Unsupervised Context-Sensitive Spelling Correction of Clinical Free-Text with Word and Character N-Gram Embeddings

Pieter Fivez[*] **Simon Šuster**[*†] **Walter Daelemans**[*]

[*]CLiPS, University of Antwerp, Antwerp, Belgium
[†]Antwerp University Hospital, Edegem, Belgium
firstname.lastname@uantwerpen.be

Abstract

We present an unsupervised context-sensitive spelling correction method for clinical free-text that uses word and character n-gram embeddings. Our method generates misspelling replacement candidates and ranks them according to their semantic fit, by calculating a weighted cosine similarity between the vectorized representation of a candidate and the misspelling context. We greatly outperform two baseline off-the-shelf spelling correction tools on a manually annotated MIMIC-III test set, and counter the frequency bias of an optimized noisy channel model, showing that neural embeddings can be successfully exploited to include context-awareness in a spelling correction model. Our source code, including a script to extract the annotated test data, can be found at https://github.com/pieterfivez/bionlp2017.

1 Introduction

The genre of clinical free-text is notoriously noisy. Corpora contain observed spelling error rates which range from 0.1% (Liu et al., 2012) and 0.4% (Lai et al., 2015) to 4% and 7% (Tolentino et al., 2007), and even 10% (Ruch et al., 2003). This high spelling error rate, combined with the variable lexical characteristics of clinical text, can render traditional spell checkers ineffective (Patrick et al., 2010).

Recently, Lai et al. (2015) have achieved nearly 80% correction accuracy on a test set of clinical notes with their noisy channel model. However, their model does not leverage any contextual information, while the context of a misspelling can provide important clues for the spelling correction process, for instance to counter the frequency bias of a context-insensitive corpus frequency-based system. Flor (2012) also pointed out that ignoring contextual clues harms performance where a specific misspelling maps to different corrections in different contexts, e.g. *iron deficiency due to ~~enemia~~ → anemia* vs. *fluid injected with ~~enemia~~ → enema*. A noisy channel model like the one by Lai et al. will choose the same item for both corrections.

Our proposed method exploits contextual clues by using neural embeddings to rank misspelling replacement candidates according to their semantic fit in the misspelling context. Neural embeddings have recently proven useful for a variety of related tasks, such as unsupervised normalization (Sridhar, 2015) and reducing the candidate search space for spelling correction (Pande, 2017).

We hypothesize that, by using neural embeddings, our method can counter the frequency bias of a noisy channel model. We test our system on manually annotated misspellings from the MIMIC-III (Johnson et al., 2016) clinical notes. In this paper, we focus on already detected non-word misspellings, i.e. where the misspellings are not real words, following Lai et al.

2 Approach

2.1 Candidate Generation

We generate replacement candidates in 2 phases. First, we extract all items within a Damerau-Levenshtein edit distance of 2 from a reference lexicon. Secondly, to allow for candidates beyond that edit distance, we also apply the Double Metaphone matching popularized by the open source spell checker Aspell.[1] This algorithm converts lexical forms to an approximate phonetic consonant skeleton, and matches all Dou-

[1] http://aspell.net/metaphone/

Proceedings of the BioNLP 2017 workshop, pages 143–148,
Vancouver, Canada, August 4, 2017.

ble Metaphone representations within a Damerau-Levenshtein edit distance of 1. As reference lexicon, we use a union of the UMLS® SPECIALIST lexicon[2] and the general dictionary from Jazzy[3], a Java open source spell checker.

2.2 Candidate Ranking

Our setup computes the cosine similarity between the vector representation of a candidate and the composed vector representations of the misspelling context, weights this score with other parameters, and uses it as the ranking criterium. This setup is similar to the contextual similarity score by Kilicoglu et al. (2015), which proved unsuccessful in their experiments. However, their experiments were preliminary. They used a limited context window of 2 tokens, could not account for candidates which are not observed in the training data, and did not investigate whether a bigger training corpus leads to vector representations which scale better to the complexity of the task.

We attempt a more thorough examination of the applicability of neural embeddings to the spelling correction task. To tune the parameters of our unsupervised context-sensitive spelling correction model, we generate tuning corpora with self-induced spelling errors for three different scenarios following the procedures described in section 3.2. These three corpora present increasingly difficult scenarios for the spelling correction task. **Setup 1** is generated from the same corpus which is used to train the neural embeddings, and exclusively contains corrections which are present in the vocabulary of these neural embeddings. **Setup 2** is generated from a corpus in a different clinical subdomain, and also exclusively contains in-vector-vocabulary corrections. **Setup 3** presents the most difficult scenario, where we use the same corpus as for Setup 2, but only include corrections which are not present in the embedding vocabulary (OOV). In other words, here our model has to deal with both domain change and data sparsity.

Correcting OOV tokens in Setup 3 is made possible by using a combination of word and character n-gram embeddings. We train these embeddings with the fastText model (Bojanowski et al., 2016), an extension of the popular Word2Vec model (Mikolov et al., 2013), which creates vector representations for character n-grams and sums these with word unigram vectors to create the final word vectors. FastText allows for creating vector representations for misspelling replacement candidates absent from the trained embedding space, by only summing the vectors of the character n-grams.

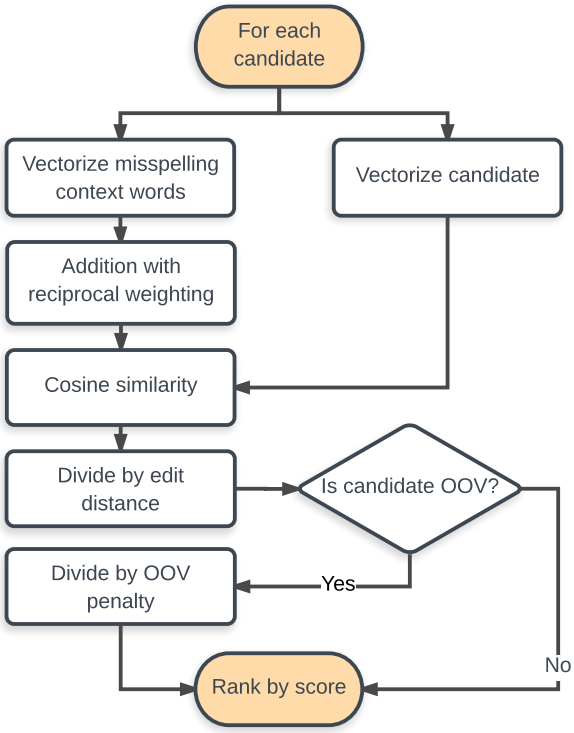

Figure 1: The final architecture of our model. It vectorizes every context word on each side within a specified scope if it is present in the vector vocabulary, applies reciprocal weighting, and sums the representations. It then calculates the cosine similarity with each candidate vector, and divides this score by the Damerau-Levenshtein edit distance between the candidate and misspelling. If the candidate is OOV, the score is divided by an OOV penalty.

We report our tuning experiments with the different setups in 4.1. The final architecture of our model is described in Figure 1. We evaluate this model on our test data in section 4.2.

3 Materials

We tokenize all data with the Pattern tokenizer (De Smedt and Daelemans, 2012). All text is lowercased, and we remove all tokens that include anything different from alphabetic characters or hyphens.

[2] https://lexsrv3.nlm.nih.gov/LexSysGroup/Projects/lexicon/current/web/index.html

[3] http://jazzy.sourceforge.net

3.1 Neural embeddings

We train a fastText skipgram model on 425M words from the MIMIC-III corpus, which contains medical records from critical care units. We use the default parameters, except for the dimensionality, which we raise to 300.

3.2 Tuning corpora

In order to tune our model parameters in an unsupervised way, we automatically create self-induced error corpora. We generate these tuning corpora by randomly sampling lines from a reference corpus, randomly sampling a single word per line if the word is present in our reference lexicon, transforming these words with either 1 (80%) or 2 (20%) random Damerau-Levenshtein operations to a non-word, and then extracting these misspelling instances with a context window of up to 10 tokens on each side, crossing sentence boundaries. For **Setup 1**, we perform this procedure for MIMIC-III, the same corpus which we use to train our neural embeddings, and exclusively sample words present in our vector vocabulary, resulting in 5,000 instances. For **Setup 2**, we perform our procedure for the THYME (Styler IV et al., 2014) corpus, which contains 1,254 clinical notes on the topics of brain and colon cancer. We once again exclusively sample in-vector-vocabulary words, resulting in 5,000 instances. For **Setup 3**, we again perform our procedure for the THYME corpus, but this time we exclusively sample OOV words, resulting in 1,500 instances.

3.3 Test corpus

No benchmark test sets are publicly available for clinical spelling correction. A straightforward annotation task is costly and can lead to small corpora, such as the one by Lai et al., which contains just 78 misspelling instances. Therefore, we adopt a more cost-effective approach. We spot misspellings in MIMIC-III by looking at items with a frequency of 5 or lower which are absent from our lexicon. We then extract and annotate instances of these misspellings along with their context, resulting in 873 contextually different instances of 357 unique error types. We do not control for the different genres covered in the MIMIC-III database (e.g. physician-generated progress notes vs. nursing notes). However, in all cases we make sure to annotate actual spelling mistakes and typos as opposed to abbreviations and shorthand, resulting in instances such as ~~*phebilitis*~~ → *phlebitis* and ~~*sympots*~~ → *symptoms*. We provide a script to extract this test set from MIMIC-III at `https://github.com/pieterfivez/bionlp2017`.

4 Results

For all experiments, we use accuracy as the metric to evaluate the performance of models. Accuracy is simply defined as the percentage of correct misspelling replacements found by a model.

4.1 Parameter tuning

To tune our model, we investigate a variety of parameters:

Vector composition functions

(a) addition
(b) multiplication
(c) max embedding by Wu et al. (2015)

Context metrics

(a) Window size (1 to 10)
(b) Reciprocal weighting
(c) Removing stop words using the English stop word list from scikit-learn (Pedregosa et al., 2011)
(d) Including a vectorized representation of the misspelling

Edit distance penalty

(a) Damerau-Levenshtein
(b) Double Metaphone
(c) Damerau-Levenshtein + Double Metaphone
(d) Spell score by Lai et al.

We perform a grid search for Setup 1 and Setup 2 to discover which parameter combination leads to the highest accuracy averaged over both corpora. In this setting, we only allow for candidates which are present in the vector vocabulary. We then introduce OOV candidates for Setup 1, 2 and 3, and experiment with penalizing them, since their representations are less reliable. As these representations are only composed out of character n-gram vectors, with no word unigram vector, they are susceptible to noise caused by the particular nature of the n-grams. As a result, sometimes the semantic similarity of OOV vectors to other vectors can be inflated in cases of strong orthographic overlap.

Table 1: Correction accuracies for our 3 tuning setups.

	Setup 1	Setup 2	Setup 3
Context	90.24	88.20	57.00
Noisy Channel	85.02	85.86	39.73

Since OOV replacement candidates are more often redundant than necessary, as the majority of correct misspelling replacements will be present in the trained vector space, we try to penalize OOV representations to the extent that they do not cause noise in cases where they are redundant, but still rank first in cases where they are the correct replacement. We tune this OOV penalty by maximizing the accuracy for Setup 3 while minimizing the performance drop for Setup 1 and 2, using a weighted average of their correction accuracies.

All parameters used in our final model architecture are described in Figure 1. The optimal context window size is 9, whereas the optimal OOV penalty is 1.5.

To compare our method against a reference noisy channel model in the most direct and fair way, we implement the ranking component of Lai et al.'s model in our pipeline (**Noisy Channel**), and optimize it with the same MIMIC-III materials that we use to train our embeddings. We perform the optimization by extracting corpus frequencies, which are used to estimate the prior probabilities in the ranking model, from this large data containing 425M words. In comparison, Lai et al.'s own implementation uses corpus frequencies extracted from data containing only 107K words, which is a rather small amount to estimate reliable prior probabilities for a noisy channel model. In the optimized setting, our context-sensitive model (**Context**) outperforms the noisy channel for each corpus in our tuning phase, as shown in Table 1.

4.2 Test

Table 2 shows the correction accuracies for **Context** and **Noisy Channel**, as compared to two baseline off-the-shelf tools. The first tool is HunSpell, a popular open source spell checker used by Google Chrome and Firefox. The second tool is the original implementation of Lai et al.'s model, which they shared with us. The salient difference in performance with our own implementation of their noisy channel model highlights the influence

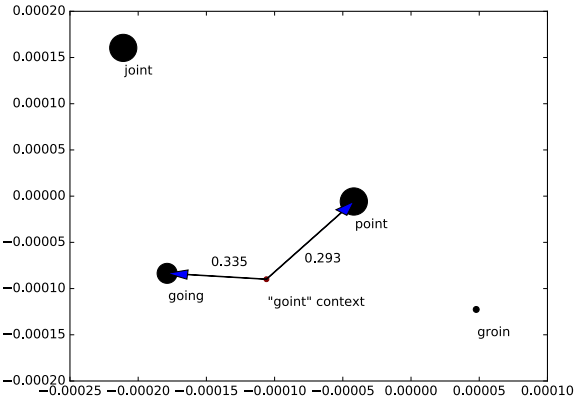

Figure 2: 2-dimensional t-SNE projection of the context of the test misspelling "goint" and 4 replacement candidates in the trained vector space. Dot size denotes corpus frequency, numbers denote cosine similarity. The misspelling context is "new central line lower extremity bypass with sob now [goint] to [be] intubated". While the noisy channel chooses the most frequent "point", our model correctly chooses the most semantically fitting "going".

of training resources and tuning decisions on the general applicability of spelling correction models.

The performance of our model on the test set is slightly held back by the incomplete coverage of our reference lexicon. Missing corrections are mostly highly specialized medical terms, or inflections of more common terminology. Table 2 shows the scenario where these corrections are added to the reference lexicon, leading to a score which is actually higher than those for the tuning corpora.

To analyze whether our context-sensitive model successfully counters the frequency bias of our optimized noisy channel model, we divide the instances of the test set into three scenarios according to the relative frequency of the correct replacement compared to the other replacement candidates. In cases where the correct replacement is the most or second most frequent candidate, the noisy channel scores slightly better. In all other cases, however, our method is more stable. Figure 2 visualizes an example.

Nevertheless, some issues have to be raised. First of all, for the cases with low relative frequency of the correct replacement, the small sample size should be kept in mind: the difference between both models concerns 6 correct instances on

Table 2: The correction accuracies for our test set, evaluated for two different scenarios. *Off-the-shelf*: gives the accuracies of all off-the-shelf tools. *With completed lexicon*: gives the accuracies of our implemented models for the scenario where correct replacements missing from the lexicon are included in the lexicon before the experiment.

Evaluation	HunSpell	Lai et al.	Context	Noisy Channel
OFF-THE-SHELF	52.69	61.97	88.21	87.85
WITH COMPLETED LEXICON			93.02	92.66

a total of 243. While the difference is very pronounced in the much larger tuning corpora, the artificial nature of those corpora does not lead to strong evidence. Moreover, considering the similarity of the general performance of both models on the test set, more test data is needed to make a strong empirical claim about this specific aspect of our model.

While we have optimized the prior probabilities of Lai et al.'s ranking model, the posterior probabilities are still estimated with Lai et al.'s rudimentary spell score, which is a weighted combination of Damerau-Levenshtein and Double Metaphone edit distance. While this error model leads to a noisy channel model which is robust in performance, as shown by our test results, an empirical error model derived from a large confusion matrix can for example help correct the instance described in Figure 2, by capturing that the word-final transformation $t \rightarrow g$ is more probable than the word-initial transformation $g \rightarrow p$. As of now, however, such a resource is not available for the clinical domain.

The errors that our model makes concern, predictably, misspellings for which the contextual clues are too unspecific or misguiding. These cases remain challenging for the concept of our method. While our tuning experiments have explicitly tried to maximize the scope and efficiency of our model, there is still room for improvement, especially for OOV corrections, even as we handle them more effectively than context-insensitive frequency-based methods.

5 Conclusion and future research

In this article, we have proposed an unsupervised context-sensitive model for clinical spelling correction which uses word and character n-gram embeddings. This simple ranking model, which can be tuned to a specific domain by generating self-induced error corpora, counters the frequency bias of a noisy channel model by exploiting contextual clues. As an implemented spelling correction tool, our method greatly outperforms two baseline off-the-shelf spelling correction tools, both a broadly used and a domain-specific one, for empirically observed misspellings in MIMIC-III.

Future research can investigate whether our method transfers well to other genres and domains. Secondly, handling corrections which are not observed in any training data still proves to be a tough task, which might benefit from new conceptual insights. Lastly, it is worthwhile to investigate how our model interacts with the misspelling detection task compared to existing models.

6 Acknowledgements

This research was carried out in the framework of the Accumulate VLAIO SBO project, funded by the government agency Flanders Innovation & Entrepreneurship (VLAIO). We would also like to thank Stéphan Tulkens for his logistic support with coding.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606* .

Tom De Smedt and Walter Daelemans. 2012. Pattern for Python. *Journal of Machine Learning Research* 13:2031–2035.

Michael Flor. 2012. Four types of context for automatic spelling correction. *TAL* 53(3):61–99.

Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* 3.

Halil Kilicoglu, Marcelo Fiszman, Kirk Roberts, and Dina Demner-Fushman. 2015. An ensemble method for spelling correction in consumer health questions.

AMIA Annual Symposium Proceedings pages 727–73.

Kenneth H. Lai, Maxim Topaz, Foster R. Goss, and Li Zhou. 2015. Automated misspelling detection and correction in clinical free-text records. *Journal of Biomedical Informatics* 55:188–195.

Hongfang Liu, Stephen T. Wu, Dingcheng Li, Siddharta Jonnalagadda, Sunghwan Sohn, Kavishwar Wagholikar, Peter J. Haug, Stanley M. Huff, and Christopher G Chute. 2012. Towards a semantic lexicon for clinical natural language processing. *AMIA Annual Symposium Proceedings* .

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Proceedings of Workshop at International Conference on Learning Representations* .

Harshit Pande. 2017. Effective search space reduction for spell correction using character neural embeddings. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* pages 170–174.

J. Patrick, M. Sabbagh, S. Jain, and H. Zheng. 2010. Spelling correction in clinical notes with emphasis on first suggestion accuracy. *2nd Workshop on Building and Evaluating Resources for Biomedical Text Mining* pages 2–8.

Fabrian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and et al. 2011. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research* 12:2825–2830.

Patrick Ruch, Robert Baud, and Antoine Geissbühler. 2003. Using lexical disambiguation and named-entity recognition to improve spelling correction in the electronic patient record. *Artificial Intelligence in Medicine* 29:169–184.

Vivek Kumar Rangarajan Sridhar. 2015. Unsupervised text normalization using distributed representations of words and phrases. *Proceedings of NAACL-HLT 2015* pages 8–16.

William F. Styler IV, Steven Bethard, Sean Finan, Martha Palmer, Sameer Pradhan, Piet C de Groen, Brad Erickson, Timothy Miller, Chen Lin, Guergana Savova, and James Pustejovsky. 2014. Temporal annotation in the clinical domain. *Transactions of the Association for Computational Linguistics* 2:143–154.

Herman D. Tolentino, Michael D. Matters, Wikke Walop, Barbara Law, Wesley Tong, Fang Liu, Paul Fontelo, Katrin Kohl, and Daniel C. Payne. 2007. A UMLS-based spell checker for natural language processing in vaccine safety. *BMC Medical Informatics and Decision Making* 7(3).

Yonghui Wu, Jun Xu, Yaoyun Zhang, and Hua Xu. 2015. Clinical abbreviation disambiguation using neural word embeddings. *Proceedings of the 2015 Workshop on Biomedical Natural Language Processing (BioNLP)* pages 171–176.