Reversing Morphological Tokenization in English-to-Arabic SMT

Mohammad Salameh[†] **Colin Cherry**[‡] **Grzegorz Kondrak**[†]

[†]Department of Computing Science
University of Alberta
Edmonton, AB, T6G 2E8, Canada
{msalameh,gkondrak}@ualberta.ca

[‡]National Research Council Canada
1200 Montreal Road
Ottawa, ON, K1A 0R6, Canada
Colin.Cherry@nrc-cnrc.gc.ca

Abstract

Morphological tokenization has been used in machine translation for morphologically complex languages to reduce lexical sparsity. Unfortunately, when translating into a morphologically complex language, recombining segmented tokens to generate original word forms is not a trivial task, due to morphological, phonological and orthographic adjustments that occur during tokenization. We review a number of detokenization schemes for Arabic, such as rule-based and table-based approaches and show their limitations. We then propose a novel detokenization scheme that uses a character-level discriminative string transducer to predict the original form of a segmented word. In a comparison to a state-of-the-art approach, we demonstrate slightly better detokenization error rates, without the need for any hand-crafted rules. We also demonstrate the effectiveness of our approach in an English-to-Arabic translation task.

1 Introduction

Statistical machine translation (SMT) relies on tokenization to split sentences into meaningful units for easy processing. For morphologically complex languages, such as Arabic or Turkish, this may involve splitting words into morphemes. Throughout this paper, we adopt the definition of tokenization proposed by Habash (2010), which incorporates both morphological segmentation as well as orthographic character transformations. To use an English example, the word *tries* would be morphologically tokenized as *"try + s"*, which involves orthographic changes at morpheme boundaries to match the lexical form of each token. When translating into a tokenized language, the tokenization must be reversed to make the generated text readable and evaluable. Detokenization is the process of converting tokenized words into their original orthographically and morphologically correct surface form. This includes concatenating tokens into complete words and reversing any character transformations that may have taken place.

For languages like Arabic, tokenization can facilitate SMT by reducing lexical sparsity. Figure 1 shows how the morphological tokenization of the Arabic word وسيمنعهم "and he will prevent them" simplifies the correspondence between Arabic and English tokens, which in turn can improve the quality of word alignment, rule extraction and decoding. When translating from Arabic into English, the tokenization is a form of preprocessing, and the output translation is readable, space-separated English. However, when translating from English to Arabic, the output will be in a tokenized form, which cannot be compared to the original reference without detokenization. Simply concatenating the tokenized morphemes cannot fully reverse this process, because of character transformations that occurred during tokenization.

The techniques that have been proposed for the detokenization task fall into three categories (Badr et al., 2008). The simplest detokenization approach concatenates morphemes based on token markers without any adjustment. Table-based detokenization maps tokenized words into their surface form with a look-up table built by observing the tokenizer's in-

Proceedings of the NAACL HLT 2013 Student Research Workshop, pages 47–53,
Atlanta, Georgia, 13 June 2013.

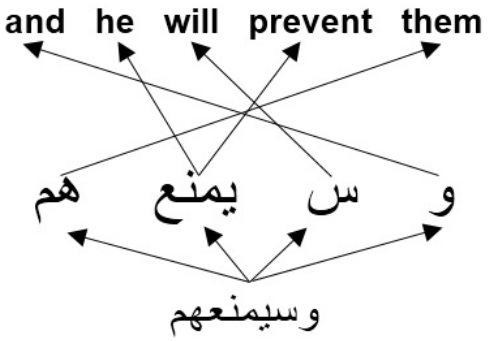

Figure 1: Alignment between tokenized form of *"wsymnςhm"* وسيمنعهم and its English translation.

put and output on large amounts of text. Rule-based detokenization relies on hand-built rules or regular expressions to convert the segmented form into the original surface form. Other techniques use combinations of these approaches. Each approach has its limitations: rule-based approaches are language specific and brittle, while table-based approaches fail to deal with sequences outside of their tables.

We present a new detokenization approach that applies a discriminative sequence model to predict the original form of the tokenized word. Like table-based approaches, our sequence model requires large amounts of tokenizer input-output pairs; but instead of building a table, we use these pairs as training data. By using features that consider large windows of within-word input context, we are able to intelligently transition between rule-like and table-like behavior.

Our experimental results on Arabic text demonstrate an improvement in terms of sentence error rate[1] of 11.9 points over a rule-based approach, and 1.1 points over a table-based approach that backs off to rules. More importantly, we achieve a slight improvement over the state-of-the-art approach of El Kholy and Habash (2012), which combines rules and tables, using a 5-gram language model to disambiguate conflicting table entries. In addition, our detokenization method results in a small BLEU improvement over a rule-based approach when applied to English-to-Arabic SMT.

[1]Sentence error rate is the percentage of sentences containing at least one error after detokenization.

2 Arabic Morphology

Compared to English, Arabic has rich and complex morphology. Arabic base words inflect to eight features. Verbs inflect for aspect, mood, person and voice. Nouns and adjectives inflect for case and state. Verbs, nouns and adjectives inflect for both gender and number. Furthermore, inflected base words can attract various optional clitics. Clitical prefixes include determiners, particle proclitics, conjunctions and question particles in strict order. Clitical suffixes include pronominal modifiers. As a result of clitic attachment, morpho-syntactic interactions sometimes cause changes in spelling or pronunciations.

Several tokenization schemes can be defined for Arabic, depending on the clitical level that the tokenization is applied to. In this paper, we use Penn Arabic Treebank (PATB) tokenization scheme, which El Kholy and Habash (2012) report as producing the best results for Arabic SMT. The PATB scheme detaches all clitics except for the definite article *Al* ال. Multiple prefix clitics are treated as one token.

Some Arabic letters present further ambiguity in text.[2] For example, the different forms of Hamzated Alif "أ إ آ" are usually written without the Hamza "ء". Likewise, when the letter Ya *'Y'* ي is present at the end of the word, it is sometimes written in the form of "Alif Maqsura" letter *'ý'* ى. Also, short vowels in Arabic are represented using diacritics, which are usually absent in written text. In order to deal with these ambiguities in SMT, normalization is often performed as a preprocessing step, which usually involves converting different forms of Alif and Ya to a single form. This decreases Arabic's lexical sparsity and improves SMT performance.

3 Related Work

Sadat and Habash (2006) address the issue of lexical sparsity by presenting different preprocessing schemes for Arabic-to-English SMT. The schemes include simple tokenization, orthographic normalization, and decliticization. The combination of these schemes results in improved translation out-

[2]We use Habash-Soudi-Buckwalter transliteration scheme (Habash, 2007) for all Arabic examples.

put. This is one of many studies on normalization and tokenization for translation from Arabic, which we will not attempt to review completely here.

Badr et al. (2008) show that tokenizing Arabic also has a positive influence on English-to-Arabic SMT. They apply two tokenization schemes on Arabic text, and introduce detokenization schemes through a rule-based approach, a table-based approach, and a combination of both. The combination approach detokenizes words first using the table, falling back on rules for sequences not found in the table.

El Kholy and Habash (2012) extend Badr's work by presenting a larger number of tokenization and detokenization schemes, and comparing their effects on SMT. They introduce an additional detokenization schemes based on the SRILM *disambig* utility (Stolcke, 2002), which utilizes a 5-gram untokenized language model to decide among different alternatives found in the table. They test their schemes on naturally occurring Arabic text and SMT output. Their newly introduced detokenization scheme outperforms the rule-based and table-based approaches introduced by Badr et al. (2008), establishing the current state-of-the-art.

3.1 Detokenization Schemes in Detail

Rule-based detokenization involves manually defining a set of transformation rules to convert a sequence of segmented tokens into their surface form. For example, the noun "*llrŷys*" للرئيس "to the president" is tokenized as *"l+ Alrŷys"* (*l+* "to" *Alrŷys* "the president") in the PATB tokenization scheme. Note that the definite article "*Al*" ال is kept attached to the noun. In this case, detokenization requires a character-level transformation after concatenation, which we can generalize using the rule:

$$l{+}Al \rightarrow ll.$$

Table 1 shows the rules provided by El Kholy and Habash (2012), which we employ throughout this paper.

There are two principal problems with the rule-based approach. First, rules fail to account for unusual cases. For example, the above rule mishandles cases where "*Al*" ال is a basic part of the stem and not the definite article *"the"*. Thus, 'l+ AlςAb' (*l+* "to" *Alς Ab* "games") is erroneously detokenized to *llEAb* للعاب instead of the correct form is "*lAlςAb*" لالعاب. Second, rules may fail to handle sequences produced by tokenization errors. For example, the word "*bslTħ*" بسلطة "with power" can be erroneously tokenized as *"b+slT+h"*, while the correct tokenizations is *"b+slTħ"*. The erroneous tokenization will be incorrectly detokenized as *"bslTh"*.

Rule	Input	Output
l+Al+l? → ll	l+ Alrŷys	llrŷys
ħ+(*pron*) → t(*pron*)	Abnħ+hA	AbnthA
y+(*pron*) → A(*pron*)	Alqy+h	AlqAh
'+(*pron*) → ŷ	AntmA'+hm	AntmAŷhm
y+y → y	ςyny+y	ςyny
n+n → n	mn+nA	mnA
mn+m → mm	mn+mA	mmA
ςn+m → ςm	ςn+mA	ςmA
An+lA → AlA	An+lA	AlA

Table 1: Detokenization rules of El Kholy and Habash (2012), with examples. *pron* stands for pronominal clitic.

The table-based approach memorizes mappings between words and their tokenized form. Such a table is easily constructed by running the tokenizer on a large amount of Arabic text, and observing the input and output. The detokenization process consults this table to retrieve surface forms of tokenized words. In the case where a tokenized word has several observed surface forms, the most frequent form is selected. This approach fails when the sequence of tokenized words is not in the table. In morphologically complex languages like Arabic, an inflected base word can attrract many optional clitics, and tables may not include all different forms and inflections of a word.

The SRILM-disambig scheme introduced by El Kholy and Habash (2012) extends the table-based approach to use an untokenized Arabic language model to disambiguate among the different alternatives. Hence, this scheme can make context-dependent detokenization decisions, rather than always producing the most frequent surface form. Both the SRILM-disambig scheme and the table-based scheme have the option to fall back on either rules or simple concatenation for sequences missing from the table.

4 Detokenization as String Transduction

We propose to approach detokenization as a string transduction task. We train a discriminative transducer on a set of tokenized-detokenized word pairs. The set of pairs is initially aligned on the character level, and the alignment pairs become the operations that are applied during transduction. For detokenization, most operations simply copy over characters, but more complex rules such as *l+ Al* $\rightarrow$ *ll* are learned from the training data as well.

The tool that we use to perform the transduction is DIRECTL+, a discriminative, character-level string transducer, which was originally designed for letter-to-phoneme conversion (Jiampojamarn et al., 2008). To align the characters in each training example. DIRECTL+ uses an EM-based M2M-ALIGNER (Jiampojamarn et al., 2007). After alignment is complete, MIRA training repeatedly decodes the training set to tune the features that determine when each operation should be applied. The features include both n-gram source context and HMM-style target transitions. DIRECTL+ employs a fully discriminative decoder to learn character transformations and when they should be applied. The decoder resembles a monotone phrase-based SMT decoder, but is built to allow for hundreds of thousands of features.

The following example illustrates how string transduction applies to detokenization. The segmented and surface forms of *bbrAςthm* ببراعتهم "with their skill" constitute a training instance:

$$b+\_brA\varsigma\hbar\_+hm \rightarrow bbrA\varsigma thm$$

The instance is aligned during the training phase as:

b+	_b	r	A	ς	ħ_	+	h	m
|	|	|	|	|	|	|	|	|
b	b	r	A	ς	t	ϵ	h	m

The underscore "_" indicates a space, while "ϵ" denotes an empty string. The following operations are extracted from the alignment:

b+ $\rightarrow$ b, _b $\rightarrow$ b, r $\rightarrow$ r, A $\rightarrow$ A, E $\rightarrow$ E, p_ $\rightarrow$ t,
+ $\rightarrow \epsilon$, h $\rightarrow$ h, m $\rightarrow$ m

During training, weights are assigned to features that associate operations with context. In our running example, the weight assigned to the b+ $\rightarrow$ b operation accounts for the operation itself, for the fact that the operation appears at the beginning of a word, and for the fact that it is followed by an underscore; in fact, we employ a context window of 5 characters to the left or right of the source substring "b+", creating a feature for each n-gram within that window.

Modeling the tokenization problem as string transduction has several advantages. The approach is completely language-independent. The context-sensitive rules are learned automatically from examples, without human intervention. The rules and features can be represented in a more compact way than the full mapping table required by table-based approaches, while still elegantly handling words that were not seen during training. Also, since the training data is generalized more efficiently than in simple memorization of complete tokenized-detokenized pairs, less training data should be needed to achieve good accuracy.

5 Experiments

This section presents two experiments that evaluate the effect of the detokenization schemes on both naturally occurring Arabic and SMT output.

5.1 Data

To build our data-driven detokenizers, we use the Arabic part of 4 Arabic-English parallel datasets from the Linguistic Data Consortium as training data. The data sets are: Arabic News (LDC2004T17), eTIRR (LDC2004E72), English translation of Arabic Treebank (LDC2005E46), and Ummah (LDC2004T18). The training data has 107K sentences. The Arabic part of the training data constitutes around 2.8 million words, 3.3 million tokens after tokenization, and 122K word types after filtering punctuation marks, Latin words and numbers (refer to Table 2 for detailed counts).

For training the SMT system's translation and reordering models, we use the same 4 datasets from LDC. We also use 200 Million words from LDC Arabic Gigaword corpus (LDC2011T11) to generate a 5-gram language model using SRILM toolkit (Stolcke, 2002).

We use NIST MT 2004 evaluation set for tuning (1075 sentences), and NIST MT 2005 evaluations set for testing (1056 sentences). Both MT04 and MT05 have multiple English references in order to evaluate Arabic-to-English translation. As we are translating into Arabic, we take the first English

Data set	Before	After
training set	122,720	61,943
MT04	8,201	2,542
MT05	7,719	2,429

Table 2: Type counts before and after tokenization.

translation to be our source in each case. We also use the Arabic halves of MT04 and MT05 as development and test sets for our experiments on naturally occurring Arabic. The tokenized Arabic is our input, with the original Arabic as our gold-standard detokenization.

The Arabic text of the training, development, testing set and language model are all tokenized using MADA 3.2 (Habash et al., 2009) with the Penn Arabic Treebank tokenization scheme. The English text in the parallel corpus is lower-cased and tokenized in the traditional sense to strip punctuation marks.

5.2 Experimental Setup

To train the detokenization systems, we generate a table of mappings from tokenized forms to surface forms based on the Arabic part of our 4 parallel datasets, giving us complete coverage of the output vocabulary of our SMT system. In the table-based approaches, if a tokenized form is mapped to more than one surface form, we use the most frequent surface form. For out-of-table words, we fall back on concatenation (in T) or rules (in T+R). For SRILM-Disambig detokenization, we maintain ambiguous table entries along with their frequencies, and we introduce a 5-gram language model to disambiguate detokenization choices in context. Like the table-based approaches, the Disambig approach can back off to either simple concatenation (T+LM) or rules (T+R+LM) for missing entries. The latter is a re-implementation of the state-of-the-art system presented by El Kholy and Habash (2012).

We train our discriminative string transducer using word types from the 4 LDC catalogs. We use M2M-ALIGNER to generate a 2-to-1 character alignments between tokenized forms and surface forms. For the decoder, we set Markov order to one, joint n-gram features to 5, n-gram size to 11, and context size to 5. This means the decoder can utilize contexts up to 11 characters long, allowing it to effectively memorize many words. We found these settings using grid search on the development set, NIST MT04.

Detokenization	WER	SER	BLEU
Baseline	1.710	34.3	26.30
Rules (R)	0.590	14.0	28.32
Table (T)	0.192	4.9	28.54
Table + Rules (T+R)	0.122	3.2	28.55
Disambig (T+LM)	0.164	4.1	28.53
Disambig (T+R+LM)	0.094	2.4	28.54
DIRECTL+	0.087	2.1	28.55

Table 3: Word and sentence error rate of detokenization schemes on the Arabic reference text of NIST MT05. BLEU score refers to English-Arabic SMT output.

For the SMT experiment, we use GIZA++ for the alignment between English and tokenized Arabic, and perform the translation using Moses phrase-based SMT system (Hoang et al., 2007), with a maximum phrase length of 5. We apply each detokenization scheme on the SMT tokenized Arabic output test set, and evaluate using the BLEU score (Papineni et al., 2002).

5.3 Results

Table 3 shows the performance of several detokenization schemes. For evaluation, we use the sentence and word error rates on naturally occurring Arabic text, and BLEU score on tokenized Arabic output of the SMT system. The baseline scheme, which is a simple concatenation of morphemes, introduces errors in over a third of all sentences. The table-based approach outperforms the rule-based approach, indicating that there are frequent exceptions to the rules in Table 1 that require memorization. Their combination (T+R) fares better, leveraging the strengths of both approaches. The addition of SRILM-Disambig produces further improvements as it uses a language model context to disambiguate the correct detokenized word form. Our system outperforms SRILM-Disambig by a very slight margin, indicating that the two systems are roughly equal. This is interesting, as it is able to do so by using only features derived from the tokenized word itself; unlike SRILM-Disambig, it has no access to the surrounding words to inform its decisions. In ad-

dition, it is able to achieve this level of performance without any manually constructed rules.

Improvements in detokenization do contribute to the BLEU score of our SMT system, but only to a point. Table 3 shows three tiers of performance, with no detokenization being the worst, the rules being better, and the various data-driven approaches performing best. After WER dips below 0.2, further improvements seem to no longer affect SMT quality. Note that BLEU scores are much lower overall than one would expect for the translation in the reverse direction, because of the morphological complexity of Arabic, and the use of one (as opposed to four) references for evaluation.

5.4 Analysis

The sentence error rate of 2.1 represents only 21 errors that our approach makes. Among those 21, 11 errors are caused by changing *p* to *h* and vice versa. This is due to writing *p* and *h* interchangeably. For example, *"AjmAly+h"* was detokenized as *"AjmAlyħ"* اجمالية instead of "AjmAlyh" اجماليه. Another 4 errors are caused by the lack of diacritization, which affects the choice of the Hamza form. For example,*"bnAŵh"* بناؤه, *"bnAŷh"* بنائه and *"bnA'h"* بناءه ("its building") are 3 different forms of the same word where the choice of Hamza ء is dependent on its diacritical mark or the mark of the character that precedes it. Another 3 errors are attributed to the case of the nominal which it inflects for. The case is affected by the context of the noun which DIRECTL+ has no access to. For example, *"mfkry+hm"* ("thinkers/Dual-Accusative") was detokenized as *"mfkrAhm"* مفكراهم (Dual-Nominative) instead of "mfkryhm" مفكريهم. The last 3 errors are special cases of *"An +y"* which can be detokenized correctly as either *"Any"* اني or *"Anny"* انني.

The table-based detokenization scheme fails in 54 cases. Among these instances, 44 cases are not in the mapping table, hence resolving back to simple concatenation ended with an error. Our transduction approach succeeds in detokenizing 42 cases out of the 54. The majority of these cases involves changing *p* to *h* and vice versa, and changing *l+Al* to *ll*. The only 2 instances where the tokenized word is in the mapping table but DIRECTL+ incorrectly detokenizes it are due to hamza case and p to h case described above. There are 4 instances of the same word/case where both the table scheme and DIRECTL+ fails due to error of tokenization by MADA, where the proper name *qwh* قوه is erroneously tokenized as *qw+p*. This shows that DIRECTL+ handles the OOV words correctly.

The Disambig(T+R+LM) erroneously detokenizes 27 instances, where 21 out of them are correctly tokenized by DIRECTL+. Most of the errors are due to the Hamza and p to h reasons. It seems that even with a large size language model, the SRILM utility needs a large mapping table to perform well. Only 4 instances were erroneously detokenized by both Disambig and DIRECTL+ due to Hamza and the case of the nominal.

The analysis shows that using small size training data, DIRECTL+ can achieve slightly better accuracy than SRILM scheme. The limitations of using table and rules are handled with DIRECTL+ as it is able to memorize more rules.

6 Conclusion and Future Work

In this paper, we addressed the detokenization problem for Arabic using DIRECTL+, a discriminative training model for string transduction. Our system performs the best among the available systems. It manages to solve problems caused by limitations of table-based and rule-based systems. This allows us to match the performance of the SRILM-disambig approach without using a language model or handcrafted rules. In the future, we plan to test our approach on other languages that have morphological characteristics similar to Arabic.

References

Ibrahim Badr, Rabih Zbib, and James Glass. 2008. Segmentation for English-to-Arabic statistical machine translation. In *Proceedings of ACL*, pages 153–156.

Ahmed El Kholy and Nizar Habash. 2012. Orthographic and morphological processing for English-Arabic statistical machine translation. *Machine Translation*, 26(1-2):25–45, March.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. Mada+tokan: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In *Proceedings of*

the Second International Conference on Arabic Language Resources and Tools.

Nizar Habash. 2007. Arabic morphological representations for machine translation. In *Arabic Computational Morphology: Knowledge-based and Empirical Methods*.

Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Hieu Hoang, Alexandra Birch, Chris Callison-burch, Richard Zens, Rwth Aachen, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Brooke Cowan, Wade Shen, Christine Moran, and Ondrej Bojar. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session*, pages 177–180.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and HMMs to letter-to-phoneme conversion. In *Proceedings of NAACL-HLT*, pages 372–379.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2008. Joint processing and discriminative training for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 905–913.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Fatiha Sadat and Nizar Habash. 2006. Combination of Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1–8.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Intl. Conf. Spoken Language Processing*, pages 901–904.