A Pipeline Approach to Supervised Error Correction for the QALB-2014 Shared Task

Nadi Tomeh[†] **Nizar Habash**[‡] **Ramy Eskander**[*] **Joseph Le Roux**[†]
{nadi.tomeh,leroux}@lipn.univ-paris13.fr[†]
nizar.habash@nyu.edu[‡], ramy@ccls.columbia.edu[*]
[†]Université Paris 13, Sorbonne Paris Cité, LIPN, Villetaneuse, France
[‡]Computer Science Department, New York University Abu Dhabi
[*]Center for Computational Learning Systems, Columbia University

Abstract

This paper describes our submission to the ANLP-2014 shared task on automatic Arabic error correction. We present a pipeline approach integrating an error detection model, a combination of character- and word-level translation models, a reranking model and a punctuation insertion model. We achieve an F_1 score of 62.8% on the development set of the QALB corpus, and 58.6% on the official test set.

1 Introduction

Devising algorithms for automatic error correction generated considerable interest in the community since the early 1960s (Kukich, 1992) for at least two reasons. First, typical NLP tools lack in robustness against errors in their input. This sensitivity jeopardizes their usefulness especially for unedited text, which is prevalent on the web. Second, automated spell and grammar checkers facilitate text editing and can be of great help to non-native speakers of a language. Several resources and shared tasks appeared recently, including the HOO task (Dale and Kilgarriff, 2010) and the CoNLL task on grammatical error correction (Ng et al., 2013b). In this paper we describe our participation to the first shared task on automatic error correction for Arabic (Mohit et al., 2014).

While non-word errors are relatively easy to handle, the task is more challenging for grammatical and semantic errors. Detecting and correcting such errors require context-sensitive approaches in order to capture the dependencies between the words of a text at various lexical and semantic levels. All the more so for Arabic which brings dependence down to the morphological level (Habash, 2010).

A particularity interesting approach to error correction relies on statistical machine translation (SMT) (Brockett et al., 2006), due to its context-sensitivity and data-driven aspect. Therefore, the pipeline system which we describe in Section 2 has as its core a phrase-based SMT component (PBSMT) (Section 2.3). Nevertheless, several factors may hinder the success of this approach, such as data sparsity, discrepancies between translation and error correction tasks, and the difficulty of incorporating context-sensitive features into the SMT decoder.

We address all these issues in our system which achieves a better correction quality than a simple word-level PBSMT baseline on the QALB corpus (Zaghouani et al., 2014) as we show in our experiments in Section 3.

2 Pipeline Approach to Error Correction

The PBSMT system accounts for context by learning, from a parallel corpus of annotated errors, mappings from erroneous multi-word segments of text to their corrections, and using a language model to help select the suitable corrections in context when multiple alternatives are present. Furthermore, since the SMT approach is data-driven, it is possible to address multiple types of errors at once, as long as examples of them appear in the training corpus. These errors may include non-word errors, wrong lexical choices and grammatical errors, and can also handle normalization issues (Yvon, 2010).

One major issue is data sparsity, since large amount of labeled training data is necessary to provide reliable statistics of all error types. We address this issue by backing-off the word-level PB-SMT model with a character-level correction component, for which richer statistics can be obtained.

Another issue may stem from the inherent difference in nature between error correction and translation. Unlike translation, the input and output vocabularies in the correction task overlap significantly, and the majority of input words are typically correct and are copied unmodified to the output. The SMT system should handle correct words by selecting their identities from all possible options, which may fail resulting in over-correction. To help the SMT decoder decide, we augment our pipeline with a problem zone detection component, which supplies prior information on which input words need to be corrected.

The final issue concerns the difficulty of incorporating features that require context across phrase boundaries into the SMT decoder. A straightforward alternative is to use such features to rerank the hypotheses in the SMT n-best hypotheses lists.

Since punctuation is particularity noisy in Arabic data, we add a specialized punctuation insertion component to our pipeline, depicted in Figure 1.

2.1 Error Detection

We formalize the error detection problem as a sequence labeling problem (Habash and Roth, 2011). Errors are classified into substitution, insertion and deletion errors. Substitutions involve an incorrect word form that should be replaced by another correct form. Insertions are words that are incorrectly added into the text and should be deleted. Deletions are simply missing words that should be added.

We group all error classes into a simple binary problem tag: a word from the input text is tagged as "PROB" if it is the result of an insertion or a substitution of a word. Deleted words, which cannot be tagged themselves, cause their adjacent words to be marked as PROB instead. In this way, the subsequent components in the pipeline can be alerted to the possibility of a missing word via its surroundings. Any words not marked as PROB are given an "OK" tag.

Gold tags, necessary for training, can be generated by comparing the text to its correction using some sequence alignment technique, for which we use SCLITE (Fiscus, 1998).

For this task, we use Yamcha (Kudo and Matsumoto, 2003) to train an SVM classifier using morphological and lexical features. We employ a quadratic polynomial kernel. The static feature window context size is set to +/- 2 words; the previous two (dynamic) predicted tags are also used as features.

The feature set includes the surface forms and their normalization after "Alef", "Ya" and digit normalization, the POS tags and the lemmas of the words. These morphological features are obtained using MADA 3.0 (Habash et al., 2009).[1] We also use a set of word, POS and lemma 3-gram language models scores as features. These LMs are built using SRILM (Stolcke, 2002).

The error detection component is integrated into the pipeline by concatenating the predicted tags with the words of the input text. The SMT model uses this additional information to learn distinct mappings conditional on the predicted correctness of words.

2.2 Character-level Back-off Correction

Each word that is labeled as error (PROB) in the output of the error detection component is mapped to multiple possible corrections using a weighted finite-state transducer similar to the transducers used in speech recognition (Mohri et al., 2002). The WFST, for which we used OpenFST (Allauzen et al., 2007), operates on the character level, and the character mapping is many-to-many (similar to the phrase-based SMT framework).

The score of each proposed correction is a combination of the scores of character mappings used to build it. The list is filtered using WFST scores and an additional character-level LM score. The result is a list of error-tagged words and their correction suggestions, which constitutes a small on-the-fly phrase table used to back-off primary PB-SMT table.

During training, the mapping dictionary is learned from the training after aligning it at the character level using SCLITE. Mapping weights are computed as their normalized frequencies in the aligned training corpus.

2.3 Word-level PBSMT Correction

We formalize the correction process as a phrase-based statistical machine translation problem (Koehn et al., 2003), at the word-level, and solve

[1]We did not use MADAMIRA (the newest version of MADA) since it was not available when this component was built.

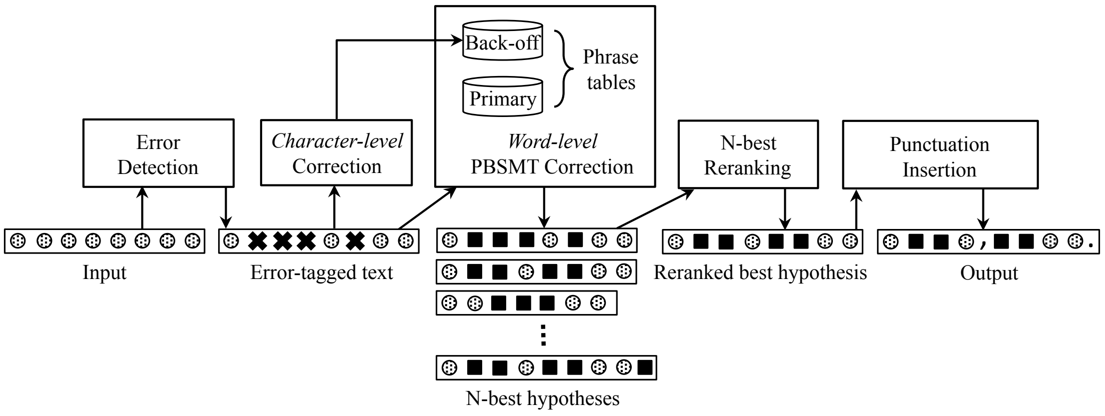

Figure 1: Input text is run through the error detection component which labels the problematic words. The labeled text is then fed to the character-level correction components which constructs a back-off phrase table. The PBSMT component then uses two phrase tables to generate n-best correction hypotheses. The reranking component selects the best hypothesis, and pass it to the punctuation insertion component in order to produce the final output.

it using Moses, a well-known PBSMT tool (Koehn et al., 2007). The decoder constructs a correction hypothesis by first segmenting the input text into phrases, and mapping each phrase into its best correction using a combination of scores including a context-sensitive LM score.

Unlike translation, error correction is mainly monotonic, therefore we set disallow reordering by setting the distortion limit in Moses to 0.[2]

When no mapping can be found for a given phrase in the primary phrase table, the decoder looks it up in the back-off model. The decoder searches the space of all possible correction hypotheses, resulting from alternative segmentations and mappings, and returns the list of n-best scoring hypotheses.

2.4 N-best List Reranking

In this step, we combine LM information with linguistically and semantically motivated features using learning to rank methods (Tomeh et al., 2013). Discriminative reranking (Liu, 2009) allows each hypothesis to be represented as an arbitrary set of features without the need to explicitly model their interactions. Therefore, the system benefits from global and potentially complex features which are not available to the baseline decoder.

Each hypothesis in an n-best list is represented by a d-dimensional feature vector. Word error rate (WER) is computed for each hypotheses by comparing it to the reference correction. The resulting scored n-best list is used for supervised training of a reranking model. We employ a pairwise approach to ranking which takes pairs of hypotheses as instances in learning, and formalizes the ranking problem as pairwise classification.

For this task we use RankSVM (Joachims, 2002) which is a method based on Support Vector Machines (SVMs). We use only linear kernels to keep complexity low. We use a rich set of features including LM scores on surface forms, POS tags and lemmas. We also use a feature based on a global model of the semantic coherence of the hypotheses (Tomeh et al., 2013). The new top ranked hypothesis is the output of this step which is then fed to the next component.

2.5 Punctuation Insertion

We developed a model that predicts the occurrence of periods and commas in a given Arabic text. The core model is a decision tree classifier trained on the QALB parallel training data using WEKA (Hall et al., 2009). For each space between two words, the classifier decides whether or not to insert a punctuation mark, using a window size of three words surrounding the underlying space.

The model uses the following features:

- A class punctuation feature, that is whether to insert a period, a comma or none at the current space location;
- The part-of-speech of the previous word;
- The existence of a conjunctive or connective proclitic in the following word; that is a “wa”

[2]Only 0.14% of edits in the QALB corpus are actually reordering.

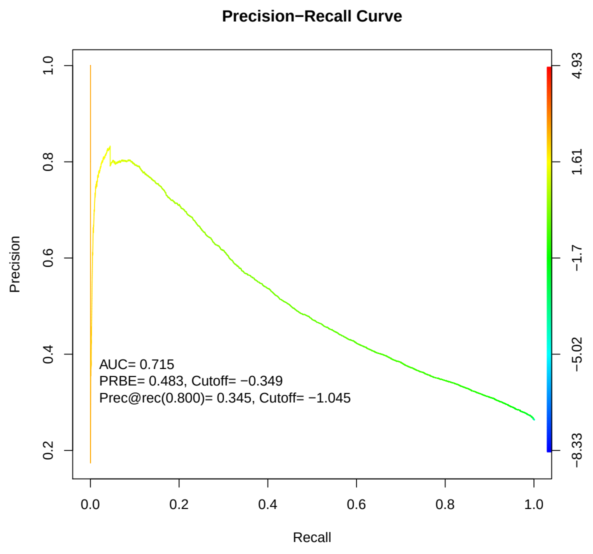

Figure 2: Evaluation of the error detection component. AUC: Area Under the Curve, PRBE: precision-recall break-even point. Classifier thresholds are displayed on the right vertical axis.

or "fa" proclitic that is either a conjunction, a sub-conjunction or a connective particle.

We obtain POS and proclitic information using MADAMIRA (Pasha et al., 2014). The output of this component is the final output of the system.

3 Experiments

All the models we use in our pipeline are trained in a supervised way using the training part of the QALB corpus (Zaghouani et al., 2014), while we reserve the development part of the corpus for testing.

3.1 Error detection

We evaluate the error detection binary classifier in terms of standard classification measures as shown in Figure 2. Each point on the curve is computed by selecting a threshold on the classifier score.

The threshold we use correspond to recall equal to 80%, at which the precision is very low which leaves much room for improvement in the performance of the error detection component.

3.2 Character-level correction

We evaluate the character-level correction model by measuring the percentage of erroneous phrases that have been mapped to their in-context reference corrections. We found this percentage to be 41% on QALB dev data. We limit the size of such phrases to one in order to focus on out-of-vocabulary words.

3.3 Punctuation insertion

To evaluate the punctuation insertion independently from the pipeline, we first remove the periods and commas from input text. Considering only the locations where periods and commas exist, our model gives a recall of 49% and a precision of 53%, giving an F_1-score of 51%.

When we apply our punctuation model in the correction pipeline, we find that it is always better to keep the already existing periods and commas in the input text instead of overwriting them by the model prediction.

While developing the model, we ran experiments where we train the complete list of features produced by MADAMIRA; that is part-of-speech, gender, number, person, aspect, voice, case, mood, state, proclitics and enclitics. This was done for two preceding words and two following words. However, the results were significantly outperformed by our final set-up.

3.4 The pipeline

The performance of the pipeline is evaluated in terms of precision, recall and F_1 as computed by the M^2 Scorer (Dahlmeier and Ng, 2012b). The results presented in Table 1 show that a simple PBSMT baseline achieves relatively good performance compared to more sophisticated models. The character-level back-off model helps by improving recall at the expense of decreased precision. The error detection component hurts the performance which could be explained by its intrinsic bad performance. Since more investigation is needed to clarify on this point, we drop this component from our submission. Both reranking and punctuation insertion improve the performance.

Our system submission to the shared task (back-off+PBSMT+Rank+PI) resulted in an F_1 score of 58.6% on the official test set, with a precision of 76.9% and a recall of 47.3%.

4 Related Work

Both rule-based and data-driven approaches to error correction can be found in the literature (Sidorov et al., 2013; Berend et al., 2013; Yi et al., 2013) as well as hybridization of them (Putra and Szabo, 2013). Unlike our approach, most of

System	PR	RC	F_1
PBSMT	75.5	49.5	59.8
backoff+PBSMT	74.1	51.8	60.9
ED+backoff+PBSMT	61.3	45.4	52.2
backoff+PBSMT+Rank	**75.7**	52.1	61.7
backoff+PBSMT+Rank+PI	74.9	**54.2**	**62.8**

Table 1: Pipeline precision, recall and F_1 scores. ED: error detection, PI: punctuation insertion.

the proposed systems build distinct models to address individual types of errors (see the CoNLL-2013, 2014 proceedings (Ng et al., 2013a; Ng et al., 2014), and combine them afterwords using Integer Linear Programming for instance (Rozovskaya et al., 2013). This approach is relatively time-consuming when the number of error types increases.

Interest in models that target all errors at once has increased, using either multi-class classifiers (Farra et al., 2014; Jia et al., 2013), of-the-shelf SMT techniques (Brockett et al., 2006; Mizumoto et al., 2011; Yuan and Felice, 2013; Buys and van der Merwe, 2013; Buys and van der Merwe, 2013), or building specialized decoders (Dahlmeier and Ng, 2012a).

Our system addresses the weaknesses of the SMT approach using additional components in a pipeline architecture. Similar work on word-level and character-level model combination has been done in the context of translation between closely related languages (Nakov and Tiedemann, 2012). A character-level correction model has also been considered to reduce the out-of-vocabulary rate in translation systems (Habash, 2008).

5 Conclusion and Future Work

We described a pipeline approach based on phrase-based SMT with n-best list reranking. We showed that backing-off word-level model with a character-level model improves the performance by ameliorating the recall of the system.

The main focus of our future work will be on better integration of the error detection model, and on exploring alternative methods for combining the character and the word models.

Acknowledgments

This material is partially based on research funded by grant NPRP-4-1058-1-168 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

Nizar Habash performed most of his contribution to this paper while he was at the Center for Computational Learning Systems at Columbia University.

References

Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. Openfst: A general and efficient weighted finite-state transducer library. In *CIAA*, pages 11–23.

Gabor Berend, Veronika Vincze, Sina Zarrieß, and Richárd Farkas. 2013. Lfg-based features for noun number and article grammatical errors. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 62–67, Sofia, Bulgaria, August. Association for Computational Linguistics.

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting esl errors using phrasal smt techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 249–256, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jan Buys and Brink van der Merwe. 2013. A tree transducer model for grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 43–51, Sofia, Bulgaria, August. Association for Computational Linguistics.

Daniel Dahlmeier and Hwee Tou Ng. 2012a. A beam-search decoder for grammatical error correction. In *EMNLP-CoNLL*, pages 568–578.

Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *HLT-NAACL*, pages 568–572.

Robert Dale and Adam Kilgarriff. 2010. Helping our own: Text massaging for computational linguistics as a new shared task. In *INLG*.

Noura Farra, Nadi Tomeh, Alla Rozovskaya, and Nizar Habash. 2014. Generalized character-level spelling error correction. In *ACL (2)*, pages 161–167.

Jon Fiscus. 1998. Speech Recognition Scoring Toolkit (SCTK). National Institute of Standard Technology (NIST). `http://www.itl.nist.gov/iad/mig/tools/`.

Nizar Habash and Ryan M. Roth. 2011. Using deep morphology to improve automatic error detection in arabic handwriting recognition. In *Proceedings of* *the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 875–884, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. MADA+TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*. The MEDAR Consortium, April.

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*. Morgan & Claypool Publishers.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November.

Zhongye Jia, Peilu Wang, and Hai Zhao. 2013. Grammatical error correction as multiclass classification with single model. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 74–81, Sofia, Bulgaria, August. Association for Computational Linguistics.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-based Translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL)*, pages 127–133, Edmonton, Canada.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Comput. Surv.*, 24(4):377–439, December.

Tie-Yan Liu. 2009. *Learning to Rank for Information Retrieval*. Now Publishers Inc., Hanover, MA, USA.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *IJCNLP*, pages 147–155.

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.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *ACL (2)*, pages 301–305.

Hwee Tou Ng, Joel Tetreault, Siew Mei Wu, Yuanbin Wu, and Christian Hadiwinoto, editors. 2013a. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, Sofia, Bulgaria, August.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013b. The conll-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria, August. Association for Computational Linguistics.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant, editors. 2014. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, Baltimore, Maryland, June.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T. 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 *LREC*, pages 1094–1101.

Desmond Darma Putra and Lili Szabo. 2013. Uds at conll 2013 shared task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 88–95, Sofia, Bulgaria, August. Association for Computational Linguistics.

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.

Grigori Sidorov, Anubhav Gupta, Martin Tozer, Dolors Catala, Angels Catena, and Sandrine Fuentes. 2013. Rule-based system for automatic grammar correction using syntactic n-grams for english language learning (l2). In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 96–101, Sofia, Bulgaria, August. Association for Computational Linguistics.

Andreas Stolcke. 2002. SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, volume 2, pages 901–904, Denver, CO.

Nadi Tomeh, Nizar Habash, Ryan Roth, Noura Farra, Pradeep Dasigi, and Mona Diab. 2013. Reranking with linguistic and semantic features for arabic optical character recognition. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 549–555, Sofia, Bulgaria, August. Association for Computational Linguistics.

Bong-Jun Yi, Ho-Chang Lee, and Hae-Chang Rim. 2013. Kunlp grammatical error correction system for conll-2013 shared task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 123–127, Sofia, Bulgaria, August. Association for Computational Linguistics.

Zheng Yuan and Mariano Felice. 2013. Constrained grammatical error correction using statistical machine translation. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 52–61, Sofia, Bulgaria, August. Association for Computational Linguistics.

François Yvon. 2010. Rewriting the orthography of sms messages. *Natural Language Engineering*, 16:133–159, 3.

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, May. European Language Resources Association (ELRA).