QCMUQ@QALB-2015 Shared Task: Combining Character level MT and Error-tolerant Finite-State Recognition for Arabic Spelling Correction

Houda Bouamor[1], **Hassan Sajjad**[2], **Nadir Durrani**[2] **and Kemal Oflazer**[1]
[1]**Carnegie Mellon University in Qatar**
hbouamor@qatar.cmu.edu, ko@cs.cmu.edu
[2]**Qatar Computing Research Institute**
{hsajjad,ndurrani}@qf.org.qa

Abstract

We describe the CMU-Q and QCRI's joint efforts in building a spelling correction system for Arabic in the QALB 2015 Shared Task. Our system is based on a hybrid pipeline that combines rule-based linguistic techniques with statistical methods using language modeling and machine translation, as well as an error-tolerant finite-state automata method. We trained and tested our spelling corrector using the dataset provided by the shared task organizers. Our system outperforms the baseline system and yeilds better correction quality with an F-score of 68.12 on L1-test-2015 testset and 38.90 on the L2-test-2015. This ranks us 2nd in the L2 subtask and 5th in the L1 subtask.

1 Introduction

With the increased usage of computers in the processing of various languages comes the need for correcting errors introduced at different stages. Hence, the topic of text correction has seen a lot of interest in the past several years (Haddad and Yaseen, 2007; Rozovskaya et al., 2013). Numerous approaches have been explored to correct spelling errors in texts using NLP tools and resources (Kukich, 1992; Oflazer, 1996). The spelling correction for Arabic is an understudied problem in comparison to English, although small amount of research has been done previously (Shaalan et al., 2003; Hassan et al., 2008). The reason for this is the complexity of Arabic language and unavailability of language resources. For example, the Arabic spell checker in Microsoft Word gives incorrect suggests for even simple errors. First shared task on automatic Arabic text correction (Mohit et al., 2014) has been established recently. Its goal is to develop and evaluate spelling correction systems for Arabic trained either on naturally occurring errors in text written by humans or machines. Similar to the first version, in this task participants are asked to implement a system that takes as input MSA (Modern Standard Arabic) text with various spelling errors and automatically correct it. In this year's edition, participants are asked to test their systems on two text genres: (i) news corpus (mainly newswire extracted from Aljazeera); (ii) a corpus of sentences written by learners of Arabic as a Second Language (ASL). Texts produced by learners of ASL generally contain a number of spelling errors. The main problem faced by them is using Arabic with vocabulary and grammar rules that are different from their native language.

In this paper, we describe our Arabic spelling correction system. Our system is based on a hybrid pipeline which combines rule-based techniques with statistical methods using language modeling and machine translation, as well as an error-tolerant finite-state automata method. We trained and tested our spelling corrector using the dataset provided by the shared task organizers Arabic (Rozovskaya et al., 2015). Our systems outperform the baseline and achieve better correction quality with an F-score of 68.42% on the 2014 testset and 44.02 % on the L2 Dev.

2 Data Resources

QALB: We trained and evaluated our system using the data provided for the shared task and the m2Scorer (Dahlmeier and Ng, 2012). These datasets are extracted from the QALB corpus of human-edited Arabic text produced by native speakers, non-native speakers and machines (Zaghouani et al., 2014). The corpus contains a large

	Train 14	%	Dev 14	%	L2 Train	%	L2 Dev	%
Word Count	925,643	-	48,471	-	51,483	-	29,475	-
Total Errors	306,757	33.14	16,659	34.37	13,206	25.65	7,293	24.74
Word errors	187,040	60.97	9,878	59.30	9,417	71.30	5,193	71.20
Punctuation Errors	618,886	39.03	6,781	40.70	3,789	28.70	2,100	28.79
Error per type								
Split	10,869	3.48	612	3.67	255	1.93	110	1.51
Add_before	99,258	32.36	5,704	34.24	3,721	28.17	2,067	28.34
Delete	6,778	2.21	338	2.03	576	4.36	324	4.44
Edit	169,769	55.34	8,914	53.51	8,009	60.64	4,434	60.79
Merge	18,267	5.95	994	5.97	662	5.01	380	5.21
Add_after	20	0.01	2	0.01	1	-	-	-
Move	427	0.14	13	0.08	132	0.9	102	1.39

Table 1: Statistics on Error Types in the QALB 2014 and 2015 datasets

dataset of manually corrected Arabic sentences. QALB covers a variety of errors, and is not just limited to typical spelling errors. For instance, train and dev-2014 data and up to 28% on the 2015 data provided in this Shared Task (See Table[1] 1).

Arabic Wordlist for Spellchecking: We used a list of 9-million Arabic words (Attia et al., 2012). The words are automatically generated from the AraComLex open-source finite state transducer. The entire list is validated against Microsoft Word spell checker.[2]

Monolingual Arabic corpus: Additionally, we used the GigaWord Arabic corpus and the News commentary corpus as used in state-of-the-art English-to-Arabic machine translation system (Sajjad et al., 2013b) to build different language models (character-level and word-level LMs). The complete corpus consists of 32 million sentences and approximately 1,700 million tokens. Due to computational limitations, we were able to train our language model only on 60% of the data which we randomly selected from the whole corpus.

3 Our Approach

Our automatic spelling corrector consists of a hybrid pipeline that combines five different and complementary approaches: (i) a morphology-based corrector; (ii) a rule-based corrector; (ii) an SMT(statistical machine translation)-based corrector; and (d) an error-tolerant finite-state automata approach.

Our system design is motivated by the diversity of the errors contained in our train and dev datasets (See Table 1). It was very challenging to design one system to handle all of the errors. We propose several expert systems each tacking a different kind of spelling errors. For example, we built a character-level machine translation system to handle cases of space insertion and deletion affecting non-clitics, as this part is specifically treated by the rule-based module. To cover some remaining character-level spelling mistakes, we use a Finite-State-Automata (FSA) approach. All our systems run on top of each other, gradually correcting the Arabic text in steps.

3.1 MADAMIRA Corrections (Morph)

MADAMIRA (Pasha et al., 2014) 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.

Similar to Jeblee et al. (2014), we used the corrections proposed by MADAMIRA and apply them to the data. We show in Section 4 that while the correction candidate proposed by MADAMIRA may not be necessarily correct, it performs at a very high precision.

[1]Part of the statistics reported in Table 1 is taken from Diab et al. (2014)

[2]The list is freely available at: `http://sourceforge.net/projects/arabic-wordlist/`

Original	**Source**	ألذي شاهدته في أليوتوب هو ان ...‘
	Target	الذي شاهدته في اليوتوب هو أن ...
	English	*which I have seen in Youtube is that*
Characters	**Source**	أ ل ذ ي # ش ا ه د ت ه #ف ي #أ ل ي و ت و ب #ه و #ا ن # ...
	Target	ا ل ذ ي #ش ا ه د ت ه #ف ي #ا ل ي و ت و ب #ه و #أ ن # ...

Table 2: Preparing the training and tuning and test corpus for alignment

3.2 Rule-based Corrector (Rules)

The MADAMIRA corrector described above does not handle splits and merges; In addition to that, we use the rule-based corrector described in (Rozovskaya et al., 2014). The rules were created through analysis of samples of the 2014 training data. We also apply a set of rules to reattach clitics that may have been split apart from the base word. After examining the train dataset, we realized that 95% of word merging cases involve "و/w/'and'" attachment. Furthermore, we removed duplications and elongations by merging a sequence of two or more of the same character into a single instance.

3.3 Statistical Machine Translation Models

An SMT system translate sentence from one language into another. An alignment step learns mapping from source into target. A phrase-based model is subsequently learned from the word-alignments. The phrase-based model along with other decoding features, such as language and reordering models[3] are used to decode the test sentences. We will use the SMT framework for spell checker where error sentences act as our source and corrections act as a target in the training data.

Phrase-based error correction system (PBMT): The available training data from the shared task consists of parallel sentences. We build a phrase-based machine translation using it. Since the system learns at phrase-level, we hope to identify and correct different errors, especially the ones that were not captured by MADAMIRA.

Character-based error correction system (CBMT): There has been a lot of work in using character-based models for Arabic transliteration to English (Durrani et al., 2014c) and for conversion of Arabic dialects into MSA and vice verse (Sajjad et al., 2013a; Durrani et al., 2014a). The conversion of Arabic dialects to MSA at character-level can be seen as a spelling correction task where small character-level changes are made to convert a dialectal word into an MSA word. We also formulate our correction problem as a character-level machine translation problem, where the pre-processed incorrect Arabic text is considered as the source, and our target is the correct Arabic text provided by the Shared task organizers.

The goal is to learn correspondences between errors and their corrections. All the train data is used to train our the phrase-based model. We treat sentences as sequences of characters instead, as shown in Table 2. Our intuition behind using such model is that it may capture and correct: (i) split errors, occurring due to the deletion of a space between two words, and (ii) merge errors occurring due to the insertion of a space between two words by mistake; (iii) common spelling mistakes (hamzas, yas, etc).

We used the Moses toolkit (Koehn et al., 2007) to create a word and character levels model built on the best pre-processed data (mainly the feat14 tokens extracted using MADAMIRA described in 3.1). We use the standard setting of MGIZA (Gao and Vogel, 2008) and the grow-diagonal-final as the symmetrization heuristic (Och and Ney, 2003) of MOSES to get the character to character alignments. We build a 5-gram word and character language models using KenLM (Heafield, 2011).

3.4 Error-tolerant FST (EFST)

We adapted the error-tolerant recognition approach developed by Oflazer (1996). It was originally designed for the analysis of the agglutinative morphology of Turkish words and used for dictionary-based spelling corrector module. This error-tolerant finite-state recognizer identifies the strings that deviate mildly from a regular set of

[3]See (Durrani et al., 2014b) for more on state-of-the-art PBSMT and features used within.

	Alj-test-2014			**L2-dev-2015**		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
	Single Systems					
Morph	**78.33**	31.27	44.69	46.46	12.97	20.28
Rules	56.92	8.51	14.81	55.84	3.02	5.72
PBMT	73.29	50.18	59.58	53.20	21.10	30.34
CBMT	71.96	57.74	64.07	57.60	29.57	39.07
EFST	38.05	26.94	38.05	47.24	8.21	13.99
	System Combinations					
Morph+PBMT	72.94	55.14	62.80	56.55	24.57	34.26
Morph+CBMT	71.22	60.18	65.24	58.12	30.46	39.98
Morph+EFST	72.19	35.05	47.19	42.49	14.24	21.34
Morph+CBMT+Rules	70.45	65.55	67.91	58.21	34.35	43.20
Morph+CBMT+Rules+EFST	70.14	**66.79**	**68.42**	58.73	35.20	44.02

Table 3: System results on the QALB 2014 test set (left) and L2 dev set (right).

strings recognized by the underlying FSA. For example, suppose we have a recognizer for a regular set over a, b described by the regular expression (aba + bab)*, and we want to recognize the inputs that are slightly corrupted, for example, abaaaba may be matched to abaaba (correcting for a spurious a), or babbb may be matched to babbab (correcting for a deletion), or ababba may be matched to either abaaba (correcting a b to an a) or to ababab (correcting the reversal of the last two symbols). This method is perfect for handling mainly transposition errors resulting from swapping two letters , or typing errors of neighboring letters in the keyboard.

We use the Foma library (Hulden, 2009) to build the finite-state tranducer using the Arabic Word-list as a dictionary.[4] For each word, our system checks if the word is analyzed and recognized by the finite-state transducer. It then generates a list of correction candidates for the non-recognized ones. The candidates are words having an edit distance lower than a certain threshold. We score the different candidates using a LM and consider the best one as the possible correction for each word.

4 Evaluation and Results

We experimented with different configurations to reach an optimal setting when combining different modules. We evaluated our system for precision, recall, and F measure (F1) against the devset reference and the test 2014 set. Results for various system configurations on the L2 dev and test 2014 sets are given in Table 3. The results clearly show different modules are complementry. For instance, combining Morph and PBMT improves the results by +3.22 compared to only using the PBMT model, on last year's test set.

We achieved our best F-measure value with the following configuration: using CBMT system after applying the clitic re-attachment rules. These were then passed through the EFST. Using this combination we are able to correct 66.79% of the errors on the 2014 test set with a precision of 70.14%. Our system outperforms the baseline for the L2 data as well with an F-measure of 44.02% compared to (F1=20.28% when we use the Morph module).

[4]Foma is an open-source finite-state toolkit that implements the Xerox lexc and xfst utilities.

5 QCMUQ@QALB-2015 Results

We present here the official results of our system (Morph+CBMT+Rules+EFST) on the 2015 QALB test set (Rozovskaya et al., 2015). The official results of our QCMUQ are presented in Table 4. These results rank us 2nd in the L2 subtask and 5th in the L1 subtask.

	P	**R**	**F1**
L1-test-2015	71.39	65.13	**68.12**
L2-test-2015	50.37	31.68	**38.90**

Table 4: The QCMUQ Official results on the 2015 test set.

6 Conclusion and Future work

We described our system for automatic Arabic text correction. Our system combines rule-based methods with statistical techniques based on SMT framework and LM-based scoring. We additionally used finite-state-automata to do corrections. Our best system outperforms the baseline with an F-score of 68.12 on L1-test-2015 testset and 38.90 on the L2-test-2015. In the future, we want to focus on correcting punctuation errors, to produce a more accurate system. We plan to experiment with different combination methods similar to the ones used for combining MT outputs.

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

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.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better Evaluation for Grammatical Error Correction. In *NAACL HLT '12 Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572.

Mona Diab, Mohammed Attia, and Al-Badrashiny Mohamed. 2014. GWU-HASP: Hybrid Arabic Spelling and Punctuation Corrector. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, Doha, Qatar.

Nadir Durrani, Yaser Al-Onaizan, and Abraham Ittycheriah. 2014a. Improving Egyptian-to-English SMT by mapping Egyptian into MSA. In *Computational Linguistics and Intelligent Text Processing*, pages 271–282, Khatmandu, Nepal. Springer Berlin Heidelberg.

Nadir Durrani, Barry Haddow, Philipp Koehn, and Kenneth Heafield. 2014b. Edinburgh's phrase-based machine translation systems for WMT-14. In *Proceedings of the ACL 2014 Ninth Workshop on Statistical Machine Translation*, WMT '14, pages 97–104, Baltimore, MD, USA.

Nadir Durrani, Hassan Sajjad, Hieu Hoang, and Philipp Koehn. 2014c. Integrating an unsupervised transliteration model into statistical machine translation. In *Proceedings of the 15th Conference of the European Chapter of the ACL*, EACL '14, Gothenburg, Sweden.

Qin Gao and Stephan Vogel. 2008. Parallel Implementations of Word Alignment Tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57. Association for Computational Linguistics.

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.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Workshop on Statistical Machine Translation (WMT'11)*, Edinburgh, UK.

Mans Hulden. 2009. Foma: A finite-state compiler and library. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, EACL '09, pages 29–32, Stroudsburg, PA, USA. Association for Computational Linguistics.

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.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Christopher Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.

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.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. In *Computational Linguistics*, page 1951.

Kemal Oflazer. 1996. Error-tolerant Finite-state Recognition with Applications to Morphological Analysis and Spelling Correction. *Comput. Linguist.*, 22(1):73–89, March.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan M Roth. 2014. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Language Resources and Evaluation Conference*, LREC '14, pages 1094–1101, Reykjavik, Iceland.

Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, and Dan Roth. 2013. The University of Illinois System in the CoNLL-2013 Shared Task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 13–19, Sofia, Bulgaria, August. Association for Computational Linguistics.

Alla Rozovskaya, Nizar Habash, Ramy Eskander, Noura Farra, and Wael Salloum. 2014. The columbia system in the qalb-2014 shared task on arabic error correction. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 160–164, Doha, Qatar, October. Association for Computational Linguistics.

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.

Hassan Sajjad, Kareem Darwish, and Yonatan Belinkov. 2013a. Translating dialectal Arabic to English. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, ACL '13, pages 1–6, Sofia, Bulgaria.

Hassan Sajjad, Francisco Guzmn, Preslav Nakov, Ahmed Abdelali, Kenton Murray, Fahad Al Obaidli, and Stephan Vogel. 2013b. QCRI at IWSLT 2013: Experiments in Arabic-English and English-Arabic spoken language translation. In *Proceedings of the 10th International Workshop on Spoken Language Technology*, IWSLT '13, Heidelberg, Germany.

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.

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.