Joint Alignment of Segmentation and Labelling for Arabic Morphosyntactic Taggers

Abdulrahman Alosaimy *scama@leeds.ac.uk*
School of Computing
University of Leeds
Leeds, LS2 9JT, UK
College Of Computer and Information Sciences
Imam Muhammad Ibn Saud Islamic University
Riyadh 13318, Saudi Arabia

Eric Atwell *e.s.atwell@leeds.ac.uk*
School of Computing
University of Leeds
Leeds, LS2 9JT, UK

Abstract

We present and compare three methods of alignment between morphemes resulting from four different Arabic POS-taggers as well as one baseline method using only provided labels. We combined four Arabic POS-taggers: MADAMIRA (MA), Stanford Tagger (ST), AMIRA (AM), Farasa (FA); and as the target output used two Classical Arabic gold standards: Quranic Arabic Corpus (QAC) and SALMA Standard Arabic Linguistics Morphological Analysis (SAL). We justify why we opt to use label for aligning instead of word form. The problem is not trivial as it is tackling six different tokenisation and labelling standards. The supervised learning using a unigram model scored the best segment alignment accuracy, correctly aligning 97% of morpheme segments. We then evaluated the alignment methods extrinsically, in terms of their effect in improving accuracy of ensemble POS-taggers, merging different combinations of the four Arabic POS-taggers. Using the best approach to align input POS taggers, ensemble tagger has correctly segmented and tagged 88.09% of morphemes. We show how increasing the number of input taggers raise the accuracy, suggesting that input taggers make different errors.

Keywords: Arabic, POS-Tagging, Segmentation, Tokenisation, Morphological Alignment.

1. INTRODUCTION

We want a Part of Speech (POS) tagger for Classical Arabic, the language of the Quran and other Arabic texts from 7th to 9th centuries CE. We have Gold Standard samples from the Quran manually POS-tagged using two alternative POS-tagging schemes, and we want to extend one or both of these POS-tagging schemes to other Classical Arabic texts including Hadith. We want to combine existing POS-taggers for Modern Standard Arabic (MSA), adapted to input Classical Arabic words and texts, and to output Classical Arabic POS-tags. Several POS-taggers have been developed for Modern Arabic, but they do not conform to shared standards in morphological segmentation or morphosyntactic tagsets for labelling. In machine learning, ensemble methods have proven to be more effective than an individual algorithm. To combine POS-taggers, methods for alignment of segmentation and labelling in parallel is a necessity.

In addition to combining taggers, when evaluating an automatic part-of-speech (POS) tagging, the segmentation scheme of words of the gold standard (the sequence of morphemes) should match the segmentation scheme of the tagger [1]. For example, if the gold-standard corpus strips the suffix in *he's*; a tagger should strip it too. In this paper, we introduce several approaches that align the two sequences of morphemes.

Sequence alignment is a well-known problem in several computational fields. It is the process of identifying tokens that correspond in some manner in the source and the target sequences. Bitext word alignment in Machine Translation is an example that identifies translation relationships between words to limit or constrain the set of translation rules learned from a bilingual parallel corpus.

The alignment problem can be formally defined as the following: having two sequences of tagged words $A = \{a_1, a_2, \dots\}$ and $B = \{b_1, b_2, \dots\}$ where a_i is vector that represent a word in a sentence and $\forall\, a \in A$; $M_a = \{m_1, m_2, \dots\}$ is the sequence of morphemes in that word, the problem is to find $m \rightarrow n, m \in M_a, n \in M_b, a \rightarrow b$. The result of the mapping is a set of pairs: $C = \{(m_i, n_j), \dots\}, m_i \in M_a, n_j \in M_b$. Indices in the pairs appear just once, limiting pairs to 1-1 mappings. In other words, the result of the alignment is a subset of the bipartite graph G=(V, E) where each edge $e = (m, n)$ and each vertex is a leaf vertex. We used Needleman–Wunsch algorithm [2] to compute the optimal global alignment between two sequences of tags using a variety of scoring matrices.

Two sequences of morphemes are regionally-aligned: words (delimited by a space) are aligned to their corresponding words on the other sequence, i.e. There is already an existing alignment mapping of: $a \rightarrow b, a \in A, b \in B$. Such existing alignment should raise the baseline accuracy as the number of possible mapping is constrained.

To illustrate the problem, the word (ولقد, *walaqado*, and indeed) has two possible tokenisations shown in figure 1. Two gold standard corpora (SAL, QAC) segmented the word into three segments, and four taggers (MA, ST, AM, FA) segmented it into two segments. This tokenisation problem can vary from tagging compound names (with one tag) to tagging a single word with multiple segments and hence tags (clitics, punctuation, dates, numbers, etc.) Therefore, it is necessary to align the results of those taggers for proper evaluation and voting. Inferring that `part_verb`, in Fig 1, should be aligned with `CERT` but not `EMPH` requires prior mapping knowledge.

```
MA  1-0  waconj
MA  1-1  laqad part_verb
ST  1-0  w     CC
ST  1-1  lqd   RP
AM  1-0  w     CC
AM  1-1  lqd   RP
FA  1-0  w+    CONJ
FA  1-1  lqd   PART
SAL 1-0  wa    p--c--
SAL 1-1  la    p--b--
SAL 1-2  qado  p--b--
QAC 1-0  wa    CONJ
QAC 1-1  la    EMPH
QAC 1-2  qado  CERT
```

Segment Form	MA	ST	AM	FA	SAL	QAC
w+	conj	CC	CC	CONJ	p--c--	CONJ
la+	---	---	---	---	p--z--	EMPH
qado	part_verb	RP	RP	PART	p--b--	CERT

FIGURE 1: A sample of aligned POS tags of one word that has two/three morphemes. The output of four taggers and two gold standard analyses is on the left, and how POS tags should be aligned is on the right.

In this paper, we compare three approaches that align segmentation using only provided labels. We are aligning four taggers' output to two gold standard corpora using:

1. **Rule-based** Manually mapping tagsets from each one to the others, then aligning matched tags,
2. **Unsupervised** Learning the alignment based on how probably two tags appear in the same word, or
3. **Supervised** Predicting the alignment using a parallel corpus of manually aligned tags

4. **Baseline** Aligning *base* or primary morpheme of the word, then aligning affixes starting from the closest ones to the primary morpheme one-by-one.

We compare the results of four POS-taggers: MADAMIRA (MA) [3], Stanford Tagger (ST) [4], AMIRA (AM) [5], Farasa (FA) [6] with two Classical Arabic gold standards: Quranic Arabic Corpus (QAC) [7] and SALMA Standard Arabic Linguistics Morphological Analysis (SAL) [8]. We show the effect of this morpheme-based alignment on building an ensemble POS tagger of the four taggers and evaluate these alignments in this manner as well.

2. RELATED WORK

In the GRACE campaign of evaluating French POS-taggers, taggers from competitors were using different POS tags and segmentation. The alignment was done based on the word form not the tags of the words [9]. Automatic Mapping Among Lexico-Grammatical Annotation Models (AMALGAM) [10; 11] unifies the tokenisation schemes by providing an alternative one to be used before the tagging step. We cannot preprocess the text before tagging in this way as there is no standard tokenisation for our input taggers.

In MorphoChallenge 2009 [12], participating unsupervised taggers was treated as black-box and no alignment was used. Instead, taggers were evaluated based on the "similarity" to a gold-standard corpus. Such evaluation is word-based and cannot be used for our ensemble taggers.

The most relevant work is [13] which combined three taggers (AMIRA toolkit, MADA+TOKAN toolkit, and authors’ maximum-likelihood tagger). AMIRA segmentation scheme differs that other input taggers. To recover from this mismatch, the authors used the gold standard tags as a pivot to align AMIRA output to the other; this enabled them to comparatively evaluate taggers against the gold standard text, but makes the ensemble tagger unusable for unseen texts.

3. CHARACTER-BASED ALIGNMENT CHALLENGES

Character-based alignment uses the word form of the segments provided by segmenter/tagger to align the output. Segments are aligned when they have most similar word form. This approach was used in the GRACE campaign to align several participating taggers using a word-based "diff" tool. We opt not to use character-based alignment for several reasons.

POS taggers for Arabic can be classified into two categories:

1. Word-based Taggers: where the word is given a compound tag with no explicit mark for segmentation in the form; i.e. the alignment between segments of the input word and the tags in the compound tag is not shown. Examples include: MADAMIRA toolkit and Microsoft POS tagger. Tagset for enclitics might even be different than ones for the stem.

2. Segment-based Taggers: Each segment is clearly defined with its morphological information. Some taggers mark enclitics by adding plus sign to indicate that it was split off from previous/next segment. Examples include: Stanford POS tagger, Farasa POS tagger, and AMIRA 2.0 tagger.

We are not able to use the character-based approach for aligning segments between taggers because some of the input taggers to the ensemble tagger were word-based, namely MADAMIRA. An example of word-based tagger is tagging the word: (كُنَّا, *kuna~A*, we were) which was given `V.Dual.Plu.Pst.Act*Subj.Plu.1` tag. Even though the tag indicates that there are two segments in the word: `V` and `Subj` (separated by the star sign), there is no mark on where the word form should be separated; i.e. the segment form is missing.

Segment-based taggers have their own issues. When a word is split off into segments, the segments might require some modification to recover their original form. There are at least four reasons for such differences:

1. Taa Marbouta Letter: Ending Taa Marbouta is converted to normal Taa when concatenated to another segment, as it never appears in starting/middle state. Splitting off segments might require recovering the Taa Marbouta letter.

2. Maddah Diacritic: which is originally constructed from two letters: (“أ+ا”, *>a+A*, Hamza with fatha and Alif). For example, questioning Alif is converted into Alif with Maddah when concatenated to word with starting Alif. When splitting off segments, the Maddah diacritic should be return to its original two letters.

3. Concatenation of Prepositional Lam and The Determiner Al: drops the Alif of the determiner and in special cases drop both letters. Recovering those dropped letters depends on the context.

4. Consonant Gemination Mark (i.e. Shaddah): which indicates consonant doubling of the letter. However, it happens that the gemination is caused by attaching a clitic to the word; thus, the letter correlates with both segments. For example, possessive Yaa is converted into a consonant gemination mark when attached to nominal with an ending *y*. Prepositional *mino* when concatenated with relative *maA* is shortened as "*mimaA*".

These differences result in different forms of the same segment between taggers; for example, "*wa+mi+mA*" is morpheme-based aligned as following:

	MA	**ST**	**AM**	**FA**
wa	wa/conj	w/conj	w/CC	w/CONJ
mi	mino/prep	m/IN	mmA/NN	mmA/part
m~aA	mA/rel	mA/WP	-	-

TABLE 1: Different Recovery of Word’s Segments.

This illustrates incompatible segmentation schemas, and more importantly it shows that MA recovered `mino` original form and therefore an extra letter (`no`) is added. QAC, the gold standard corpus, is segment-based tagged but converted letters were not recovered after splitting off segments. As a result, a few segments do not have a segment form (as the segment form was part of another segment, e.g. possessive Yaa).

In addition, not all taggers report the segment fully voweled (with diacritics). Back to "*wa+mi+mA*" example, we can see that only MX reports segments diacritics (letters: *a, o, i, u*).

Furthermore, taggers do not always follow the same procedure of normalization. Stanford POS tagger, for example, by default normalizes all Alif shapes into the normal Alif. This results in mismatch between characters when aligning using character-based approach.

Lastly, the script used by the QAC corpus differ significantly from the script used by input taggers. QAC corpus used Othmani script, where 43.16% of the verses and 52.80% of the words are written differently from a version written using Modern Standard Arabic script. It requires special handling and manual verification to convert it to the modern orthographical standard Arabic script. As taggers assume text to be written in modern standard orthography, we used Tanzil Project[1] to retrieve an authenticated modern script version of the Quran text.

As a result of the above difficulties, we chose to align taggers using the labels provided instead of aligning them using the word form.

[1] http://tanzil.net/

4. CORPUS

The gold standard corpus we used is the 29th chapter of the holy Quran annotated by QAC [7] and SALMA [8]. It is nearly 1000 words, morphologically tokenized to produce 1709 (QAC) or 1942 (SALMA) morphemes. Other taggers produce different number of morphemes: 1615, 1448, 1426, 1409 tags for FA, ST, MA, and AM respectively. This shows that both QAC and SALMA used more fine-grained segmentation schemes as both were annotated with respect to traditional Arabic grammar system (Iraab). This also shows that segmentation varies widely between different Arabic POS-taggers.

The tagsets used by the two projects differ: The QAC tagset is more syntactically-driven while the SALMA tagset is more focused on the internal morphology of the words and has more morphological features (total of 22 features). QAC tagset is designed only for the Quran, thus it does not have tags for punctuation or formulae for example. In terms of basic tags, the possible number of part of speech categories (without any associated features such as gender or person) in QAC is 37: 9 tags for nominals, one for verbs, and 34 tags for particles. The SALMA tagset is more fine-grained (77 tags): 34 for nouns, one for verbs, 22 for particles, and 20 for residuals (others). Irrelevant tags (e.g. punctuation) were excluded.

Tagset of the four taggers range from 16 (FA) to 26 (AM, ST) to 59 tags (MA)[2]. The mapping is only required for a subset of these tags, as some are irrelevant to the Quran special text (punctuations, date, latin, etc). However, we do not have control of these taggers to exclude them from prediction. The SALMA tagset is much more fine-grained in the nouns, i.e. `NOUN` tag can be mapped to at least ten tags in SAL. Similarly, QAC is fine-grained in particles and adverbs.

We ran input taggers to re-tag the corpus which results in a sequence of tagged morphemes. These morphemes were manually aligned by the first author to SALMA (1942 morphemes) and QAC (1709 morphemes) to construct the reference data for evaluating the approaches. A single morpheme can be aligned to only one: if a tag can be aligned to multiple tags, we chose the most suitable tag based on the meaning as shown on Fig. 1.

5. ALIGNMENTS METHODS

5.1. Baseline Alignment

This approach is a simple method to jointly align the output of two sequences. Assuming taggers will produce a sequence of morphemes with one morpheme marked as a `primary'. Formally, let the primary morpheme be: $a_i \in A, b_j \in B$ the result of the mapping is the set of pairs: $C = \{.., (a_{i-2}, b_{j-2}), (a_{i-1}, b_{j-1}), (a_i, b_j), (a_{i+1}, b_{j+1}), (a_{i+2}, b_{j+2}), ..\}$.

To illustrate this method: Assume we have two sequences:

```
A={conj, prep, verb}
B={pc,pj,ra,vc,rb}
```

where `verb` and `vc` are the primary morphemes. The alignment result will be:

```
          | conj    | prep   | verb* |
pc        | pj      | ra     | vc*   |  rb
```

This method suffers from exceptions to three assumptions:

1. Standard Definition of Primary Morphemes: An example that illustrate the lack of this standard is the case of `PREP + PRON` which is common in Arabic; an example is (فيه, *fyh*, "in it").

[2] MA uses a separate tagset for enclitcs (24) and proclitics (28) and its tagset is designed for Egyptian Arabic dialect as well. This number is after excluding such tags and merging clitics with main tags.

2. A Word has Only One Primary Stem: which is invalid when two morphemes are equal in rank. (إِنَّمَا, *<in\~amaA*, but) was segmented by QAC into two primary morphemes: *<in\~a/ACC* + *maA/PREV*.

3. Taggers will Mark One Morpheme As Primary: Several taggers do not (e.g. ST). To overcome this issue, we gave every tag a priority and rank word's morpheme accordingly, and mark the top-ranking morpheme as primary. This might as well solve the two problems mentioned earlier; we can give a higher priority to `PRON` than `PREP`, to `ACC` than `PREV`.

In addition, the noticeable difference in the segmentation schemes makes this baseline algorithm not efficient, so we investigate three different approaches to improve the alignment.

5.2. Needleman–Wunsch Algorithm

For the following alignment approaches, we used Needleman–Wunsch algorithm [2] to compute the optimal global alignment between two sequences of tags. Needleman–Wunsch algorithm is a dynamic programming algorithm that maximizes a score computed by summing the weights of matches and penalizing for each gap inserted. The alignment depends on:

1. the penalty associated with an insertion of a gap, and
2. the weights associated with a match.

Needleman–Wunsch alignment is projective, i.e. there is no two mappings such that:

1. $\mathrm{m_i} \rightarrow \mathrm{m_j}$ where $\mathrm{m_i} \in \mathrm{M_a}$ and $\mathrm{m_j} \in \mathrm{M_b}$ and $\mathrm{i} < j$, and
2. $\mathrm{m_i} \rightarrow \mathrm{m_j}$ where $\mathrm{m_i} \in \mathrm{M_a}$ and $\mathrm{m_j} \in \mathrm{M_b}$ and $\mathrm{j} < i$.

This is helpful in our case as taggers produce tags in same order.

We adapted a scoring system which counts not only the cost of operation, but also the two tokens involved in alignment using the similarity matrix $S^{A,B}$. Matching between `noun` and `N` may be given full score, but matching between `noun` and `proper_noun` may be given a lower score. $S^{A,B} = (0 \leq \mathrm{m_{i,j}} \leq 1)$

We used three approaches to construct the similarity matrix $S^{A,B}$. The rule-based approach relies on the hand-crafted mapping rules. Using an aligned corpus to infer the rules using unigrams and bigrams, we construct matrices for the data-driven approach.

5.3. Rules-based Alignment

In this approach, rules that map one tagset to the other guide the alignment algorithm which uses these rules to constrain the alignment to only mapped pairs (if such exist).

The task was done by two linguists with background in teaching Arabic as a second language and pursing a PhD degree in computational linguistics. Mapping one tagset to another tag set requires thorough understanding of both tagsets. Using a tool, designed especially for mapping, and with the following in hand:

1. a description of each tag (extracted from the manual or the paper of the tagset),
2. in-context examples of the tag, and
3. some statistical information about the target tag (no. of inward maps, probability of such tagging without aligning),

a linguist selects all possible tags in SALMA tagset to map to, for each tag in our four POS-taggers.

Among possible mappings from MA tagset (59 tags) to SALMA tagset (77 tags), 228 were selected: 130 by both, 33 and 65 by each linguist. Some tags (e.g. "date", "currency", and not-separated affixes like Taa Marbouta, feminine suffix) were not mapped. The average number of mappings for one tag in MADAMIRA is 1.88-2.57, in SALMA is 1.98-2.15 for each linguist respectively. This indicates that the mapping between the two tagsets are mostly n-n mapping. The scoring matrix $S^{A,B}$ is constructed as follows:

$$s_{i,j}^{A,B} = s_{j,i}^{A,B} = \begin{cases} 1 \; if \; f(i,j) = 2 \\ 0.5 \;\; if \; f(i,j) = 1 \\ 0 \; otherwise \end{cases}$$

Where $f(i,j)$ is the number of mappings from tag i to tag j.

5.4. Data-driven Supervised Alignment

The second approach uses an aligned corpus to learn the probability of aligning one morpheme in one sequence to another using its information. We used our parallel annotated and aligned corpus (PAC)[3]. Incorrectly-tagged words were marked and skipped from learning. The annotation process was adaptive as each sentence's annotation is used to predict future annotations.

To construct the scoring matrix, we use normalized weighted count unigram and bigram. While these scoring systems are simple and do not account for the context, they are fast to adapt each alignment correction that the annotators make.

5.5. Unsupervised Alignment

In this approach, the word alignment task in Statistical Machine Translation was used as a model. Given a bilingual parallel corpus aligned on the sentence level, the word alignment task is to find the best alignment between words in the source language and corresponding ones in the target language. Similarly, we align tags that correspond in meaning in the source and target of morphemes of word-level parallel aligned corpus.

Using our PAC corpus, we use *fast_align* method [14] which uses expectation-maximization algorithm to maximize the likelihood of a parallel corpus.

The alignment output is not necessarily projective, as *fast_align* method is designed for word alignment of a bilingual parallel corpus. By using a priority that favours alignments that are close to "diagonal", we could recover the projectivity property.

Post-processing the output was necessary to convert n-n mappings to one-to-many mapping. Among the *m* possible mappings, and rather than basically choose the first one, we pick the most confident mapping, i.e. the most-frequent pair in the whole training.

6. EVALUATION:

6.1. Fraction of Correctly Aligned Morphemes

We evaluate different approaches of alignment using an *intrinsic* metric: The accuracy of aligning morphemes.

Using 80-20 split for training and testing, we report overall accuracy: the fraction of morphemes that have been correctly aligned to the PAC gold-standard corpus. An incorrect alignment will cause at least a doubled penalty in this metric.

[3] http://github.com/aosaimy/ch29_pac

Mapping	Ru	unigram	bigram	baseline	unsup	unsup*
AM → QA	0.91	0.97	0.83	0.95	0.9	0.91
AM → SW	0.9	0.96	0.72	0.94	0.83	N/A
FA → QA	0.91	0.99	0.84	0.95	0.95	0.95
FA → SW	0.97	0.99	0.95	0.95	0.96	N/A
MA → QA	0.92	0.95	0.81	0.91	0.92	0.92
MA → SW	0.93	0.94	0.71	0.9	0.83	N/A
ST → QA	0.94	0.98	0.82	0.92	0.89	0.9
ST → SW	0.93	0.96	0.72	0.91	0.82	N/A

TABLE 2: The accuracy of aligning morphemes using five approaches of alignments.

Since there is a chance that a tool incorrectly tagged a word and this word will contribute to the error rate of the alignment, we decided to mark those words and exclude them from our evaluation. We exclude them in the training part of the data-driven approaches as well. While sometimes just one morpheme is marked incorrectly, we mark the whole word as incorrect and exclude all its morphemes.

We performed 5-fold cross-validation for the supervised approach. The unsupervised model used the full unaligned corpus for training. However, evaluation is based on the same test portions as the supervised approach. In *unsup** column, we report the accuracy of unsupervised learning from a larger training data (nine times original size), and accuracy has increased by around 0.5-1%.

The results in table 2 show that the unigram model outperforms all other models in all our tagsets mappings. Even though the bigram model uses more context information to predict alignment, the bigram model suffered from insufficient training corpus. We might later adapt the bigram model to back off to unigram, or more generally Katz's backoff model [15]. We can see that aligning taggers to SALMA is more difficult than to QAC. This is because QAC uses segmentation and labelling schemes more compatible with input taggers. However, FA seems to be more compatible with SALMA than QAC. The unsupervised method suffered from the post processing step which converts n-n mappings to 1-1. Both basic and most-confident suffer from cases where a tag is more associated with another tag, e.g. verbs frequently collocate with pronouns. For example, using the basic method, the tag verb would have paired with `ra` (imperfect particle) instead of `vc` (imperfect verb). Since the pair (`verb`, `vc`) is more common, most-confident method will pick this pair instead. While the most-confident method should improve the accuracy, it fails to choose the right pair when there are affixes that appear more than their stem. For example, the tag `noun` was paired with `nu` (active paritical noun) and `rm` (masculine plural sound suffix) but since SALMA tagset is finer grained, noun can be mapped to at least 15 possible tags, which lowers the probability of `noun` → `nu`. Thus, this method chose the incorrect pair (`noun`, `rm`).

6.2. Evaluation: Effect of Alignment on Ensemble POS Tagging

We evaluated different approaches to alignment using an *extrinsic* metric: The accuracy of POS tagging each morpheme with an ensemble model using different alignments of input taggers. In this evaluation, we used the QAC POS-tagging of several chapters from the Quran (2,3,4). We used our PAC corpus (chapter 29) for training the unigram approach, and we compare alignment methods with an ensemble tagger with "no" alignment; i.e. morphemes were aligned using their natural order.

We extend the alignment algorithm to work for multiple input taggers. We used a simple method: having two sets: aligned and non-aligned, we sequentially align one from non-aligned with last-added tagger in aligned set.

Formally, let T be the set of taggers and P be the set of aligned taggers initialised by randomly adding one tagger from T to it. Then, we select and align a randomly picked tool from $T - P$ and align it with p_{m-1} then add it to P. While this greedy algorithm does not ensure optimal multi-sequence alignment, it performs well enough in our parallel corpus and its decrease in accuracy seems negligible: 0.025 on the last-added tagger of 4-tagger ensemble. Errors of prior alignment are propagated to next pair. The decrease in accuracy was caused mainly from incorrectly labelled words; i.e. aligning two incompatible outputs.

We used Random Forest implemented in WEKA toolkit [16] for classification. Only aligned labels were provided to the classifier. We do not edit mislabelled segments, nor ignore them in training. Note that our individual data points were assumed to be independent, and we rely on input taggers to consider context for classification. A sample of the input to the classifier is Table 3. MA, AM, FA and ST are four POS-taggers in the ensemble tagger, and QA is Class to be predicted.

MA	AM	FA	ST	QA
verb	VBD	V	VBD	V
prep	NN	PREP	IN	P
2ms_pron	PRP	PRON	IN	PRON
det	DET	DET	DT	DET
noun	NN	NOUN	NN	N
prep	IN	PREP	IN	P
det	-----	DET	DT	DET
noun	NN	NOUN	NN	N

TABLE 3: A sample of input to ensemble POS tagger.

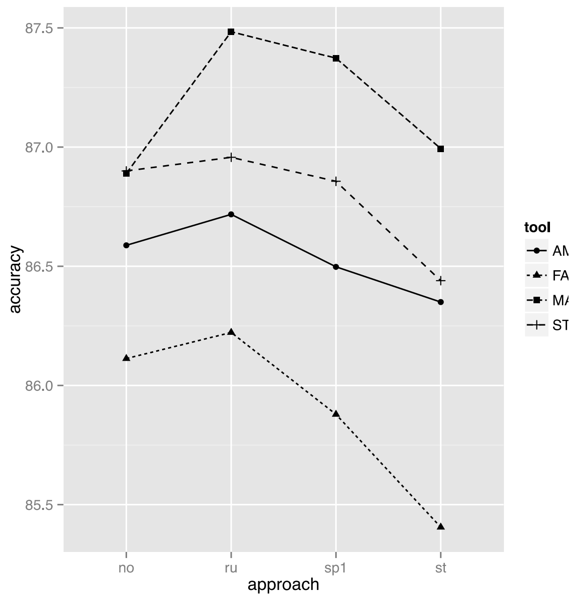

FIGURE 2: The average accuracy of each input tagger against different alignment approaches.

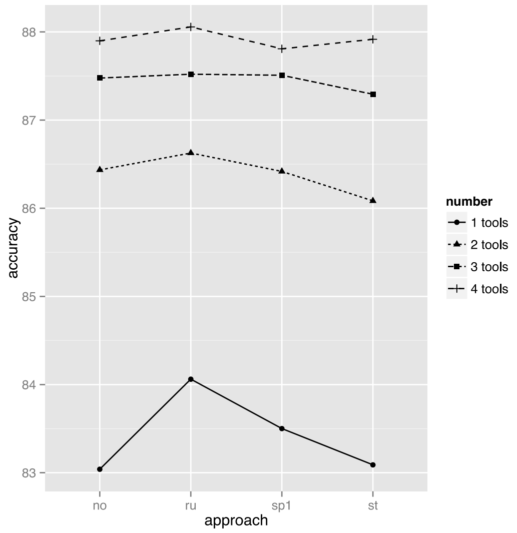

FIGURE 3: The effect of increasing the number of input taggers against different alignment approaches.

The results show that as we increase the number of taggers, the accuracy improves, see Fig 3. We can see that the effect of alignment decreases as we increase the number of input taggers though. Errors generated from the greedy method might have cancelled the gain of more taggers in the ensemble tagger. The ensemble tagger improved the accuracy over the best input tagger by at least 1.7%. The best ensemble tagger was an ensemble of AM, ST, and MA taggers with an accuracy 88.09%, 88.07%, 87.88%, 87.74% (using unigram, rule-based, baseline, and without any alignment respectively). However, the ensemble of all four input taggers performed a little bit worse: 87.80%, 88.06%, 87.92% and 87.90%.

The alignment between taggers seems to increase the tagger performance slightly. Overall, the average improvement in accuracy is 0.01 and 0.036 for unigram and rule-based respectively. The fact that rule-based performed better than unigram does not contradict the intrinsic evaluation, as we were eliminating words that were incorrectly labelled in intrinsic evaluation. Training dataset for ensemble tagger is considerably larger than one used in intrinsic evaluation. Unseen tags might contribute to the difference as well.

Input taggers differ in their contribution to the ensemble tagger. Fig 2 shows the average accuracy of all combinations of ensemble taggers that include the selected input tagger. It shows that MA contributes the most to the ensemble, and alignment improved its accuracy noticeably. This contribution might be due to its fine-grained tagset. While FA used a more fine-grained segmentation scheme, its small tagset makes it less helpful to the ensemble tagger.

One major disadvantage of this alignment is the dropping of segments that never appear in input taggers. One example is EMPH tag, which was used in QAC to mark the EMPH enclitic in verbs (See Fig. 1). Input taggers never segment this enclitic, instead tag it as a part of the verb.

6.3. Comparative Evaluation of The Ensemble POS Tagger

We compare the accuracy of our ensemble tagger with other related POS taggers. Alashqar [17] used six different taggers (Unigram, Bigram, Trigram, Brill, HMM, and TnT) trained on the Quranic Arabic Corpus. The best accuracy achieved was 80.4% using full QAC tagset. However, his result is not directly comparable since it uses different version of QAC. QAC in version prior to 0.4 is word-based annotated but in the version 0.4 is morpheme-based annotated. Alabbas [13] combined AMIRA, MADA, and a maximum-likelihood tagger to build an ensemble tagger. The work is not directly comparable as the test tagset is different; the author used ensemble tagger on Modern Standard Arabic compared to our Classical Arabic corpus. In addition, authors assumed gold-standard segmentation when aligning AMIRA output to the test dataset. Thus, we found ourselves limited to the available POS taggers. POS taggers (MX, ST, AM, FA) used different test dataset and different tagset when reporting their accuracy, which prevent us from making a direct comparison with our ensemble system. In addition, they are designed for tagging Modern Standard Arabic scripts. However, we post-process their results to match our test dataset by training a simple model (Random Forest using WEKA toolkit) to predict QAC tag using the input tool tag and the word form. The model of MX, ST, AM, and FA taggers correctly converted 83.86%, 84.78%, 83.28%, and 80.22% respectively. This should as well give a rough estimation on how likely an input tagger can help our ensemble tagger.

7. CONCLUSION

We presented and compared three methods of alignment between morpheme segmentations generated by different Arabic POS-taggers. We show how the problem is not trivial as it is dealing with different tokenisation and labelling standards. The supervised learning approach using a unigram model had the best alignment accuracy. The small size of our training corpus may be a reason why a bigram model scored lower alignment accuracy; however, the corpus was clearly at least large enough to train a useful unigram model. The rule-based approach also scored lower alignment accuracy, suggesting that it is very difficult even for Arabic linguistics experts to formulate rules for mapping between different detailed POS-tagging schemes. We used these alignment methods to build an ensemble tagger, and reported the accuracy of different ensembles based on these alignment methods. We show that using alignment improved the ensemble POS-tagger accuracy by 3.6%, a significant improvement in the domain of POS-tagging.

For future work, we might consider adding morphological features to the alignment methods, especially for the unsupervised approach. We might try the character-based approach after unifying the segment forms between input taggers to jointly do segmentation and labelling to overcome the missing segments problem. Also, we plan to extend our gold standard corpus, to include other Classical Arabic sources including Hadith; and to apply our POS-tagger on a larger scale, to accurately POS-tag a larger Corpus of Classical Arabic.

8. ACKNOWLEDGMENTS

Abdulrahman Alosaimy owes his deepest gratitude to Al-Imam University in Riyadh, Saudi Arabia for funding this research and his PhD study. Eric Atwell is grateful for EPSRC grant EP/K015206/1 Natural Language Processing Working Together with Arabic and Islamic Studies. We thank Abdullah Alfaifi (Al-Imam University, Riyadh and University of Leeds) and Ayman Alghamdi (Umm Al-Qura University, Mecca and University of Leeds) for volunteering their Arabic linguistics expertise in advising our research and mapping tagsets. We thank also the developers of POS-taggers MADAMIRA (MA) [3], Stanford Tagger (ST) [4], AMIRA (AM) [5], Farasa (FA) [6] and two Classical Arabic gold standards: Quranic Arabic Corpus (QAC) [7] and SALMA Standard Arabic Linguistics Morphological Analysis (SAL) [8] for making their tools and resources available for use.

9. REFERENCES

[1] Paroubek, P. “Evaluating Part-of-Speech Tagging and Parsing Patrick Paroubek”. Evaluation of Text and Speech Systems, 2007

[2] Needleman, S.B., and C.D. Wunsch. “A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins”. Journal of Molecular Biology. vol. 48, 1970, pp. 443–53

[3] Pasha, A., M. Al-Badrashiny, M. Diab, A. El Kholy, R. Eskander, N. Habash, and others. “Madamira: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic”. in Proceedings of the Language Resources and Evaluation Conference (LREC), Reykjavik, Iceland, 2014

[4] Toutanova, K., D. Klein, and C.D. Manning. “Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network”. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1 (NAACL ’03), 2003, pp. 252–59

[5] Diab, M. “Second Generation AMIRA Tools for Arabic Processing: Fast and Robust Tokenization, POS Tagging, and Base Phrase Chunking”. ed. by Khalid Choukri and Bente Maegaard. Conference on Arabic Language Resources and Tools, 2009pp. 285–88

[6] Zhang, Y., C. Li, R. Barzilay, and K. Darwish. “Randomized Greedy Inference for Joint Segmentation, POS Tagging and Dependency Parsing”. Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 42–52

[7] Dukes, K., E. Atwell, and N. Habash. “Supervised Collaboration for Syntactic Annotation of Quranic Arabic”. Language Resources and Evaluation, 2013

[8] Sawalha, M., E. Atwell, and M. a M. Abushariah. “SALMA: Standard Arabic Language Morphological Analysis”. 2013 1st International Conference on Communications, Signal Processing and Their Applications, ICCSPA, 2013, 2013

[9] Adda, G., J. Mariani, J. Lecomte, P. Paroubek, and M. Rajman. “The GRACE French Part-of-Speech Tagging Evaluation Task.”. International Conference on Language Resources and Evaluation, Granada, May. vol. 1 1998, pp. 433–441

[10] Hughes, J., C. Souter, and E. Atwell. “Automatic Extraction of Tagset Mappings from Parallel-Annotated Corpora”, 1995, pp. 8

[11] Atwell, E., J. Hughes, and C. Souter. “AMALGAM: Automatic Mapping Among Lexico-Grammatical Annotation Models”. Proceedings of ACL Workshop on The Balancing Act: Combining Symbolic and Statistical Approaches to Language, 1994, pp. 11–20

[12] M. Kurimo S. Virpioja, V.T.E.A. “Overview and Results of Morpho Challenge 2009”. Access Evaluation, 2009

[13] Alabbas, M.A.S. “Textual Entailment for Modern Standard Arabic”, 2013

[14] Dyer, C., V. Chahuneau, and N.A. Smith. “A Simple, Fast, and Effective Reparameterization of Ibm Model 2”, 2013

[15] Katz, S., L. Lamel, and G. Adda. “Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer”. IEEE Transactions on Acoustics, Speech, and Signal Processing. vol. 35, 1987,pp. 400–401

[16] Breiman, L. “Random Forests”. Machine Learning. vol. 45, 2001, pp. 5–32

[17] Alashqar, A.M. “A Comparative Study on Arabic POS Tagging Using Quran Corpus”. Informatics and Systems (INFOS), 2012, pp. NLP-29-NLP-33