UMMU@QALB-2015 Shared Task: Character and Word level SMT pipeline for Automatic Error Correction of Arabic Text

Fethi Bougares[1] **and Houda Bouamor**[2]
[1]**Laboratoire d'Informatique de l'Université du Maine**
[2]**Carnegie Mellon University in Qatar**
fethi.bougares@lium.univ-lemans.fr, hbouamor@qatar.cmu.edu

Abstract

In this paper we present the LIUM (Laboratoire d'Informatique de l'Universit du Maine) and CMU-Q (Carnegie Mellon University in Qatar) joint submission in the Arabic shared task on automatic spelling error correction. Our best system is a sequential combination of two statistical machine translation systems (SMT) trained on top of the MADAMIRA output. The first is a Character-based one, used to produce a first correction at the character level. Characters are then glued to form the input to the second system working at the Word level. This sequential combination achieves an F_1 score of (**69.42**) that is better than the best F_1 score reported on the 2014 test set (**67.91**). The UMMU best submission to the QALB-15 shared task is **ranked first** over 10 submission on the L2 test condition and **second** over 12 submission on the L1 testsset.

1 Introduction

Errors such as incorrect spelling, word choice, or grammar, limit the effectiveness of NLP models: language errors are problematic when provided as input to NLP systems, which are often not robust enough to handle unexpected variations. The difficulty of spelling errors are language-dependent: the more complex the orthography, morphology, or syntax of a language, the more likely it is to have errors in aspects requiring complex human/machine processing. For morphologically rich languages such as Arabic, spelling errors are very frequent, even among native speakers (Shaalan, 2005). This is because Modern Standard Arabic (MSA), the unifying language of formal text, is not the native language of any Arab (Habash et al., 2008).[1] Arabic word morphology is agglutinative: particles and pronouns are written as part of a word (Habash, 2010). This adds an additional challenge to the writer (native or non native) and could be a principal source of spelling mistakes.

In this paper, we describe an approach performing a sequential combination of two statistical machine translation systems for automatic spelling error correction for Arabic. Our system learns models of correction by training on paired examples of errors and their corrections. The training, tuning and test data are provided by the Shared task organizers. Compared to the first edition of this shared task, this year's version proposes two sub-tasks tackling two text genres: (1) news corpus (news articles extracted from Aljazeera); (2) a corpus of sentences written by learners of Arabic as a Second Language. These two corpora are extracted from the QALB corpus (Zaghouani et al., 2014). We tested our system and showed that it performs well on both corpora.

The remainder of this paper is organized as follows. First, we review the main previous efforts for automatic spelling correction, in Section 3. We then give an overview of the various spelling mistakes done while writing an Arabic text, in Section 4. In Section 5, we detail our error correction system. We present in section 6 the results obtained for the different experiments we conducted using the shared task 2015 dev set. Before concluding, we section 7 details the UMMU official results on QALB-15 test set.

[1]MSA is a language that children learn at school and not innately from their parents

Proceedings of the Second Workshop on Arabic Natural Language Processing, pages 166–172,
Beijing, China, July 26-31, 2015.

2 QALB Shared Task Description

The goal of the QALB shared task is developing the of an automatic system for Arabic Error Correction. The QALB-2015 task is the extension of the first QALB shared task (Mohit et al., 2014) that took place last year. The QALB-2014 addressed errors in comments written to Aljazeera articles by native Arabic speakers (Zaghouani et al., 2014). This year's competition includes two tracks, and, in addition to errors produced by native speakers, also includes correction of texts written by learners of Arabic as a foreign language (L2) (Zaghouani et al., 2015). The native track includes Alj-train-2014, Alj-dev-2014, Alj-test-2014 texts from QALB-2014. The L2 track includes L2-train-2015 and L2-dev-2015. This data was released for the development of the systems. The systems were scored on blind test sets Alj-test-2015 and L2-test-2015.

3 Related Work

Automatic error detection and correction include automatic spelling checking, grammar checking and post-editing. Numerous approaches (both supervised and unsupervised) have been explored to improve the fluency of the text and reduce the percentage of out-of-vocabulary words using NLP tools, resources, and heuristics, e.g., morphological analyzers, language models, and edit-distance measure (Kukich, 1992; Oflazer, 1996; Zribi and Ben Ahmed, 2003; Shaalan et al., 2003; Haddad and Yaseen, 2007; Hassan et al., 2008; Habash, 2008; Shaalan et al., 2010). There has been a lot of work on error correction for English (e.g., (Golding and Roth, 1999)).

For Arabic, this issue was studied in various directions and in different research work. In 2003, Shaalan et al. (2003) presented work on the specification and classification of spelling errors in Arabic. Later on, Haddad and Yaseen (2007) presented a hybrid approach using morphological features and rules to fine tune the word recognition and non-word correction method. In order to build an Arabic spelling checker, Attia et al. (2012) developed semi-automatically, a dictionary of 9 million fully inflected Arabic words using a morphological transducer and a large corpus. They then created an error model by analyzing error types and by creating an edit distance ranker. Finally, they analyzed the level of noise in different sources of data and selected the optimal subset to train their system. Alkanhal et al. (2012) presented a stochastic approach for spelling correction of Arabic text. They used a context-based system to automatically correct misspelled words. First of all, a list is generated with possible alternatives for each misspelled word using the Damerau-Levenshtein edit distance, then the right alternative for each misspelled word is selected stochastically using a lattice search, and an n-gram method. Shaalan et al. (2012) trained a Noisy Channel Model on word-based unigrams to detect and correct spelling errors. Dahlmeier and Ng (2012) built specialized decoders for English grammatical error correction.

More recently, (Pasha et al., 2014) created MADAMIRA, a system for morphological analysis and disambiguation of Arabic, this system can be used to improve the accuracy of spelling checking system especially with Hamza spelling correction. A statistical machine translation model to train an error correction system was presented recently by Jeblee et al. (2014). In contrast to their approach, our system combines two level MT models: character level, then a word level.

4 Spelling errors in Arabic

Three types of Arabic word misspellings are defined in the literature: *typographic*, *cognitive* and *phonetic* errors (Haddad and Yaseen, 2007). The typographic errors corresponding to single word error misspelling represent 80% of all misspelling errors in Arabic (Ben Hamadou, 1994). Based on this study, the most common typographic editing errors that can be found in any Arabic text are the following:

***Substitution*:** approximately, 41.5% of errors belong to substitution errors. In the case of (/لغب → لعب/, "he played"), for example, the letter /ع, E/ is mistakenly substituted by /غ, g/, which results in an incorrect word.

***Deletion*:** approximately 23% of single errors are deletion errors. For example in /فخ → فتح/, "he opens", the letter /ت, t/ had been missed leading to an erroneous word.

***Insertion*:** approximately 15% of the errors are insertion errors. For example, (in /المددرسة →

المدرسة/, "the school", the letter /د, d/ is erroneously inserted twice.

***Transposition*:** swapping two letters represents 4% of the errors. i.e., (/تتشرك → تشترك/, "shares"), the letters /ش, $/ and /ت, t/ are swapped.
In addition to misspelling errors, different editing errors can occur in writing. Some can result in :

***Merge*:** words accidentally merged with surrounding words: (الأمم المتحدة→الأمـمالمتحدة, United Nations)

***Split*:** words mistakenly splitted (فقال→ففال, "he said")

***Cognitive and phonetic errors*:** words generally coming from Arabic dialects: (لكن→لاكن, "but"),

5 SMT system for error correction

We formulate the error correction task as a translation problem where the source part is the text to be corrected and the target part is the correct text. Let's assume that we want to correct an erroneous sentence e to a correct sentence c, and $f_i(e, c)$ is such a model which calculates a probability that c is the correction of e. The goal of the system is to find the correction c^* defined as :

$$c^* = \arg\max_c \ p(c|e)$$
$$= \arg\max_c \ p(e|c)p(c)$$

$p(e|c)$ is estimated in a translation model and $p(c)$ is the target-side language model. The $argmax$ is the task of the decoder and it represent the search for the best hypothesis in the space of possible correction c. The translation system is trained using the well known *MOSES* toolkit (Koehn et al., 2007). The system is constructed using data produced for the QALB shared task and described in Table 1 as follows: First, we generate the correct sentences[2] of the QALB training corpus then translation and reordering models are trained. The language model (LM) is trained on the correct side of the QALB data and a selected part of the Arabic Gigaword corpus.

In order to select the most appropriate amount of monolingual data, we employ data selection techniques based on cross-entropy criterion using *Xenc*[3] (Rousseau, 2013). The selected data is determined in such a way that the corresponding LM minimize the perplexity calculated on the development set. Selected part of each monolingual corpus is used to train an interpolated n-gram [4] back-off target LM using SRILM toolkit (Stolcke et al., 2011) with Kneser-Ney smoothing.

In this work, we propose to use two SMT systems trained with different translation unit (words and characters) as described previously. This was motivated by our intuition that each system will target a different pattern of errors and their combination may outperform the single system performance.

6 Experiments and results

We train four models depending on the used training unit and the nature source side (with or without pre-processing). Each system is evaluated independently and best systems are combined.

6.1 Data description

All our models are built using training, development and testing data provided by the shared task organizers and described in Table 1.

	# sentences	# tokens
Alj-train-2014	19,411	1.1M
L2-train-2015	310	46.3k
Alj-dev-2014	1,017	58.9K
Alj-test-2014	968	56.1k
L2-dev-2015	154	26.3k

Table 1: Train, dev and test data distribution

6.2 Baseline: MADAMIRA corrections

MADAMIRA (**?**) is a tool, originally designed for morphological analysis and disambiguation of MSA and dialectal Arabic texts. MADAMIRA employs different features to select, for each word in context, a proper analysis and performs Alif and Ya spelling correction for the phenomena associated with its letters.

The task organizers provided the shared task data preprocessed with MADAMIRA, including all of the features generated by the tool for every word. Similarly to Jeblee et al. (2014), we use the

[2] for that we used a modified version of the *m2scorer* script that could be distributed

[3] https://github.com/rousseau-lium/XenC

[4] we trained a 4-gram LM for word level and 9-gram LM for character level system

corrections proposed by MADAMIRA and apply them to the data. Table 2 gives the detailed scores obtained using MADAMIRA correction. While the candidates obtained may not necessarily be correct, MADAMIRA performs at a very high precision.

	Precision	Recall	F1
Alj-dev-2014	77.47	32.10	45.40
Alj-test-2014	**78.33**	31.27	44.69
L2-dev-2015	46.46	12.97	20.28

Table 2: F1-score on Dev14, Test14 and L2Dev obtained using MADAMIRA correction

6.3 SMT on raw data

We present here the results we obtained using the wod-level and character-level systems trained on raw non processed data. As shown in Table 3 and Table 4, these systems outperform the MADAMIRA baseline.

	Precision	Recall	F1
Alj-dev-2014	72.69	53.00	61.31
Alj-test-2014	73.80	52.69	61.48
L2-dev-2015	53.40	21.54	30.70

Table 3: Word level SMT for spelling error correction.

It is interesting to note that our character-level system performs better than the word-level one, both on the dev (66.12 *vs.* 61.31) and test sets. This could be explained by the fact that character level system takes advantage from its finer granularity.

	Precision	Recall	F1
Alj-dev-2014	73.19	60.29	66.12
Alj-test-2014	74.22	59.84	66.26
L2-dev-2015	57.08	29.34	38.76

Table 4: Character level SMT error correction.

6.4 SMT on MADAMIRA pre-processed data

The results obtained using MADAMIRA correction candidates (see Table 2) makes it a good start point which one can exploit in order to improve our SMT correction systems. For this we used MADAMIRA as pre-processing step of the SMT training data. Indeed, we re-train our systems over the MADAMIRA pre-processed data. Results for the character-level and word-level systems are presented in Table 5 and Table 6.

	Precision	Recall	F1
Alj-dev-2014	73.72	56.08	63.71
Alj-test-2014	74.33	55.84	63.77
L2-dev-2015	56.79	25.97	35.64

Table 5: Word level SMT error correction with MADAMIRA pre-process

	Precision	Recall	F1
Alj-dev-2014	74.15	61.86	67.45
Alj-test-2014	75.02	61.39	67.53
L2-dev-2015	58.55	30.51	40.12

Table 6: Character level SMT error correction with MADAMIRA pre-process.

As we expected, this combination yields better results (F-score of 40.12 on the L2-dev-2015 data set *vs.* 38.76, when using only the character-level system). It is not surprising that the character level system gives better results than the word level one when trained on the MADAMIRA pre-processed data (66.12 *vs.* 63.71 on Alj-dev-2014).

6.5 Sequential combination

Although the character level system outperforms the word level one, we still want to take benefits from the higher modeling level of word based system. For this we propose two combination setups: (i) *top-down sequential combination* and (ii) *bottom-up sequential combination*. [5] Both combination are performed using data pre-processed with MADAMIRA.

6.5.1 Top-down combination

In this setup, we first use the word-based SMT system. Then we re-translate its outputs using the character level system. The results obtained are given in Table 7. This combination yeilds better results than when using the character level system only (See Table 4).

6.5.2 Bottom-up combination

The Bottom-up combination consists in using the word-level system to re-translate the character

[5] We refer to word to be the top level since characters are of a finer granularity

	Precision	Recall	F1
Alj-dev-2014	69.96	63.18	66.40
Alj-test-2014	70.88	62.59	**66.48**
L2-dev-2015	53.61	31.58	**39.74**

Table 7: Top-down sequential combination

level outputs. Results are shown in Table 8. We obtain our best F1-scores with this setup.

	Precision	Recall	F1
Alj-dev-2014	71.63	66.68	**69.07**
Alj-test-2014	72.77	66.36	**69.42**
L2-dev-2015	56.72	34.80	**43.13**

Table 8: Bottom up sequential combination.

7 UMMU@QALB-2015 Results

In this section, we present the official results of our system on the 2015 QALB test set (Rozovskaya et al., 2015). We submitted two outputs UMMU-1 and UMMU-2. The UMMU-1 is the result of our best system on the dev data (see Table 8 for UMMU-1 dev results) and the UMMU-2 is the output of the Character level SMT without combination (see table 6 for UMMU-2 dev results). The official results of UMMU primary and secondary submissions are respectively presented on table 9 and 10. According to the results presented on Table 9 our system is ranked first in the L2 subtask and second in the L1.

	P	R	F1
L1-test-2015	70.28	71.93	**71.10**
L2-test-2015	54.12	33.26	**41.20**

Table 9: The UMMU Official results on the 2015 test set. First column shows the system rank according to the $F1$ score.

	P	R	F1
L1-test-2015	72.69	67.52	**70.01**
L2-test-2015	55.83	29.47	**38.58**

Table 10: The UMMU-2 results on the 2015 testset.

Table 10 gives the results of the UMMU-2 submission. With regards to our UMMU-1 results we note that our character-level system has a higher precision and lower recall in both subtask. These findings show that our word-level system, when applied on the character-level outputs, improves the recall but decrease the precision. Thus, a better combination of our systems may improve the final F_1 score by avoiding the precision drop.

8 Conclusion and Future Work

We described our submission in the Arabic shared task on automatic spelling error correction. Our system is a sequential combination of two statistical machine translation systems (SMT). First a Character-based SMT system is used to perform lower level correction. Characters outputs of this systems are then glued and used as the input to the higher level system working at the Word level. This sequential combination allows to achieve a F_1 score of **71.10** on L1-test-2015 and **41.20** on L2-test-2015, which ranks us **2nd** in the L1 subtask and **1st** in the L2 subtask. We submitted is a three-stage system that benefits from a MADAMIRA pre-processing, a low level character based SMT system and a higher word-level SMT system. We showed the complementarity of the three stages. We also showed that at each step we our F1-score was improved. In future work, we would like to investigate the possibility of adding an additional layer that uses a neural network language model to estimate the probability in a continuous space and gives better generalization to unseen events.

Acknowledgements

This publication was partly made possible by grants NPRP-4-1058-1-168 from the Qatar National Research Fund (a member of the Qatar Foundation).

References

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

Mohammed Attia, Pavel Pecina, Younes Samih, Khaled Shaalan, and Josef van Genabith. 2012. Improved Spelling Error Detection and Correction for Arabic. In *Proceedings of COLING 2012: Posters*, pages 103–112, Mumbai, India.

Abdelmajid Ben Hamadou. 1994. The Phases of Computational Analysis of Arabic Towards Detecting and Correcting Errors. In *Proceedings of the Second Conference for Arabization of Computers, in Arabic*.

Daniel Dahlmeier and Hwee Tou Ng. 2012. A Beam-Search Decoder for Grammatical Error Correction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 568–578, Jeju Island, Korea.

A. R. Golding and D. Roth. 1999. A Winnow Based Approach to Context-Sensitive Spelling Correction. *Machine Learning*, 34(1-3):107–130.

Nizar Habash, Owen Rambow, Mona Diab, and Reem Kanjawi-Faraj. 2008. Guidelines for Annotation of Arabic Dialectness. In *Proceedings of the LREC Workshop on HLT & NLP within the Arabic world*, Marrakech, Morocco.

Nizar Habash. 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.

Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*, volume 3. Morgan & Claypool Publishers.

Bassam Haddad and Mustafa Yaseen. 2007. Detection and Correction of Non-words in Arabic: a Hybrid Approach. *International Journal of Computer Processing of Oriental Languages*, 20(04):237–257.

Ahmed Hassan, Sara Noeman, and Hany Hassan. 2008. Language Independent Text Correction using Finite State Automata. In *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 913–918, Hyderabad, India.

Serena Jeblee, Houda Bouamor, Wajdi Zaghouani, and Kemal Oflazer. 2014. Cmuq@qalb-2014: An smt-based system for automatic arabic error correction. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 137–142, Doha, Qatar, October. Association for Computational Linguistics.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual meeting-association for computational linguistics*, volume 45, page 2.

Karen Kukich. 1992. Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys (CSUR)*, 24(4):377–439.

Behrang Mohit, Alla Rozovskaya, Nizar Habash, Wajdi Zaghouani, and Ossama Obeid. 2014. The First QALB Shared Task on Automatic Text Correction for Arabic. In *Proceedings of EMNLP Workshop on Arabic Natural Language Processing*, Doha, Qatar, October.

Kemal Oflazer. 1996. Error-Tolerant Finite-State Recognition with Applications to Morphological Analysis and Spelling Correction. *Computational Linguistics*, 22(1):73–89.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland.

Anthony Rousseau. 2013. Xenc: an open-source tool for data selection in natural language processing. *Prague Bulletin of Mathematical Linguistics*, 100:73–82.

Alla Rozovskaya, Houda Bouamor, Nizar Habash, Wajdi Zaghouani, Ossama Obeid, and Behrang Mohit. 2015. The Second QALB Shared Task on Automatic Text Correction for Arabic. In *Proceedings of ACL Workshop on Arabic Natural Language Processing*, Beijing, China, July.

Khaled Shaalan, Amin Allam, and Abdallah Gomah. 2003. Towards Automatic Spell Checking for Arabic. In *Proceedings of the 4th Conference on Language Engineering, Egyptian Society of Language Engineering (ELSE)*, Cairo, Egypt.

Khaled Shaalan, Rana Aref, and Aly Fahmy. 2010. An Approach for Analyzing and Correcting Spelling Errors for Non-native Arabic Learners. In *Proceedings of The 7th International Conference on Informatics and Systems, INFOS2010, the special track on Natural Language Processing and Knowledge Mining*, pages 28–30, Cairo, Egypt.

Khaled Shaalan, Mohammed Attia, Pavel Pecina, Younes Samih, and Josef van Genabith. 2012. Arabic Word Generation and Modelling for Spell Checking. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 719–725, Istanbul, Turkey.

Khaled F Shaalan. 2005. Arabic GramCheck: A Grammar Checker for Arabic. *Software: Practice and Experience*, 35(7):643–665.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. SRILM at sixteen: Update and outlook. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop (ASRU)*, Décembre.

Wajdi Zaghouani, Behrang Mohit, Nizar Habash, Ossama Obeid, Nadi Tomeh, Alla Rozovskaya, Noura Farra, Sarah Alkuhlani, and Kemal Oflazer. 2014. Large Scale Arabic Error Annotation: Guidelines and Framework. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland.

Wajdi Zaghouani, Nizar Habash, Houda Bouamor, Alla Rozovskaya, Behrang Mohit, Abeer Heider, and Kemal Oflazer. 2015. Correction annotation for non-native arabic texts: Guidelines and corpus. In *Proceedings of The 9th Linguistic Annotation Workshop*, pages 129–139, Denver, Colorado, USA, June. Association for Computational Linguistics.

Chiraz Zribi and Mohammed Ben Ahmed. 2003. Efficient Automatic Correction of Misspelled Arabic Words Based on Contextual Information. In *Proceedings of the Knowledge-Based Intelligent Information and Engineering Systems Conference*, pages 770–777, Oxford, UK.