Large Scale Arabic Error Annotation: Guidelines and Framework

Wajdi Zaghouani[1], Behrang Mohit[1], Nizar Habash[2], Ossama Obeid[1], Nadi Tomeh[3], Alla Rozovskaya[2], Noura Farra[2], Sarah Alkuhlani[2] and Kemal Oflazer[1]

[1]**Carnegie Mellon University in Qatar**
{wajdiz,behrang,oobeid}@cmu.edu, ko@cs.cmu.edu
[2]**Center for Computational Learning Systems, Columbia University**
{habash,alla,noura,salkuhlani}@ccls.columbia.edu
[3]**Université Paris 13, Sorbonne Paris Cité**
nadi.tomeh@lipn.univ-paris13.fr

Abstract

We present annotation guidelines and a web-based annotation framework developed as part of an effort to create a manually annotated Arabic corpus of errors and corrections for various text types. Such a corpus will be invaluable for developing Arabic error correction tools, both for training models and as a gold standard for evaluating error correction algorithms. We summarize the guidelines we created. We also describe issues encountered during the training of the annotators, as well as problems that are specific to the Arabic language that arose during the annotation process. Finally, we present the annotation tool that was developed as part of this project, the annotation pipeline, and the quality of the resulting annotations.

Keywords: Error Annotation, Arabic, Guidelines

1. Introduction

The Arabic language presents many challenges for natural language processing (NLP), especially because of its orthographic conventions and morphological complexity (Habash, 2010). Learners of Arabic as a second language (L2) have to face the problem of grammatical complexity, adapt to a different script and a different vocabulary. Even native Arabic speakers have trouble managing the complex vocabulary and grammar primarily because the spoken dialects of Arabic are quite different from Modern Standard Arabic (MSA). These linguistic factors contribute to the proliferation of errors in written Arabic texts produced by both native and non-native speakers. Consequently, the development of an Arabic error correction system is an important goal. Unfortunately, existing systems are not able to address many kinds of errors, as the current techniques are limited in their scope and accuracy (Shaalan et al., 2012). The majority of automatic spell checkers for Arabic perform error correction out of context (Rytting et al., 2011), although some approaches also consider contextual information (Alkanhal et al., 2012) or conduct spelling modification as part of other applications (Habash, 2008). Overall there is a lot that can be done to improve existing methods of automatic error correction for Arabic.

This paper presents the work carried out within the framework of the Qatar Arabic Language Bank (QALB) project, a large-scale error annotation effort that aims to create a manually corrected corpus of errors for a variety of Arabic texts (the target size is 2 million words).[1] The present work was motivated mainly by the lack of equivalent large-scale corpora covering various text genres. Moreover, most of the available corpora are usually designed for language learner studies and their annotation is not always suitable for the purpose of building automatic correction systems. The resulting QALB corpus can be used for building Arabic error correction tools and as a gold standard in the evaluation of error correction systems. To the best of our knowledge, this is the first published error annotation effort for Arabic of this scale.

The corpus that we present contains a variety of text genres and sources. In order to address the problem of error sparsity, we selected texts that are likely to have a high error density. These include: (a) user comments on news websites, (b) native speaker essays (c) non-native speaker essays, (d) machine translation output.

Building of any manually annotated corpus presents a variety of challenges. To address these challenges, we created comprehensive annotation guidelines that were used by a team consisting of 10 junior annotators and a lead annotator. To ensure a maximal agreement among the annotators, various training sessions were provided and regular inter-annotator agreement measures were performed to check the annotation quality.

Overall, the QALB corpus has the following features that distinguish it from other annotation projects:

- **Aim:** designed mainly as a resource for developing automatic correction systems.
- **Size:** 2 million words.
- **Text types:** news comments, essays authored by native and non-native speakers, machine translation output.

[1] http://nlp.qatar.cmu.edu/qalb/

- **Annotation history:** history of all of the correction actions is provided.

- **Annotation tool:** the annotation tool we built is designed to manage large-scale, distributed annotation projects.

At the time of writing this paper, 1.2 million words were annotated out of the 2 million words targeted. Most of the data annotated so far will be available to the participants of the shared task on error correction for Arabic that will take place during the EMNLP 2014 Arabic NLP workshop.

In the next sections, we briefly review related work (Section 2), describe the corpus, guidelines, and error types (Sections 3-5), and present our annotation logistics and workflow (Section 6). Finally, we present the annotation web interface and output format (Section 7) and an evaluation of the annotation quality (Section 8).

2. Related Work

Currently available manually corrected corpora are generally limited when it comes to the size and the genre of data. In fact, most of the available data is limited to corpora produced by L2 learners. Several corpora of learners of English annotated for errors are publicly available (Rozovskaya and Roth, 2010; Yannakoudakis et al., 2011; Dahlmeier et al., 2013), ranging in size between 60,000 words and more than one million. Another example of a corpus larger than 100K words is the 500K word Chyby corpus of native Czech errors (Pala et al., 2003; Bušta et al., 2009). Dickinson and Ledbetter (2012) annotated errors in student essays written by learners of Hungarian from three proficiency levels at Indiana University. The annotation was performed using EXMARaLDA (Extensive Markup Language for Discourse Annotation), a freely available tool that allows multiple and concurrent annotations (Schmidt, 2010). Student errors were marked according to various categories such as phonology, spelling, agreement and derivation errors.

For Arabic, Abuhakema et al. (2008) annotated for errors a small corpus of 9,000 words of non-native text. Alkanhal et al. (2012) created a manually corrected corpus of 65,000 words consisting of essays written in Arabic by students from two universities. This corpus includes two sources of errors: actual spelling mistakes generated by the students and transcription mistakes generated by the transcribers. Farwaneh and Tamimi (2012) created a 50,000 word corpus of essays produced by L2 and heritage students of Arabic, collected over 15 years of teaching.

More recently, a notable work in progress has been initiated by Alfaifi and Atwell (2012) with the goal of building a 282,000 word Arabic learner corpus. The corpus consists of written and spoken materials produced by native and non-native learners of Arabic from pre-university and university levels. The authors plan to tag errors in the corpus according to a labeling system inspired by Abuhakema et al. (2008). The actual errors will be corrected and the corpus will be annotated with part-of-speech (POS) tags.

3. Corpus Description

The building of the QALB corpus is currently in progress. The resulting corpus will contain texts from a variety of sources, thereby allowing for the development of a robust error correction system. Below we describe how the data for annotation is selected. The *user comments* portion of the corpus is provided by Al Jazeera, an Arabic news channel. The comments are randomly selected from the available comments related to news stories. The *native student essays* data is extracted from the corpus created by Alkanhal et al. (2012). Note that in order to be consistent with our annotation schema, we do not use annotations provided with the corpus but only the source (uncorrected) texts. For the *non-native data*, we selected the Arabic Learners Written Corpus (Farwaneh and Tamimi, 2012). The data is categorized by the student level (beginner, intermediate, advanced), learner type (L2 vs. heritage), and essay type (description, narration, instruction). Finally, we plan to include more data from the following two sources: first, some *machine translation* data will be selected from various Wikipedia introductory sections translated from English to Arabic automatically using Google Translate; and second, we plan to annotate both the native and the non-native data from the Arabic corpus created by Alfaifi and Atwell (2012).

4. Development of the Guidelines

Annotation guidelines typically document the core of the annotation policy. Our guidelines detail the types of errors that are targeted and describe the process of how to correct them, including how to deal with borderline cases. Many annotated examples are provided in the guidelines to illustrate the various annotation rules and exceptions.

We adopted an iterative approach for developing the guidelines, which includes multiple revisions and updates needed to reach a consistent set of directions. For instance, several changes to the guidelines were needed to address the issue of dialect words correction, and whether or not to correct or ignore certain word categories. Specifically we took the following steps:

1. Arabic writing errors were defined and classified.
2. Although we have one main set of guidelines, specific additions were developed to address unique aspects of specific text types, e.g., native speaker comments as opposed to non-native essays or machine translation.
3. Text samples from each text type were annotated as part of an annotation trial to help us revise and improve the guidelines. For news comments, the sample consisted of 50 documents.
4. In order to help our annotators with some complex Arabic spelling correction rules, we compiled a summary of the most common spelling rules as an appendix to the guidelines.

The annotation guidelines will be published as a technical report and will be available in the near future on the QALB project web page.

5. The Definition of Errors in the Guidelines

Errors in any natural language can be defined as a deviation from the standard language norms in grammar, syntax, punctuation, etc. They can also be classified according to basic types: omission, addition, substitution, or related to word order. In order to help the annotators understand the types of errors to be corrected, we defined these types of errors in the annotation guidelines. It should be noted that the task of the annotators is limited to correcting the errors, and there is no requirement to indicate the error category. Furthermore, to reduce over-correction and improve annotation consistency, the annotators were asked not to make modifications pertaining to changing informal or colloquial writing style, which is considered by some to be less acceptable than formal style.

We group the errors to be corrected into seven main categories: spelling, punctuation, word choice, morphology, syntax, proper names and dialectal word usage.

5.1. Spelling Errors

Spelling errors occur when at least one of the characters in a word is deleted or substituted for another character, or when an extra character is inserted. Some of these errors result in non-words and some results in semantically incorrect words in context. For example, the non-word طيطمقراطية *TyTmqrATyħ*[2] should be corrected to ديمقراطية *dymqrATyħ* 'democracy'. Another frequent source of spelling errors is the merge of two words which generates a non-word. As an example, we consider the two words رئيس الجمهورية *rŷys Aljmhwryħ* 'the president of the republic'; when merged together, they result in the wrong word رئيسالجمهورية *rŷysAljmhwryħ*. The annotator should add a space to split the word in such cases.

The following four types of spelling error are especially common in Arabic:

1. The *Hamza* (glottal stop) consonant has seven possible forms which Arabic writers often confuse.

Arabic:	أ	إ	آ	ا	ؤ	ئ	ء
Transliteration:	*Â*	*Ǎ*	*Ā*	*A*	*ŵ*	*ŷ*	*'*

2. The *Alif-Maqsura* ى *ý* and *Ya* ي *y* are frequently inverted or confused, e.g., compare علي *ʕly* 'Ali' with على *ʕlý* 'above'.

3. The Ta-Marbuta ة *ħ* and the Ha ه *h* are frequently confused, e.g., compare مكتبة *mktbħ* 'library' with مكتبه *mktbh* 'his office'.

4. Errors caused by erroneous pressing of a neighboring keyboard keys, e.g., خرير *xryr* 'trickling' versus حرير *Hryr* 'silk'.

[2]Arabic transliteration is presented in the Habash-Soudi-Buckwalter scheme (Habash et al., 2007): (in alphabetical order) *AbtθjHxdðrzsšSDTDŠʕγfqklmnhwy* and the additional symbols: ' ء, Â أ, Ǎ إ, Ā آ, ŵ ؤ , ŷ ئ , ħ ة, ý ى.

5.2. Punctuation Errors

The rules of punctuation vary with language and register. Moreover, aspects of punctuation use vary from author to author, and can be considered a stylistic choice. While punctuation in the English or French language is guided by a series of grammar-related rules, in other languages such as Arabic, punctuation is a recent innovation as pre-modern Arabic did not use punctuation (Zaki, 1995). According to Awad (2013), there is an inconsistency in the punctuation rules and usage in Arabic, and omitting the punctuation marks is a very frequent error. Punctuation errors are especially present in student essays and online news comments. This is mainly due to the fact that some punctuation mark rules are not clearly defined in Arabic writing references. Table 1 shows an example of two punctuation errors and their corrections.

Error	أحب السفر كل ،[1] صيف ولكن لن أسافر هذا العام *ÂHb Alsfr kl ,[1] Syf wlkn ln ÂsAfr hðA AlʕAm*
Edit	أحب السفر كل صيف ولكن لن أسافر هذا العام .[2] *ÂHb Alsfr kl Syf wlkn ln ÂsAfr hðA AlʕAm.*[2]
English	I like to travel every summer but I won't this year.

Table 1: Example of two punctuation errors. A comma is used in the wrong place (superscript 1) and the sentence does not end with a period (superscript 2).

We created a set of simple rules for correcting punctuation. For example, our Arabic comma correction rules state the following four uses as valid: (1) to separate between coordinated and related sentences, usually between short sentences, in order to specify that there is a continuation in the topic; (2) during enumeration to avoid repetition; (3) to provide an explanation or a definition of the previous word; and (4) to separate between parts of the conditional sentences.

5.3. Word Choice Errors

This category includes the use of an incorrect word. We made it clear in the guidelines that only wrong word choices are considered for correction, while style changes should not be made since the goal is not to correct or improve the writing style of the text. Word choice errors are especially common among Arabic L2 students. For example, the word أمر *Âmr* 'matter' is replaced by the wrong word أمل *Âml* 'hope' in the sentence shown in Table 2. Although these errors can be explained as spelling errors (akin to the examples in Section 5.1.), they tend to be less common and harder to spot without the larger context.

Error	هذا **أمل** مهم جدا. *hðA Âml mhm jdA.*
English	This is a very important **hope**.
Edit	هذا **أمر** مهم جدا. *hðA Âmr mhm jdA.*
English	This is a very important **matter**.

Table 2: Example of a word choice error.

5.4. Morphology Errors

In this category, the errors are usually related to an incorrect derivation or inflection, or incorrect templatic or concatenative morphology. The annotator should be aware of the Arabic morphological inflection rules and their exceptions in order to be able to correct this type of errors. For example, the word سألو *sÂlw* 'they asked' uses the wrong form of the third person plural suffix واو+ *+wA* and should be corrected to سألوا *sÂlwA*.

5.5. Syntactic Errors

These errors include wrong agreement in gender, number, definiteness or case, as well as wrong case assignment, wrong tense use, wrong word order, and missing word or redundant/extra words. For each of these errors, illustrative examples are provided in the guidelines. The example in Table 3 shows an agreement error.

Error	أنا أدرس في الجامعة **الجديد**[1] *ÂnA Âdrs fy AljAmçħ* ***Aljdyd***[1]
Edit	أنا أدرس في الجامعة **الجديدة**[2] *ÂnA Âdrs fy AljAmçħ* ***Aljdyd****ħ*[2]
English	I study in the new university

Table 3: Example of a syntactic error. There is an agreement error in gender in the word الجديد *Aljdydħ* 'new' (superscript 1). The error is corrected in (superscript 2) by making the word feminine.

5.6. Proper Name Errors

In the current guidelines, we set specific rules to correct errors that occur in spelling proper names of persons, organizations, and locations, especially those of foreign origin which could be incorrectly transliterated or are transliteratable in different ways. The guidelines specify that when in doubt of a particular named entity spelling, the annotators should verify the most commonly used variation in sources such as the Arabic Wikipedia.[3] If the text uses one of multiple widely acceptable transliterations, the annotators should not modify the word. Examples of named entity errors that require correction include جورج بش *γwrγ bš* 'George Bush' which should be corrected to جورج بوش. Examples of acceptable proper name variations include فلادمير بوتين *flAdmyr bwtyn* and فلادمير بوتن *flAdmyr bwtn* 'Vladimir Putin'.

5.7. Dialectal Usage Correction

In our current annotation task we neither address dialectal Arabic spelling normalization (Eskander et al., 2013), nor do we do a systematic translation of dialectal words into MSA (Salloum and Habash, 2011; Salloum and Habash, 2013). We recognize that the Arabic language is in a diglossic situation and borrowing is frequent. We are more interested in reducing various spelling inconsistencies that do occur. Most of the texts provided for annotation are in MSA, but dialectal words are used sometimes. Some dialectal words are morphologically related to an equivalent MSA word while others are not. Overall, we classify dialectal word issues into five categories which are partly inspired by Habash et al. (2008): dialectal lexical choice, pseudo-dialectal lexical choice, morphological choice, phonological choice and closed class dialectal words. Only the last three categories are considered for correction.

Lexical choice In this category, words are generally specific to a region and are not morphologically related to an MSA word, e.g., كرهبة *krhbħ* 'car' in Tunisian Arabic. The annotators are asked to skip such words and not modify them.

Pseudo-dialectal lexical choice In this category, MSA words appear dialectal because they are infrequently used in MSA, but they are listed in MSA dictionaries. Examples include زعلان *zçlAn* 'sad' and أتوبيس *Âtwbys* 'bus'.[4]

Morphological choice In this category, we have dialectal words that are related to MSA words but use dialectal morphology, e.g., بيكتبوا *byktbwA* 'they write', which we correct to MSA يكتبون *yktbwn* or يكتبوا *yktbwA* depending on the verb mood in context.

Phonological choices In this category, dialectal words are phonologically related to MSA words with a slight regional variation like the Levantine word جزور *jzwr* 'roots', which would be corrected to MSA جذور *jðwr*.

Closed class dialectal word In this category, a limited number of frequent closed class words like pronouns or verbal particles are mapped to MSA. Examples include the Levantine relativizer الي *Aly* 'who, which' which we map to the appropriate MSA relativizer in context such as الذي *Alðy* 'who [masc.sg.]' or التي *Alty* 'who [fem.sg.]'.

5.8. Additional Annotation Rules

In addition to the different types of errors that need to be corrected, we specified some general rules on how and when to correct errors. The top two rules are as follows. First, texts should be corrected with a minimum number of edits. The goal is to correct the errors and not the style of writing. And secondly, if the text is highly dialectal (more than a third of the words), or it contains some dialectal words in a dialect not familiar to the annotator, then the annotator is expected to flag the segment to be considered by the annotation manager for review (see Section 6. for more details on the flag feature). We expect that as we extend the guidelines to work on machine generated text or L2 speaker texts, additional general rules may be needed.

[3] http://ar.wikipedia.org/

[4] We advise our annotators to check carefully for such cases using online resources such as `http://www.almaany.com`.

6. The Annotation Logistics and Workflow

The annotation of a large scale corpus requires the involvement of multiple annotators. In our project, the annotation effort is led by an annotation manager, and the team also consists of junior annotators and a programmer.

6.1. The Annotation Manager

The annotation manager is responsible for the whole annotation task. This includes compiling the corpus, the annotation of the gold standard Inter-Annotator Agreement (IAA) portion of the corpus, writing the annotation guidelines, hiring and training the annotators, evaluating the quality of the annotation, monitoring and reporting on the annotation progress, and designing the annotation tool specifications with the programmer.

6.2. Annotator Training

To ensure the suitability of the annotators for the error correction task, we selected only university level annotators with strong Arabic language background including some Arabic teaching experience, similar annotation experience or having a university degree in Arabic language or literature. Furthermore, the annotators were tested in an Arabic language screening test (syntax and spelling related questions). Once selected, the annotators were trained over a period of two to three weeks. During the training period, the annotators completed a course consisting of ten annotation tasks, guideline reading and several meetings with the annotation manager.

6.3. Interface Implementation

The interaction between the annotation manager and the programmer can be described as an agile project management, an iterative method of determining requirements for engineering development projects in a highly flexible and interactive manner. The annotation manager participated in the whole design of the annotation framework and was responsible for testing and reporting bugs or tool enhancement. Moreover, the annotation manager compiled and relayed to the programmer the annotators' feedback after various iterations of the annotation tool testing to improve the annotation speed and user interaction aspects of the tool.

6.4. Annotation Management

To ensure the quality of our annotated corpus, we followed three main steps with each newly hired annotator. After an initial training phase, the annotator's work is closely monitored in a revision phase. Afterwards, the annotator can join the production phase. Details on each phase are explained below.

Training phase During this phase, newly hired annotators annotate a small set of texts. They get used to the corpus, the texts, the guidelines and the annotation tool. The annotation manager monitors the annotation quality and speed of each annotator.

Revision phase In the revision phase, the annotation manager performs a detailed analysis of the annotations, especially the error analysis, the flagged unresolved annotation cases and annotator's feedback. Moreover, the annotation manager revises the guidelines. Extra training may be required for some annotators depending on their background and experience.

Production phase In the production phase, the annotation manager dispatches the tasks to the annotators. The annotators usually work independently of each other, but regular annotator meetings are held with the annotation manager, to discuss the various issues raised during the correction of the texts. The annotation manager controls the quality of the produced annotation. When the annotators are not certain how to correct a particular word, they may use the flag word feature, which will alert the annotation manager so he can correct the flagged words later on. Furthermore, a communication message board is provided as a space for the annotators to post their questions, add comments, report issues and get feedback from the annotation manager as well as the other annotators. We encouraged the annotators to use this way of communication in order to keep track of all the issues faced and to have an interaction archive that can be used later on to improve the current version of the guidelines.

6.5. The Annotation Workflow

The annotation manager is the main person responsible of the annotation workflow. Once the corpus files are selected, the annotation manager uploads the files into the annotation system to create a new project. Once uploaded, the files are automatically white-space and punctuation tokenized and processed through a tool that automatically attempts to correct common spelling errors.

The tool we use is MADA (Habash and Rambow, 2005; Habash et al., 2009), a system for morphological analysis and disambiguation of Arabic. MADA uses a morphological analyzer to produce, for each input word, a list of analyses specifying every possible morphological interpretation of that word, covering all morphological features of the word. MADA then applies a set of models to produce a prediction, per word in-context, for different morphological features, such as POS, lemma, gender, number or person. The robust design of MADA allows it to consider different possible spellings of words, especially relating to Ya/Alif-Maqsura, Ha/Ta-Marbuta and Hamzated Alif forms, which are very common error sources. MADA selects the correct form in context, thus correcting for these errors which are often connected to lemma choice or morphology.[5]

Afterwards, the annotation manager assigns the tasks to the available annotators as a first human pass. Once the text is submitted and there are no active flags during the annotation, the files are archived and ready for delivery. Otherwise, the annotation manager performs a second human pass on the flagged files.

[5]We used version 3.2 of MADA. In the future, we plan to use the most recent version of MADA, called MADAMIRA 1.0 (Pasha et al., 2014).

Figure 1: The annotation interface with a sample merge operation

```
<ACTION_HISTORY>
  <ACTION actionType="edit" annotatorID="23" newText="في" passNum="1" tokenID="4" />
  <ACTION actionType="edit" annotatorID="23" newText="الأسود" passNum="1" tokenID="9" />
  <ACTION actionType="edit" annotatorID="23" newText="الجماعي" passNum="1" tokenID="22" />
  <ACTION actionType="edit" annotatorID="23" newText="وإياكم" passNum="1" tokenID="28" />
  <ACTION actionType="edit" annotatorID="23" newText="الأمراض" passNum="1" tokenID="30" />
  <ACTION actionType="edit" annotatorID="23" newText="والآفات" passNum="1" tokenID="31" />
  <ACTION actionType="edit" annotatorID="23" newText="أذهب" passNum="1" tokenID="38" />
  <ACTION actionType="edit" annotatorID="23" newText="اليأس" passNum="1" tokenID="39" />
  <ACTION actionType="edit" annotatorID="23" newText="وأشفي" passNum="1" tokenID="40" />
  <ACTION actionType="edit" annotatorID="23" newText="أنت" passNum="1" tokenID="41" />
  <ACTION actionType="edit" annotatorID="23" newText="الشافي" passNum="1" tokenID="42" />
  <ACTION actionType="edit" annotatorID="23" newText="لأشفاء" passNum="1" tokenID="43" />
  <ACTION actionType="edit" annotatorID="23" newText="إلا" passNum="1" tokenID="44" />
  <ACTION actionType="edit" annotatorID="23" newText="أحفظ" passNum="1" tokenID="52" />
  <ACTION actionType="edit" annotatorID="23" newText="إخواننا" passNum="1" tokenID="53" />
  <ACTION actionType="edit" annotatorID="23" newText="وأحفظنا" passNum="1" tokenID="57" />
  <ACTION actionType="edit" annotatorID="23" newText="وأرحم" passNum="1" tokenID="58" />
</ACTION_HISTORY>
</DOCUMENT>
```

Figure 2: An annotated document in XML format

7. The Annotation Tool and Output Format

The Annotation Tool Our framework (Obeid et al., 2013) has three core components: the Application Programming Interface (API) server, the annotation interface, and the admin interface. The annotation interface is used by annotators to correct assigned documents. The admin interface is used by the lead annotator to manage annotators and documents, assign tasks, evaluate IAA, and monitor the overall progress of the annotation process. The API server manages all annotation and user data and serves requests from the admin and annotation interfaces. The annotation interface first displays a list of tasks assigned to an annotator. The annotator selects a task and is then presented with the annotation window (Figure 1), where corrections are performed. The annotation window displays tokens in separate boxes. Each box can either be dragged or double-clicked. Moving tokens around is achieved by dragging and dropping tokens to the desired location. Merging is achieved by dragging a token slightly to the left or to the right. Double clicking on a token presents a dialog box with a text input. The text input contains the current value of the clicked token and can be used to modify the token's text. Adding a space between the characters of a token performs a split. Annotators cannot modify a token and split it at the same time. This allows us to track individual changes so that we have a consistent action history. Finally, for cases the annotator is not sure how to correct a word, a flag word feature can be used. Once a word is flagged, the annotation manager will be notified and the word can be corrected later on in a second annotation pass.

The Annotation Output Format We record for each annotated file the list of actions taken by the annotator (edit, add, delete, split, merge, and move). These actions operate on one or two tokens depending on the action. We also supply token alignments starting from after document tokenization and before MADA to after human annotation. Figure 2 shows and example of how the annotation actions are kept in the XML annotation export file.

8. Evaluation

We evaluate two aspects of the QALB corpus: the quality of the input text before annotation, and inter-annotator agreement on the corrections. The results in this section are computed using 200 randomly selected files, with a total of 10,288 words. Each of these files is corrected by at least three different annotators.

8.1. Quality of Input Text

Quality of the input text is estimated by comparing it to the gold annotation by the lead annotator. We use word error rate (WER), computed as the percentage of erroneous words in the input text, shown in the first row of Table 4. Recall that to optimize the manual annotation process, the input text undergoes an automatic correction phase using MADA, which attempts to correct simple error such as Ta-Marbuta, Ya and Hamza errors. To assess the contribution of MADA, we evaluate its output by computing its WER with respect to the gold corrections, and compare it to the WER in the input text. Table 4 (row 2) shows that many errors in the input text are taken care of automatically by MADA. MADA annotation helped reduce the errors by approximately 9% absolute (and 27% relative) when compared to the original unannotated text on all words. Punctuation errors make up around 8.5% of all errors. If we ignore punctuation errors, MADA's contribution is close to 42% relative reduction in error.

Text	all words	no punctuation
Original (unannotated)	31.8%	23.3%
MADA output	23.0%	13.4%

Table 4: Word Error Rate with respect to the lead annotator's correction (Gold). In the "no punctuation" setting, all punctuation marks are ignored.

8.2. Inter-Annotator Agreement

We evaluate the IAA to quantify the extent to which independent annotators, excluding the lead annotator, trained using our guidelines, agree on the needed correction upon evaluating the same input text (after automatic correction with MADA). A high level of agreement between the annotators indicates that the annotations are reliable and the guidelines are useful in producing homogeneous and consistent data.

We measure IAA using two approaches. The first approach consists in averaging WER over all pairs of annotations (AWER). The higher the WER between two annotations, the lower is their agreement. The second approach consists in formulating the annotation process as assigning *labels* to the *units* of the input text. IAA is then computed by comparing the sets of labels produced by the different annotators. The units are the words of the text, while the labels are actions needed to correct each word, and thus contain {`keep, delete, replace-with(w)`} where $w \in$ vocabulary. Each annotation is aligned to the input text using minimum edit distance in order to extract the assigned labels for each annotator. We compare the obtained label assignments using pairwise percent agreement averaged over all pairs of annotations (APP). The pairwise percent agreement (also called observed agreement) is computed as the percentage of times two annotators assign the same label to a unit. The results are presented in Table 5. The high level of agreement obtained suggests that the annotators produce consistently similar results under the proposed guidelines.

IAA measure	all words	no punctuation
AWER	11.3%	3.8%
APP	93.0%	98.1%

Table 5: Inter-annotator agreement on "all words" and "no punctuation" (punctuation ignored) in terms of average WER (AWER; lower is better) and average pairwise percent agreement (APP; higher is better).

9. Current Status and Future Directions

The annotation process in the QALB project is still in progress. A portion of the corpus will be released to the participants of the Arabic error correction shared task at the EMNLP 2014 Arabic NLP workshop. Currently, the annotated corpus contains a total of 1.2M words mostly from Al Jazeera News user comments (1,170,000 words) and some native students essays (31,600 words). We will continue annotating user comments to reach 1.5M words. In the next few months, we will extend our annotation guidelines to work for non-native students essays and machine translation output, and add close to 0.5M words of annotations from these text types.

10. Conclusion

We presented a large-scale error annotation effort for Arabic, including guideline development and an annotation framework. The QALB corpus will include various data sources: news comments, machine translation output, native and non native students essays. We described the methods used to create the annotation guidelines and how the annotation effort was organized, from annotation logistics to the annotation production pipeline. Moreover, we described features of our annotation tool and how annotator feedback during a pilot annotation experiment improved the tool design and guidelines. The high agreement results obtained by the annotators during the inter-annotator agreement experiment demonstrate a good understanding and application of the guidelines. This corpus will provide researchers with a new resource for building Arabic error detection and correction tools.

11. Acknowledgements

We thank anonymous reviewers for their valuable comments and suggestions. We also thank all our dedicated annotators: Hoda Fathy, Dhoha Abid, Mariem Fekih, Anissa Jrad, Hoda Ibrahim, Noor Alzeer, Samah Lakhal, Jihene Wefi, Elsherif Mahmoud and Hossam El-Husseini. We thank Al Jazeera News (and especially, Khalid Judia) for providing the user comments portion of our corpus. This publication was made possible by grants NPRP-4-1058-1-168 and YSREP-1-018-1-004 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

12. References

Abuhakema, Ghazi, Faraj, Reem, Feldman, Anna, and Fitzpatrick, Eileen. (2008). Annotating an Arabic learner corpus for error. In *Proceedings of LREC*.

Alfaifi, Abdullah and Atwell, Eric. (2012). Arabic Learner Corpora (ALC): A taxonomy of coding errors. In *The 8th International Computing Conference in Arabic*.

Alkanhal, Mohamed I, Al-Badrashiny, Mohamed A, Alghamdi, Mansour M, and Al-Qabbany, Abdulaziz O. (2012). Automatic Stochastic Arabic Spelling Correction With Emphasis on Space Insertions and Deletions. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(7):2111–2122.

Awad, Dana. (2013). La ponctuation arabe : histoire et règles.

Bušta, Jan, Hlavácková, Dana, Jakubícek, Miloš, and Pala, Karel. (2009). Classification of Errors in Text. *RASLAN 2009*, page 109.

Dahlmeier, D., H.T. Ng, Hwee, and Wu, S.M. (2013). Building a large annotated corpus of learner english: The NUS corpus of learner english. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, Atlanta, Georgia, June. Association for Computational Linguistics.

Dickinson, Markus and Ledbetter, Scott. (2012). Annotating errors in a Hungarian learner corpus. In *Proceedings of LREC*.

Eskander, Ramy, Habash, Nizar, Rambow, Owen, and Tomeh, Nadi. (2013). Processing Spontaneous Orthography. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.

Farwaneh, Samira and Tamimi, Mohammed. (2012). Arabic Learners Written Corpus: A Resource for Research and Learning. *The Center for Educational Resources in Culture, Language and Literacy.*

Habash, Nizar and Rambow, Owen. (2005). Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan.

Habash, Nizar, Soudi, Abdelhadi, and Buckwalter, Tim. (2007). On Arabic Transliteration. In van den Bosch, A. and Soudi, A., editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.

Habash, N., Rambow, O., Diab, M., and Kanjawi-Faraj, R. (2008). Guidelines for Annotation of Arabic Dialectness. In *Proceedings of the LREC Workshop on HLT & NLP within the Arabic world*.

Habash, Nizar, Rambow, Owen, and Roth, Ryan. (2009). Mada+tokan: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In *Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR), Cairo, Egypt*, pages 102–109.

Habash, Nizar. (2008). Four Techniques for Online Handling of Out-of-Vocabulary Words in Arabic-English Statistical Machine Translation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 57–60, Columbus, Ohio.

Habash, Nizar. (2010). *Introduction to Arabic Natural Language Processing*. Morgan & Claypool Publishers.

Obeid, Ossama, Zaghouani, Wajdi, Mohit, Behrang, Habash, Nizar, Oflazer, Kemal, and Tomeh, Nadi. (2013). A web-based annotation framework for large-scale text correction. In *The Companion Volume of the Proceedings of IJCNLP 2013: System Demonstrations*, Nagoya, Japan, October.

Pala, Karel, Rychlỳ, Pavel, and Smrž, Pavel. (2003). Text corpus with errors. In *Text, Speech and Dialogue*, pages 90–97. Springer.

Pasha, Arfath, Al-Badrashiny, Mohamed, Kholy, Ahmed El, Eskander, Ramy, Diab, Mona, Habash, Nizar, Pooleery, Manoj, Rambow, Owen, and Roth, Ryan. (2014). Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *In Proceedings of the 9th International Conference on Language Resources and Evaluation*, Reykjavik, Iceland.

Rozovskaya, A. and Roth, D. (2010). Annotating esl errors: Challenges and rewards. In *Proceedings of the NAACL Workshop on Innovative Use of NLP for Building Educational Applications*, 6.

Rytting, C., Buckwalter, T., Wayland, S., Hettick, C., Rodrigues, P., and Zajic, D. (2011). Spelling correction for dialectal Arabic dictionary lookup. *ACM Transactions of Asian Language Information Processing*, 10(1):1–15.

Salloum, Wael and Habash, Nizar. (2011). Dialectal to Standard Arabic Paraphrasing to Improve Arabic-English Statistical Machine Translation. In *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, pages 10–21, Edinburgh, Scotland.

Salloum, Wael and Habash, Nizar. (2013). Dialectal Arabic to English Machine Translation: Pivoting through Modern Standard Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.

Schmidt, Thomas. (2010). Linguistic tool development between community practices and technology standards. In *Proceedings of the LREC Workshop Language Resource and Language Technology Standards state of the art, emerging needs, and future developments*.

Shaalan, Khaled F., Attia, Mohammed, Pecina, Pavel, Samih, Younes, and van Genabith, Josef. (2012). Arabic word generation and modelling for spell checking. In *LREC*, pages 719–725.

Yannakoudakis, H., Briscoe, T., and Medlock, B. (2011). A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 180–189, Portland, Oregon, USA, June. Association for Computational Linguistics.

Zaki, Ahmad. (1995). at-tarqim wa alamatu-hu fi al-lugati al-arabiyya, maktabat al-matbuat al islamiyya bi halab.