Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 3895–3904
Marseille, 11–16 May 2020

Morphological Analysis and Disambiguation for Gulf Arabic: The Interplay between Resources and Methods

Salam Khalifa, Nasser Zalmout, Nizar Habash
Computational Approaches to Modeling Language (CAMeL) Lab
New York University Abu Dhabi
{salamkhalifa, nasser.zalmout, nizar.habash}@nyu.edu

Abstract

In this paper we present the first full morphological analysis and disambiguation system for Gulf Arabic. We use an existing state-of-the-art morphological disambiguation system to investigate the effects of different data sizes and different combinations of morphological analyzers for Modern Standard Arabic, Egyptian Arabic, and Gulf Arabic. We find that in very low settings, morphological analyzers help boost the performance of the full morphological disambiguation task. However, as the size of resources increase, the value of the morphological analyzers decreases.

Keywords: Gulf Arabic, Morphology, Disambiguation

1. Introduction

Despite the many advances in the field in Natural Language Processing (NLP) for Arabic, many dialectal Arabic varieties are lagging behind. Modern Standard Arabic (MSA), the official language of the Arab world, is well studied in NLP and has an abundance of resources including corpora and tools. On the other hand, most Arabic dialects are considered under-resourced, with the exception of Egyptian Arabic (EGY).

One of the main challenges for Arabic NLP is data sparsity due to morphological richness and the lack of orthography standards. To address such challenges, recent NLP models use deep learning architectures that have access to character and subword-level information. Such models are well equipped to model some aspects of morphology implicitly as part of an end-to-end system without requiring explicit feature engineering. However, these models are very data-intensive, and do not scale down well in the case of low-resource languages. Moreover, some morphological behaviors can be complicated and irregular at times. This makes morphology more challenging to capture implicitly while modeling other tasks. This further highlights the importance of explicit morphological modeling for languages that are morphologically-rich and low-resource in particular.

Morphological analyzers are dictionary-like resources that provide all the potential analyses for a given word out-of-context. Ideal morphological analyzers are expected to return all possible analyses of a given word (modeling ambiguity), along with covering all the different inflected forms of a word lemma (modeling richness). The quality of the morphological analyzers varies drastically based on the method used to create them. Higher quality analyzers are carefully built using existing linguistic dictionaries, and are manually checked. On the other end of the spectrum, morphological analyzers created automatically (e.g., extracted from annotated text or seed dictionaries) can have a lower quality depending on the quantity and quality of data used or the methods employed in creating them.

Morphological disambiguation is the process of providing the most probable morphological analysis in context for a given word. This task is achieved by either ranking the output of a morphological analyzer or through an end-to-end system that generates a single answer.

In this paper, we focus on Gulf Arabic (GLF), a morphologically rich and resource poor Arabic dialect. We aim to benchmark full morphological analysis and disambiguation for GLF using state-of-the-art approaches for the first time. As part of this work, we investigate the relationship between the size of the training data and the different available analyzers with respect to the disambiguation model.

The rest of the paper is structured as follows. We present related work in Section 2, and briefly discuss the linguistic background of GLF and its challenges in Section 3. We present the morphological analyzer creation process in Section 4. In Sections 5 and 6, we present our experimental setup and evaluation, respectively. We conclude and outline future work in Section 7.

2. Related Work

In the past two decades, there have been a lot of efforts on morphological modeling for Arabic, as it proved to be useful in a number downstream NLP tasks (Sadat and Habash, 2006; El Kholy and Habash, 2012; Halabi, 2016; Baly et al., 2017). In this section we review early efforts on morphological modeling of MSA and dialectal Arabic. We then present the latest neural-based contributions with special interest in Arabic.

Modern Standard Arabic Morphological Modeling One of the earliest morphological tagging systems for Arabic was presented by Khoja (2001); it was based on a corpus of 50,000 words. Later, the LDC released the Penn Arabic Treebank (PATB) (Maamouri et al., 2004), which was substantially larger, and supported many further efforts on Arabic morphological modeling. The PATB relied on the existence of the Buckwalter Morphological Analyzer (BAMA) (Buckwalter, 2004) and its later version the Standard Arabic Morphological Analyzer (SAMA) (Maamouri et al., 2010). Among such efforts, are MADAMIRA (Pasha et al., 2014) and it predecessors MADA (Habash and Rambow, 2005; Roth et al., 2008; Habash et al., 2009; Habash et al., 2013) and AMIRA (Diab et al., 2004; Diab et al., 2007). MADAMIRA uses a morphological analyzer and SVMs-based taggers for different features, along with n-gram language models for lemmatization and diacritization. More recent efforts include Farasa (Abdelali et al., 2016; Darwish and Mubarak, 2016) and YAMAMA (Khalifa et al., 2016b), which are out-of-context taggers/segmenters that use different techniques to achieve reasonable performance and fast running time.

Adaptation of MSA Tools and Resources for Dialectal Arabic A number of approaches attempt to exploit linguistic similarity between MSA and Arabic dialects to build dialect tools using existing MSA resources (Duh and Kirchhoff, 2005; Habash and Rambow, 2006; Zribi et al., 2013; Salloum and Habash, 2014; Hamdi et al., 2015; Albogamy and Ramsay, 2015; Eskander et al., 2016). MAGEAD is a morphological analyzer that models Arabic dialects together with MSA using a common multi-tape finite-state-machine framework (Habash and Rambow, 2006). Zribi et al. (2013) adapt an MSA analyzer, Al-Khalil (Boudlal et al., 2010), to Tunisian Arabic, where they modify the derivation patterns and add Tunisian-specific roots and patterns. Eskander et al. (2016), on the other hand, presented a paradigm completion approach to generate morphological analyzers for low-resource dialects using morphologically annotated corpora. They then make use of available MSA and EGY analyzers as backoff.

Dialect-Specific Contributions Al-Sabbagh and Girju (2012) presented a POS annotated data set and tagger for EGY. Habash et al. (2012) presented CALIMA, a morphological analyzer for EGY, which was built by extending the Egyptian Colloquial Arabic Lexicon (ECAL) (Kilany et al., 2002). The LDC has also released the EGY treebank (ARZATB) (Maamouri et al., 2012). The treebank is morphologically annotated in a similar style to the PATB. The aforementioned MADAMIRA and YAMAMA systems were extended to EGY using CALIMA (Habash et al., 2012) and ARZATB (Maamouri et al., 2012). Jarrar et al. (2014) released the Curras Corpus of Palestinian Arabic, also annotated in the PATB morphology style. Darwish et al. (2018) used a CRF model for a multi-dialect POS tagging, using a small annotated Twitter corpus for several dialects. Erdmann et al. (2019) developed a de-lexical segmentation tool for dialectal content. Their model is mainly unsupervised, relying on a small grammar of closed-class affixes. Alshargi et al. (2019) presented morphologically annotated corpora for seven different dialects.

Regarding GLF specifically, Khalifa et al. (2017) presented a morphological analyzer for Gulf verbs, covering segmentation, POS, and lemmatization details for Gulf verbal paradigms. Khalifa et al. (2018) also presented a large-scale morphologically annotated corpus of Emirati Arabic, extracted from online novels, with about 200K words. The annotation includes tokenization, POS, lemmatization, English glosses and dialect identification, as the corpus includes traces of other dialects, along with MSA content.

So far, the mentioned efforts regarding disambiguation suffer from two main issues. First, they require explicit feature engineering which can lead to over fitting and not being able to generalize to new dialects. Second, those systems rely heavily on pre-existing morphological analyzers to generate the final answers. This reliance limits the system's performance to the quality of the analyzer, especially when generating open class features such as lemmas. In this work we use state-of-the-art neural architectures that have the ability to model morphologically rich and complex languages such as Arabic and its different verities.

Neural-based contributions for Arabic Morphology Neural-based contributions for Arabic NLP are relatively scarce and specific to individual tasks rather than full morphological disambiguation. Among the contributions that utilize morphological structures to enhance the neural models in different NLP tasks, we note Guzmán et al. (2016) for machine translation, and Abandah et al. (2015) for diacritization. Shen et al. (2016) applied their Bi-LSTM morphological disambiguation model on MSA, but did not present any improvements over the state-of-the-art. Heigold et al. (2016) developed character-based neural models for morphological tagging for 14 different languages, including Arabic, using the UD treebank. Samih et al. (2017a) used a Bi-LSTM-CRF architecture and pre-trained character embeddings for the segmentation of EGY tweets. They then build up on this approach using a similar architecture for segmentation in multiple dialects, through combining the training datasets for the different dialects, and train a unified segmentation model. They report the results using both an SVM-Rank and Bi-LSTM-CRF models (Samih et al., 2017b). Darwish et al. (2017) use Bi-LSTM models to train a POS tagger, and compared it against SVM-based model. The SVM model in their system outperformed the neural model, even with incorporating pre-trained embeddings. Other notable contributions include the work of Inoue et al. (2017), who used multi-task learning to model fine-grained POS tags, using the individual morphosyntactic features. More recently, Zalmout and Habash (2017) presented the first neural based full morphological disambiguation system for Arabic. Alharbi et al. (2018) also use a Bi-LSTM model for GLF POS tagging, with good results.

In this work, we introduce the first morphological analysis and disambiguation system for GLF. We base our work on Zalmout and Habash (2019), and we use the data from Khalifa et al. (2018).

3. Relevant Linguistic Background

Gulf Arabic We follow the definition of GLF mentioned in (Khalifa et al., 2016a), as the variety of Arabic spoken by indigenous populations residing the six countries of the Gulf Cooperation Council (GCC): Saudi Arabia, United Arab Emirates, Qatar, Kuwait, Bahrain, and Oman. In this work, the GLF data used in this work is of the Emirati dialect specifically.

Arabic NLP Challenges Similar to MSA and other dialects, GLF is morphologically rich; a single lemma can

mςqwlħ hðy Hbh AlwHyd? معقولة هذي حبه الوحيد ؟ Is this really his only love?													
Diac	**Gloss**	**Analysis (condensed)**	**pos**	**asp**	**per**	**gen**	**num**	**prc3**	**prc2**	**prc1**	**prc0**	**enc0**	
Hib∼+ah	love,kiss him	VERB.C2MS+PRON.3MS	verb	c	2	m	s	0	0	0	0	3ms_dobj	
Hab∼+ah	loved,kissed him	VERB.P3MS+PRON.3MS	verb	p	3	m	s	0	0	0	0	3ms_dobj	
Hub∼+ah	his love	NOUN.MS+PRON.3MS	noun	na	na	m	s	0	0	0	0	3ms_poss	
Hab∼aħ	pill, seed	NOUN.FS	noun	na	na	f	s	0	0	0	0	0	
Hab∼+ah	his seeds	NOUN.MS+PRON.3MS	noun	na	na	m	s	0	0	0	0	3ms_poss	

Table 1: Possible analyses of the word حبه *Hbh*. The correct analysis is highlighted in gray. The annotations are represented using condensed tags, we converted them to the verbose representation of 10 morphological non-lexical features.

have a large number of forms realizing different combinations of morphological inflections and cliticized particles. Arabic orthography adds a lot of ambiguity due to the common omission of short-vowel (and other) diacritics. Table 1 demonstrates a set of possible morphological analyses for the word حبه *Hbh*[1] in GLF.

Dialectal Differences and NLP Arabic dialects differ significantly from each other and from MSA to the point that using MSA tools and resources for processing dialects is not sufficient. Jarrar et al. (2014) report that MADAMIRA MSA (Pasha et al., 2014) only correctly analyzes 64% of Palestinian Arabic words, compared with 78% using MADAMIRA EGY. Eryani et al. (2020) reports that the average vocabulary overlap between any pair of Arabic city dialects from the MADAR parallel dialectal corpus (Bouamor et al., 2018), is about 36%.[2]

Dialectal Orthography Dialectal Arabic has no standard orthography. A word can be spelled differently depending on the writer's decision to either fall back to the MSA cognate or to spell phonetically. In extreme cases, a word can have more than twenty different spelling variations (Habash et al., 2018). This is very challenging due to the inconsistency and therefore more sparsity in the data. In this work, we use text that has been normalized according to the Conventional Orthography for Dialectal Arabic (CODA) (Habash et al., 2018). CODA provides a set of guidelines and rules to help create spelling conventions for Arabic dialects. For more details about specific decisions on GLF, see (Khalifa et al., 2016a). For the full latest set of guidelines, see (Habash et al., 2018).

Morphological Representations There are different approaches to represent morphological analysis for a word depending on the task in hand. From the annotation point of view, the data is often annotated in a way that guarantees annotation efficiency, hence, the representation of tags used is often more readable and explainable to the human annotator. The data we use from Khalifa et al. (2018) was annotated using the MADARi morphological annotation interface (Obeid et al., 2018). Where for each word in context the annotators were asked to produce a CODA spelling, then tokenize and assign a single condensed tag that includes the core POS along with the functional morphological features for each token.

Computationally, a more verbose representation of the analysis is usually used to have more control over modeling choices. In this work, we follow the same format used in previous efforts (Habash, 2007; Pasha et al., 2014), where we have a vector like representation of feature value pairs. We therefore map from the condensed representation to the more verbose one. This mapping includes POS tags, clitics position mapping, and stemming.

Table 1 shows the two different representations side by side for the different analyses. We use a vector of 10 non-lexical morphological features which are POS, aspect (**asp**), person (**per**), gender (**gen**), number (**num**),[3] four proclitcis (**prc0, prc1, prc2, prc3**), and one enclitic (**enc0**). We use the POS tagset introduced in (Habash et al., 2013) which consists of 36 tags.

4. Morphological Analyzer Creation through Paradigm Completion

Morphological analyzers are rich resources that can be used as lexical and grammatical references. In a number of popular tools for Arabic morphological disambiguation, the task is defined as an in-context ranking of the out-of-context morphological analyses produced by an analyzer (Habash et al., 2009; Pasha et al., 2014). There are different ways of building morphological analyzers: manually, automatically, or semi-automatically. In this work, we use two manually created morphological analyzers for MSA and EGY. However, for GLF, we use the approach of *paradigm completion* as demonstrated by Eskander et al. (2013a) and Eskander et al. (2016).

Paradigm completion makes use of the templatic nature of Arabic to model morphology through roots and patterns. The approach mainly aims at completing the inflectional classes (ICs) generated from available morphological annotations. The set of inflectional forms for a given lexeme is called a paradigm. The completion is performed on the level of POS tags present in the data, where all the possible morphosyntactic feature combinations from the words that share the same POS tag are collected. This set of potential feature combinations represent the slots for inflected forms in the ICs. Using this set to indicate the potential slots, the algorithm goes through each of the lemmas in the

[1]Arabic script transliteration is presented in the Habash-Soudi-Buckwalter transliteration scheme (Habash et al., 2007).

[2]The specific city dialects were of Beirut, Cairo, Doha, Tunis and Rabat (Eryani et al., 2020).

[3]We use form-based not functional gender and number features in this work (Alkuhlani and Habash, 2011).

Split	Sentences	Tokens	Types	$\frac{Analyses}{Type}$	$\frac{Analyses}{Lemma}$
TRAIN	12,274	162,031	20,079	1.28	3.69
DEV	1,499	20,198	5,090	1.14	2.53
TEST	1,452	20,100	4,980	1.15	2.48
ALL	15,225	202,329	22,924	1.29	3.89

Table 2: Statistics on TRAIN, DEV, and TEST in terms of number of sentences, tokens, and types, as well as an ambiguity measure (analyses per type) and a richness measure (analyses per lemma). Note that 'ALL' represents the statistics on the entire corpus as a whole and not the sum of the splits.

dataset and fills the corresponding slot in the IC using all the inflected forms of that lemma in the dataset. The slots include information on the prefixes, stems, and suffixes for each lemma. After this process, many slots will still be empty; so the algorithm automatically completes the ICs to fill in the missing slots, and obtains all inflections of all the lexemes. The resulting paradigms and ICs are then combined into a morphological analyzer.

We used paradigm completion by Eskander et al. (2016) *as is* in this work, where the only input is the training data and the output is a morphological analyzer. We used the training data (TRAIN, see next section) as the input to the paradigm completion pipeline after filtering all sentences that are not marked as GLF. We also filtered out potential annotation mistakes that would propagate throughout the paradigm completion process. We identified some errors automatically: for words that share the same lemma, gloss, POS, and morphological features we chose the entries that had the highest stem count throughout the text.

The resulting analyzer has the same basic structure of the Buckwalter analyzer (Buckwalter, 2004): it consists of three tables for complex prefixes, complex suffixes, and stems, and three tables representing the compatibility between prefixes, suffixes, and stems (Habash, 2007).

5. Experimental Setup

We describe in this section all the relevant details for the experiments we conducted.

5.1. Data

Corpus In this work, we use the Annotated Gumar Corpus (Khalifa et al., 2018), which is a portion of the Emirati Arabic subset of the Gumar corpus (Khalifa et al., 2016a). The text was manually annotated for full morphology, which includes all morphological features in addition to lemmatization, tokenization, CODA spelling annotation, English gloss, and sentence level dialect tagging. The data comes in eight documents, where each document is roughly 25,000 words and represents a portion (the first 25,000 words on the sentence cut) of a complete novel.

Splits We split the data into three sets: training, development, and testing, henceforth TRAIN, DEV, and TEST, respectively. Given the nature of the data, where each document comes from a different novel written by a different author, there are a number of ways to split. An intuitive way is to split on the novel level, but because of the varying styles of the novels, the trained model could be heavily biased towards a certain style. Therefore, we split the data equally across the eight documents, this way we ensure fair coverage of different styles across the splits. From each document, the first 80% is TRAIN, the following 10% is DEV, and the last 10% is TEST, where the portions for each split are concatenated together. Table 2 shows the statistics on sentence and word token count of the different splits. The table also includes a the average number of analyses per type (ambiguity), and the average number of analyses per lemma (richness). Table 2 illustrates that TRAIN is representative of the entire dataset in terms of ambiguity and richness measures.

Word Embeddings For pre-trained embeddings, we used FastText (Bojanowski et al., 2016) trained on the full Gumar corpus (Khalifa et al., 2016a) which contains about 100 million tokens of Gulf Arabic.

CODA The Annotated Gumar Corpus (Khalifa et al., 2018) provides the raw text as well as the CODA version of the text. The word error rate of the raw text against CODA is 24.9% of which 21.2% is due to substitutions, 2.5% to insertions, and 1.1% to deletions. All of the reference annotations are linked to the CODA version of the text. As a result, full processing of raw text must include an initial conversion into CODA, comparable to the work of Eskander et al. (2013b) and Eryani et al. (2020), and related to general Arabic spelling correction efforts (Mohit et al., 2014; Watson et al., 2018). Since spelling modifications, especially insertions and deletions, will lead to a more complex full morphological evaluation process, we leave this effort to future work. In this paper, all of our results assume starting with CODA text.

5.2. Morphological Analyzers

We used three morphological analyzers for MSA, EGY, and GLF.

- **MSA-MA**$_{Manual}$ For MSA, we used the Standard Arabic Morphological Analyzer (SAMA) (Graff et al., 2009), a manually created morphological analyzer.
- **EGY-MA**$_{Manual}$ For EGY, we used CALIMA Egyptian (Habash et al., 2012), also a manually created morphological analyzer.
- **GLF-MA**$_{PC}$ For GLF, we used the analyzer described in Section 4, which was automatically created through paradigm completion.

We expect the different analyzers to exhibit different qualities based on the approach and the data that were used to build them. Both MSA and EGY analyzers are custom-built with high coverage compared to the GLF analyzer that is automatically generated from the training data as explained in Section 4. Non-GLF features in the output of the MSA and EGY analyzers are dropped to make the output compatible with GLF features we model. These include case, state, mood, voice and additional enclitics. We experiment with using no analyzers, only **GLF-MA**$_{PC}$, and extending it with non GLF analyzers, by taking the union of outputs of different analyzers.

5.3. Disambiguation Models

We report performance on two disambiguation models.

MLE First is a Maximum Likelihood Estimation (MLE) model based on TRAIN, where each word is assigned the most frequent full analysis; and out-of-vocabulary (OOV) words are treated as proper nouns and assigned default gender and number features (i.e., NOUN_PROP.MS).

Neural Joint Model Second is the full neural morphological tagger from Zalmout and Habash (2019), which is a joint-modeling approach for lemmatization, diacritization, and normalization (modeled at the character level) and non-lexical morphological features (modeled at the word level). We used a modified sequence-to-sequence architecture, where some components of the encoder are shared between a tagger, for the morphological features, and the encoder-decoder architecture, for the lemma and other lexicalized features. We also used separate decoders for the different lexical features that share the same encoder and trained jointly using a shared loss function.

Because of the corpus' conversational style (Khalifa et al., 2016a), some portions of the text had particularly long sentences. Therefore, we split sentences in a cascading fashion with a length of 200 words and a buffer of 10 words at the beginning of the new sentence from the previous one to maintain contextual integrity.

We consider three setups for using the morphological analyzers with the **Neural Joint Model**.

- **No Analyzer**: This is the most basic mode for **Neural Joint Model**. The system generates all the lexical and non-lexical features directly.
- +EMBED: We use morphological analyzers to get the potential candidates for each morphological feature, and embed them as part of the input to the model. This approach helps the system narrow down the space of potential tags to the ones that are considered likely through the analyzer.
- +EMBED+RANK: In addition to embedding the potential candidate features, the system ranks the potential analyses produced by the morphological analyzer. The predicted analysis from the model is used to evaluate each of the potential analyses obtained from the morphological analyzer. The analysis with the highest matching score, weighted through pre-tuned weight values, is returned as the overall chosen analysis.

5.4. Metrics

To evaluate our performance we compute the accuracy in terms of the following metrics:

- FULL: The overall accuracy of the full analysis, i.e., POS, morphological features, and lemma.
- TAGS: The accuracy of the combined set of the 10 non-lexical morphological features described in Section 3.
- LEX: The accuracy of matching the fully diacritized lemma.
- POS: The accuracy of matching the POS, described in Section 3.
- SEG: The accuracy of all five clitic features (four proclitics and one enclitic). This measure estimates the segmentation performance.

6. Results

Next, we evaluate the performance of the above mentioned models and analyzers on the GLF dataset.

6.1. Baselines

As an initial experiment, assuming that we do not have any GLF training resources, we measure the performance of the **Neural Joint Model**+EMBED+RANK model trained on MSA, and on EGY, and using their respective analyzers, **MSA-MA**$_{Manual}$ and **EGY-MA**$_{Manual}$. We ignore non-GLF features such as MSA case and voice; as well as EGY additional enclitics. Table 3 shows the results in the different metrics. Although both models perform poorly on GLF, the EGY trained model outperforms the MSA trained model. This behavior is expected since dialects are generally closer to each other than to MSA.

	FULL	TAGS	LEX	POS	SEG
EGY Model	**39.5**	**52.6**	**63.1**	**74.9**	74.8
MSA Model	37.8	50.0	60.1	69.9	**75.6**

Table 3: DEV set baseline results of the **Neural Joint Model**+EMBED+RANK trained on MSA and EGY and using their respective analyzers.

6.2. Morphological Analyzers & Disambiguation Models

We experimented with different combinations of morphological analyzers: no analyzer, **GLF-MA**$_{PC}$, and **GLF-MA**$_{PC}$ extended with MSA and EGY analyzers (**MSA-MA**$_{Manual}$ and **EGY-MA**$_{Manual}$, respectively). When using a morphological analyzer with **Neural Joint Model**, we include the results for both settings, +EMBED and +EMBED+RANK. Table 4 shows the results. The different analyzers provide minor or no improvements over the **Neural Joint Model** alone when embedding the candidate tags. On the other hand, the ranking approach reduces the accuracy drastically for different combinations of analyzers.

This behavior might suggest that for relatively higher-resource dialects, the model is capable of identifying the inflectional relationships between the different surface forms without having to rely on an explicit paradigm completion process, nor external analyzers. We can also observe that LEX scores in particular had the biggest drop when we used the ranking approach. This suggests that constraining the

Analyzer	Model	FULL	TAGS	LEX	POS	SEG
No Analyzer	**MLE**	84.7	85.9	89.4	88.9	90.1
	Neural Joint Model	89.3	93.1	93.1	96.8	**97.5**
GLF-MA$_{PC}$	**Neural Joint Model**+EMBED	**89.7**	93.0	**93.3**	96.6	97.3
	Neural Joint Model+EMBED+RANK	84.8	92.2	88.4	95.0	96.3
GLF-MA$_{PC}$+**MSA-MA**$_{Manual}$	**Neural Joint Model**+EMBED	89.6	**93.3**	93.2	96.8	**97.5**
	Neural Joint Model+EMBED+RANK	86.4	92.4	90.7	95.9	97.4
GLF-MA$_{PC}$+**MSA-MA**$_{Manual}$+**EGY-MA**$_{Manual}$	**Neural Joint Model**+EMBED	89.4	93.1	93.2	**96.9**	**97.5**
	Neural Joint Model+EMBED+RANK	87.7	92.4	92.2	96.2	97.4

Table 4: DEV results of the various morphological analyzer configurations for GLF, using **GLF-MA**$_{PC}$ first, then using external morphological analyzers from MSA and EGY.

system to the lemmas in the analyzer rather than the generated lemmas is not beneficial. We investigate those issues in more detail next, where we run the **Neural Joint Model** with different combinations of analyzers on a learning curve of the training data size, where we control the amount of GLF data that the model has access to.

6.3. Training Data Learning Curve

Next, we report the results on the relationship between the training data size, and the morphological analyzer and disambiguation model choice. This gives insights on the performance under different degrees of data availability. Our hypothesis is that the role of morphological analyzers depends on the amount of training data. With less training data we expect to see more added value; and with more training data, we except less added value. However, since paradigm completion relies on training data, it is possible that the effect of morphological analyzers created through paradigm completion may be muted all along.

To investigate this hypothesis, we control the amount of data that the model has access to. We train the **Neural Joint Model** on portions of the available training data through reducing the size by a factor of two each time and selecting the data samples randomly. For each fraction, we run the paradigm completion on the corresponding amount of data. We also incorporate the existing **MSA-MA**$_{Manual}$ and **EGY-MA**$_{Manual}$ analyzers at each run of the model, since these resources are independent of the amount of training data available for GLF.

Table 5 shows the results for different training data sizes. We use the FULL metric only for this evaluation. We observe three regions of different performance behaviors. In the very low setting (5k-10K), using the combination of the different analyzers in the +EMBED+RANK model configuration outperforms all the others. In the medium setting (20k-40k), the combination of analyzers still helps but only when embedding the candidate analysis. Finally, in the highest setting (>80k), the **GLF-MA**$_{PC}$ slightly outperforms the **Neural Joint Model** with no analyzer, and the **Neural Joint Model** with the combination of analyzers.

We can see that morphological analyzers are generally helpful. Embedding candidate tags is helpful when using any combinations of the analyzers. The ranking approach, on the other hand, is more helpful in smaller sizes of the training data, but as the size increases, the ranking approach seems to be lagging even behind the **Neural Joint Model** with no analyzer. It seems that with more training data, the **Neural Joint Model** is producing better analyses than the limited morphological analyzers. Hence, in higher-resource settings, the morphological analyzer actually constraints the overall modeling capacity of the system, rather than improving it when used for ranking.

6.4. Discussion

Our experiments show the clear relationship between the different combinations of resources and model configuration, where more training data helps when available in large amounts, otherwise, high-coverage morphological analyzers boosts the performance in very low settings.

However, when we examine the performance closely, we notice that lemmatization in particular seems to suffer a lot when +RANK is introduced as shown in Table 4. To investigate more, we evaluate the lemmatization accuracy in particular across the different training data sizes in the learning curve experiments. We contrast the lemmatization accuracy LEX values against the FULL metric. We also compare the LEX values to the results of the TAGS metric, which evaluates the non-lexical features only. The results illustrated in Figure 1 confirm our intuition regarding the lemmatization behavior. The gap between the LEX and TAGS values increases when using more training data. Increasing the training data enhances the modeling capacity in general. But the limited number of different lemmas in the morphological analyzer, and having to choose an analysis from the analyzer in the ranking step, prevents further improvement. We also observe that the FULL values are highly correlated with LEX. So the drop in the FULL accuracy when using the ranking approach, compared to the **Neural Joint Model** alone or embedding the candidate tags, can be attributed to the drop in the lemmatization accuracy compared to those models.

This observation motivates future efforts on automatically creating high-coverage morphological analyzers from available training data. This is in contrast to the paradigm completion approach described in Section 4, where it is aimed at expanding the coverage for the non-lexical features through filling the missing slots in the inflectional classes.

Analyzer	Model	5K	10K	20K	40K	80K	162K
No Analyzer	**MLE**	64.9	69.7	74.9	78.7	81.9	84.7
	Neural Joint Model	69.7	75.2	82.4	85.6	88.1	89.3
GLF-MA$_{PC}$	**Neural Joint Model**+EMBED	71.1	75.9	82.9	85.5	88.4	**89.5**
	Neural Joint Model+EMBED+RANK	73.1	76.9	80.9	82.5	82.0	84.9
GLF-MA$_{PC}$+MSA-MA$_{Manual}$+EGY-MA$_{Manual}$	**Neural Joint Model**+EMBED	72.2	76.9	**83.8**	**86.5**	**88.5**	89.2
	Neural Joint Model+EMBED+RANK	**73.8**	**77.2**	81.8	82.2	83.0	87.1

Table 5: DEV results in the FULL metric on a learning curve of the training data size against different morphological analyzers and disambiguation models.

Analyzer	Model	FULL	TAGS	LEX	POS	SEG
No Analyzer	**MLE**	84.2	85.9	88.9	88.8	90.0
	Neural Joint Model	88.7	**92.9**	92.6	**96.9**	97.2
GLF-MA$_{PC}$	**Neural Joint Model**+EMBED	**89.2**	**92.9**	**93.1**	96.7	**97.3**

Table 6: TEST results on the different baselines and the best performing system.

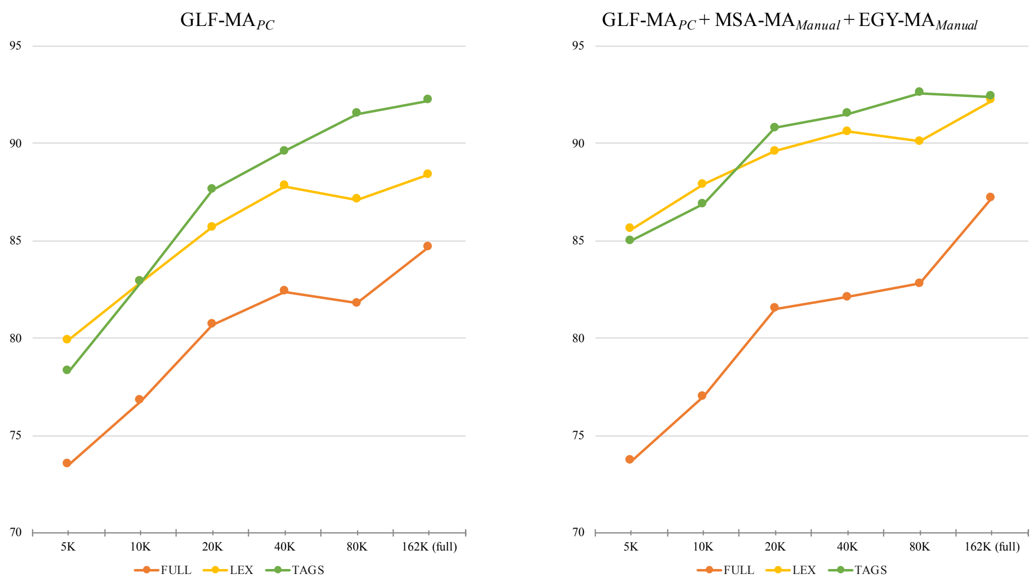

Figure 1: The relationship between lemmatization (LEX) accuracy and the overall accuracy (FULL) of the model at different data sizes. The chart also shows the relationship between lemmatization and accuracy of the non-lexical features (TAGS).

6.5. Blind Test

Finally, we apply our baseline systems and best performing setup from Table 4 (according to the FULL metric) on the TEST set. Table 6 shows the results for the different metrics. The results are consistent with our previous experiments, where using the morphological analyzer with the **Neural Joint Model** helps the overall performance.

7. Conclusion and Future Work

We presented a morphological analysis and disambiguation system for Gulf Arabic. We experimented using different combinations of morphological analyzers, disambiguation models, and training data sizes. There are several takeaways from this work. In lower-resource settings, the state-of-the-art neural approach suffers severely, and adding a morphological analyzer generated from the same training data alone boosts the performance by 3.4% at the lowest setting. Morphological analyzers are most beneficial at lower settings because of their lexical coverage. On the other hand, non-lexical features are bounded in the language and therefore can be captured easily. This suggests that lexically rich analyzers can benefit full morphological disambiguation at very low resource settings. Moreover, using morphological analyzers to provide candidates as additional embedding features is more helpful than using it for potential answers since it can restrict the space of possible answers depending on the quality of the analyzer.

In the future we plan to provide benchmarks for other low resource dialects and investigate more ways to enhance the coverage of automatically generated analyzers.

Acknowledgments

We would like to thank Ramy Eskander for help using the paradigm completion tool. The second author was supported by the New York University Abu Dhabi Global PhD Student Fellowship program. We also gratefully acknowledge the support and resources from the High Performance Computing Center at New York University Abu Dhabi.

Bibliographical References

Abandah, G. A., Graves, A., Al-Shagoor, B., Arabiyat, A., Jamour, F., and Al-Taee, M. (2015). Automatic diacritization of Arabic text using recurrent neural networks. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(2):183–197.

Abdelali, A., Darwish, K., Durrani, N., and Mubarak, H. (2016). Farasa: A Fast and Furious Segmenter for Arabic. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 11–16, San Diego, California.

Al-Sabbagh, R. and Girju, R. (2012). A supervised POS tagger for written Arabic social networking corpora. In Jeremy Jancsary, editor, *Proceedings of the Conference on Natural Language Processing (KONVENS)*, pages 39–52. ÖGAI. Main track: oral presentations.

Albogamy, F. and Ramsay, A. (2015). POS tagging for Arabic tweets. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 1–8, Hissar, Bulgaria, September. INCOMA Ltd. Shoumen, BULGARIA.

Alharbi, R., Magdy, W., Darwish, K., Abdelali, A., and Mubarak, H. (2018). Part-of-speech tagging for Arabic gulf dialect using bi-lstm. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Alkuhlani, S. and Habash, N. (2011). A Corpus for Modeling Morpho-Syntactic Agreement in Arabic: Gender, Number and Rationality. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, Portland, Oregon, USA.

Alshargi, F., Dibas, S., Alkhereyf, S., Faraj, R., Abdulkareem, B., Yagi, S., Kacha, O., Habash, N., and Rambow, O. (2019). Morphologically annotated corpora for seven Arabic dialects: Taizi, sanaani, najdi, jordanian, syrian, iraqi and Moroccan. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 137–147, Florence, Italy, August. Association for Computational Linguistics.

Baly, R., Hajj, H., Habash, N., Shaban, K. B., and El-Hajj, W. (2017). A sentiment treebank and morphologically enriched recursive deep models for effective sentiment analysis in arabic. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(4):23.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Bouamor, H., Habash, N., Salameh, M., Zaghouani, W., Rambow, O., Abdulrahim, D., Obeid, O., Khalifa, S., Eryani, F., Erdmann, A., and Oflazer, K. (2018). The MADAR Arabic Dialect Corpus and Lexicon. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Boudlal, A., Lakhouaja, A., Mazroui, A., Meziane, A., Bebah, M., and Shoul, M. (2010). Alkhalil Morpho Sys1: A morphosyntactic analysis system for Arabic texts. In *Proceedings of the International Arab Conference on Information Technology*, pages 1–6.

Buckwalter, T. (2004). Buckwalter Arabic Morphological Analyzer Version 2.0. LDC catalog number LDC2004L02, ISBN 1-58563-324-0.

Darwish, K. and Mubarak, H. (2016). Farasa: A new fast and accurate Arabic word segmenter. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Portorož, Slovenia.

Darwish, K., Mubarak, H., Abdelali, A., and Eldesouki, M. (2017). Arabic pos tagging: Don't abandon feature engineering just yet. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 130–137, Valencia, Spain.

Darwish, K., Mubarak, H., Abdelali, A., Eldesouki, M., Samih, Y., Alharbi, R., Attia, M., Magdy, W., and Kallmeyer, L. (2018). Multi-dialect Arabic pos tagging: A CRF approach. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Diab, M., Hacioglu, K., and Jurafsky, D. (2004). Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 149–152, Boston, MA.

Diab, M., Hacioglu, K., and Jurafsky, D., (2007). *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, chapter Automated Methods for Processing Arabic Text: From Tokenization to Base Phrase Chunking. Springer Netherlands, kluwer/springer edition.

Duh, K. and Kirchhoff, K. (2005). POS tagging of dialectal Arabic: a minimally supervised approach. In *Proceedings of the Workshop on Computational Approaches to Semitic Languages (CASL)*, pages 55–62, Ann Arbor, Michigan.

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

Erdmann, A., Khalifa, S., Oudah, M., Habash, N., and Bouamor, H. (2019). A little linguistics goes a long way: Unsupervised segmentation with limited language specific guidance. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 113–124, Florence, Italy, August. Association for Computational Linguistics.

Eryani, F., Habash, N., Bouamor, H., and Khalifa, S. (2020). A Spelling Correction Corpus for Multiple Arabic Dialects. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Marseille, France.

Eskander, R., Habash, N., and Rambow, O. (2013a). Automatic extraction of morphological lexicons from morphologically annotated corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1032–1043, Seattle, Washington, USA.

Eskander, R., Habash, N., Rambow, O., and Tomeh, N. (2013b). Processing spontaneous orthography. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 585–595, Atlanta, Georgia.

Eskander, R., Habash, N., Rambow, O., and Pasha, A. (2016). Creating resources for Dialectal Arabic from a single annotation: A case study on Egyptian and Levantine. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 3455–3465, Osaka, Japan.

Graff, D., Maamouri, M., Bouziri, B., Krouna, S., Kulick, S., and Buckwalter, T. (2009). Standard Arabic Morphological Analyzer (SAMA) Version 3.1. Linguistic Data Consortium LDC2009E73.

Guzmán, F., Bouamor, H., Baly, R., and Habash, N. (2016). Machine translation evaluation for Arabic using morphologically-enriched embeddings. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 1398–1408, Osaka, Japan. The COLING 2016 Organizing Committee.

Habash, N. and Rambow, O. (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, pages 573–580, Ann Arbor, Michigan.

Habash, N. and Rambow, O. (2006). MAGEAD: A morphological analyzer and generator for the Arabic dialects. In *Proceedings of the International Conference on Computational Linguistics and the Conference of the Association for Computational Linguistics (COLING-ACL)*, pages 681–688, Sydney, Australia.

Habash, N., Soudi, A., and Buckwalter, T. (2007). On Arabic Transliteration. In A. van den Bosch et al., editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, pages 15–22. Springer, Netherlands.

Habash, N., Rambow, O., and Roth, R. (2009). MADA+TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In Khalid Choukri et al., editors, *Proceedings of the International Conference on Arabic Language Resources and Tools*, Cairo, Egypt. The MEDAR Consortium.

Habash, N., Eskander, R., and Hawwari, A. (2012). A Morphological Analyzer for Egyptian Arabic. In *Proceedings of the Workshop of the Special Interest Group on Computational Morphology and Phonology (SIGMORPHON)*, pages 1–9, Montréal, Canada.

Habash, N., Roth, R., Rambow, O., Eskander, R., and Tomeh, N. (2013). Morphological Analysis and Disambiguation for Dialectal Arabic. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Atlanta, Georgia.

Habash, N., Eryani, F., Khalifa, S., Rambow, O., Abdulrahim, D., Erdmann, A., Faraj, R., Zaghouani, W., Bouamor, H., Zalmout, N., Hassan, S., shargi, F. A., Alkhereyf, S., Abdulkareem, B., Eskander, R., Salameh, M., and Saddiki, H. (2018). Unified guidelines and resources for Arabic dialect orthography. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Habash, N. (2007). Arabic Morphological Representations for Machine Translation. In Antal van den Bosch et al., editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Kluwer/Springer.

Halabi, N. (2016). *Modern standard Arabic phonetics for speech synthesis*. Ph.D. thesis, University of Southampton.

Hamdi, A., Nasr, A., Habash, N., and Gala, N. (2015). POS-tagging of Tunisian dialect using standard Arabic resources and tools. In *Proceedings of the Second Workshop on Arabic Natural Language Processing*, pages 59–68, Beijing, China, July. Association for Computational Linguistics.

Heigold, G., Genabith, J. v., and Neumann, G. (2016). Scaling character-based morphological tagging to fourteen languages. In *Proceedings of the International Conference on Big Data (Big Data)*, pages 3895–3902.

Inoue, G., Shindo, H., and Matsumoto, Y. (2017). Joint Prediction of Morphosyntactic Categories for Fine-Grained Arabic Part-of-Speech Tagging Exploiting Tag Dictionary Information. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, pages 421–431, Vancouver, Canada.

Jarrar, M., Habash, N., Akra, D., and Zalmout, N. (2014). Building a Corpus for Palestinian Arabic: A Preliminary Study. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 18–27, Doha, Qatar.

Khalifa, S., Habash, N., Abdulrahim, D., and Hassan, S. (2016a). A Large Scale Corpus of Gulf Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Portorož, Slovenia.

Khalifa, S., Zalmout, N., and Habash, N. (2016b). Yamama: Yet another multi-dialect Arabic morphological analyzer. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 223–227, Osaka, Japan.

Khalifa, S., Hassan, S., and Habash, N. (2017). A Morphological Analyzer for Gulf Arabic Verbs. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, Valencia, Spain.

Khalifa, S., Habash, N., Eryani, F., Obeid, O., Abdulrahim, D., and Kaabi, M. A. (2018). A morphologically annotated corpus of emirati Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Khoja, S. (2001). APT: Arabic Part-of-Speech Tagger. In *Proceedings of NAACL Student Research Workshop (SRW)*, pages 20–26, Pittsburgh.

Kilany, H., Gadalla, H., Arram, H., Yacoub, A., El-Habashi, A., and McLemore, C. (2002). Egyptian Colloquial Arabic Lexicon. LDC catalog number LDC99L22.

Maamouri, M., Bies, A., Buckwalter, T., and Mekki, W. (2004). The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *Proceedings of the International Conference on Arabic Language Resources and Tools*, pages 102–109, Cairo, Egypt.

Maamouri, M., Graff, D., Bouziri, B., Krouna, S., Bies, A., and Kulick, S. (2010). Standard Arabic morphological analyzer (sama) version 3.1. *Linguistic Data Consortium, Catalog No.: LDC2010L01*.

Maamouri, M., Bies, A., Kulick, S., Tabessi, D., and Krouna, S. (2012). Egyptian Arabic Treebank DF Parts 1-8 V2.0 - LDC catalog numbers LDC2012E93, LDC2012E98, LDC2012E89, LDC2012E99, LDC2012E107, LDC2012E125, LDC2013E12, LDC2013E21.

Mohit, B., Rozovskaya, A., Habash, N., Zaghouani, W., and Obeid, O. (2014). The first QALB shared task on automatic text correction for Arabic. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 39–47, Doha, Qatar.

Obeid, O., Khalifa, S., Habash, N., Bouamor, H., Zaghouani, W., and Oflazer, K. (2018). MADARi: A Web Interface for Joint Arabic Morphological Annotation and Spelling Correction. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

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 Language Resources and Evaluation Conference (LREC)*, pages 1094–1101, Reykjavik, Iceland.

Roth, R., Rambow, O., Habash, N., Diab, M., and Rudin, C. (2008). Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, Columbus, Ohio.

Sadat, F. and Habash, N. (2006). Combination of Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proceedings of the International Conference on Computational Linguistics and the Conference of the Association for Computational Linguistics (COLING-ACL)*, pages 1–8, Sydney, Australia.

Salloum, W. and Habash, N. (2014). ADAM: Analyzer for Dialectal Arabic Morphology. *Journal of King Saud University - Computer and Information Sciences*, 26(4):372–378.

Samih, Y., Attia, M., Eldesouki, M., Abdelali, A., Mubarak, H., Kallmeyer, L., and Darwish, K. (2017a). A neural architecture for dialectal Arabic segmentation. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 46–54, Valencia, Spain.

Samih, Y., Eldesouki, M., Attia, M., Darwish, K., Abdelali, A., Mubarak, H., and Kallmeyer, L. (2017b). Learning from relatives: Unified dialectal Arabic segmentation. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 432–441, Vancouver, Canada, August. Association for Computational Linguistics.

Shen, Q., Clothiaux, D., Tagtow, E., Littell, P., and Dyer, C. (2016). The role of context in neural morphological disambiguation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 181–191, Osaka, Japan. The COLING 2016 Organizing Committee.

Watson, D., Zalmout, N., and Habash, N. (2018). Utilizing character and word embeddings for text normalization with sequence-to-sequence models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 837–843, Brussels, Belgium, October-November. Association for Computational Linguistics.

Zalmout, N. and Habash, N. (2017). Don't throw those morphological analyzers away just yet: Neural morphological disambiguation for Arabic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 704–713, Copenhagen, Denmark.

Zalmout, N. and Habash, N. (2019). Joint diacritization, lemmatization, normalization, and fine-grained morphological tagging. *arXiv preprint arXiv:1910.02267*.

Zribi, I., Ellouze Khemakhem, M., and Hadrich Belguith, L. (2013). Morphological analysis of Tunisian dialect. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 992–996, Nagoya, Japan, October. Asian Federation of Natural Language Processing.