Investigating the Impact of Various Partial Diacritization Schemes on Arabic-English Statistical Machine Translation

Sawsan Alqahtani, Mahmoud Ghoneim, Mona Diab
{sawsanq, mghoneim, mtdiab}@gwu.edu
Department of Computer Science
The George Washington University, USA
Washington, DC 20052

Abstract

Most diacritics in Arabic represent short vowels. In Arabic orthography, such diacritics are considered optional. The absence of these diacritics naturally leads to significant word ambiguity to top the inherent ambiguity present in fully diacritized words. Word ambiguity is a significant impediment for machine translation. Despite the ambiguity presented by lack of diacritization, context helps ameliorate the situation. Identifying the appropriate amount of diacritic restoration to reduce word sense ambiguity in the context of machine translation is the object of this paper. Diacritic marks help reduce the number of possible lexical word choices assigned to a source word which leads to better quality translated sentences. We investigate a variety of (linguistically motivated) partial diacritization schemes that preserve some of the semantics that in essence complement the implicit contextual information present in the sentences. We also study the effect of training data size and report results on three standard test sets that represent a combination of different genres. The results show statistically significant improvements for some schemes compared to two baselines: text with no diacritics (the typical writing system adopted for Arabic) and text that is fully diacritized.

1 Introduction

Resolving natural language ambiguity is at the crux of the NLP enterprise. Ambiguity refers to the problem of possibly having different interpretations for different segments (words, phrases, etc.) of a sentence. Languages such as Arabic, Hebrew and Persian are typically written in a manner that exacerbates this ambiguity problem and increases the homograph rate by underspecifying some of the letters such as short vowels and consonantal gemination, which in turn increases the effect of having multiple interpretations for the same word. This renders text even more ambiguous than typically expected.

While context helps native speakers of the language resolve some of the ambiguity, context alone does not always produce adequate clarity for interpretation. The problem is further complicated in Arabic by the fact that there are no native speakers of Modern Standard Arabic (MSA), which is the language used in education and formal settings. Instead, speakers of Arabic converse in various dialects of Arabic which are at times starkly different from MSA.

One solution for this problem is diacritic restoration, or diacritization, which refers to rendering the underspecified diacritics explicit in the text. We investigate the problem of diacritization within the context of Arabic-to-English Statistical Machine Translation (SMT) system. We address the problem in MSA texts, the majority of which are underspecified for these diacritic marks. We focus here on the most prominent Arabic diacritics which are short vowels (i, u, a), the syllable boundary marker, known as sukoon (o), indefiniteness marker, known as nunation (F, K, N), and the consonantal doubling marker (gemination) known as shadda (∼)[1]. In this study, we aim to investigate what is the appropriate level and type of diacritic restoration that would have the biggest impact on natural language understanding as tested and evaluated via machine translation. Hence we experiment with various diacritization schemes based on lexical and/or syntactic information.

This current work is a follow on to the pilot work presented in (Diab et al., 2007). However it is different in the following respects: 1- we explore automatically diacritized data; 2- we define more schemes that target both lexical and/or syntactic properties of the Arabic language. 3- we test the robustness of our observations taking into consideration varying training size and cross genre evaluation.

2 Related Work

Automatic Arabic diacritization has been addressed thoroughly in (Zitouni et al., 2006; Elshafei et al., 2006; Nelken and Shieber, 2005; Habash and Rambow, 2007; Pasha et al., 2014; Maamouri et al., 2008). Full diacritization indicates rendering the text with all the most prominent diacritics, namely (a, i, u, o, ∼).[2] Initial efforts in automatic diacritization include rule-based approaches to add all diacritics in the texts (Debili and Achour, 1998; El-Imam, 2004); however, it is expensive to maintain these rules to be generalized for unseen instances.

Most studies focused on full diacritic restoration. For Automatic Speech Recognition (ASR), (Vergyri and Kirchhoff, 2004; Ananthakrishnan et al., 2005) perform full diacritization on MSA speech transcripts for language modeling. They show that developing ASR models on fully diacritized datasets improves performance significantly. Supervised classifiers such as Hidden Markov Model (HMM) and Maximum Entropy (MaxEnt) have been employed for diacritization (Gal, 2002; Bebah et al., 2014; Zitouni and Sarikaya, 2009; Zitouni et al., 2006). In a study conducted by Ananthakrishnan et al. (2005), the researchers use MaxEnt trained on MSA with lexical and n-gram features to improve ASR. Another study uses decision trees and stochastic language models to fully diacritize texts in order to render graphemes to synthesized speech (Cherif et al., 2015). The Buckwalter Arabic Morphological Analysis (BAMA) (Buckwalter, 2002) system has been used along with a single tagger or a language model to select amongst the diacritized analyses in context to render text fully diacritized (Vergyri and Kirchhoff, 2004; Ananthakrishnan et al., 2005).

In Marton et al. (2010), the authors show that some inflectional and lexical related morphological features improve the performance of syntactic parsing in Arabic. Although Marton et al. (2010) have not used diacritics directly in their work, they use the same essential information that is used to diacritize Arabic texts. Diab et al. (2007) not only investigate the impact of full diacritization on Statistical Machine Translation (SMT) but also introduce the notion of partial diacritization. They also show that several schemes have a small positive effect albeit not significant on SMT performance over none and full diacritization despite the significant increase in the number of types. Although the results in Diab et al. (2007) are not statistically significant, they provide directions of research that we can exploit to increase the performance of Arabic related NLP applications. In a study conducted by AlHanai and Glass (2014), three partial diacritic schemes have been defined and compared to both fully and non-diacritized versions of the

[1] We use Buckwalter Transliteration encoding: http://www.qamus.org/transliteration.htm

[2] In some studies such as (Habash and Rambow, 2007; Pasha et al., 2014), they also address hamza restoration.

words. In their study, it is found that fully-diacritized text without gemination have statistically better performance than fully diacritized texts including gemination in ASR application. Our work follows the same general procedure as (Diab et al., 2007; AlHanai and Glass, 2014) where we study the impact of some aspects of diacritization information in NLP applications, SMT in particular.

For Arabic reading comprehension, Hermena et al. (2015) studies the impact of partial diacritics in improving Arabic speakers' reading comprehension. Their study shows the effectiveness of having some level of diacritization between none and fully diacritized forms that help the readers disambiguate homographs that cannot be understood by the surrounding contexts. This shows the importance of having accurate automatic partial diacritization not only in improving different NLP applications but also to diacritize texts to help readers understand Arabic texts better. Having the goal of helping other researchers develop partial diacritization, Bouamor et al. (2015) has conducted a pilot study that minimally diacritize the dataset to reduce lexical ambiguity and help generate models to find an optimal level of diacritization in some NLP applications. Although the result of this minimally-diacritized annotation has been highly affected by the annotators' subjectivity and background, it has shown some promising results for future studies.

The idea of integrating Word Sense Disambiguation (WSD) technologies into the SMT framework has been studied previously, tackling different aspects of the phenomenon and showing statistically significant improvement integrating explicit WSD into the SMT system (Chan et al., 2007; Carpuat and Wu, 2007; Yang and Kirchhoff, 2012; Aminian et al., 2015). Mainly, WSD integration improves the ability of the system to choose the target translation if it has been incorporated efficiently. Carpuat and Wu (2007) show an improvement in Chinese-to-English SMT system in eight different automatic evaluation metrics when they integrate WSD in their translation system at decode time. They use the same parallel corpus used for training and the phrase translation table generated by the SMT tool to disambiguate senses of the words by using the aligned phrases in the target language. All of the previous work incorporates features that help disambiguate senses in a supervised or unsupervised manner to generate better quality translation. Some of these studies change the SMT pipeline to integrate WSD but others implement it as a pre-processing step at decode time. In this study, we have the same goal as theirs which is to appropriately select the correct sense of a target word at decode time. We implement this by adding a certain amount of diacritics in Arabic as preprocessing in the data preparation step. Thus, the translation quality is not only enhanced by the appropriate choice of target word but also by the fact that the word alignment procedure is improved.

3 Scheme Extraction

We investigate the impact of various partial diacritization schemes on SMT application. We compare their performance against two baselines, specifically FULL diacritization where all the diacritics are present and NONE where no diacritics are present. Similar to the extraction strategy of (Diab et al., 2007; AlHanai and Glass, 2014), each of these schemes is identified from fully diacritized Arabic datasets. Additionally, the extraction process of some schemes involves the full morphological analysis of the words' part of speech and their lemmas. To identify these morphological features, we use MADAMIRA, a morphological analyzer and disambiguator for the Arabic language (Pasha et al., 2014). The quality of diacritization schemes rely on the performance of the automatic diacritization to predict diacritics. It is important to note that we rely on the underlying diacritized lemma form for ensuring extraction accuracy.

(Diab et al., 2007) define six different diacritization schemes based on their usage prominence in the Arabic Treebank (ATB) (Maamouri et al., 2008). Namely, they are fully diacritized (FULL), passive voice diacritic marks (PASS), consonant doubling or gemination (GEM), presence of the syllable boundary marker sukoon (SUK), syntactic case and mood diacritics (CM), and the case of no diacritization (NONE). In this study, we adopt the same previously mentioned schemes in addition to introducing several new ones: FULL-CM, PASS+CM, PASS+GEM, SUK+GEM, PASS+SUK, PASS+SUK+GEM, FULL-CM-PASS, TANWEEN.[3] The following is a detailed explanation of these diacritic schemes.

The schemes are linguistically-motivated reflecting lexical, syntactic, or both types of information. The Arabic sentences are written in Buckwalter Transliteration[4] and are tokenized according to the ATB style (Arabic TreeBank Tokenization). It is crucial to note that if the word is not affected by the defined diacritic pattern, we remove all of its diacritics (i.e. NONE scheme).

Baselines: NONE: indicates that no diacritics are kept at all in the sentence, including the removal of the naturally occurring diacritics.

e.g. w+ ADAft An Tbyb AEln wfAp AlmEtql , bEd An HAwl AtxA* kl AlAjrA'At AllAzmp l+ AnqA* +h .

FULL: indicates that all diacritics are kept in the sentence.

e.g. wa+ AaDAfat Aan∼a TabiybAF AaEolana wafApu AlmuEotaqalu , baEoda Aano HAwala Ait∼ixA*a kul∼i AlAijorA'Ati All∼Azimapi li+ AinoqA*i +hu .

Singleton Schemes (Lexical): SUK: is an explicit marking of the absence of a short vowel typically between syllables, known as sukoon. We keep sukoon in the word whenever it is present in the underlying lemma.

e.g. w+ ADAft An Tbyb AEoln wfAp AlmEotql , bEod Ano HAwl AtxA* kl AlAjorA'At AllAzmp l+ AnoqA* +h .

GEM: renders explicit the doubling of consonants (shaddah or gemination) whenever the shaddah is present in the underlying lemma of the word.

e.g. w+ ADAft An∼ Tbyb AEln wfAp AlmEtql , bEd An HAwl At∼xA* kl∼ AlAjrA'At AllAzmp l+ AnqA* +h .

Singleton Schemes (Inflectional): CM (Case and Mood): reflects syntactic case and mood on nominals and verbs, respectively. We keep the last diacritic marker whenever the part of speech explicitly indicates the presence of case or mood.

e.g. w+ ADAft An TbybAF AEln wfApu AlmEtqlu , bEda An HAwl AtxA*a kl∼i AlAjrA'Ati AllAzmpi l+ AnqA*i +h .

PASS (passivization): indicates that the diacritic(s) reflecting passive voice on verbs are the only markers kept.

e.g. w+ qAlt AlqyAdp Aljnwbyp b+ myAmy fy byAn , An jvmAn Almtwfy s+ yurAEay wfq AltqAlyd w+ AlAErAf Aldynyp , Alty yntmy l +hA .

Singleton Schemes (Both): TANWEEN: reflects syntactic case and indefiniteness on nominals. We keep all tanween marks (K, F, N).

e.g. w+ ADAft An TbybAF AEln wfAp AlmEtql , bEd An HAwl AtxA* kl AlAjrA'At AllAzmp l+ AnqA* +h .

Combined Schemes (Lexical): SUK+GEM: Combines SUK and GEM diacritic schemes.

e.g. w+ ADAft An∼ Tbyb AEoln wfAp AlmEotql , bEod Ano HAwl At∼xA* kl∼ AlAjorA'At AllAzmp l+ AnoqA* +h .

[3] Naming Convention: Similar to the mathematical operations, the symbol [-] indicates that we remove the diacritics of the scheme presented after the symbol from the scheme presented before the symbol. The symbol [+] indicates that we add the diacritic scheme for each of these schemes to define a new one.

[4] More description can be found in http://www.qamus.org/transliteration.htm

FULL-CM-PASS: indicates that all diacritics are kept in the word except the syntactic level diacritics.

e.g. wa+ qAlat AlqiyAdap Aljanuwbiy∼ap bi+ mayAmiy fiy bayAn , Aino juvomAn Almutawaf∼ay sa+ yrAEy wifoq Alt∼aqAliyd wa+ AlAaEorAf Ald∼iyniy∼ap , Al∼atiy yanotamiy li +hA .

Combined Schemes (Inflectional): PASS+CM: combines the properties of PASS and CM schemes.

e.g. w+ qAlt AlqyAdpu Aljnwbypu b+ myAmy fy byAnK , An jvmAnu Almtwfy s+ yurAEay wfqa AltqAlydi w+ AlAErAfi Aldynypi , Alty yntmy l +hA .

Combined Schemes (Both): FULL-CM: the same as FULL but we remove the CM related diacritics from the word.

e.g. wa+ AaDAfat Aan∼a Tabiyb AaEolana wafAp AlmuEotaqal , baEod Aano HAwala Ait∼ixAV kul AlAijorACAt All∼Azimap li+ AinoqAV +hu .

PASS+GEM: Combines the features of PASS and GEM schemes.

e.g. w+ qAlt AlqyAdp Aljnwby∼p b+ myAmy fy byAn , An jvmAn Almtwf∼y s+ yurAEy wfq AltqAlyd w+ AlAErAf Aldyny∼p , Al∼ty yntmy l +hA .

PASS+SUK: Combines the features of PASS and SUK schemes.

e.g. w+ qAlt AlqyAdp Aljnwbyp b+ myAmy fy byAn , Ano jvomAn Almtwfy s+ yurAEay wfoq AltqAlyd w+ AlAErAf Aldynyp , Alty ynotmy l +hA .

PASS+SUK+GEM: Combines PASS, SUK, and GEM schemes.

e.g. w+ qAlt AlqyAdp Aljnwby∼p b+ myAmy fy byAn , Ano jvomAn Almtwf∼y s+ yurAEay wfoq AltqAlyd w+ AlAErAf Aldyny∼p , Al∼ty ynotmy l +hA .

As indicated previously, the goal of the various schemes is to reduce the number of possible choices for translating a sentence by distinguishing meanings at the word level which in turn affect the phrase level. For example, the word 'bEd' can be understood as baEod (Adv), buEod (Noun), baEuda (Verb), baEida (Verb), buEida (Verb-Passive), baE∼ada (Verb), buE∼ida (Verb-Passive) and biEad∼i (Prep+Noun) which means "after;post, yet", "dimension; distance; remoteness", "became distant (Aspect:State)", "became distant (Aspect:Action)", "kept distant to something", "displace; exclude; alienate", "became displaced; excluded; alienated", "by counting", respectively. The diacritic schemes defined here segment the space in different ways to reduce some aspects of this ambiguity. For example, SUK segments the space into two subgroups where both 'baEod' and 'buEod' share the form 'bEod' while others remain 'bEd'. Similarly, GEM creates three segments. It uniquely identifies 'biEad∼i' as 'bEd∼' solving its ambiguity and groups 'baE∼ada' and 'buE∼ida' as 'bE∼d' (still ambiguous, although to a lesser extent) and the remaining five words as 'bEd'. Adding PASS to GEM will solve the ambiguity regarding 'baE∼ada' and 'buE∼ida' (become 'bE∼d' and 'buE∼id', respectively).

While FULL solves all ambiguity, it actually over specifies every word by including Case and Mood which in turn increases data sparsity. For example the word 'buEod' can take the following forms 'buEoda', 'buEodi', 'buEodu', 'buEodK', 'buEodN', 'buEod' according to the FULL scheme. FULL-CM would decrease this sparseness but again there is still some redundancy (e.g. 'baEod' can take the forms 'baEodu' 'baEodi' 'baEoda' as the last diacritic is neither a Case nor a Mood marker). The ability of these schemes to disambiguate is sensitive to the genre of the text where some variations might not appear or become rarely used. Also sparsity would be a limiting factor for small training data sizes. The objective of this study is to explore the appropriate level of diacritization information that reduces ambiguity to practical levels within the context of SMT. As it is hard to define these practical levels, we report results on different training data sizes and using test sets that exhibit different combinations of genres. It is important to note that this study does not aim at distinguishing the different possible senses of the word with diacritics specified (fully diacritized). To illustrate, the word 'buEod' may mean dimension, distance or remoteness; this level of disambiguation cannot be addressed using diacritics alone but depends on context which is not addressed directly by this work, but is assumed to be taken care of through the SMT pipeline as a whole.

4 Experimental Setup

4.1 Dataset

To train the SMT model, we use an Arabic-English parallel dataset which includes 60M tokens and is created from 53 LDC (Linguistic Data Consortium) catalogs. This dataset includes multiple genres such as newswire, broadcast news, broadcast conversations, newsgroups, and weblogs. We use three different test datasets from multiple genres: NIST 2009, 2006, and 2005 Open Machine Translation Evaluation,[5] which correspond to MT09, MT06, and MT05, respectively. MT09 is 41,640 tokens from weblogs and newswires; MT06 consists of 49,154 tokens from newswire, broadcast news, and weblogs; MT05 consists of 33,407 tokens from newswire. We use NIST 2008 Open Machine Translation Evaluation (MT08),[6] which is 45,555 tokens from newswire and web collection genres, for tuning. It is important to note that the number of types varies across diacritic schemes as opposed to the number of tokens which is consistent for all schemes. Table 1 shows the number of types for each of the train and test datasets associated with each diacritic scheme.

For both Arabic and English datasets, we separate punctuations and numbers from the text and convert them to standard forms (PUNC and NUM) in order to reduce the number of types and errors to some extent. We use the morphological analyzer toolkit, MADAMIRA (Pasha et al., 2014) to tokenize the Arabic side of the parallel dataset according to Arabic Treebank tokenization (ATB) style (Maamouri et al., 2004). All diacritic patterns have the same exact preprocessing; the only difference is the number and the type of diacritics added to the dataset. For the English side of the parallel dataset, we tokenize the dataset using Tree Tagger (Schmid, 1995) and lowercase all letters.

Diacritic Pattern	No. of Types	Type Increase %	No. of Types	Type Increase %
	(Train)	(Train)	(MT09/MT06/MT05)	(MT09/MT06/MT05)
NONE	303,049	-	8,562 / 9,205 / 6,128	-
FULL	432,832	42.83	11,072 / 12,027 / 7,966	29.32 / 30.66 / 29.99
SUK	306,648	1.19	8,644 / 9,324 / 6,186	0.96 / 1.29 / 0.95
GEM	308,424	1.77	8,638 / 9,312 / 6,175	0.89 / 1.16 / 0.77
CM	414,615	36.81	10,936 / 11,845 / 7,868	27.73 / 28.68 / 28.39
PASS	306,003	0.97	8,603 / 9,237 / 6,155	0.48 / 0.35 / 0.44
TANWEEN	342,025	12.86	9,363 / 10,134 / 6,720	9.36 / 10.09 / 9.66
SUK+GEM	311,024	2.63	8,702 / 9,400 / 6,220	1.64 / 2.12 / 1.50
FULL-CM-PASS	329,123	8.60	8,912 / 9,611 / 6,337	4.09 / 4.41 / 3.41
PASS+CM	417,876	37.89	10,969 / 11,869 / 7,892	28.11 / 28.94 / 28.79
FULL-CM	329,632	8.77	8,939 / 9,652 / 6,359	4.40 / 4.86 / 3.77
PASS+GEM	311,202	2.69	8,676 / 9,344 / 6,201	1.33 / 1.51 / 1.19
PASS+SUK	309,499	2.13	8,683 / 9,353 / 6,211	1.41 /1.61 / 1.35
PASS+SUK+GEM	313,788	3.54	8,739 / 9,429 / 6,245	2.07 / 2.43 / 1.91

Table 1: This table shows the number of types for each diacritic scheme for test and train datasets. Type Increase columns indicate the percentage of increase in the number of types compared to NONE.

[5] Catalog Numbers: LDC2010T23 (MT09), LDC2010T17 (MT06), LDC2010T14 (MT05).

[6] Catalog Number: LDC2010T21.

4.2 SMT System

We train standard phrase-based SMT system using Moses toolkit version 2.1.1 (Koehn et al., 2007). The parallel corpus is word aligned using GIZA++ (Och and Ney, 2003) with a maximum sentence length of 250 words. The phrase tables contains up to 8-words phrases. We use SRILM (Stolcke et al., 2002) to build 5-gram language model with modified Kneser-Ney smoothing (James, 2000). Our language modeling data consists of the English Gigaword 5th edition LDC2011T07 and the English side of the training datasets. The best weight parameters are tuned using the Minimum Error Rate Training (MERT) algorithm (Och, 2003) to maximize BLEU score (Papineni et al., 2002). To account for optimization algorithm instability, we replicate optimization three times per experiment. We use bootstrap resampling and approximate randomization (Clark et al., 2011) to statistically test for significant differences using two evaluation metrics: BLEU (Papineni et al., 2002) and TER (Snover et al., 2006). As BLEU reflects a bias toward fluency in the target language and TER identifies the least post editing, they capture complementary aspects of the translation. We consider NONE and FULL as the baselines which show the the impact of under- and over-specification of the diacritics. As discussed before, NONE accounts for the dataset without any diacritics added (consonants only) which is the default setting for most current SMT systems whereas FULL shows the impact of all automatically generated lexical and syntactic diacritic marks.

5 Results & Discussion

Diacritic Pattern	**BLEU ↑**	**TER ↓**	**BLEU ↑**	**TER ↓**	**BLEU ↑**	**TER ↓**
Dataset	**MT09**		**MT06**		**MT05**	
Baselines						
NONE	47.0 $\diamond$	45.5	25.4	56.5	27.9	48.0
FULL	46.7	45.4	25.3	56.3	27.9	47.7 $\bullet$
Singleton Schemes (*Lexical*)						
SUK	47.0 $\diamond$	45.5	25.4	56.5	27.8	48.1
GEM	47.2 $\bullet$ $\diamond$	45.3 $\bullet$	25.5 $\diamond$	56.0 $\bullet$ $\diamond$	27.9	47.6 $\bullet$
Singleton Schemes (*Inflectional*)						
CM	46.9	45.5	25.2	56.7	27.8	48.1
PASS	47.1 $\diamond$	45.4	25.4	56.0 $\bullet$ $\diamond$	27.9	47.8 $\bullet$
Singleton Schemes (*Both*)						
TANWEEN	46.7	45.9	25.3	56.7	27.9	48.1
Combined Schemes (*Lexical*)						
SUK+GEM	47.2 $\diamond$	45.4	25.4	56.2 $\bullet$	27.9	48.3
FULL-CM-PASS	47.4 $\bullet$ $\diamond$	45.2 $\bullet$ $\diamond$	25.5 $\bullet$ $\diamond$	55.9 $\bullet$ $\diamond$	28.0	47.7 $\bullet$
Combined Schemes (*Inflectional*)						
PASS+CM	47.0 $\diamond$	45.5	25.3	56.5	27.7	48.0
Combined Schemes (*Both*)						
FULL-CM	47.2 $\bullet$ $\diamond$	45.3	25.5 $\bullet$ $\diamond$	56.2 $\bullet$	28.0	47.9
PASS+GEM	47.5 $\bullet$ $\diamond$	45.1 $\bullet$ $\diamond$	25.7 $\bullet$ $\diamond$	56.0 $\bullet$ $\diamond$	28.1 $\bullet$ $\diamond$	47.6 $\bullet$
PASS+SUK	47.3 $\bullet$ $\diamond$	45.3 $\bullet$	25.3	56.4	27.9	48.0
PASS+SUK+GEM	47.1 $\diamond$	45.3	25.3	56.3 $\bullet$	27.9	47.8 $\bullet$

Table 2: This table shows the SMT performance using BLEU and TER evaluation metrics. The symbol $\bullet$ indicates statistically significant improvement compared to NONE; the symbol $\diamond$ indicates statistically significant improvement compared to FULL.

The main goal of this study is to investigate the relative performance between the different diacritic schemes on the Arabic-English SMT. We use p-value $<$0.05 as the level of significance. Generally speaking, PASS+GEM, which involves both lexical and inflectional information, has significantly higher results than both baselines across both metrics; exception can be found in MT05 using TER metric where the PASS+GEM has comparable performance to FULL. FULL-CM-PASS, which includes all semantic distinguishing diacritics, has also significantly higher results in MT09 and MT06, compared to the baselines across all metrics. In MT05, FULL-CM-PASS significantly outperforms NONE using the TER metric only; however, it has comparable performance to both baselines using the BLEU metric. Additionally, GEM has a considerably significant performance in a fair number of experiments. TANWEEN and CM are the worst performing models because it could not outperform the baselines in any of the datasets using any of the metrics. This is expected since both CM and TANWEEN reflect a relatively low lexical semantic variation compared to other schemes.

Furthermore, although the explicit marker for the absence of diacritic, SUK, in the Arabic vocabulary plays a major role in distinguishing meaning, it does not yield competitive results against either baseline except in one experiment. The same finding can be observed in PASS+CM which covers inflectional properties only. Comparing the baselines with each other, NONE seems to have higher results using the BLEU metric although the increase of the performance is significant only in MT09. On the other hand, FULL yields higher results than NONE using TER metric; the increase in performance is also not significant except in MT05.

	Type OOV Rate			**Number of Types**	**Token OOV Rate**		
Diacritic Pattern	**MT09**	**MT06**	**MT05**	**(MT09/MT06/MT05)**	**MT09**	**MT06**	**MT05**
NONE	2.51%	4.36%	1.29%	8,562 / 9,205 / 6,128	1.06%	1.30%	0.31%
FULL	2.83%	4.37%	1.39%	11,072 / 12,027 / 7,966	1.29%	1.58%	0.40%
Singleton Schemes (*Lexical*)							
SUK	2.50%	4.31%	1.28 %	8,644 / 9,324 / 6,186	1.06%	1.31%	0.31%
GEM	2.54%	4.37%	1.31%	8,638 / 9,312 / 6,175	1.07%	1.32%	0.32%
Singleton Schemes (*Inflectional*)							
CM	2.68%	4.20%	1.28%	10,936 / 11,845 / 7,868	1.25%	1.52%	0.37%
PASS	2.52%	4.43%	1.28 %	8,603 / 9,237 / 6,155	1.07%	1.32%	0.31%
Singleton Schemes (*Both*)							
TANWEEN	2.51%	4.24%	1.29%	9,363 / 10,134 / 6,720	1.10%	1.36%	0.34%
Combined Schemes (*Lexical*)							
SUK+GEM	2.54%	4.33%	1.30%	8,702 / 9,400 / 6,220	1.08%	1.32%	0.32%
FULL-CM-PASS	2.72%	4.46%	1.36%	8,912 / 9,611 / 6,337	1.13%	1.37%	0.33%
Combined Schemes (*Inflectional*)							
PASS+CM	2.69%	4.28%	1.28%	10,969 / 11,869 / 7,892	1.25%	1.54%	0.37%
Combined Schemes (*Both*)							
FULL-CM	2.73%	4.46%	1.35%	8,939 / 9,652 / 6,359	1.13%	1.37%	0.33%
PASS+GEM	2.55%	4.44%	1.31%	8,676 / 9,344 / 6,201	1.08%	1.33%	0.32%
PASS+SUK	2.51%	4.38%	1.27%	8,683 / 9,353 / 6,211	1.07%	1.32%	0.31%
PASS+SUK+GEM	2.55%	4.40%	1.30%	8,739 / 9,429 / 6,245	1.08%	1.33%	0.32%

Table 3: This table shows the rate of type and token OOV in the test sets for each diacritic scheme. For convenience, we also add the number of types (the same information as Table 1). The number of tokens for each dataset is: 41,640 tokens for MT09, 49,154 tokens for MT06, and 33,407 tokens for MT05.

Obviously, the number of type increase for each diacritic scheme in each dataset follows the same trend. NONE has the lowest number of types while FULL has the highest; the remaining diacritic schemes lie in between. When we compare the number of types and the performance of the diacritic schemes, we can see that there is a level of number of types between NONE and FULL that achieves good performance in distinguishing lexical meaning. This suggests that the increase of the number of types to some extent between NONE and FULL is acceptable with adequate amount of tokens/types in the training phase. This increase must provide appropriate lexical signals to enhance the overall performance because providing redundant lexical/inflectional signals may also degrade the performance. The number of types in the best performing diacritic schemes is closer to NONE as we can see from Table 1. On the other hand, we can observe that schemes that did not contribute to distinguishing the meaning have relatively high number of types compared to NONE (i.e. the number of types in such schemes is close to that in FULL). The increase in out of vocabulary (OOV) tokens follows the same trend as the number of types as shown in Table 3. They have comparative rate that ranges between 1.06% to 1.29% in MT09, 1.30% to 1.58% in MT06, and 0.31% to 0.40% in MT05. Because all diacritic schemes have the same number of tokens, the slight increase in OOV rate shows the impact of the diacritic scheme coverage on the test dataset.

Looking at the performance of the diacritic schemes in each dataset, it is observed that MT05 has not been affected by any of them. The only diacritic scheme that outperforms both baselines in this dataset is PASS+GEM with a 0.2 BLEU point improvement. However, although the results in MT05 are not significant, almost all diacritic schemes have comparable performance to NONE and FULL. Using the BLEU metric, MT09 benefits the most from these diacritics schemes: GEM, FULL-CM-PASS, FULL-CM, PASS+GEM, and PASS+SUK. It is unclear whether the genres in each dataset plays a major role here; MT05 is extracted from newswire collection only while MT09 and MT06 includes both newswire and web collection. Moreover, We can observe from Table 1 that the number of types in MT09 and MT06 considerably higher than MT05 which may also be a factor.

The overall performance for the defined diacritic schemes shows potential improvement which would enhance the SMT system performance to some degree. Because the phrase-based SMT system implicitly takes context of the word into its consideration, we believe that developing more sophisticated schemes that recognize context would have even a more significant impact on SMT performance especially in distinguishing words with much less sparsity.

Label	**NONE**	**GEM**
Source Sentence	E$ kl lHZp kAn +hA Axr lHZp fy HyAp +k	E$ kl∼ lHZp kAn∼ +hA Axr lHZp fy HyAp +k
Target Sentence	live every moment was the last moment in your life	live every moment as if it were the last moment in your life
Gold References	1: Live each moment as though it is the last moment in your life . 2: Live every moment as if it was the last moment of your life . 3: Live each moment as though it were the last moment of your life . 4: Live every moment as if it's the last moment of your life .	
Label	**NONE**	**FULL-CM**
Source Sentence	w+ hl wDEt qmp brwksyl AlAxyrp Hd l+ Hlm Aldstwr AlAwrwby ?	wa+ halo waDaEat qim∼ap bruwkosiyl AlAaxiyrap Had∼ li+ Hulom Ald∼usotuwr AlAuwruwb∼iy ?
Target Sentence	do you put the recent summit in brussels according to the dream of the european constitution	and whether the recent brussels summit put an end to the dream of the european constitution
Gold References	1: and has the latest brussels summit put an end to the dream of a european constitution 2: did the latest summit in brussels put an end to the dream of a european constitution 3: did the recent brussels summit put an end to the dream of a european constitution 4: has the last brussels summit put an end to the dream of a european constitution	

Table 4: This table shows the outputs for NONE and GEM systems for the first sentence; the second sentence shows the output for NONE and FULL-CM, along with their gold references.

Table 4 shows an output example from GEM and PASS+GEM systems compared to the baseline NONE to illustrate our intuition behind specifying partial diacritic schemes. The word 'kAn' [7] may have the meanings 'was' or 'as if it was/were' as produced by NONE and GEM respectively. Adding ∼ to this word makes the system consider 'kAn' and 'kAn∼' as distinct words with different lexical meanings. Although FULL produces the same translation output for the first sentence as GEM, FULL's overall performance is less than most of the diacritic patterns and it causes the dataset to be extremely sparse. For the second sentence, we compare the target translation for NONE and FULL-CM. NONE considers different sense for the word "wDEt" which is "waDaEota" (means 'you put') whereas the correct sense for this word is

[7] We normalize all forms of letter Alef (A,|,<,>) to 'A' during our preprocessing. Also, words in the datasets may exhibit tokenization and preprocessing errors generated by MADAMIRA such as the word 'kAn' which should have been tokenized as 'k+ An".

”waDaEat” (means ’it/she puts’) as appears in FULL-CM. Additionally, NONE translation of ”Hd” into ’according’ is not the accurate choice whereas the lexical target choice has been accurately predicated in the FULL-CM (’an end’).

5.1 Training Size Effect

To tease apart the training size effect on performance, we conduct the same experiments using 30% and 50% of tokens in the training dataset chosen randomly, and then compare their performance with the full training size. Table 5 shows the training number of types for 30% and 50% of the dataset for each diacritic scheme. Results in Table 6 illustrates the drop in the overall performance using smaller amount of training data for almost all the experimental conditions. As we decrease the training size, the influential triggers in the diacritic patterns become less impactful compared to NONE and do not contribute to the performance except in few cases. Most of the diacritic schemes in MT05 outperform FULL, as we decrease the training size, which is expected since FULL introduces the most sparsity in the experimental set up.

Diacritic Pattern	100%	50%	30%
NONE	303,049	232,282	186,791
FULL	432,832	336,951	266,819
SUK	306,648	235,322	187,858
GEM	308,424	236,682	188,899
CM	414,615	322,722	258,234
PASS	306,003	234,613	187,256
TANWEEN	342,025	264,595	211,293
SUK+GEM	311,024	238,878	190,627
FULL-CM-PASS	329,123	253,209	201,258
PASS+CM	417,876	325,260	260,187
FULL-CM	329,632	253,751	201,804
PASS+GEM	311,202	238,882	190,570
PASS+SUK	309,499	237,575	189,568
PASS+SUK+GEM	313,788	241,058	192,282

Table 5: This table shows the number of types for each diacritic scheme in the training data for each proportional experiment.

Reducing to 50% of the training data size, we still maintain some experimental conditions that achieve significantly higher results than the baselines. GEM and PASS+SUK+GEM provides significant higher results than both baselines across all datasets. FULL-CM-PASS also outperforms both baselines in MT09 and MT06 only. The number of diacritic schemes that outperform NONE in MT05 increases compared to using the full training data. For the 30% training condition, there are four diacritic schemes that significantly outperform NONE in MT09; namely, SUK, GEM, PASS, and their combination PASS+SUK+GEM. None of the diacritic schemes achieve significantly higher results than NONE in both MT06 and MT05, except FULL-CM in MT05 with 0.2 improvement. MT05 and MT06 actually show significantly worse results for several of the diacritic schemes that involve inflectional diacritics: FULL, CM, PASS+CM in case of both datasets in addition to PASS and PASS+GEM in MT06. We can observe that each proportion of the dataset exhibits different diacritic schemes that outperform the baselines with FULL-CM-PASS always outperforming both baselines although not consistently statistically significant.

5.2 Automatic Diacritization Performance

The impact of the diacritic schemes on the Arabic-to-English SMT performance is highly dependent on the accuracy of the automatic Arabic diacritization toolkit in addition to its performance in part of speech tagging. Evaluating their performance in the newswire genre, MADAMIRA reports an overall error rate of 13.7% for FULL diacritization and tokenization

Dataset	**MT09**			**MT06**			**MT05**		
Diacritic Pattern	**100%**	**50%**	**30%**	**100%**	**50%**	**30%**	**100%**	**50%**	**30%**
NONE	47.0 ◇	46.9 ◇	45.2 ◇	25.4	25.3 ◇	24.3 ◇	27.9	28.0	27.3 ◇
FULL	46.7	46.3	44.4	25.3	24.9	23.2	27.9	27.8	26.6
SUK	47.0 ◇	47.1 ◇	46.0 ●◇	25.4	25.4 ◇	24.4 ◇	27.8	28.2 ●◇	27.3 ◇
GEM	47.2 ●◇	47.2 ●◇	45.5 ●◇	25.5 ◇	25.6 ●◇	24.2 ◇	27.9	28.3 ●◇	27.3 ◇
CM	46.9	46.4	44.4	25.2	24.8	23.9 ◇	27.8	27.7	27.0 ◇
PASS	47.1 ◇	46.9 ◇	45.6 ●◇	25.4	25.2 ◇	24.0 ◇	27.9	28.1 ◇	27.2 ◇
TANWEEN	46.7	46.8 ◇	45.1 ◇	25.3	25.3 ◇	24.1 ◇	27.9	27.9	27.3 ◇
SUK+GEM	47.2 ◇	47.0 ◇	45.3 ◇	25.4	25.1 ◇	24.3 ◇	27.9	28.2 ◇	27.3 ◇
FULL-CM-PASS	47.4 ●◇	47.2 ●◇	45.4 ◇	25.5 ●◇	25.5 ●◇	24.3 ◇	28.0	28.1 ◇	27.4 ◇
PASS+CM	47.0 ◇	46.3	44.4	25.3	24.5	23.7 ◇	27.7	27.8	26.9 ◇
FULL-CM	47.2 ●◇	46.9 ◇	45.2 ◇	25.5 ●◇	25.2 ◇	24.3 ◇	28.0	28.1 ◇	27.5 ●◇
PASS+GEM	47.5 ●◇	47.3 ●◇	45.3 ◇	25.7 ●◇	25.2 ◇	23.9 ◇	28.1 ●◇	28.0	27.1 ◇
PASS+SUK	47.3 ●◇	46.8 ◇	45.5 ◇	25.3	25.0	24.3 ◇	27.9	28.1	27.4 ◇
PASS+SUK+GEM	47.1 ◇	47.2 ●◇	45.53 ●◇	25.3	25.5 ●◇	24.2 ◇	27.9	28.3 ●◇	27.4 ◇

Table 6: This table shows the SMT performance using BLEU for each test dataset according to the different training sizes. For convenience, we repeat the numbers for the whole dataset (100%). The symbol ● indicates statistically significant improvement compared to NONE; the symbol ◇ indicates statistically significant improvement compared to FULL.

and 3.9% for part of speech tagging error rate. To assess the performance of MADAMIRA in diacritization for each of the schemes, we use LDC Arabic Treebank (ATB) corpora from some genres that are used in our SMT model, which is approximately 500,510 tokens: broadcast news which includes ATB 5, 7, 10, and 12 in addition to web collection data which includes ATB 6 and 11. This corpora provides many gold morphological features including diacritization and tokenization. We exclude newswire collection from our evaluation because MADAMIRA is trained on this genre; thus, we try to avoid the performance bias towards newswire.

Diacritic Pattern	NONE	FULL	SUK	GEM	CM	PASS	TANWEEN
Diacritization Error	-	9.35%	0.73%	0.67%	6.77%	0.17%	2.82%
Diacritic Pattern	SUK+GEM	FULL-CM-PASS	PASS+CM	FULL-CM	PASS+GEM	PASS+SUK	PASS+SUK+GEM
Diacritization Error	1.17%	3.42%	6.9%	3.62%	0.82%	0.87%	1.3%

Table 7: This table shows MADAMIRA diacritization error for each of the linguistically motivated schemes. The diacritization error excludes the tokenization error rate and considers the word to be a match if they have the same tokenization and diacritization.

In this evaluation, we consider the words in both the automatically tagged corpus and the gold data to be a match if they have the exact same tokenization and diacritization. The tokenization F1 score for this dataset is 96.40% which is the same tokenization performance score for all schemes. Table 7 shows the diacritization error for each of the diacritization schemes which takes the frequency of words that are affected by the associated diacritic pattern into its consideration. We can observe from Table 7 that FULL has the highest error rate followed by CM and PASS+CM. Removing CM (or CM+PASS) related diacritics from FULL substantially decreases the error rate compared to FULL. SUK, GEM, and PASS related scheme have the highest performance. Hence, we can say that case and mood related diacritics are the hardest diacritics to predict using MADAMIRA; this suggests a strong correlation with the underperformance of such schemes that involve CM on SMT. As we discussed previously, diacritic schemes are built from automated sources for tokenization, diacritization, part of speech tagging, and lemmatization. This reliance on external morphological analyzers and disambiguators easily propagates errors to higher levels especially in pipelined set ups. However, it is also not realistic to rely on gold sources as they are expensive and limited in their sizes.

6 Conclusion

We investigated the impact of various linguistically motivated partial diacritization schemes on Arabic-to-English SMT performance. We find that PASS+GEM and FULL-CM-PASS have statistically higher results than two robust baselines using several SMT metrics despite significant increase in the number of types in the data sets which indicates that such diacritization schemes are capturing and modeling important information at a more appropriate level of granularity. There are other diacritic schemes that perform well in some of the evaluation metrics and datasets. Additionally, having a relatively large dataset has a significant impact on performance. Moreover improving the underlying diacritization technology will probably have a significant impact on performance.

7 Acknowledgements

We thank Abdelati Hawwari and the three anonymous reviewers for their valuable comments and suggestions. This publication was made possible by grant NPRP 6-1020-1-199 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the author[s].

References

AlHanai, T. and Glass, J. (2014). Lexical modeling for Arabic ASR: A systematic approach. *Proceedings of INTERSPEECH*.

Aminian, M., Ghoneim, M., and Diab, M. (2015). Unsupervised false friend disambiguation using contextual word clusters and parallel word alignments. *Syntax, Semantics and Structure in Statistical Translation*, page 39.

Ananthakrishnan, S., Narayanan, S., and Bangalore, S. (2005). Automatic diacritization of Arabic transcripts for automatic speech recognition. In *Proceedings of the 4th International Conference on Natural Language Processing*, pages 47–54.

Bebah, M., Amine, C., Azzeddine, M., and Abdelhak, L. (2014). Hybrid approaches for automatic vowelization of Arabic texts. *International Journal on Natural Language Computing (IJNLC)*.

Bouamor, H., Zaghouani, W., Diab, M., Obeid, O., Oflazer, K., Ghoneim, M., and Hawwari, A. (2015). A pilot study on Arabic multi-genre corpus diacritization annotation. In *ANLP Workshop 2015*, page 80.

Buckwalter, T. (2002). Buckwalter Arabic morphological analyzer version 1.0. *Linguistic Data Consortium*.

Carpuat, M. and Wu, D. (2007). Improving statistical machine translation using word sense disambiguation. In *EMNLP-CoNLL*, volume 7, pages 61–72.

Chan, Y. S., Ng, H. T., and Chiang, D. (2007). Word sense disambiguation improves statistical machine translation. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 33.

Cherif, W., Madani, A., and Kissi, M. (2015). Towards an efficient opinion measurement in Arabic comments. *Procedia Computer Science*, 73:122–129.

Clark, J. H., Dyer, C., Lavie, A., and Smith, N. A. (2011). Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 176–181. Association for Computational Linguistics.

Debili, F. and Achour, H. (1998). Voyellation automatique de l'arabe. In *Proceedings of the Workshop on Computational Approaches to Semitic Languages*, pages 42–49. Association for Computational Linguistics.

Diab, M., Ghoneim, M., and Habash, N. (2007). Arabic diacritization in the context of statistical machine translation. In *Proceedings of MT-Summit*.

El-Imam, Y. A. (2004). Phonetization of Arabic: Rules and algorithms. *Computer Speech & Language*, 18(4):339–373.

Elshafei, M., Al-Muhtaseb, H., and Alghamdi, M. (2006). Statistical methods for automatic diacritization of Arabic text. In *The Saudi 18th National Computer Conference. Riyadh*, volume 18, pages 301–306.

Gal, Y. (2002). An HMM approach to vowel restoration in Arabic and Hebrew. In *Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages*, pages 1–7. Association for Computational Linguistics.

Habash, N. and Rambow, O. (2007). Arabic diacritization through full morphological tagging. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 53–56. Association for Computational Linguistics.

Hermena, E. W., Drieghe, D., Hellmuth, S., and Liversedge, S. P. (2015). Processing of Arabic diacritical marks: Phonological–syntactic disambiguation of homographic verbs and visual crowding effects. *American Psychological Association*.

James, F. (2000). Modified Kneser-Ney smoothing of n-gram models. *Research Institute for Advanced Computer Science, Tech. Rep. 00.07*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Maamouri, M., Bies, A., Buckwalter, T., and Mekki, W. (2004). The penn Arabic treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, volume 27, pages 466–467.

Maamouri, M., Bies, A., and Kulick, S. (2008). Enhancing the Arabic treebank: A collaborative effort toward new annotation guidelines. In *LREC*.

Marton, Y., Habash, N., and Rambow, O. (2010). Improving Arabic dependency parsing with lexical and inflectional morphological features. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 13–21. Association for Computational Linguistics.

Nelken, R. and Shieber, S. M. (2005). Arabic diacritization using weighted finite-state transducers. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 79–86. Association for Computational Linguistics.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics.

Pasha, A., Al-Badrashiny, M., Diab, M., Kholy, A. E., Eskander, R., Habash, N., Pooleery, M., Rambow, O., and Roth, R. (2014). MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. European Language Resources Association (ELRA).

Schmid, H. (1995). Improvements in part-of-speech tagging with an application to German. In *Proceedings of the ACL SIGDAT-Workshop*.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, volume 200.

Stolcke, A. et al. (2002). SRILM-an extensible language modeling toolkit. In *Interspeech*, volume 2002, page 2002.

Vergyri, D. and Kirchhoff, K. (2004). Automatic diacritization of Arabic for acoustic modeling in speech recognition. In *Proceedings of the Workshop on Computational Approaches to Arabic Script Based Languages*, pages 66–73. Association for Computational Linguistics.

Yang, M. and Kirchhoff, K. (2012). Unsupervised translation disambiguation for cross-domain statistical machine translation. In *Proceedings of Association for Machine Translation in the Americas*.

Zitouni, I. and Sarikaya, R. (2009). Arabic diacritic restoration approach based on maximum entropy models. *Computer Speech & Language*, 23(3):257–276.

Zitouni, I., Sorensen, J. S., and Sarikaya, R. (2006). Maximum entropy based restoration of Arabic diacritics. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 577–584. Association for Computational Linguistics.