, 3038 baseform types, and 1800 root types.<sup>5</sup>

#### 3. Related Work

Morphological analysis enumerates all possible lemmatizations (baseforms) for the stem of a given word token (along with the token's segmentation and morphological attributes) and can therefore be used as a component of corpus-dictionary linkage. Kiraz (2000) built a finite state morphological analyzer for Syriac and Arabic, but unfortunately the implementation is no longer available. Tim Buckwalter's morphological analyzer for Arabic is perhaps the best known analyzer for a Semitic language.<sup>6</sup> However, traditional morphological analysis only enumerates possible baseform links and does not rank them.

Morphological disambiguation extends morphological analysis by ranking candidate analyses based on some measure of the correctness of the analysis given word context. To assist annotators with automatic stem-baseform link proposals, a morphological disambiguation system is necessary. Habash and Rambow (2005) built an Arabic morphological disambiguation system that uses a machinelearned classifier for each morphological attribute. They present methods for combining the best individual morphological tags into a joint tag that is comparable to the quality of morphological analyses obtained from a morphological analyzer. They achieve a morphological tag accuracy of 97.6%. Recently, Lindgren (2011) built a morphological analyzer for classical Syriac called dkrMorph, which provides some basic disambiguation functionality by ranking candidate analyses according to frequency of occurrence in a training set. Despite the simplicity of its ranking algorithm, dkrMorph reports 94.42% accuracy on a heldout set of stem-baseform links; 99.75% accuracy on known stems seen in the training data and 46.20% on unknown stems encountered only in the heldout set.

McClanahan et al. (2010) built a probabilistic morphological disambiguation system for classical Syriac called SyroMorph. SyroMorph's segmentation module uses a conditional Markov model that interpolates distinct models for known and unknown word types. For known word types, a maximum entropy (maxent) classifier disambiguates among known segmentations for the word. For unknown word types, a sequence-labeling approach is employed using a maximum entropy Markov model to decide whether each character in the word is in the prefix, in the stem, or in the suffix. As noted in Section 1, the segmentation model achieves 98.87% overall accuracy on the Syriac PNT, making Syriac segmentation a mostly solved problem. SyroMorph's stem-baseform linker uses a hybrid model that interpolates a maxent classifier for known word types with a discriminative lemmatizer for unknown word types. The lemmatizer is Morfette, introduced by Chrupała et al. (2008), which will be discussed further in Section 4. When the stem-baseform linkage module is trained and tested on heldout stem-baseform data, Syro-Morph achieves 96.19% accuracy overall, 98.05% accuracy on known stems, and 78.40% accuracy on unknown stems. This work by McClanahan et al. provides a point of comparison for the work reported in this paper.

DirecTL+ is a general discriminative string transducer which has been applied to the letter-to-phoneme problem (Jiampojamarn et al., 2008), and name transliteration (Jiampojamarn et al., 2009). Jiampojamarn et al. (2007) recognized the potential to use DirecTL+ for morphological analysis, but to our knowledge we are the first ones to do so. We describe our application of DirecTL+ to the stembaseform linkage task in Section 4.

Kilgarriff (2005) describes how dictionaries can be curated using the Word Sketch Engine (Kilgarriff and Rundell, 2002). A *word sketch* summarizes the words that cooccur with a particular target word in a corpus, organized by the frequency of the collocations and morphological attributes of co-occurring words. Word sketches offer a lexicographer information with which to manually build and refine a dictionary but do not automatically link word occurrences to existing natural dictionary entries.

### 4. Models and Features

As noted in the introduction, word tokens and their baseforms are often similar, possibly even sharing some substring(s). Thus, string transduction models can be used to perform lemmatization. However, to disambiguate among possible baseforms for a given stem, the transducer will need to take a word's local context (sentence) into account. The Morfette model, which has state-of-the-art results for stem-baseform linkage in classical Syriac, is a discriminative lemmatizer that accommodates the transduction of words in isolation or in the context of neighboring words. The DirecTL+ model is a general discriminative string transducer. Inasmuch as DirecTL+ has no built-in notion of words or sentential context, we present a preprocessing step that combines multiple stem-baseform linkage problems into a single sentence-level string transduction problem. The following sub-sections describe Morfette and DirecTL+ in more detail, including an enumeration of the word and local-context features the model implementations presently provide.

#### 4.1. Morfette

The Morfette approach to lemmatization uses a maximum entropy classifier to predict an edit script for a given stem; the script is then applied to the stem to produce the baseform (Chrupała et al., 2008). The original Morfette implementation uses a minimum edit script zero-indexed from the end of the input string (Chrupała et al., 2008); e.g., for the stem-baseform pair (*mlk*, *mlk'*) the edit script would be <(Insert, ', 0)> to transform *mlk* to *mlk'*. McClanahan et al. (2010)'s Morfette implementation uses a minimum edit script zero-indexed from the beginning of the input string; e.g., to transform *mlk* to *mlk'* the edit script is <(Insert, ',

<sup>&</sup>lt;sup>5</sup>All counts are for non-diacritized units even though the PNT has diacritized word tokens.

<sup>&</sup>lt;sup>6</sup>http://www.qamus.org/

2)>. The edit operators considered by the maximum entropy model are learned from labeled stem-baseform pairs. Not all of the edit operators in the model will apply to each input unit, thus the decoder disregards non-applicable operators for a given context; e.g., say the edit script <(Delete, ', 0), (Delete, w, 2)>, learned from the stem-baseform pair ('*zqwp*, *zqp*), had high probability for the stem *mlk* but the script would not be applicable because *mlk* does not have an *aleph* (') at position 0 or a *waw* (w) at position 2.

Individual Morfette models can operate on stems in isolation, using stem features to predict edit script classes. When the Morfette model is composed into a conditional Markov model, features from neighboring words (local context) can additionally be employed, giving the model the potential to disambiguate stems based on their usage in a sentence.

SyroMorph's Morfette implementation provides the feature templates summarized in Table 2. All of the feature templates are employed to collectively extract a sparse feature vector. Word context feature templates that only extract information from the input stem include WORD, WORDLEN, CHARACTER, PREFIX, and SUFFIX. The WORD template extracts the stem itself as a feature. The WORDLEN template extracts the length of the stem as a feature. We corrected the CHARACTER template to extract the characters in the stem as features; it had been miscoded to emit word.length copies of the same feature, thus being redundant with the WORDLEN feature. The PREFIX template extracts the prefixes of length [1, min(5, word.length)] for stems longer than one character. Similarly, the SUFFIX template extract the suffixes of length [1, min(10, word.length)] for stems longer than one character. (Note that these features are not to be confused with the prefix and suffix morphemes of a word token.)

Feature templates that operate on the local context (the stem and its surrounding stems) include EOS and PREV-TAGS. The EOS (End Of Sentence) template extracts the position of the word from the end of the sentence if its position is less than the configured Markov order. The PREV-TAGS template extracts sequences of baseforms from the stem's preceding stems; extracted sequences are of length 1 to the configured Markov order (inclusive).

### 4.2. DirecTL+

The DirecTL+ approach to string transduction uses a joint discriminative hidden Markov model and monotone phrasal decoder (Jiampojamarn et al., 2008; Jiampojamarn et al., 2010). As a pre-processing step before training the model, the input and output data must be aligned. Jiampo-jamarn et al. (2007) implemented a monotone (no crossing) many-to-many (M2M) alignment algorithm that determines the mapping between character subsequences in the input to character subsequences in the output. The algorithm is parameterized by the maximum number of characters allowed in any subsequence, or chunk, on the input and output sides, and whether characters on the input or output side can be deleted. The many-to-many aligned data is then used to train DirecTL+. The discriminative HMM(Collins, 2002) employs the Viterbi algorithm to predict the best output

	Name	Template	Example	
W	WORD	word	mlk	
W	WORDLEN	word.length	3	
W	CHARACTER	character	$\{m, l, k\}$	
W	PREFIX	length, prefix	$\{m, ml\}$	
		string		
W	SUFFIX	length, suffix	$\{k, lk\}$	
		string		
L	EOS	distance from	N/A*	
		end		
L	PREVTAGS	(Markov or-	$\{(1, pyltws),$	
		der, tag, tag,	(2, pyltws, l)	
		, tag)		

Table 2: Morfette feature templates for SyroMorph 2.2. Here *word* means stem. W indicates word context features; L indicates local context features. Examples are for stem *mlk* in the context of John 19:15 assuming Markov order 2.

string for a given input string. The monotone phrasal decoder works in conjunction with the discriminative HMM to chunk individual characters into subsequences consistent with the chunks in the many-to-many aligned training data.

Adapting DirecTL+ to stem-baseform linkage, we build two kinds of input-output string pairs for training: isolated word pairs, and sentence pairs. The isolated word string pairs consist of just the stem-baseform pair for a single token; for example, after M2M processing (see Section 5) the input string for mlk is m-l.k and the output string for mlk' is *m-l.k.*' ("-" delimits chunks and "." indicates characters combined into chunks). To enable DirecTL+ to lemmatize stems given their local context, we first pre-process the isolated word string pairs with the M2M tool, and then concatenate the word alignments into a sentence string. The input string is thus a stem sentence and the output string is a baseform sentence. This method of pre-processing ensures that DirecTL+ will always predict a baseform sentence with one and only one baseform for each stem in the input because the item delimiter chunk (#) is always mapped to itself; an alternate method of first concatenating stems and baseforms into strings and then running the M2M tool to learn alignments allows the item delimiter to be chunked with other characters and thus does not guarantee that DirecTL+ will predict one and only one baseform for each stem.

At test time DirecTL+ will predict the transduction to the output string as well as the chunking of the input string, so the input string does not need to be pre-processed by the M2M utility but rather only broken down into individual characters. When testing lemmatization for stems in their local context, the stems in a sentence again need to be concatenated together.

DirecTL+ provides the features summarized in Table 3, which can be seen as word or local context features given the particular pre-processing of the data. The n-gram context template extracts all n-grams of input chunks of size [1, n] within a window c of the current chunk. The transition template extracts the output chunk for the current input chunk's preceding chunk. The linear chain template pairs

the output chunk for the input chunk's preceding chunk with each of the n-grams extracted by the n-gram context template. The joint n-gram template extracts n-grams of size [1, n] of input-output-chunk pairs for chunks preceding the current chunk. With these features and the monotone phrasal decoder that explores possible chunks, DirecTL+ captures much of the same information as the WORD, CHARACTER, PREFIX, and SUFFIX feature templates in Morfette. DirecTL+ cannot model the WORDLEN feature.

#### 5. Experimental Design

To determine how well DirecTL+ performs stembaseform linkage we compare it against Morfette and a strong baseline stem-baseform linker. We build Morfette and DirecTL+ models that operate on stems in isolation, only using word context features, which we denote Morfette-W and DirecTL+-W. Similarly, we build models that operate on stems in their local context, denoted Morfette-L and DirecTL+-L. This section describes the parameterization and training of each model. The accuracy values of the models are then evaluated and compared to one another in the next section.

The models are trained and tested on non-diacritized stem-baseform data from the Syriac PNT. The gold-standard stems are used as test input, as opposed to running SyroMorph's segmentation module and taking its predicted stems as test input. To conduct better error analysis on unknown stems we use a 60-20-20 training-validation-test split of the data, whereas previous evaluations of stembaseform linkage on the PNT have used an 80-10-10 split (McClanahan et al., 2010; Lindgren, 2011). Using 60% of the data for training instead of 80% should not hurt model performance too much, since training on 35% of the data yields high performance on a variety of morphological tasks on the PNT (McClanahan et al., 2010). 10-fold cross validation experiments are run over a combination of the training and development test data.

The baseline model memorizes the most frequently used baseforms for each stem in the training data and then predicts the most frequent baseform for a given known stem in the test data. For unknown stems the baseline model simply predicts the stem itself as the baseform.

For both the Morfette-W and Morfette-L models we use all word context feature templates, and for Morfette-L, all local context feature templates provided by SyroMorph 2.2. We use all of the extracted features instead of selecting only rare or non-rare features. We allow the log likelihood maximization of the maximum entropy model to run until improvements are less than  $10^{-6}$ . The decoder beam size is set to 5. Though the maximum entropy models in Morfette-W operate on words in isolation, the models are still composed in a conditional Markov model in the code so as to get a baseform prediction for each stem in the sentence.

To determine a good many-to-many alignment of the Syriac data we ran the M2M alignment tool, trained DirecTL+ on the aligned output using a minimal set of ngram features (context size of 3), and then compared the accuracy achieved by DirecTL+ given the different parameterizations of the M2M tool. We conducted a search over the cross-product of the maxX [1,5], maxY [1,5], delX (yes/no), and delY (yes/no) parameters of M2M. The maxX parameter is the maximum number of characters allowed in an input chunk. Similarly, the maxY parameter is the maximum number of characters allowed in an output chunk. The delX parameter controls whether chunks can be deleted from the input, which is represented by mapping the input chunk to a *null* output chunk. Similarly, the delY parameter controls whether chunks can be deleted from the output, which is represented by mapping a *null* input chunk to the output chunk; deleting an output chunk is the same as inserting an input chunk. The best parameters turned out to be maxX = 3, maxY = 3, delX = yes, delY = no, based on performance on the validation set.

Using the M2M pre-processed data, DirecTL+ models were then built adding one feature template at a time and searching for good parameters for the , again based on performance on the validation set. For the DirecTL+-W model the best feature template configuration was to include n-grams with a context window of size 5 (allowing up to 11-grams), transition features (Markov order of 1 hard-coded), linear chain features, and joint n-gram features with n = 2. DirecTL+ always employs a beam search when joint n-gram features are used. We kept the default beam size of 20. For the DirecTL+-L the best feature template configuration was to include n-grams with a context window of size 4 (allowing up to 9-grams), transition features, and linear chain features. We did not include joint n-gram features since they did not improve accuracy.

### 6. **Results**

To understand the expected performance of the models we performed 10-fold cross validation on the combined training and validation data. Table 4 reports the models' overall, known, and unknown accuracies and correct and total counts. All of the models were tested on the same partitioning of the data with the exception of DirecTL+-W.<sup>7</sup> Each of the models achieves above 90% overall accuracy, including the baseline model, and each discriminative model performs better than the baseline except for DirecTL+-W. The accuracy gains by Morfette-W, Morfette-L, and DirecTL+-L come from their better generalization to unknown stems. The baseline model generalizes to unknown stems at about 20% accuracy, whereas all of the discriminative models generalize at about 60% accuracy, including DirecTL+-W. Indeed, DirecTL+-W achieves the highest mean unknown accuracy, but its high variance makes it less stable than the other discriminative models. As expected, data-driven discriminative generalization works better than the heuristic of predicting unknown stems to be their own baseforms. Considering known stems, while all the discriminative models achieve more than 93%, none achieve better accuracy than 98.29% as achieved by the baseline approach of memorizing the most frequent baseform for each seen stem.

We evaluated the models on the blind test data to ensure that the expected performance seen in cross validation was consistent with the generalized performce, and to compare

<sup>&</sup>lt;sup>7</sup>DirecTL+-W was run on a distinct partition from a different random seed due to experimenter's error.

Name	Word Context Example	Local Context Example
n-gram context		
1-grams	'. <i>z</i> . <i>q</i> , <i>w</i> , <i>p</i>	, m, l.k, #, '.z.q, w, p,
2-grams	('.z.q, w), (w, p)	, (m, l.k), (l.k, #), (#, '.z.q), ('.z.q, w), (w, p),
transition		
order-1	<i>z.q</i>	z.q
linear chain		
1-grams	$[<'.z.q, w, p> \times z.q]$	$[<, m, l.k, #, '.z.q, w, p,> \times z.q]$
2-grams	$[<('.z.q, w), (w, p)> \times z.q]$	$[<, (m, l.k), (l.k, #), (#, '.z.q), ('.z.q, w), (w, p), > \times z.q]$
joint n-gram		
1-grams	['. <i>z</i> . <i>q</i> , <i>z</i> . <i>q</i> ]	, [m, m], [l.k, l.k.'], [#, #], ['.z.q, z.q],
2-grams	none	$\dots, ([m, m], [l.k, l.k.']), ([l.k, l.k.'], [#, #]), ([#, #], ['.z.q, z.q]), \dots$

Table 3: DirecTL+ feature templates examples. Examples given are relative to the second input chunk, w, in the M2M pre-processed stem '.*z.q-w-p* in its context of John 19:15. Bigrams are indicated by (*chunk-1*, *chunk-2*); unigrams have no surrounding parentheses; n-grams where n > 2 are not shown. Pairings of input and output chunks are indicated by [*input-chunk*, *output-chunk*]. The notation " $<\bullet> \times$  *chunk*" indicates an element-wise pairing of the elements in  $<\bullet>$  with the *chunk*.

	All Acc.	All Correct	Known Acc.	Known Correct	Unk. Acc.	Unk. Correct
Total Count	N/A	$*8756 \pm 118$	N/A	$*8411 \pm 105$	N/A	$*345 \pm 20$
Baseline	$95.31 \pm 0.34$	$8347 \pm 109$	$\textbf{98.29} \pm \textbf{0.18}$	$8267 \pm 104$	$23.05 \pm 4.03$	$80 \pm 14$
Morfette-W	$96.14 \pm 0.29$	$8419 \pm 113$	$97.62 \pm 0.25$	$8211 \pm 104$	$60.00 \pm 2.91$	$207 \pm 16$
Morfette-L	$95.96 \pm 0.31$	$8402 \pm 113$	$97.42 \pm 0.25$	$8194 \pm 104$	$60.33 \pm 2.62$	$208 \pm 15$
DirecTL+-W*	$92.67 \pm 0.94$	$8115\pm127$	$93.93 \pm 0.66$	$7904 \pm 107$	$61.40 \pm 12.51$	$211\pm51$
DirecTL+-L	$\textbf{96.71} \pm \textbf{0.26}$	$8468 \pm 106$	$98.24 \pm 0.20$	$8263 \pm 103$	$59.36 \pm 2.65$	$205 \pm 15$

Table 4: Mean and std. dev. values for accuracy and related counts with 10-fold cross validation. Measures are per stem. Max. values in bold. Unknown values for Morfette-L in bold due to its stability. \*All models were cross-validated on the same data partition except for DirecTL+-W, which has comparable total count values: All Total Count  $8756 \pm 91$ , Known Total Count  $8414 \pm 91$ , Unknown Total Count  $342 \pm 17$ .

with previously reported results. Table 5 reports accuracies, as well as correct and total counts. Of the models tested in this work, DirecTL+-L achieves the best overall accuracy at 96.62% and known stem accuracy at 98.30%, while DirecTL+-W does best on unknown stems with 61.78%. Compared to previously reported results in similar circumstances, DirecTL+-L still does best overall. SyroMorph's hybrid Morfette model (as opposed to its plain Morfette model) has the best reported unknown accuracy of 78.40%; however, training and testing scenarios differ here as well: we trained on 80% and tested on 20% of the data, while the original SyroMorph was trained on 90% and tested on 10% of the data. The latter difference in data configuration may explain the difference in known accuracy: using a larger training set makes more of the stems known and unambiguous at test-time.

DirecTL+-L's superior performance as measured by accuracy, is tempered by the time required to train the model. DirecTL+-L took roughly 25 times longer to train (100 minutes) than it did to train Morfette-L (3.7 minutes), which is likely due to the fact that DirecTL+ does Viterbi decoding during its online discriminative training. The long training time might be suitable for stem-baseform linkage in batch, offline processing for small data sets, but it would scale poorly to our intended machine assistance scenario where the stem-baseform linkage model is iteratively refined as data is annotated. Morfette-W therefore appears to be better suited to provide dynamically adaptive machine assistance as it achieves the second highest overall accuracy but can be trained more quickly.

For further comparison, the results reported for dkr-Morph (Lindgren, 2011) in terms of known accuracy are higher than ours, but a number of differences make the results difficult to compare. First and foremost, the task is not the same: dkrMorph uses the PNT dataset directly from SE-DRA3 *without* the additional suffix segmentation information employed by our models and *with* diacritics. Also, the dataset partition is not the same: dkrMorph trained on 90% and tested on 10%, 10,960 cases, of the PNT dataset. Overall accuracy was 94.42%, and accuracy on known cases was reported to be 99.75%, while accuracy on unknown cases was 47.20%.

One possible approach to achieve even better accuracy, and therefore better pre-annotations for use in machine-assisted corpus-dictionary linkage, is to combine the strengths of the different models. For example, combine the memorization of the baseline model—that does well on seen stems—with the generalization and disambiguation abilities of the discriminative models for unseen and ambiguous stems respectively. SyroMorph's hybrid Morfette model takes this approach and achieves (now) the second highest reported overall accuracy on the PNT and the highest unknown accuracy. We do not combine models in this work, but we do conduct error analysis to better understand

	All Acc.	All Correct	Known Acc.	Known Correct	Unk. Acc.	Unk. Correct
Total Count	N/A	22,080	N/A	21,269	N/A	811
Baseline	95.57	21,101	98.20	20,887	26.39	214
Morfette-W	96.04	21,205	97.48	20,372	58.32	473
Morfette-L	96.06	21,209	97.47	20,730	59.06	479
DirecTL+-W	92.76	20,481	93.94	19,980	61.78	501
DirecTL+-L	96.92	21,399	98.30	20,907	60.67	492
SyroMorph	96.19	N/R	98.05	N/R	78.40	N/R

Table 5: Accuracy and related counts per model on blind test data. Measures are per stem. Max. values in bold. SyroMorph results as reported by McClanahan et al. (2010) using a test set with 11,290 stems. N/R indicates *not reported*.

the strengths and weaknesses of each model.

Returning to the 10-fold cross validation results on the training and validation data, we examine the true labels on the test data (per fold, with aggregate statistics reported) and categorize the stems into the following error cases:

- **known stable unambiguous (KSU)**: the stem has the same *one* baseform in the training and test data
- **known latent ambiguous (KLA)**: the stem has one baseform in the training data, but has one or more additional baseforms in the test data
- **known stable ambiguous (KSA)**: the stem has the same two or more baseforms in the training and test data
- **known dynamic ambiguous (KDA)**: the stem has two are more baseforms in the training data, and has one ore more additional baseforms in the test data
- **unknown unambiguous (UU)**: the stem has one baseform in the test data
- **unknown ambiguous (UA)**: the stem has two or more baseforms in the test data

Table 6 reports the 10-fold cross validation performance of the Baseline, Morfette-L and DirecTL+-L models for each error case, due to their superior performance.

The known stable unambiguous case is the largest, accounting for about 80% of all of the test stems in each fold. Out of the unambiguous stems in the training data (KSU + KLA), 99.1% remain unambiguous at test time, while 0.9% have new additional baseforms. The baseline model achieves perfect memorized accuracy on the KSU stems, and a little above 50% accuracy on KLA stems but with a standard deviation of 25%, indicating that the novel ambiguous baseforms occur on average about two out of four times in the test data. Morfette-L and DirecTL+-L come close to perfect accuracy for the KSU case as well; however, more analysis must be done to determine if training the discriminative models on the high volume of KSU stems hinders the models' ability to generalize in the ambiguous or unknown cases as it might try to focus on memorizing stem-baseform mappings instead. Morfette-L and DirecTL+-L models perform better than the baseline on the KLA case, indicating that they can generalize a little bit to novel baseforms for seen stems.

The known stable and dynamic ambiguous cases (KSA and KDA) are harder than the known unambiguous (KSU + KLA) cases. The baseline model achieves 90.70% on KSA stems which indicates that most occurrences of an

ambiguous stem use the most frequent baseform seen during training. The fact that Morfette-L performs at 90%, slightly worse than the baseform, implies that it also tends to predict the most frequent baseform. The DirecTL+-L model achieves better accuracy than the baseline model, at 93.48%, indicating that it is able to disambiguate among some stems' known baseforms. Similar patterns are seen in the KDA case, although the percentage of test stems that are even more ambiguous than in the training data is only 0.15% so it is unclear whether DirecTL+-L is generalizing to capture novel test-time baseforms or just disambiguating among known baseforms.

Morfette-L and DirecTL+-L are able to generalize much better than the baseline model for unknown unambiguous stems, which is consistent with the overall unknown performance observed before. However, on unknown ambiguous stems there is no apparent winner among the three models given the fact that the number of ambiguous unseen stems is very small. Although the UA error case is small (0.58% of all unknown tokens, 0.02% overall), further analysis might reveal whether the baseline model and discriminative models use effectively the same generalization hypothesis to predict the stem itself as the baseform, or whether they, in fact, use complementary hypotheses.

### 7. Conclusion

We evaluated several models for stem-baseform linkage, a form of corpus-dictionary linkage. We adapted the DirecTL+ discriminative string transducer and compared it with the state-of-the-art Morfette lemmatizer and with a strong baseline. Both Morfette with word or local context features and DirecTL+ with local context features perform better than the baseline in overall accuracy. DirecTL+ with local context achieves about 96.92% overall accuracy on the Syriac Peshitta New Testament, the best reported accuracy on this data set.

In future work we will explore more fully the interactive, online setting in which the stem-baseform linkage model is refined while expert annotators provide additional links between the corpus and the dictionary. Furthermore, we intend to explore hybrid models for this linkage task by combining the best transductive models with the strong memorizing baseline. From our error analysis it is clear that with sufficient baseform-linked (labeled) data most of the remaining unlinked stems are known and unambiguous, so incorporating a memorization module in the hybrid would result in high baseline accuracy. Moreover, known ambigu-

	KSU Acc.	KSU Correct	KLA Acc.	KLA Correct	KSA Acc.	KSA Correct
Total Count	N/A	$7050\pm101$	N/A	$61 \pm 40$	N/A	$1297\pm55$
Baseline	$100.00\pm0.00$	$\textbf{7050} \pm \textbf{101}$	$51.10 \pm 24.97$	$39 \pm 42$	$90.70 \pm 0.98$	$1176\pm57$
Morfette-L	$99.04 \pm 0.21$	$6983 \pm 100$	$53.59 \pm 23.29$	$40 \pm 42$	$90.17 \pm 0.93$	$1170\pm57$
DirecTL+-L	$99.40 \pm 0.08$	$7008\pm99$	$55.41 \pm 22.51$	$41 \pm 42$	$93.48 \pm 0.86$	$1212 \pm 56$
	KDA Acc.	KDA Correct	UU Acc.	UU Correct	UA Acc.	UA Correct
Total Count	N/A	$3.4 \pm 2.7$	N/A	$344 \pm 20$	N/A	$1.3\pm1.1$
Baseline	$50.44 \pm 15.60$	$1.8\pm1.7$	$23.03 \pm 4.00$	$79 \pm 14$	$27.77 \pm 31.03$	$0.40\pm0.70$
Morfette-L	$46.87 \pm 23.34$	$1.7\pm1.8$	$60.41 \pm 2.64$	$208 \pm 14$	$36.11 \pm 28.70$	$0.50 \pm 0.71$
DirecTL+-L	$53.77 \pm 17.72$	$1.9\pm1.7$	$59.47 \pm 2.61$	$205\pm15$	$27.77 \pm 31.03$	$0.40\pm0.70$

Table 6: Mean and std. dev. values for accuracy and related counts with 10-fold cross validation. Measures are per stem token. Maximum mean values bolded per error case. The unknown unambiguous values for Morfette-L are also bolded as it is the most stable (i.e., has high mean and small variance).

ous stems and unknown stems can be handled in the hybrid model by a discriminative model (or models) using local context. As the human annotators work through the corpus, many stems will be exposed as ambiguous or more ambiguous given novel baseform links from the human annotators. To assist the annotators in identifying potential ambiguity, the hybrid model will also need a component model to predict novel baseforms not currently in the labeled data, or at least not currently paired with a particular stem type, so as to detect latent and dynamic stem ambiguity.

We are also intrigued by the root-and-pattern features of dkrMorph and look forward to exploring those features further in the discriminative framework.

To provide machine assistance for corpus-dictionary linkage, we will combine the hybrid stem-baseform linker model with the SyroMorph segmentation model, likewise adapted to operate in an iterative transductive fashion, and a model to disambiguate baseforms among dictionary entries with homographic headwords. This pipeline will then be used in a live application to assist Syriac scholars link each word in the corpus of Ephrem the Syrian's writings to Jessie Payne Smith's Compendious Syriac Dictionary.

#### 8. References

- Carmen, M., Felt, P., Haertel, R., Lonsdale, D., McClanahan, P., Merkling, O., Ringger, E., and Seppi, K. (2010).
  Tag dictionaries accelerate manual annotation. In *Proceedings of LREC-10*, Valletta, Malta. ELRA.
- Chrupała, G., Dinu, G., and van Genabith, J. (2008). Learning morphology with Morfette. In *Proceedings of LREC-08*, Marrakech, Morocco. ELRA.
- Collins, M. (2002). Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of the EMNLP-*02, pages 1–8, Stroudsburg, PA. ACL.
- Felt, P., Ringger, E. K., Seppi, K., Heal, K. S., Haertel, R. A., and Lonsdale, D. (2013). Evaluating machineassisted annotation in under-resourced settings. *Lan*guage Resources and Evaluation.
- Habash, N. and Rambow, O. (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of ACL-05*, pages 573–580, Stroudsburg, PA. ACL.

Jiampojamarn, S., Kondrak, G., and Sherif, T. (2007).

Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *Proceeding of NAACL-07*, pages 372–379, Rochester, NY. ACL.

- Jiampojamarn, S., Cherry, C., and Kondrak, G. (2008). Joint processing and discriminative training for letter-tophoneme conversion. In *Proceedings of ACL-08: HLT*, pages 905–913, Columbus, OH. ACL.
- Jiampojamarn, S., Bhargava, A., Dou, Q., Dwyer, K., and Kondrak, G. (2009). DirecTL: a language independent approach to transliteration. In *Proceedings of NEWS-09*, pages 28–31, Suntec, Singapore. ACL.
- Jiampojamarn, S., Cherry, C., and Kondrak, G. (2010). Integrating joint n-gram features into a discriminative training framework. In *Proceedings of NAACL-10*, pages 697–700, Los Angeles, CA. ACL.
- Kilgarriff, A. and Rundell, M. (2002). Lexical profiling software and its lexicographic applications - a case study. In *Proceedings of EURALEX-02*, pages 807–818, København, Denmark. Center for Sprogteknologi.
- Kilgarriff, A. (2005). Linking dictionary and corpus. In Proceedings of ASIALEX-05, Singapore.
- Kiraz, G. A. (1994). Automatic concordance generation of Syriac texts. *Orientalia Christiana Analecta*, 247:461– 475.
- Kiraz, G. A. (2000). Multitiered nonlinear morphology using multitape finite automata: A case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105.
- Lindgren, L. (2011). dkrMorph: A Syriac morphological analyzer. Bachelor's thesis, Uppsala University, Department of Informatics and Media.
- McClanahan, P., Busby, G., Haertel, R., Heal, K., Lonsdale, D., Seppi, K., and Ringger, E. (2010). A probabilistic morphological analyzer for Syriac. In *Proceedings of EMNLP-10*, pages 810–820, Cambridge, MA. ACL.
- Payne Smith, J. (1903). A compendious Syriac dictionary founded upon the Thesaurus Syriacus of R. Payne Smith.
- Ringger, E., Carmen, M., Haertel, R., Seppi, K., Lonsdale, D., McClanahan, P., Carroll, J., and Ellison, N. (2008). Assessing the costs of machine-assisted corpus annotation through a user study. In *Proceedings of LREC-08*, pages 3318–3324, Marrakech, Morocco. ELRA.