# Clear-cut methodology for Arabic OCR and post-correction with low technical skilled annotators

Alicia González Martínez
Universität Hamburg
ERC-Project COBHUNI
Edmund-Siemers Allee 1
Hamburg 20146
alicia.gonzalez@uni-hamburg.de

Tillmann Feige
Universität Hamburg
ERC-Project COBHUNI
Edmund-Siemers Allee 1
Hamburg 20146
tillmann.feige@uni-hamburg.de

Thomas Eich
Universität Hamburg
ERC-Project COBHUNI
Edmund-Siemers Allee 1
Hamburg 20146
thomas.eich@uni-hamburg.de

## ABSTRACT

This paper describes an efficient and straightforward methodology for OCR-ing and post-correcting Arabic text material on Islamic embryology collected for the COBHUNI project. As the target texts of the project include diverse diachronic stages of the Arabic language, the team of annotators for performing the OCR post-correction requires well-trained experts on language skills. While technical skills are also desirable, highly trained language experts typically lack enough technical knowledge. Furthermore, a relatively small portion of the target texts needed to be OCR-ed, as most of the material was already on some digital form. Thus, the OCR task could only require a small amount of resources in terms of time and work complexity. Both the low technical skills of the annotators and the resource constraints made it necessary for us to find an easy-to-develop and suitable workflow for performing the OCR and post-correction tasks. For the OCR phase, we chose Tesseract Open Source OCR Engine, because it achieves state-of-the-art levels of accuracy. For the post-correction phase, we decided to use the Proofread Page extension of the MediaWiki software, as it strikes a perfect balance between usability and efficiency. The post-correction task was additionally supported by the implementation of an error checker based on simple heuristics. The application of this methodology resulted in the successful and fast OCR-ing and post-correction of a corpus of 36,132 tokens.

## CCS CONCEPTS

• **Information systems** → **Information integration**; • **Applied computing** → *Optical character recognition*; Arts and humanities;

## KEYWORDS

Optical character recognition, Arabic optical text recognition, post-correction, error checking, Arabic, Classical Arabic, Modern Standard Arabic, Arabic digital humanities


*DATeCH2017, June 01-02, 2017, Göttingen, Germany*

ACM ISBN 978-1-4503-5265-9/17/06…$15.00
https://doi.org/http://dx.doi.org/10.1145/3078081.3078103

**ACM Reference format:**
Alicia González Martínez, Tillmann Feige, and Thomas Eich. 2017. Clear-cut methodology for Arabic OCR and post-correction with low technical skilled annotators. In *Proceedings of DATeCH2017, Göttingen, Germany, June 01-02, 2017,* 4 pages.
https://doi.org/http://dx.doi.org/10.1145/3078081.3078103

## 1 INTRODUCTION

Arabic literary and cultural heritage stands out for containing one of the most productive and valuable collections of libraries in the world. Traditionally, researchers used to focus their attention on relatively small amounts of texts to carry out their studies. However, in recent years, a lot of digitalization projects have emerged. These projects have made it possible to research huge libraries of accumulative knowledge right away [2]. One of these new projects that stands out in the field of Arabic Corpus linguistics is the Open Arabic project, developed by Maxim Romanov in Leipzig University [6]. This project aims at building a corpus of pre-modern texts in Arabic to encourage computational analysis of the Arabic written tradition. Up to now, the size of the corpus reaches the amount of 64,448,901 words [6]. To OCR the collection of texts, the project has used a novel software called Kraken, developed by Benjamin Kiessling [3], which relies on neural networks and has proven to be very successful in dealing with different Arabic scripts [7]. Yet, there is still a lot of work to be done–a vast amount of texts are waiting to be digitalised.

## 2 THE COBHUNI PROJECT

The COBHUNI project aims at diversifying our understanding of how pre-natal life is conceptualized in texts of Islamic normativity. To achieve this, we built a corpus of texts related with Islamic embryology and, in a following step, annotated them with semantic information. The annotation process is still an ongoing task.

### 2.1 The Arabic language

The target texts for building the COBHUNI corpus are exclusively in Arabic language. Yet, as the project spans over many different periods of the Islamic history, we are going to encounter instances of both classical Arabic (CA) and Modern Standard Arabic (MSA). CA dates back to the early stages of Islam and constitutes the traditional language variety of the Islamic high culture. One special type of CA is Quranic Arabic (QA), the language of the Holy Quran. In turn, MSA is the language currently used in formal registers of society. In general, CA differs from MSA more in the lexical stratum than in the grammar [8]. The Arabic writing system uses a cursive

**Table 1: Size of the corpus**

| Sources | No. tokens | | No.types | |
|---|---|---|---|---|
| | Complete | COBHUNI | Complete | COBHUNI |
| *altafsir.com* | 12,601,880 | 1,306,233 | 466,145 | 125,024 |
| *hadith.al-islam.com* | 11,482,139 | 144,122 | 313,846 | 22,608 |
| physical books | 36,132 | | 8,038 | |
| TOTAL | 24,120,151 | 11,486,487 | | |

script with the special feature that consonants and long vowels are always written down, while short vowels are optional–they may be written or not–the only exception to this being QA, in which all vowels are expected to be written in an effort to minimise semantic ambiguity. As a consequence of this, Arabic texts exhibit an artificially high type token ratio.

The intrinsic heterogeneity of the language in this project made it necessary to have annotators with high language skills that also had a background on Islamic embryology.

## 2.2 The COBHUNI Corpus

The texts of the corpus have been collected from different sources and they rely on diverse areas of knowledge, from religious literature to medical or legal topics. As we already pointed out, the project has also a diachronic dimension, so it includes different synchronic stages of Arabic, ranging from old CA to MSA. Up to now, we have three main sources of texts for our corpus (see Table 1):

(1) The website http://altafsir.com, the largest online repository for Quranic commentary literature.
(2) The website http://hadith.al-islam.com, which contains one of the largest collections of *hadith*[1] material on the internet. All these texts includes vocalization.
(3) A small collection of physical books that are not available in digital form.

We decided to download all texts from the websites and then make a selection of the texts that are relevant for the COBHUNI project, based on the experience of human experts. Thus, we can have a larger corpus for general purpose research, and a specialised one directly meant for the COBHUNI project. The texts from all three sources have been sanitized and stored along with relevant metadata, i.e., name of book, chapter, and so on.

The task of digitalizing the physical books is the problem described and solved in the following sections of this paper.

# 3 THE OCR PHASE: TESSERACT

Amongst the most well known OCR systems that support Arabic language are the Tesseract Open Source OCR Engine and NovoVerus. We didn't consider Kraken, the software mentioned in the introduction, because it was not available until very recently. Tesseract was originally developed by Hewlett-Packard and it's now being developed by Google [10]. Although it is considered one of the most accurate open source OCR software products, the levels of accuracy for Arabic are not very high yet–it achieves over 80% accuracy on correct words detection [1]. This is due to the fact that Arabic uses a cursive script, which poses a significant challenge to the task of word segmentation [9]. NovoVerus is presented as an OCR solution for Middle Eastern and Asian scripts and it's supposed to achieve a high accuracy too [5]. Propriety software usually claim to achieve accuracy rates not far below 100%. Yet, their evaluation texts are usually applied on texts with simplified typesets and no vocalization. Tests using high quality scans of classical texts reach accuracy rates in the range of 65% to 75%[7].

As the propriety OCR engines didn't seem to beat in reality the levels of accuracy of open source software, we focused our attention on Tesseract and NovoVerus. We applied some tests on both systems to compare their performance in our text material. Our findings show that, while NovoVerus works better on modern texts, Tesseract surpasses it on classical texts stored in good quality images. We carried out a simple experiment using the Tesseract engine–we took 137 random characters from our data and found a rate of up to 83% of correctly recognised words. Consequently, we chose Tesseract for OCR-ing the physical books of our corpus.

We stored our scans in 300 dpi uncompressed tiff files. We discarded using 600 dpi because the images ended up being overdetailed and this tends to confuse the OCR engine. The size of the OCR subset of the corpus was 36,132 tokens.

[1] *Hadiths* are collections of texts containing the words, actions and habits of the Islamic prophet Muhammad.

**Table 2: Human correctors**

| Corrector | Native | Group |
|---|---|---|
| A | yes | 1 |
| B | no | 1 |
| C | yes | 2 |
| D | no | 2 |

# 4 THE POST-CORRECTION PHASE

For correcting the output of the previously generated OCR, we needed a team of annotators, an easy-to-use platform in which the annotators could work without much training, and a quality control system.

## 4.1 Human resources

We hired four student assistants to correct the OCR-ed subset of the corpus. All students were pursuing a Master degree in areas related to the project and they had limited technical background. They were organised in pairs, each of which included one native speaker and one non-native speaker, but fluent in Arabic (see Table 2).

## 4.2 The Wiki Proofread Page extension

For carrying out the post-correction task, we decided to use the Wiki Proofread Page extension of the web-based MediaWiki software [4]. There were several reasons for taking this decision.

- As we are working with Arabic script, multilingual support is a critical requirement for us. This reduces the list of available software we can use for the post-correction task. The MediaWiki software supports Unicode and bi-directionality.
- The MediaWiki, together with the Wiki Proofread Page extension, is an easy-to-use tool for users with limited technical background. Most people have used a wiki before, albeit not contributing.
- We were already using MediaWiki for documentation purposes.
- And last but not least, it provides a configurable front-end, so features such as font size or text alignment can be customized.

The Wiki Proofread Page renders the OCR-ed text beside the original scanned image. Our human correctors only had to edit the text–as they would do in a normal wiki–following the reference of the scan. In relation with the directionality, we only had to edit the mediawiki-localization to right-align the Arabic texts. For improving readability, we increased the font-size from 13px to 17px. This was greatly appreciated by the correctors, as Arabic fonts tend to be too small, causing usability problems and resulting in errors such as the lack of space between words.

## 4.3 An ad hoc error checker

In any manual task performed by humans, errors are unavoidable, but we can minimise their presence. Thus, to support the post-correction process we implemented an error checker based on simple heuristics. The error checker splits the text into tokens and check each token separately, excluding those tokens corresponding to punctuation. When the student assistants finished the post-correction task, we passed the corrected texts through the error checker, which yields a log file with errors and warnings [2]. Error messages indicate with total certainty that something is wrong in the text. Warning messages indicate that something in the text is suspicious and must be re-checked by the human corrector.

A warning message is given if:

- The token contains a character that doesn't belong to the Arabic charset. This checking uncovered spurious noise yielded by the OCR engine.
- The token is considered too long. Based on some tests we performed, we set the threshold in a maximum of 8-character length token excluding diacritics. This checking was very useful to discover words with no space in between them. This was one of the most common errors encountered in the texts.

An error message is given if:

**Table 3: Working packages for correctors**

| Document | No. tokens | Group | Corrector | Reviewer |
|---|---|---|---|---|
| Al-Taufi | 744 | 1 | A | B |
| Al-Mulaqqin | 1525 | 1 | A | B |
| Fakihani | 2557 | 1 | A | B |
| Farhud | 4012 | 1 | A | B |
| Fashni | 2143 | 1 | B | A |
| Ibn Hajar | 8965 | 1 | B | A |
| Ibn Rajab | 3739 | 2 | C | D |
| Munawi | 5538 | 2 | C | D |
| Qadi Iyad | 1398 | 2 | C | D |
| Qurtubi | 1666 | 2 | D | C |
| Nabrawi | 3845 | 2 | D | C |

- A letter *ta marboota* ة is found in a position other than last or penultimate within the token. This is an obvious error according to Arabic orthography.
- More than one short vowels are written together. Again, this is illegal orthography. Either they are to be replaced by a *tanwin* vowel or simplified to one occurrence.

## 4.4 The workflow

The complete workflow is as follows–we scan the texts and store them into tiff files. We take these files and OCR them using Tesseract. Then, we convert the tiff images into dvju and insert the OCR texts into the same file. In order to have all the images corresponding to the same book together, we merge them into the same dvju file. At this point, we upload the OCR text along with its image into MediaWiki. Here, we perform two stages of post-correction: first, the student assistants correct the texts assigned to them, and in a second step, they review the already corrected texts that have been assigned to the other group of students. The two correctors of each group share feedback between themselves, but not with the other group. In this way, we guarantee a blind review. The working packages of each annotator can be seen in the Table 3.

When the correction and the revision are finished, the texts are exported from the wiki to json files. This is done with a python program created specifically for this project [3]. The json files are then converted into xmi. We perform this conversion because xmi is a format supported by WebAnno, the annotation tool we are using for annotating semantically the texts. To carry out the xmi conversion, we built a java program that uses the DKPro Core toolkit, a collection of software components for natural language processing based on the Apache UIMA framework [11] [4]. Just before dumping the data into xmi format, we pass the texts through the error checker. Together with the second phase of the post-correction step, this serves as a quality control measure. The student assistants must check the resulting log file and assure there are no errors or problematic warnings. Otherwise, they must go back into the wiki, correct the errors indicated in the log file, and apply the export and the conversion again. When the error checker doesn't yield more typos to correct, the process is finished.

[2]The code for the error checker can be found at https://gitlab.com/alrazi/ini_xmiconverter/blob/master/src/main/java/ini_xmiconverter/XmiConverterOcred.java

[3]The code for this subproject can be found at https://gitlab.com/alrazi/wiki_export

[4]The code for this subproject can be found at https://gitlab.com/alrazi/ini_xmiconverter

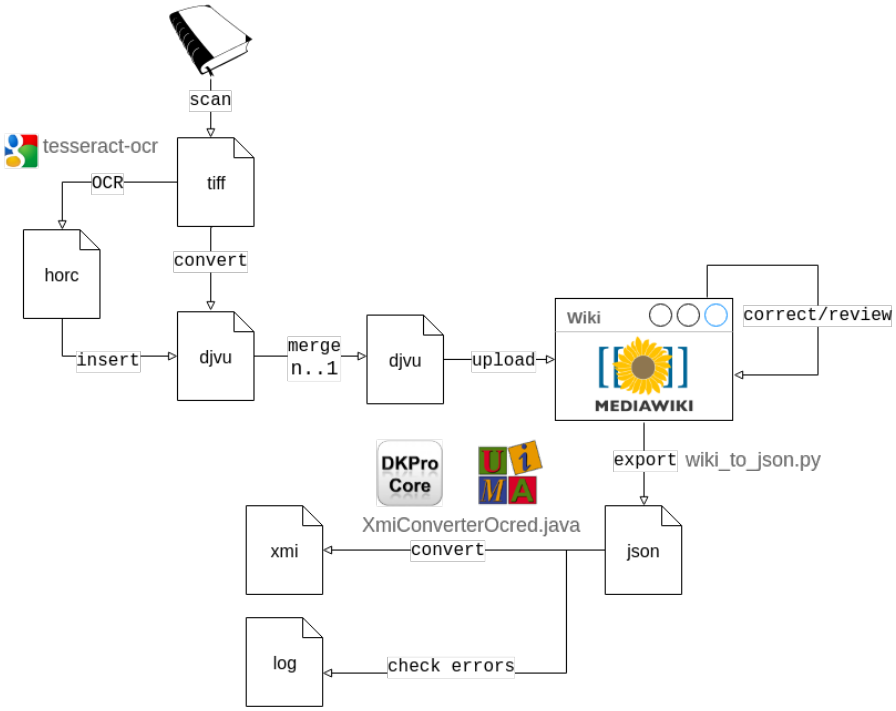


**Figure 1: Workflow of the OCR and post-correction phase of the project**

## 5 CONCLUSIONS

Arabic digital humanities is experiencing a golden era thanks to the dramatic improvements in computer technologies made in recent years. The first endeavour has obviously been directed to digitalise in a consistent and systematic way the text collections of the Arabic cultural heritage. A clear example of this is the recent creation of the OpenArabic Corpus. Yet, two main problems persist–the limited technical skills of the human correctors that prevent them to use advanced software, and the lack of universal Unicode support for bi-directional scripts, i.e., the intermingling of RLT and LTR charsets. New tools such as Kraken are starting to make the difference. We will take it into consideration for future works.

From our methodology we can conclude that the united forces of the Tesseract Engine and the Proofread Page extension of the MediaWiki software, together with a well established quality control workflow methodology for the team of correctors and an automatic error checker, turned out to be an outstanding strategy to perform fast and successful Arabic OCR and post-correction without investing too much effort developing highly sophisticated software.

## ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ ERC grant agreement n° 647490.

## REFERENCES

[1] Bare'a OCR engine. 2017. Bare'a vs Google Tesseract OCR. https://arabicocr.wordpress.com/2015/07/11/barie-vs-google-tesseract-ocr. (2017). Accessed: 2017-01-16.

[2] Stefan Jänicke, Greta Franzini, Muhammad Faisal Cheema, and Gerik Scheuermann. 2015. On Close and Distant Reading in Digital Humanities: A Survey and Future Challenges. (2015).

[3] Benjamin Kiessling. 2017. kraken 0.9.0. https://pypi.python.org/pypi/kraken. (2017). Accessed: 2017-01-16.

[4] MediaWiki. 2017. Proofread Page extension. https://www.mediawiki.org/wiki/Extension:Proofread_Page. (2017). Accessed: 2017-01-16.

[5] NovoDynamics. 2017. NovoVerus software. https://www.novodynamics.com/novoverus. (2017). Accessed: 2017-01-16.

[6] Maxim Romanov. 2017. OpenArabic Project. https://github.com/OpenArabic/Annotation. (2017). Accessed: 2017-01-16.

[7] Maxim Romanov, Matthew Thomas Miller, Sarah Bowen Savant, , and Benjamin Kiessling. 2016. Important New Developments in Arabographic Optical Character Recognition (OCR). (October 2016).

[8] Karin C Ryding. 2006. *A Reference Grammar of Modern Standard Arabic*. Cambridge University Press, New York. get-book.cfm?BookID=19243

[9] Ray Smith, Daria Antonova, and Dar-Shyang Lee. 2009. In *MOCR '09: Proceedings of the International Workshop on Multilingual OCR*. http://doi.acm.org/10/1145/1577802.1577804

[10] Ray Smith and Zdenko Podobny. 2017. Tesseract Open Source OCR Engine. https://github.com/tesseract-ocr/tesseract. (2017). Accessed: 2017-01-16.

[11] Ubiquitous Knowledge Processing Lab (UKP) at the Technische Universität Darmstadt. 2017. DKPro Core. https://dkpro.github.io/dkpro-core/. (2017). Accessed: 2017-01-16.