Part-of-Speech and Morphological Tagging of Algerian Judeo-Arabic

Ofra Tirosh-Becker[1*] Michal Kessler[2*] Oren M. Becker[3] Yonatan Belinkov[4†]

[1] The Hebrew University of Jerusalem, Israel. otirosh@mail.huji.ac.il
[2] The Hebrew University of Jerusalem, Israel. michalskessler@gmail.com
[3] Becker Consulting Ltd., Mevasseret Zion, Israel. becker.oren@gmail.com
[4] Technion – Israel Institute of Technology, Israel. belinkov@technion.ac.il

Abstract Most linguistic studies of Judeo-Arabic, the ensemble of dialects spoken and written by Jews in Arab lands, are qualitative in nature and rely on laborious manual annotation work, and are therefore limited in scale. In this work, we develop automatic methods for morpho-syntactic tagging of Algerian Judeo-Arabic texts published by Algerian Jews in the 19th–20th centuries, based on a linguistically tagged corpus. First, we describe our semi-automatic approach for preprocessing these texts. Then, we experiment with both an off-the-shelf morphological tagger, several specially designed neural network taggers, and a hybrid human-in-the-loop approach. Finally, we perform a real-world evaluation of new texts that were never tagged before in comparison with human expert annotators. Our experimental results demonstrate that these methods can dramatically speed up and improve the linguistic research pipeline, enabling linguists to study these dialects on a much greater scale.

1 Introduction

Application of Natural Language Processing (NLP) to real-world problems has been the field's goal from its early days. As algorithms advance, the contribution of NLP to real problems has become more evident and more substantial. The present study originates from a real-world challenge faced by linguists of Semitic languages, in this case researchers of the Judeo-Arabic dialects of Algeria (AJA). Their challenge, simply put, is how to scale up linguistic analyses of such dialects. Semitic languages in general, and Arabic in particular, are characterized by a very rich morphology that uses both templatic and concatenative morphemes, combined with the use of a vowelless script ("*abjad*"). This makes morphological analysis of Arabic very time-consuming even for expert linguists. Because speakers of the AJA dialects are becoming scarce, the attention of linguists in this field has shifted from fieldwork interviews with native speakers to library-based analysis of texts written in those dialects. Fortunately, vast collections of AJA texts were preserved in printed books, journals and handwritten manuscripts. Analyzing this linguistic treasure-trove, however, is proving to be challenging due to its size. The time-consuming manual annotation does not scale, and requires expertise that is hard to find.

We aim to scale up the linguistic analysis of this Arabic dialect using NLP tools. In particular, our goal is to develop an NLP tool that will assist AJA linguists in their *real-world task*, in a way that *they* will find it useful. Basing our work on the existing linguistically Tagged Algerian Judeo-Arabic (TAJA) corpus (Tirosh-Becker and Becker, 2022), we set out to develop automatic methods for morpho-syntactic tagging of such texts. Several specially designed neural network taggers and an off-the-shelf morphological tagger were experimented with, and assessed for their accuracy and likely usefulness. We also considered a hybrid human-in-the-loop approach. Finally, we carried out a *real-world evaluation* of our best performing part-of-speech (POS) taggers, applying them to untagged texts and assessing their quality via a user study with expert AJA linguists. Our experimental results demonstrate that these methods can dramatically speed up and improve the linguistic research pipeline, enabling linguists to study this language on a much greater scale.

[*] Equal contribution

[†] Supported by the Viterbi Fellowship in the Center for Computer Engineering at the Technion.

2 Linguistic Background

Judeo-Arabic (JA) lies in the intersection of Semitic languages and Jewish languages. As a Semitic language, and more specifically, an Arabic language variety, its words are generally composed of 3-letter roots, with added vowels and consonants according to pattern paradigms, as well as affixes and clitics (McCarthy, 1981). Arabic is the most widely spoken Semitic language, with 300 million native speakers (Owens, 2013). In fact, the term 'Arabic' refers both to Modern Standard Arabic (MSA) and to the Arabic dialects spoken throughout the Arab World. The two varieties of Arabic coexist in a state of diglossia (Ferguson, 1959) or continuuglossia (Hary, 2003), meaning the language varieties exist side by side, with writers or speakers shifting between varieties according to circumstance. MSA is written using the Arabic script, which is a right-to-left alphabet. Arabic dialects are usually written in Arabic script as well, but there is no standardized spelling for dialectal Arabic (Habash et al., 2012).

Arabic uses both templatic and concatenative morphemes. There are two types of templatic morphemes: roots and templates. Roots are usually three consonantal radicals that signify some abstract meaning. Roots are inserted into abstract patterns called templates.

There are two kinds of concatenative morphemes that attach to the templatic morphemes. Clitics are morphemes that have the syntactic characteristics of words, but are phonologically bound to another word (Zitouni, 2014), for example "*wa*",[1] meaning "and". Affixes are phonologically and syntactically part of the word, and often represent inflectional features, such as person, gender, number, and more.

Dialectal Arabic (DA) is a primarily spoken family of language varieties (and in modern days, widely used in written form on social media as well) that exist alongside the written MSA. DA diverges from MSA on several levels. There are differences in phonology, morphology, lexicon, and orthography (Habash et al., 2012). The regional dialects can be broken down into main groups, with one possible breakdown being Egyptian, Levantine, Gulf, Iraqi, and Maghrebi. Even within dialect groups there can be quite a lot of variance between dialects, although in many cases there is a certain level of intelligibility between speakers of different dialects, with more significant difficulty across dialect groups. Maghrebi dialects are influenced by the contact with French and Berber languages, and the Western-most varieties could be unintelligible by speakers from other regions in the Middle East, especially in spoken form (Zaidan and Callison-Burch, 2014).

[1] We use the Habash-Soudi-Buckwalter transliteration scheme (Habash et al., 2007) for Arabic text. For AJA texts, we use the common transliteration of JA; see Table 9 in the appendix.

While JA can be looked at as an ensemble of Arabic dialects, it is first and foremost a subgroup of Jewish languages. Jewish languages are a family of language varieties that developed in Jewish communities throughout the diaspora. The original language used by Jews in the Land of Israel was Hebrew, followed closely by Aramaic. As Jews spread across the world, they adopted local languages and developed distinctive varieties of these languages. Nonetheless Hebrew remained their liturgical language, even as it almost died out as a spoken language until its revival in the late 19th and early 20th centuries. Perhaps the most well-known of these Jewish languages is Yiddish, the Judeo-German language developed by Ashkenazi Jews living in Central and Eastern Europe before the Holocaust. Jewish languages vary in their distance and divergence from their non-Jewish sister languages, some being influenced by multiple languages due to language contact. Nonetheless, among the features that tie these languages together are the presence of Hebrew and Aramaic lexical components (Kahn and Rubin, 2017), the use of the Hebrew alphabet for writing, and more.

Algerian JA (AJA) is a member of the North African Judeo-Arabic dialect group, i.e., dialects spoken and written by Jews of the Maghreb. AJA is in contact with Moroccan and Tunisian Arabic dialects (both Jewish and Muslim), with French and to a lesser extent other trade languages such as Spanish and Italian, and with Hebrew and Aramaic, the historical Jewish cultural languages. In general AJA shares many characteristics with other Jewish languages, including the use of Hebrew script, presence of Hebrew and Aramaic components, and a mixture of conservative trends, vernacular features, and heterogeneous elements (Tirosh-Becker, 2012). To date, AJA has been sparsely studied by linguists. The AJA dialect of the city of Algiers was studied over a century ago by Cohen (1912), with most of the recent work on AJA published by Tirosh-Becker, focusing on Constantine, the third largest city in Algeria (Tirosh-Becker, 1988, 1989, 2011a,b, 2014). AJA research employs fieldwork interviews of informants and the study of selected written texts (e.g., Bar-Asher, 1992; Tedghi, 2012; Tirosh-Becker, 2011a,c, 2012). Regretfully, the number of AJA speakers has decreased following Algeria's independence (in 1962) and the subsequent dispersion of its Jewish communities, making fieldwork today almost impossible. Hence, this research is now shifting towards an analysis of the vast textual sources left by many of these Jewish communities, in both manuscript and print form. Most of the linguistic analyses done thus far on AJA texts have been based on single or few texts, as each study requires extended effort of poring over texts, dictionaries, and grammars. Given the size of these corpora, this is a perfect match for machine learning and NLP approaches.

3 Related Work

3.1 Arabic Corpora

Corpora for Arabic NLP are usually gathered with a specific language variety in mind, and optionally annotated with information for specific tasks. We briefly discuss here the most prominent and relevant Arabic corpora, and refer to Belinkov (2021) for a broader survey. Masader (Alyafeai et al., 2022; Altaher et al., 2022) is an online catalogue of Arabic NLP datasets.

The majority of annotated Arabic corpora are for MSA. The most prominent annotated MSA corpora are the Penn Arabic Treebank (PATB; Maamouri et al., 2004), and the Prague Arabic Dependency Treebank (PADT; Hajič et al., 2009), a dependency treebank for MSA. In addition, El-Haj and Koulali (2013) present KALIMAT, a multipurpose corpus for MSA, with over 20,000 articles and over 18 million words, annotated using existing state-of-the-art Arabic NLP tools for POS tags, morphological analyses, named entity recognition (NER), and auto-summarization.

There are also annotated corpora for DA. ATB-ARZ (Maamouri et al., 2014) is an Egyptian Arabic treebank, with 182,965 tokens after clitic splitting. This corpus is annotated for POS, morphology, gloss, and syntactic treebank, following the guidelines of the PATB. There are several corpora for dialect identification that include Algerian and other Maghrebi dialects, such as Habibi (El-Haj, 2020), a corpus of Arabic song lyrics, or QADI (Abdelali et al., 2021). Seddah et al. (2020) created the NArabizi corpus, North African Arabic written in Latin letters (commonly known as Arabizi), with 1500 sentences of annotated Algerian dialectal Arabic, with tokenization, morphological analysis, code-switching identification, syntactic annotations, and sentence-level translations in French. MADAR (Bouamor et al., 2018) has 12,000 sentences with parallel translations in mutliple dialects, including Algerian and other North African dialects, but without morphological annotations. In addition, Zribi et al. (2015) transcribed and annotated a spoken Tunisian Arabic corpus, a North African dialect that is close to Algerian Arabic. It is worth noting that many DA corpora are transcribed from audio sources and are not originally textual data.

As for Judeo-Arabic corpora, the only publicly available JA corpus to date is the Friedberg Judeo-Arabic Project,[2] with almost 4 million words from 110 pre-modern JA texts, including texts by Rav Saadia Gaon and Maimonides. The only annotation available for these words is language (Arabic, or Hebrew/Aramaic). Recently, Tirosh-Becker and Becker (2022) developed the TAJA (Tagged Algerian Judeo-Arabic) corpus, a linguistically annotated corpus of written Algerian Judeo-Arabic. This corpus is a collection of modern AJA texts published in Algeria in the late 19th and the first half of the 20th century. Section 4 provides a detailed description of the TAJA corpus, on which this paper is based.

[2] https://fjms.genizah.org/

3.2 Arabic POS Tagging and Morphological Analysis

Much of the work done on POS tagging in Arabic has used statistical methods. Diab (2009) uses an SVM classifier for choosing POS tags on MSA. MADAMIRA (Pasha et al., 2014), trained on the MSA PATB, is often used as a benchmark for Arabic POS tagging. It uses a morphological analysis component as part of the preprocessing stage, and then uses SVM and language models to predict POS tags, as well as tokenization, NER, and other tasks. Farasa is another Arabic NLP tool with support for POS tagging in MSA and DA, which is based on conditional random fields (Abdelali et al., 2016; Darwish et al., 2018). In recent work, deep neural networks have been used to train POS and morphological taggers. Plank et al. (2016) built POS taggers for 22 languages, including Arabic, using data from the Universal Dependencies project (Nivre et al., 2015). They experiment with using word embeddings, character embeddings, byte embeddings, and some combinations thereof. Their best performing model does especially well on Arabic, reaching up to 98.91% accuracy.

Works that cover DA often leverage tools developed on or for MSA. Duh and Kirchhoff (2005) propose a minimally supervised approach for POS tagging of DA that combines raw text data from several varieties of Arabic, and a morphological analyzer for MSA with no other dialect-specific tools. Habash et al. (2013) tweak the MSA morphological analyzer MADA (Roth et al., 2008) for analyzing Egyptian DA, rather than the original MSA. They achieve up to 84.5% accuracy on morphological tags and 90.1% on Penn POS tags.

Other studies that address both MSA and DA have used bi-LSTMs for morphological tagging, sometimes jointly with other tasks like diacritization (Zalmout and Habash, 2020, 2019). Very recently, Inoue et al. (2022) have shown benefits from using pre-trained Transformer language models, especially when transferring from high- to low-resource dialects or language varieties, outperforming previous approaches.

Darwish et al. (2020) introduce a robust multi-dialect POS tagging system trained on tweets from four different dialect groups. They implement two approaches: the first uses CRFs, and the second stacks layers of CNNs, recurrent neural networks (RNNs), and a CRF layer. Their dataset comprises hundreds of tweets in each dialect group, each manually segmented into tokens and clitics. They make use of stem templates and Brown clusters as features concatenated to the embeddings for classification, and achieve accuracy of up to 92.4% on the POS tagging of seen words and

source text reference	line number	context	word
Gn_avot_4:16	2	אדנייא האדי תשבאה לסקיפ׳א ʾdnyyʾ hʾdy tšbʾh lsqyfʿ	אדנייא ʾdnyyʾ
lemma/root	POS	morphological analysis 1	morphological analysis 2
דנייא dnyyʾ	noun	feminine	signular
additional tags	enclitic pronoun	comments	orthography and pronunciation
NA	NA		1. yy denotes consonantal ya' 2. Phonetic transcription of the definite article

Table 1: The general structure of a word-record in the TAJA corpus with a specific example. The words and context are stored in the word-record in their original Hebrew script; transliterations are added here for clarity.

82.9% on unseen words in Maghrebi dialects.

Given the orthographic, grammatical, and lexical differences between JA on the one side and MSA and other Arabic dialects on the other side, it is not straightforward to apply tools developed for MSA and Arabic-script DA to processing JA. Future work may investigate ways to transfer such tools or incorporate them with JA-dedicated tools. Efforts to transliterate JA texts to Arabic script (Terner et al., 2020) may assists in pursuing this direction.

3.3 Code-Switching

While there is no work known to us applying NLP to JA, there is work on code-switching, which is a significant characteristic of JA, as we noted in Section 2. Code-switching is when a speaker alternates between two or more languages or dialects in the context of a single conversation or situation. Ahmed (2018) annotates Hebrew elements in JA, capturing cases of code-switching, borrowing, and Hebrew quotations, and investigating sociolinguistic aspects in medieval JA texts. Wagner and Connolly (2018) perform a quantitative analysis of code-switching in JA texts from the Cairo Geniza.

As Çetinoğlu et al. (2016) point out, POS tagging of code-switched data is much harder than tagging monolingual texts, as models could reach 97% accuracy for the latter, but as low as 77% for the former. Attia et al. (2019) find that POS tags provide a strong signal for identifying code-switching. Just as code-switching is a major characteristic of AJA, it also characterizes other varieties of Algerian Arabic, and poses a challenge to Arabic NLP research (Riabi et al., 2021).

4 Data

This project has used the Tagged Algerian Judeo-Arabic (TAJA) corpus developed by Tirosh-Becker and Becker (2022).[3] This AJA corpus is a collection of modern AJA texts published in Algeria in the late 19th and the first half of the 20th century. The texts represent a variety of prose genres written by Algerian Jews, including:

- Bible translations, known as *šarḥ* (sg.) or *šurūḥ* (pl.).
- Translations of Hebrew post-biblical texts (such as the Mishnah, the Passover Haggadah, and liturgical poems).
- Translations of other Hebrew texts (such as Maimonides' *Mishne Torah*).
- Original writings composed in AJA, including commentaries and writings about Jewish law.
- Journalistic writings in AJA.

These texts were manually typed into computer-readable format and subsequently proofread, as Hebrew OCR (Optical Character Recognition) failed on these AJA texts. This was due not only to the less-than-favorable conditions under which the books had been stored, leaving the pages grayed and worn, but also because the fonts used in these books are not identical to standard Hebrew, as they have JA-specific adaptations, such as diacritics. Each text was manually tokenized and annotated by research assistants (RAs, usually MA or PhD candidates) in a spreadsheet, according to strict guidelines, and most were verified by a senior expert.

The digitization and annotation project spanned several years, with some dozen RAs contributing to the annotation efforts. Approximately 80% of the time spent on the creation of TAJA was dedicated to the annotation process, as the digitization is a more straightforward (though non-trivial) task.

4.1 Data Annotation

The TAJA corpus was created to be a *linguistically annotated* digital corpus of *genre-diverse written* texts (Tirosh-Becker and Becker, 2022). The basic elements in this digital corpus are the individual words. Generally speaking, the texts are split on white-spaces, though there are some multi-word expressions that are annotated as a single unit. Each word is stored in a sort of *word-record*, which places the word in its sentence-level context (as well as a reference to the full text), and provides linguistic information about its grammatical components and more (Table 1).

[3]The corpus is available through the authors.

4.1.1 Parts of Speech (POS)

Each word is tagged with a unique POS tag. The tags are drawn from a closed list of the following POS: noun, verb, particle, proper noun, relative pronoun, adjective, number, personal pronoun, demonstrative, adverb, presentative, quantifier, and acronym. POS tagging was also applied to the embedded Hebrew, Aramaic, and French words, which are identified in the TAJA corpus by code-switching tags, as these embedded words are interwoven into the syntactic fabric of JA. In almost all cases these code-switching words were nouns. See Table 10 (Appendix) for a list of valid POS tags.

4.1.2 Morphological Tags

The morphology of each word was fully analyzed by expert JA linguists. Each POS tag calls for its own set of morphological features. Given a noun, for example, we expect information about gender, number, and code-switching. The fields in our dataset in which we find this morphological information are *analysis1*, *analysis2*, *additional tags* and *enclitic pronouns*. Note that there is a clear ranking between these fields. Most of the morphological information is captured by the first two fields, *analysis1* and *analysis2*, reflecting the rich morphology of AJA, while the *additional tags* field refers to a small subset of morphological attributes that apply only to a limited number of POS tags, i.e., to verbs (combinations of person, gender, and number) or to demonstratives (proximal vs. distal). The information provided by the *enclitic pronouns* field is morphologically more restricted. Each POS tag generally has its own set of legal values for these analyses, and they do not often overlap with the legal morphological annotations of other POS tags. In fact, at times, the same linguistic information may appear in different annotation fields for different POS tags. For example, code-switching information for nouns appears in the *analysis1* field, but the same information for proper nouns appears in *analysis2*. See Tables 11–14 (Appendix) for lists of valid morphological tags for the prominent POS tags.

4.2 Corpus Statistics

TAJA is comprised of 69 spreadsheet files, which cover 16 printed texts. These include 9904 AJA sentences, with a total of 61,481 tokens. There are 17,876 different word types in the corpus, for a type–token ratio (TTR) of 0.2907. It is important to recall that AJA is a highly morphological language, with extensive use of affixes and clitics. For example, a word is marked as definite using the prefix *ʾl-*. The same is true for several prepositions, such as *b-* ("in" or "at") or *l-* ("to" or "for"). Thus, a single lemma with two different prefixes will be counted as two distinct word types, so the reported number of word types in fact represents fewer lemmas.

	Tokens	Types
Surface	20.89%	37.37%
Lemmas	5.34%	16.88%

Table 2: Out-of-vocabulary percentages for tokens and types, by surface level words and for lemmas.

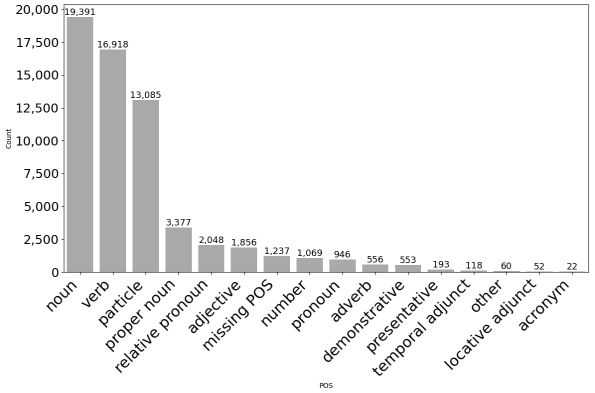

Figure 1: Part-of-speech tag distribution for the TAJA corpus.

As for the use of the term "lemma", the verbs in TAJA are tagged for the root rather than their lemma. In addition, approximately 2.1% of words in TAJA are missing the annotation in the lemma field, and are therefore left out of statistical calculations we report below at the level of lemmas. These issues limit our ability to provide accurate statistics on a lemma level.

For the 90/10 training/test split of TAJA with which we work in our experiments, we see high out of vocabulary (OOV) percentages for surface-level words (Table 2). When looking at word types, we see that more than a third of surface-level word types in the test set did not appear in the training set. Recall that this includes words that appear in the training set with an affix (such as a determiner, for example), and appear in the test set without said affix (or vice versa). We also look at the lemma OOV percentage, despite what we explained above about verbs being annotated for root instead of lemma. There is a large portion of OOV lemma types in the test set. These characteristics illustrate the diversity of the data in both lexical and surface form levels.

Finally, the data suffer from a long-tailed distribution of the annotations, a common problem in NLP. When examining the distribution of POS tags, for example, the three most common POS tags (noun, verb, and particle) account for approximately 80% of the annotations (see Figure 1), while the other 11 valid POS tags comprise only a fifth of the annotations.

4.3 Corpus Ambiguity

Before we discuss the ambiguity statistics of the TAJA corpus, we must address the noisiness of the data. Despite the laborious annotation effort (Tirosh-Becker and Becker, 2022), the data still contain problematic annotations. We discuss our attempts to clean the data below, but at this point, it is enough to know that some annotations appear with typos, or with additions that should not be there, such as question marks.

Corpus ambiguity is defined in Dermatas and Kokkinakis (1995) as the mean number of possible tags for each word of the corpus. This number can provide a signal for the difficulty of the tagging task. The corpus ambiguity of TAJA, calculated on the surface-level tokens, is 1.7497. This is high relative to the corpus ambiguity of other language corpora, as reported in Dermatas and Kokkinakis (1995), which range from 1.11 (for Dutch) to 1.69 (for French). However, as we mentioned, the noise in the annotations makes this number unreliable. For example, if a word that appears many times in the corpus appears one time with a typo in the annotation, this will raise the corpus ambiguity unjustifiably.

4.4 End-Goal: The Unannotated NAJA Corpus

In addition to TAJA, there exists a larger unannotated corpus of digitized AJA text. The New Algerian Judeo-Arabic (NAJA) corpus includes the same genres as TAJA, though differently distributed. The estimated size of NAJA is between 170k-186k tokens, almost three times as many as TAJA. It is the laborious task of manually annotating this corpus that we wish to automate away, using taggers trained on TAJA.

5 Preprocessing

In this section, we describe several challenges we faced in the preprocessing stage and the steps we took to address them.

5.1 Invalid annotations

Although the annotators were provided with the list of legal tags and legal morphological annotations, the data are rife with 'illegal' values, including mistyped tags (e.g., צל instead of מל),[4] two tags combined into one (מל+שע rather than שע), annotations with question marks and slashes (indicating that they are not confident about the tag they chose) and most often, words that are simply missing a tag.

[4]This mistyping is caused by the letter צ (ṣ) being adjacent to the letter מ (m) on the Hebrew keyboard. Table 10 (Appendix) provides the list of POS codes and their meanings.

We took a semi-automated approach for correcting as many annotations as possible. We created a mapping from misspelled or mistyped tags to the correct spelling. This was an iterative process, as at each iteration new categories of errors emerged, requiring additional consultation with JA language experts. For example, when resolving combined tags (as we described above), it is not obvious that it is desirable to drop the information represented by either of the tags. Being able to automate away the correct or obvious cases, enabled us to narrow down the number of questions we needed to bring to the experts, and conversely, having a language expert to whom we could bring the difficult questions, allowed us to ensure the annotations are as accurate as possible. Upon loading the spreadsheets and ingesting the data, we automatically convert any incorrect tags that appear in our mappings to the correct tags. These mappings catch 662 errors that are automatically corrected as part of the preprocessing stage. Using regular expressions we collected the cases of low confidence annotations (indicated by question marks or slashes in the original spreadsheets), and sent them for review by the language experts. Most of these were corrected manually in the original spreadsheets, in addition to some errors found in the enclitic pronouns, for a total of 64 manual corrections. Finally, missing annotations are represented with an underscore.

5.2 Column offsets

Another kind of noise we encountered in the annotated data are column offsets (e.g., the POS tag appears in the *analysis1* column, and so on). During preprocessing, we check automatically for such offsets in the columns, and automatically realign the annotations to their correct fields while parsing. We found 64 such cases, and fixed them automatically.

5.3 Multi-word expressions

The spreadsheet input includes the tokenization of each sentence, listing each token on a separate line, where each sentence is separated from the next by an empty line. In most cases, the tokenization is done on whitespaces. However, on various occasions, a multi-word expression appears on a single line and is annotated as a single unit. This happens most commonly with proper nouns, such as ראש השנה (*rʾš hšnh*, 'the New Year'; this is also a Hebrew construct phrase) or יהושע ולד נון (*yhwšʿ wld nwn*, 'Joshua son of Nun'), or Hebrew phrases such as the phrase בעולם הזה (*bʿwlm hzh*, 'in this world'; includes a noun) or ועשה טוב (*wʿsh ṭwv*, 'and do good'; includes a verb). These multi-word expressions are most often Hebrew phrases or terms that are embedded in the AJA text. They are treated as a single word in TAJA, because they represent a single concept or entity. How-

ever, this potentially poses a problem, as the tokenization we perform on new texts is based on white-spaces and punctuation, and therefore when coming to annotate previously unseen texts with multi-word expressions, the tagger will address each component of the phrase as its own word, and it might be considered 'out of vocabulary' as far as our tagger is concerned. However, this is a very rare phenomenon, with fewer than 100 appearances in the entire corpus, and therefore we did not split multi-word phrases in our experiments.

6 Methods

6.1 The Tasks

We formulate both part-of-speech (POS) and morphological tagging as sequence labeling tasks. In the POS tagging task, we are given an input sentence of n words, denoted by $\mathrm{x} = w_1, \ldots, w_n$, and need to find the correct sequence of tags $\mathrm{t} = t_1, \ldots, t_n$, where t_i is taken from the set of POS tags T (Table 10, Appendix). Morphological tagging is performed on the same input as the POS tagging task. In this task, there are four morphological fields (*analysis1*, *analysis2*, *additional tags*, *enclitic pronoun*) to be tagged in addition to the POS tag field: $\mathrm{t}^1 = t_1^1, \ldots, t_n^1$, $\mathrm{t}^2 = t_1^2, \ldots, t_n^2$, $\mathrm{t}^3 = t_1^3, \ldots, t_n^3$, $\mathrm{t}^4 = t_1^4, \ldots, t_n^4$. Tables 11–14 (Appendix) contain lists of valid morphological tags for the prominent POS tags.

6.2 Models

We experiment with two types of models for the sequence labeling tasks: CRFs and RNNs. CRFs (Lafferty et al., 2001) are a framework for building probabilistic models for segmenting and labeling sequence data, while relaxing strong independence assumptions made by hidden Markov models (HMMs), and avoiding certain biases that maximum entropy Markov models (MEMMs) are prone to have. Parameter estimation is done by maximum likelihood estimation and the Viterbi algorithm is used for inference. CRFs use handcrafted features, such as the preceding and succeeding words, prefixes and suffixes, and more. In this study we experiment with MarMoT, an off-the-shelf tool that implements a pruned CRF model which has performed well on Modern Standard Arabic (Müller et al., 2013).

In addition to the standard MarMoT tool, we implement our own tagging model based on long short-term memory networks (LSTM; Hochreiter and Schmidhuber, 1997), a type of RNN that is more robust to the vanishing gradients problem and performs well on sequence-level tasks. Our backbone is a bi-directional LSTM model based on the PyTorch implementation (Paszke et al., 2019). On top of that, we add a linear layer that maps the hidden representations to the output space: either the space of all POS tags or the space of each of the morphological tag classes. Below we describe several improvements to this basic architecture.

6.3 Word-based vs. Character-based

Our basic LSTM architecture receives a sentence as input, and, using an embedding matrix for the words, passes the word embedding vectors $x_1 \ldots x_n$ through the LSTM one after another. However, this method has no way to deal with out-of-vocabulary (OOV) words, which are all mapped to a single 'UNKNOWN' token, and therefore to the same word embedding. It must use contextual information alone from neighboring words. OOV words are especially common in morphologically rich languages like AJA, as is evident from the corpus statistics (Section 4.2). To account for the highly morphological nature of AJA, it is important to address the characters on an individual level, as has been shown for other languages (Dos Santos and Zadrozny, 2014; Ling et al., 2015; Ballesteros et al., 2015). Looking at characters separately from words helps tag OOV words, because we can identify certain affixes that provide a strong signal about one of the annotations. For example, words starting with אל (ʾl), א (ʾ), or ל (l)[5] are more likely to be nouns.

For this purpose, we created two character-aware models. Both models train embeddings for the characters, but use different methods to create a word representation given the character embeddings.[6] Let the k^{th} word of sentence x be $w_k = c_{k,1}, c_{k,2}, \ldots, c_{k,m}$ (for ease of notation, $c_{k,i}$ represents both characters and character embeddings). The first method builds on the idea proposed by (Luong and Manning, 2016), and passes each word w_k's characters through an inner character-LSTM. The final hidden state $h_{k,m}$ of the character-LSTM is a character-aware word representation, which is concatenated to that word's embedding x_k. The combined representation $\tilde{x}_k = (x_k, h_{k,m})$ is fed to the word-level LSTM. We call this model CHAR-LSTM.

The second method follows (Kim et al., 2016), and uses a one-dimensional convolutional neural network (CNN), with a hyperparametric number of kernels K that convolve with the matrix of each word w_k's character embeddings. We apply a $\tanh$ non-linearity to the convolution outputs, and then pool the maximal values of each output to create a single character-based representation for each word, h_k. This representation is concatenated to the word's embedding. The combined representation $\tilde{x}_k = (x_k, h_k)$ is fed to the word-level LSTM. We call this model CNN.

[5] All these forms are related to the determiner אל (ʾl).

[6] One could use pre-trained word or character embeddings, but given the relatively small size of our corpus, we do not expect this to yield substantial improvements.

6.4 Flat vs. Hierarchical vs. Multitask Learning

Our basic experimental setup is to train tagging models for each field alone, resulting in five separate models (one for POS tagging and four for the morphological fields). We consider this 'flat' tagging a sort of baseline, as we hypothesize that including information from one field can improve results when predicting another.

The next setting we explore is a hierarchical model, utilizing a simple two-tier hierarchy with POS tags at the base and morphological tags building on that. This is anchored in the tag distribution. As mentioned in Section 4.1.2, most POS tags have their own set of legal morphological analyses in each field that are not shared with other POS tags. Thus, given the POS tag for a given word, the size of the possible pool of tags in each morphological field significantly decreases. Let $\tilde{x}_k$ be the word representation of word w_k including character information, as discussed above. In this setup, we also train five separate models, but while the POS model is identical to the base model architecture, the four morphological models concatenate POS tag information to the word representations, in the form of a one-hot vector $e_{t_k} \in \{0,1\}^d$ (where t_k is the index of the POS of w_k, for some ordering of all the POS tags, and d is the size of the POS tag set). The concatenated vector $(\tilde{x}_k, e_{t_k})$ is then fed to the word-level LSTM. During training, we provide the ground truth POS tag. At inference, we use POS tags predicted by the POS tagging model.[7]

Finally, a natural approach to take when tackling several tasks that are related to one another is multitask learning (MTL; Caruana, 1997), which has previous been considered for MSA morphological tagging (Inoue et al., 2017). In this setup, we share all parameters (word and character embeddings, and hidden states) between the different tasks, except for the final linear layer that receives the hidden states as input, and returns the scores for the relevant tag space. We have one layer of this kind for each task, each with its own parameters. We average the losses of each task, and backpropogate based on the averaged loss.

7 Experiments

7.1 POS Experiments

We begin our experimentation with addressing the POS tagging task alone, in order to determine the best architecture for our base model on a simpler task before diving into the more complicated morphology task. Our initial experiments are run with a base configuration of hyperparameters loosely based on prior work (Kim et al., 2016) and general intuition. Then we conduct a hyperparameter search for the best configuration. The exact settings are provided in Appendix B.

We run all our experiments by training on 90% of the tagged data, of which we hold out 10% for early stopping of the NN model training, and testing on the remaining 10%. All results of the neural-network-based models are averaged over five runs using five different seeds, unless noted otherwise. We compare the various model results to a 'most-frequent baseline' assignment, in which we assign a word the POS with which it appears most often in the training data, and assign all OOV words the most common POS tag (noun).

Table 3 summarizes the results of the various POS tagging models. The most frequent tag baseline is quite strong, as common in POS tagging tasks. In fact, it outperforms the WORD-LSTM model. Using character information is beneficial, and the CHAR-CNN model is better able to do so than the CHAR-LSTM model. Among the neural network models, it performs best. The best performing tagger overall is the CRF-based MarMoT tool.

Model	Accuracy [%]
most frequent baseline	82.01
WORD-LSTM	78.08±1.10
CHAR-LSTM	84.42±0.80
CHAR-CNN	87.45±0.58
MarMoT	89.17

Table 3: Accuracy of the POS tagging models. Best scoring model appears in bold.

7.2 Interim Summary

We saw in our experiments above that, among our neural-network (NN) approaches, representing a word by a CNN on its characters performs better than an LSTM, or ignoring the characters altogether. We use this CHAR-CNN model for hyperparameter tuning (see Appendix B). However, we also saw that MarMoT is indeed a very strong tool, and outperforms the CHAR-CNN in this task. Therefore, we move forward to the morphological tagging using both models, the CHAR-CNN representing the NN family, and MarMoT as a strong off-the-shelf tool.

7.3 Morphology Experiments

As we just discussed, of the three neural network architectures, the CHAR-CNN model performs best, and therefore we choose this architecture as our base model as we move forward with the morphology experiments,

[7] We use this setup for simplicity and do not consider curriculum learning strategies that sample targets both from the ground truth and from the model's predictions (Zhang et al., 2019).

Model	morphology			
	analysis1	analysis2	additional tags	enclitic
most frequent baseline	72.89	76.71	87.16	94.47
CHAR-CNN				
flat	80.18±0.47	84.02±0.53	90.59±0.08	95.72±0.10
hierarchical (pred POS)	79.56±0.32	83.69±0.76	90.05±0.53	95.87±0.14
hierarchical (true POS)	88.35±0.39	92.34±0.19	94.81±0.11	96.30±0.21
MTL	78.15±0.78	83.57±0.52	89.75±0.26	94.96±0.28
MarMoT	82.32	85.55	91.69	96.38

Table 4: Morphological models results by field. Best scoring results are in bold.

using the same base configuration of hyperparameters that we used in the POS experiments. We experiment with three approaches for predicting morphological tags (Section 6.4): the flat approach trains one model per each morphological attribute, the hierarchical approach uses POS information when predicting morphology, and the multitask approach predicts all morphological attributes jointly in a multitask manner. The hierarchical model was tested in two different setups, using either the predicted POS tag or the true POS tag in order to predict the morphological tags.

Providing true POS tags is a realistic choice for a linguistic annotation pipeline, since POS annotation is much simpler than morphological annotation. One may envision a human-in-the-loop process, where humans correct initial automatically assigned POS tags, and then a morphological tagger relies on the human-corrected tags. We return to this point in the discussion (Section 9).

7.3.1 Field by Field Accuracy

Table 4 shows the morphological tagging results broken down by field. The comparison highlights that our 'hierarchical CHAR-CNN model', when based on true POS, outperformed MarMoT in the first three morphology analysis fields. The model's success ranged from almost 89% for the *analysis1* field, almost 93% for *analysis2* and almost 95% for *additional tags*. This was judged as very significant by our JA experts. Due to its morphological complexity, manually tagging these morphology fields is highly time consuming even for experienced linguists. *Enclitic pronouns*, which are morphologically more restricted, are successfully predicted by most models with an accuracy greater than 95%.

7.3.2 Overall Accuracy

We also present several alternative overall scores for each of the taggers (Table 5). The 'strict' score considers a word to be correctly tagged only if all five fields are correctly tagged. This score was judged by the JA linguists as too severe, as they see real-world usefulness even if not all of the analysis fields were correctly tagged. The 'flexible' score counts each correct tag separately and gives equal weight to each field. Finally, reflecting the importance that our JA experts assigned to each field, a 'weighted' score was calculated as well, where the vector (3, 2, 2, 1, 1), for example, emphasizes POS over *analysis1* and *analysis2*, and gives the lowest weight to the *additional tags* and the enclitic pronouns. The comparison shows that our hierarchical CHAR-CNN (true POS) model performs better than MarMoT by 2.2% and 2.8% when calculating the 'flexible' score and the 'weighted' score, respectively, while MarMoT excels by the 'strict' metric.

Model	strict	flexible	weighted
most freq	66.94	82.64	80.76
CHAR-CNN			
flat	66.71±0.34	87.58±0.07	86.32±0.08
hierarchical (pred POS)	70.91±0.67	87.35±0.35	86.12±0.40
hierarchical (true POS)	71.18±0.52	91.95±0.13	90.71±0.15
MTL	66.24±0.97	86.84±0.30	85.72±0.30
MarMoT	75.84	89.02	87.92

Table 5: Overall accuracy scores for the morphological models. The strict, flexible, and weighted (3,2,2,1,1) scores are defined in the text.

7.3.3 Accuracy for words with legal tag combinations

Another way to evaluate our results is to look for all the words for which we know the tagger went wrong somehow. Recall that each POS has a certain set of legal values in each morphological analysis field, which differs from POS to POS (some of which can be seen in Tables 11–14 (Appendix). As our taggers are given the entire tagset, regardless of each specific word's POS, they may

Model	legal tag combo accuracy [%]	illegal tag combo average no. words	illegal tag combo percent [%]
CHAR-CNN			
flat	92.40±0.15	1652.0±44.9	26.95±0.73
hierarchical (pred POS)	88.56±0.20	1082.2±81.5	17.65±1.32
hierarchical (true POS)	95.18±0.25	1070.4±57.6	17.46±0.94
MTL	89.92±0.27	1379.0±44.6	22.49±0.73
MarMoT	89.49	1003.0	16.36

Table 6: 'Flexible' model accuracy for words with legal tag combinations, after removing words flagged as illegal combinations of POS tag and morphological analyses, and the average number of illegally tagged words and their percentage of the test set.

		morphology			
Model	POS	analysis1	analysis2	additional tags	enclitic
CHAR-CNN					
flat	72.63±0.52	59.64±0.99	61.06±1.28	73.80±0.61	88.40±0.43
hier. (pred)	72.58±0.46	57.72±0.81	59.64±1.39	72.26±1.85	89.01±0.68
hier. (true)	73.96±0.61	74.91±0.54	78.42±1.09	85.39±0.33	90.87±0.49
MTL	72.86±1.03	55.33±1.60	58.59±1.53	70.98±1.25	87.20±0.70
MarMoT	71.35	55.82	59.95	75.02	89.23

Table 7: Accuracy of morphology tagging for Out of Vocabulary (OOV) words.

produce illegal tag combinations if one of the predicted morphological tags does not appear in the legal values of the word's predicted POS tag (or, in the case of the true-POS-based hierarchical model, the true POS). In Table 6, we show the 'flexible' accuracy for each model on all the words that have legal tag combinations. Note that the accuracy of the true-POS hierarchical model for such words is almost 4% higher than its performance on the entire test set.

The table also shows the number of words that were tagged with illegal tag combinations and their percentage in the test set. Several observations can be made on the basis of this analysis. First, the models with the highest percentage of illegally tagged words are the flat CHAR-CNN and the multitask model. While the reported percentage of illegally tagged words for the true-POS-based hierarchical model (17%) is slightly higher than that of MarMoT, it is within a standard deviation of the percentage of words flagged in the MarMoT run. Coupled with the significant improvement in the 'flexible' score over MarMoT, which hardly improves over its general accuracy, this is a strong indication of the benefits of the true-POS-based hierarchical model.

We concede that 17% of all words is too many to expect a JA expert to address when using an automatic system for tagging new and unannotated data; however, these findings could potentially be used in other ways as well, such as adding a step in the automatic tagging process that forces the tagger to select a legal combination of POS tag and morphological analyses, using some heuristic to determine which of the predicted annotations to follow. This being said, as we mentioned in Section 4.3, the annotations in TAJA are noisy, and as such, 12% of the words in the annotated corpus appear with invalid analyses (mostly missing analyses, some illegal combinations) to begin with.

7.3.4 Out of Vocabulary Accuracy

Another way to evaluate how useful each model is in a real-world setting is through the accuracy of morphology tagging of Out of Vocabulary (OOV) words (Table 7) – words in the test set that did not occur in the training set. OOV words accounted for 21% of the TAJA test set (1281 of 6131 words). This high percentage of OOV words is reflective of the corpus' characteristics as discussed in Section 4.2. The results of this analysis are remarkable, with the hierarchical CHAR-CNN (true POS) significantly outperforming all other models across the different morphological analysis fields by 19% for *analysis1* and *analysis2* and by 10% for *additional tags*. This is a significant and encouraging finding, because it is very likely that the percentage of OOV words will increase in the future when we apply these tools to new texts beyond the TAJA corpus, especially if these texts are of different literary genres. Despite performing well in some of the previous evaluations, MarMoT failed on the morphology analysis of OOV words. A related ob-

servation is that even the hierarchical CHAR-CNN (predicted POS) model was able to assign POS tags to OOV words slightly better than MarMoT, achieving an accuracy of 72.58% vs. MarMoT's 71.35%.

8 Real-World Evaluation

The end goal of this project is to provide AJA language experts with an automatic tagger to help them annotate large volumes of text, a task which is otherwise laborious and time-consuming when tackled manually. To evaluate such real-world usefulness of the taggers we set out to compare the performance of our two best POS models (the hierarchical CHAR-CNN based model and MarMoT) with that of manual annotation by two expert AJA linguists.

8.1 The Task

For this evaluation task we selected a subset of the above-mentioned NAJA corpus, encompassing 30 chapters from the AJA translation of Psalms that were never annotated (a total of 3817 words). The two models were first trained on the entire TAJA corpus, and then we used the models to tag these selected unannotated texts. The resulting POS predictions were then given to two AJA experts of different calibers (see below), to evaluate and score. The two experts were instructed to write corrections only if one or both of the models were wrong, and to leave the annotation blank if both were correct. This enabled us not only to evaluate the performance of our competing models, but also assess inter-annotator agreement (IAA).

8.2 Inter-Annotator Agreement

It should be noted that the two human annotators that performed this task were of different expertise levels. One annotator is a senior professor of Judeo-Arabic, with decades of experience annotating and analyzing Judeo-Arabic texts. The second annotator is a doctoral student, a research assistant who has worked for several years under that professor's tutelage. Therefore, we consider the annotations of the senior expert to be the gold-standard, whereas the annotation of the junior expert is considered to be a silver-standard.

We calculate Cohen's Kappa between the annotations of the senior expert and those provided by the junior expert, excluding all the words on which the models disagreed but one of the human annotators did not identify the correct tag. We are left with 3685 words, for which $\kappa = 0.875$.

Note, however, that while Cohen's Kappa is a symmetric score, our two human annotators are of different calibers. Hence we take the senior expert's annotation as the correct result (gold-standard), and measure the accuracy of the junior expert's annotation relative to that of the senior expert. This will later be compared to the accuracy of the automatic taggers. Calculated on the same 3685 words stated above, the junior expert's accuracy in the wild was 0.908.

8.3 POS Tagger Evaluation

The accuracy statistics for the two POS taggers was evaluated relative to the corrections of the senior expert, whose annotations are considered to be the correct ones. Despite the instruction to correct all cases where at least one of the taggers was mistaken, there were 32 cases (0.8%) where the two models disagreed, but no correction was provided. On the remaining 3785 words, the accuracy of the two models was almost the same and only slightly lower than the accuracy of the human junior expert (Table 8). The real-world usefulness of the automatic taggers is highlighted when taking into account that it took the junior expert approximately 5.5 hours to complete this relatively limited task.

MarMoT	CHAR-CNN	junior expert
88.85	88.92	90.80

Table 8: POS tagging accuracy, on Psalms 1-30, relative to the correct tagging by the senior expert.

These results can be interpreted in several ways. A favorable way to look at this is that the automatic models are almost as good as a medium-level human annotator, and are therefore invaluable to the effort of annotating large amounts of text. A less favorable view is that a less experienced human annotator is more susceptible to agree with subtle mistakes made by an automatic tagger, though they might provide the correct annotation when facing a blank page. The easiest way to confirm or reject the hypothesis that the RA is more susceptible to being led astray by the automatic annotations is to compare his accuracy on this Psalms file to a similar number of annotations he made on a completely unannotated file. Unfortunately, that breakdown is not available. However, in support of this hypothesis, we break down the mistakes made by the junior expert by whether or not the models agreed on the annotation. We see that over 75% of the junior expert's mistakes were in cases where the models agreed, and of those cases, over 70% are words where the junior expert agreed with the automatic taggers, whereas the senior expert chose a different tag. In light of these numbers, it is important to emphasize to human annotators who use the automatically generated tags that they must look at the tags with a critical eye, and not assume that the taggers "know" the truth.

As is apparent from the results, there is almost no difference in accuracy between the two models, despite the fact that the models disagree on 11.2% of annotations. The number of mistakes made by each of the models is almost equal, with MarMoT being correct on 179 words and CHAR-CNN on 182 words out of 429 words on which the models disagree, and an additional 35 words on which both models were wrong. An interesting direction for future research is to characterize the kinds of mistakes each model tends to make, and explore ways to combine their strengths. Furthermore, we note that in this real-world application to NAJA (i.e., texts that are not part of TAJA) the CHAR-CNN model performed a little better than its initial TAJA-based evaluation (88.92% vs. 87.45%, see Table 3) while the MarMoT model performed a little worse (88.85% vs. 89.17%).

9 Discussion and Conclusion

The pressing *real-world* challenge facing researchers of Algerian Judeo Arabic (AJA) dialects is how to scale up their linguistic analyses from individual texts to large textual collections. The rich morphology of Arabic (as of other Semitic languages) and scarcity of expert linguists makes this complex and time-consuming task impractical unless aided by automation. Hence, developing automatic taggers that would support *real-world* linguistic analysis at scale and prove *useful* for AJA linguists is the challenge we aim to tackle. Reflecting the linguists' challenges, we focus on the performance of the morphological tagger in tests that are predictive of the real-world setting. For this reason, we did not limit ourselves to purely automated approaches, but also explored a hybrid human–machine approach, wherein the human expert contributes to the automatic approach.

The rich morphology of Arabic and its use of morpho-syntactic affixes led us to focus on character-based models (rather than word-based models), as these can identify key morphemes that are essential for annotating OOV words. Starting from a word-based LSTM neural network architecture, we integrated character-level information via either an LSTM or a CNN. Subsequently we explored a two-tier hierarchical approach to morphological tagging with POS tags at its base and the morphology tags building on that. This hierarchy mirrors the underlying character of Arabic annotation, where each POS tag has a set of legal morphological tags. The two-tier approach also enables exploring a human-in-the-loop step in between the two tiers. Our best performing strategy, denoted AJATAG for simplicity, is now available for use by AJA linguists.[8] To evaluate the *usefulness* of the AJATAG strategy we compared it to the *off-the-shelf* POS and morphological tagger, MarMoT, which is based on CRF. All models were trained on the annotated TAJA corpus.

For the base task of POS tagging, we found that among the evaluated neural network architectures, representing a word using a CNN run on its characters performed better than an LSTM or ignoring the characters altogether. Training on the TAJA corpus, the POS accuracy of the CHAR-CNN model was $87.4 \pm 0.58\%$. This accuracy is only slightly lower than the 89.17% accuracy obtained by MarMoT for this task. The 1.5% difference suggest essentially similar performance for the two models in a real-world setting. Morphology tagging, as indicated above, is the most challenging and time-consuming task that takes up 80% of the expert linguist annotation time. Here, too, CHAR-CNN performed better than the other neural network models we explored, especially in a two-tier hierarchical approach. The accuracy of this model, denoted herein as 'hierarchical CHAR-CNN (predicted POS)', ranges from 81% to 91% for the different morphology analysis fields (*analysis1*, *analysis2*, *additional tags*). To further improve the performance, we allowed for human input between the two tiers in the form of manual correction of POS tags. Using 'true POS' assignments, instead of the predicted assignments, further improved the performance of the 'hierarchical CHAR-CNN (true POS)' morphology tagger. We denote this hybrid strategy AJATAG and have compared its performance on AJA to MarMoT. We use MarMoT as is, without modifications or adaptations to a hybrid setting, because for the linguists it is an *off-the-shelf* tool that is to be used as is.

Evaluation of the morphological tagging by AJATAG demonstrated favorable performance across multiple evaluation metrics:

- **Field-by-field accuracy** – AJATAG accuracy for the two main *analysis* fields (89.0%, 92.7%, respectively) is higher by up to 7% compared to MarMoT's accuracy (82.3%, 85.6% respectively). It should be noted that the greatest gain in accuracy is in *analysis1*, which of the morphological analysis fields is the richest and most difficult to assign. Both approaches perform well identifying the *enclitic* field with an accuracy greater than 96%.
- **Overall accuracy** – We evaluate the overall accuracy of the morphology taggers using a 'flexible' score, which best mimics real-life usefulness of the tagger as it counts each correct tag separately. The overall accuracy of AJATAG was 91.2%, a little over 2% better than MarMoT (89.0%).
- **Accuracy for words with legal tag combinations** – In TAJA each POS tag has a set of legal values for morphological tags. However, both

[8] https://github.com/technion-cs-nlp/nlp4aja

taggers end up assigning a significant percentage of the words with illegal tag combinations. It is noteworthy, however, that for words that were tagged with legal tag combinations (which are the majority at over 80%) the accuracy of AJATAG went up by 4% to 95.2%, while the accuracy of MarMoT was essentially unchanged.

- **Out of Vocabulary accuracy** – Perhaps the most important predictor for future real-world performance of any tagger is its success with words that are out of vocabulary (OOV), especially as OOV words account for 21% of the TAJA test set. When using predicted POS tags with our hierarchical CHAR-CNN model, the accuracy on the challenging *analysis1* field for OOV words was 57.72%, better than MarMoT by approximately 2% (it also performed better on POS tagging of OOV words with 72.58% vs. MarMoT's 71.35%). However, this important performance indicator is where our hybrid AJATAG strategy delivered its most important fruits. The accuracy of AJATAG in the challenging task of morphologically tagging OOV words is 74.91% and 78.42% for the *analysis1* and *analysis2* fields, respectively, which is significantly better than MarMoT's OOV tagging for these two fields (55.82% and 59.95%, respectively). AJATAG also performs much better in the *additional tags* field for OOV words (85.4% compared to MarMoT's 75.0%).

The justification for the hybrid approach explored herein is in its real-world usefulness, outside of the NLP lab. The 56%–60% accuracy of the off-the-shelf solution for the two most important morphological fields, *analysis1* and *analysis2*, when applied to OOV words is not sufficient for real linguistic work. In contrast, the hybrid AJATag strategy achieved an accuracy level of 74.91%–78.42% on morphological tagging of OOV words, which is expected to be useful for real-world applications, improving upon MarMoT by 18%–19% for this task on both analysis fields. It is reassuring that even without the added human input, our fully automated hierarchical CHAR-CNN performed better than MarMoT on POS and *analysis1* tagging of OOV words. The value of the AJATAG strategy was further confirmed by other performance indicators, including its overall accuracy and its accuracy on words with legal tag combinations, as defined above.

To assess the feasibility of the human interface element in AJATAG, we performed a real-world evaluation of this process. The first-tier POS output was given to two AJA linguists to correct, before moving on to the second-tier morphology tagging. POS tags manually corrected by a senior expert were perceived as the 'true' POS assignment, to which the performance of the automatic taggers as well as the corrections by a junior expert were compared. It is reassuring that both automated taggers, our CHAR-CNN model and MarMoT, performed well at an almost identical accuracy (~89%) relative to the 'true' POS, an accuracy quite similar to the 91% accuracy by the junior expert, who is a PhD candidate with several years of experience in AJA linguistics.

To conclude, while not perfect, the hybrid AJATAG approach provides AJA linguists with a working solution that already impacts their real-world workflow in a way that off-the-shelf tools cannot provide. In the future we plan to continue improving these tools by addressing limitations such as tagging words with illegal tag combinations. Nonetheless, we believe that even in its current form AJATAG could prove useful to linguists as they take on the task of analyzing large untagged AJA corpora. We hope that in the future we will be able to expand the utility of these tools to other Judeo-Arabic dialects.

Acknowledgements

This research was supported by the Israel Science Foundation (grant No. 1191/18). YB was supported by an Azrieli Foundation Early Career Faculty Fellowship.

References

Abdelali, Ahmed, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for Arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–16, San Diego, California. Association for Computational Linguistics.

Abdelali, Ahmed, Hamdy Mubarak, Younes Samih, Sabit Hassan, and Kareem Darwish. 2021. QADI: Arabic dialect identification in the wild. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 1–10, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Ahmed, Mohamed AH. 2018. Xml annotation of hebrew elements in judeo-arabic texts. *Journal of Jewish Languages*, 6(2):221–242.

Altaher, Yousef, Ali Fadel, Mazen Alotaibi, Mazen Alyazidi, Mishari Al-Mutairi, Mutlaq Aldhbuiub, Abdulrahman Mosaibah, Abdelrahman Rezk, Abdulrazzaq Alhendi, Mazen Abo Shal, et al. 2022. Masader plus: A new interface for exploring+ 500 arabic nlp datasets. *arXiv preprint arXiv:2208.00932*.

Alyafeai, Zaid, Maraim Masoud, Mustafa Ghaleb, and Maged S. Al-shaibani. 2022. Masader: Metadata sourcing for Arabic text and speech data resources. In

Proceedings of the Thirteenth Language Resources and Evaluation Conference, pages 6340–6351, Marseille, France. European Language Resources Association.

Attia, Mohammed, Younes Samih, Ali Elkahky, Hamdy Mubarak, Ahmed Abdelali, and Kareem Darwish. 2019. POS tagging for improving code-switching identification in Arabic. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 18–29, Florence, Italy. Association for Computational Linguistics.

Ballesteros, Miguel, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 349–359, Lisbon, Portugal. Association for Computational Linguistics.

Bar-Asher, Moshe. 1992. *La composante hebraïque du judeo-arabe algerien: communautés de Tlemcen et Aïn-Témouchent.* Magnes, Jerusalem.

Belinkov, Yonatan. 2021. Large-scale electronic corpora and the study of middle and mixed Arabic. In *Middle and Mixed Arabic over Time and across Written and Oral Genres: From Legal Documents to Television and Internet through Literature. Proceedings of the IVth AIMA International Conference (Emory University, Atlanta, GA, USA, 12–15 October 2013)*, Publications de l'Institut Orientaliste de Louvain, pages 43–67, Université catholique de Louvain, Louvain-la-Neuve. Peeters.

Bouamor, Houda, Nizar Habash, Mohammad Salameh, Wajdi Zaghouani, Owen Rambow, Dana Abdulrahim, Ossama Obeid, Salam Khalifa, Fadhl Eryani, Alexander Erdmann, et al. 2018. The MADAR Arabic dialect corpus and lexicon. In *LREC*.

Caruana, Rich. 1997. Multitask learning. *Machine Learning*, 28.

Çetinoğlu, Özlem, Sarah Schulz, and Ngoc Thang Vu. 2016. Challenges of computational processing of code-switching. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 1–11, Austin, Texas. Association for Computational Linguistics.

Cohen, Marcel. 1912. *Le parler arabe des Juifs d'Alger*. Collection linguistique, pub. par la Société de linguistique de Paris-4. H. Champion, Paris.

Darwish, Kareem, Mohammed Attia, Hamdy Mubarak, Younes Samih, Ahmed Abdelali, L. Màrquez, M. Eldesouki, and Laura Kallmeyer. 2020. Effective multi-dialectal Arabic POS tagging. *Natural Language Engineering*, 26:677 – 690.

Darwish, Kareem, Hamdy Mubarak, Ahmed Abdelali, Mohamed Eldesouki, Younes Samih, Randah Alharbi, Mohammed Attia, Walid Magdy, and Laura Kallmeyer. 2018. Multi-dialect Arabic POS tagging: A CRF approach. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Dermatas, Evangelos and George Kokkinakis. 1995. Automatic stochastic tagging of natural language texts. *Computational Linguistics*, 21(2):137–163.

Diab, Mona. 2009. Second generation AMIRA tools for Arabic processing: Fast and robust tokenization, POS tagging, and base phrase chunking. In *In 2nd International Conference on Arabic Language Resources and Tools*.

Dos Santos, Cicero and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *International Conference on Machine Learning*, pages 1818–1826. PMLR.

Duh, Kevin and Katrin Kirchhoff. 2005. POS tagging of dialectal Arabic: A minimally supervised approach. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 55–62, Ann Arbor, Michigan. Association for Computational Linguistics.

El-Haj, Mahmoud. 2020. Habibi - a multi dialect multi national Arabic song lyrics corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1318–1326, Marseille, France. European Language Resources Association.

El-Haj, Mahmoud and Rim Koulali. 2013. KALIMAT a multipurpose Arabic corpus. In *Second workshop on Arabic corpus linguistics (WACL-2)*, pages 22–25.

Ferguson, Charles A. 1959. Diglossia. *WORD*, 15(2):325–340.

Habash, Nizar, Mona Diab, and Owen Rambow. 2012. Conventional orthography for dialectal Arabic. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 711–718, Istanbul, Turkey. European Language Resources Association (ELRA).

Habash, Nizar, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. 2013. Morphological analysis and disambiguation for dialectal Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 426–432, Atlanta, Georgia. Association for Computational Linguistics.

Habash, Nizar, Abdelhadi Soudi, and Timothy Buckwalter. 2007. *On Arabic Transliteration*, volume 38, chapter 2. Springer Netherlands, Dordrecht.

Hajič, Jan, Otakar Smrž, Petr Zemánek, Petr Pajas, Jan Šnaidauf, Emanuel Beška, Jakub Kracmar, and Kamila Hassanová. 2009. Prague Arabic dependency treebank 1.0. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Hary, Benjamin. 2003. Judeo-Arabic: A diachronic reexamination. *International Journal of The Sociology of Language*, 2003:61–75.

Hochreiter, Sepp and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Inoue, Go, Salam Khalifa, and Nizar Habash. 2022. Morphosyntactic tagging with pre-trained language models for Arabic and its dialects. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1708–1719, Dublin, Ireland. Association for Computational Linguistics.

Inoue, Go, Hiroyuki Shindo, and Yuji Matsumoto. 2017. Joint prediction of morphosyntactic categories for fine-grained Arabic part-of-speech tagging exploiting tag dictionary information. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 421–431, Vancouver, Canada. Association for Computational Linguistics.

Kahn, Lily and Aaron D Rubin. 2017. *Handbook of Jewish Languages: Revised and Updated Edition*. Brill.

Kim, Yoon, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2741–2749. AAAI Press.

Lafferty, John, Andrew Mccallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Ling, Wang, Chris Dyer, Alan W Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luís Marujo, and Tiago Luís. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

Luong, Minh-Thang and Christopher D. Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063, Berlin, Germany. Association for Computational Linguistics.

Maamouri, Mohamed, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 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. Cairo.

Maamouri, Mohamed, Ann Bies, Seth Kulick, Michael Ciul, Nizar Habash, and Ramy Eskander. 2014. Developing an Egyptian Arabic treebank: Impact of dialectal morphology on annotation and tool development. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2348–2354, Reykjavik, Iceland. European Language Resources Association (ELRA).

McCarthy, John J. 1981. A prosodic theory of non-concatenative morphology. *Linguistic Inquiry*, 12:373–418.

Müller, Thomas, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Nivre, Joakim, Željko Agić, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Cristina Bosco, Sam Bowman, Giuseppe G. A. Celano, Miriam Connor, Marie-Catherine de Marneffe, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Tomaž Erjavec, Richárd Farkas, Jennifer Foster, Daniel Galbraith, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Berta Gonzales, Bruno Guillaume, Jan Hajič, Dag Haug, Radu Ion, Elena Irimia, Anders Johannsen, Hiroshi Kanayama, Jenna Kanerva, Simon Krek, Veronika Laippala, Alessandro Lenci, Nikola Ljubešić, Teresa Lynn, Christopher Manning, Cătălina Mărănduc, David Mareček, Héctor Martínez Alonso, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Shunsuke Mori, Hanna Nurmi, Petya Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Slav Petrov, Jussi Pitulainen, Barbara Plank, Martin Popel, Prokopis Prokopidis, Sampo Pyysalo, Loganathan Ramasamy, Rudolf Rosa, Shadi Saleh, Sebastian Schuster, Wolfgang Seeker, Mojgan Seraji, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Kiril Simov,

Aaron Smith, Jan Štěpánek, Alane Suhr, Zsolt Szántó, Takaaki Tanaka, Reut Tsarfaty, Sumire Uematsu, Larraitz Uria, Viktor Varga, Veronika Vincze, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. 2015. Universal dependencies 1.2. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Owens, J. 2013. *The Oxford Handbook of Arabic Linguistics*. Oxford Handbooks. Oxford University Press.

Pasha, Arfath, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 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)*, pages 1094–1101, Reykjavik, Iceland. European Language Resources Association (ELRA).

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Plank, Barbara, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics.

Riabi, Arij, Benoît Sagot, and Djamé Seddah. 2021. Can character-based language models improve downstream task performances in low-resource and noisy language scenarios? In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 423–436, Online. Association for Computational Linguistics.

Roth, Ryan, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 117–120, Columbus, Ohio. Association for Computational Linguistics.

Seddah, Djamé, Farah Essaidi, Amal Fethi, Matthieu Futeral, Benjamin Muller, Pedro Javier Ortiz Suárez, Benoît Sagot, and Abhishek Srivastava. 2020. Building a user-generated content North-African Arabizi treebank: Tackling hell. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1139–1150, Online. Association for Computational Linguistics.

Tedghi, Joseph. 2012. 'le livre de Jonas' traduit en judéo-arabe marocain par Samuel Malka: étude linguistique. In *Dynamiques langagières en Arabophonies*, pages 253–290, Zaragoza. Universidad de Zaragoza, Área de Estudios Árabes e Islámicos.

Terner, Ori, Kfir Bar, and Nachum Dershowitz. 2020. Transliteration of Judeo-Arabic texts into Arabic script using recurrent neural networks. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 85–96, Barcelona, Spain (Online). Association for Computational Linguistics.

Tirosh-Becker, Ofra. 1988. The phonology and topics in the morphology of a Judeo-Arabic translation of the book of Psalms from Constantine (Algeria) / של תרגום לספר תהילים בערבית-יהודית מקונסטנטין (אלג'יריה) פונולוגיה ופרקים במורפולוגיה. Master's thesis, The Hebrew University of Jerusalem.

Tirosh-Becker, Ofra. 1989. On the linguistic uniformity in the "Šarḥ" of the Jews of Constantine / קונסטנטין לשאלת אחדות הלשון בשרח של יהודי. *Proceedings of the World Congress of Jewish Studies /העולמי למדעי היהדות דברי הקונגרס*, .204–197:י

Tirosh-Becker, Ofra. 2011a. Old and new in the translation and commentary of Avot tractate / אבות ופירושה ישן וחדש בתרגום משנת. In *Hikrei ma'arav u-mizrah: studies in language, literature and history presented to Joseph Chetrit (Volume 1) / מוגשים ליוסף שיטרית כרך 1) חקרי מערב ומזרח: לשונות, ספרויות ופרקי תולדה*, Hikrei ma'arav u-mizrah : studies in language, literature and history presented to Joseph Chetrit / ליוסף שיטרית חקרי מערב ומזרח: לשונות, ספרויות ופרקי תולדה מוגשים. Carmel.

Tirosh-Becker, Ofra. 2011b. On dialectal roots in Judeo-Arabic texts from Constantine (east Algeria). *Revue des Études Juives*, 170:227–253.

Tirosh-Becker, Ofra. 2011c. Terms for realia in an Algerian Judeo-Arabic translation of the Hošaʿnot. *Studies in the Culture of North African Jewry*, 1:171–186.

Tirosh-Becker, Ofra. 2012. Mixed linguistic features in a Judeo-Arabic text from Algeria: The Šarḥ to the Hafṭarot from Constantine. In *Language and Nature: Papers presented to John Huehnergard on the Occasion*

of his 60th Birthday, pages 391–406, Chicago. Oxford University Press.

Tirosh-Becker, Ofra. 2014. A reflection of a linguistic reality: An Algerian Judeo-Arabic book for the new year. *Studies in the Culture of North African Jewry*, 3:193–216.

Tirosh-Becker, Ofra and Oren M. Becker. 2022. TAJA corpus: Linguistically tagged written Algerian Judeo-Arabic corpus. *Journal of Jewish Languages*, 10(1):24 – 53.

Wagner, Esther-Miriam and Magdalen Connolly. 2018. Code-switching in judaeo-arabic documents from the cairo geniza. *Multilingua*, 37(1):1–23.

Zaidan, Omar and Chris Callison-Burch. 2014. Arabic dialect identification. *Computational Linguistics*, 40:171–202.

Zalmout, Nasser and Nizar Habash. 2019. Adversarial multitask learning for joint multi-feature and multi-dialect morphological modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1775–1786, Florence, Italy. Association for Computational Linguistics.

Zalmout, Nasser and Nizar Habash. 2020. Joint diacritization, lemmatization, normalization, and fine-grained morphological tagging. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8297–8307, Online. Association for Computational Linguistics.

Zhang, Wen, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343, Florence, Italy. Association for Computational Linguistics.

Zitouni, Imed. 2014. *Natural Language Processing of Semitic Languages*. Springer.

Zribi, Inès, Mariem Ellouze, Lamia Belguith, and Philippe Blache. 2015. Spoken Tunisian Arabic corpus "STAC": Transcription and annotation. *Research in Computing Science*, 90.

A Data

In this appendix, we detail the transliteration scheme for JA texts used in this paper (Table 9). This table only covers the consonants in JA, as the pronounciation of the vowels in text is not always known.

We also show some of the tag sets used in TAJA. We detail all the POS tags (Table 10), and the morphological tags of the more prominent POS tags (Tables 11, 12, 13 and 14).

Hebrew letter	Transliteration
א	ʾ
ב	b
ג׳	ǧ
ג	ġ
ד	d
ה	h
ו	w
ז	z
ח	ḥ
ט	ṭ
י	y
כ	k
כ׳	x
ל	l
מ	m
נ	n
ס	s
ע	ʿ
פ	f
צ	ṣ
צ׳	ḍ
ק	q
ר	r
ש	š
ת	t

Table 9: Transliteration table for Hebrew (JA) letters.

POS code	Hebrew POS	POS
שע	שם עצם	noun
פע	פועל	verb
מל	מילית	particle
שת	שם תואר	adjective
שפ	שם פרטי	proper noun
מס	מספר	number
כג	כינוי גוף	pronoun
כר	כינוי רמז	demonstrative
כז	כינוי זיקה	relative pronoun
תפ	תואר הפועל	adverb
הצ	הצגה	presentative
תז	תיאור זמן	temporal adjunct
תמ	תיאור מקום	locative adjunct
רת	ראשי תיבות	acronym

Table 10: Legal POS tags in TAJA.

Nouns					
analysis1 code		analysis2 code		additional tags code	
זכר	masculine	יחיד	singular	NA	
נקבה	feminine	רבים	plural		
עברי	Hebrew	זוגי	dual		
ארמי	Aramaic				
לועזי	foreign				

Table 11: Legal morphological analyses for nouns.

Verbs					
analysis1 code		analysis2 code		additional tags code	
(בניין)	(derived stem)	(זמן)	(tense)	(גוף)	(person)
בנ1	I	עב	perfect	י1	1s
בנ2	II	עת	imperfect	יז2	2sm
בנ3	III	צו	imperative	ינ2	2sf
בנ4	IV	בפע	passive participle	יז3	3sm
בנ5	V	בפו	active participle	ינ3	3sf
בנ6	VI	מצ	verbal noun	ר1	1p
בנ7	VII	לפעול	infinitive	רז2	2pm
בנ8	VIII			רנ2	2pf
בנ10	X			רז3	3pm
בנn	passive stem			רנ3	3pf
	related to VII			יחיד	participle sm
בנt	passive stem with a t/tt prefix			יחידה	participle sf
				רבים	participle pm
				רבות	participle pf

Table 12: Legal morphological analyses for verbs.

Adjectives				
analysis1 code		analysis2 code		additional tags code
יחיד	singular masculine	עברי	Hebrew	NA
יחידה	singular feminine	ארמי	Aramaic	
רבים	plural masculine	לועזי	foreign	
רבות	plural feminine			

Table 13: Legal morphological analyses for adjectives.

Proper Nouns				
analysis1 code		analysis2 code		additional tags code
אדם	person	משוערב	Arabized	NA
מקום	place	מתורגם	translated	
עם	people	עברי	Hebrew	
האל	God			

Table 14: Legal morphological analyses for proper nouns.

B Experiments

After determining that the CHAR-CNN model is the best of the three options, we conducted hyperparameter tuning by k-fold cross-validation ($k = 5$). The hyperparameters that we wanted to test are summarized in Table 15, where the reported statistics and standard deviation are over the folds. Rather than report the mean for each hyperparameter test, we report difference between the base configuration result and the hyperparameter result. The value of each hyperparameter in the base configuration appears in parentheses following the name of the hyperparameter. As we are attempting to optimize a large number of hyperparameters, grid search was deemed unfeasible (with a Cartesian product of over 23k hyperparameter combinations). Instead, we test each hyperparameter separately against the base configuration. However, we saw no significant differences between various configurations. This is evident from the table, as in most cases, the results for the tested hyperparameters are within one standard deviation of the base configuration result. Therefore, we continue conducting all our experiments using the original base configuration.

Hyperparameter	value	micro average accuracy (mean)	std	num epochs (mean)
base configuration	**NA**	**0.8908**	**0.0058**	**13**
batch size (8)	4	-0.0023	0.0069	12.6
	16	-0.0034	0.0048	12
directions (2)	1	-0.0004	0.0042	15.2
dropout (0.5)	0.0	-0.0041	0.0036	11.6
	0.3	-0.0048	0.0083	11.8
	0.7	-0.0031	0.0079	13.4
learning rate (0.1)	0.01	-0.0007	0.0050	12.8
	0.05	+0.0009	0.0047	13.4
	0.5	-0.0022	0.0099	12.8
kernel width (6)	4	-0.0046	0.0048	14.2
	8	-0.0053	0.0086	11.2
num kernels (500)	250	-0.0054	0.0049	13.2
	1000	-0.0019	0.0099	12.4
char embedding dim (25)	10	-0.0092	0.0081	15
	50	<-0.0001	0.0045	11.2
word embedding dim (100)	50	-0.0009	0.0047	13.8
	200	+0.0013	0.0027	13.2
hidden dim (100)	50	-0.0009	0.0060	13.6
	200	-0.0006	0.0039	12.6

Table 15: Summary of hyperparameter tuning (base configuration value in parentheses.)