Improving Text Normalization via Unsupervised Model and Discriminative Reranking

Chen Li and **Yang Liu**
The University of Texas at Dallas
Computer Science Department
chenli,yangl@hlt.utdallas.edu

Abstract

Various models have been developed for normalizing informal text. In this paper, we propose two methods to improve normalization performance. First is an unsupervised approach that automatically identifies pairs of a non-standard token and proper word from a large unlabeled corpus. We use semantic similarity based on continuous word vector representation, together with other surface similarity measurement. Second we propose a reranking strategy to combine the results from different systems. This allows us to incorporate information that is hard to model in individual systems as well as consider multiple systems to generate a final rank for a test case. Both word- and sentence-level optimization schemes are explored in this study. We evaluate our approach on data sets used in prior studies, and demonstrate that our proposed methods perform better than the state-of-the-art systems.

1 Introduction

There has been a lot of research efforts recently on analysis of social media text (e.g., from Twitter and Facebook) (Ritter et al., 2011; Owoputi et al., 2013; Liu et al., 2012b). One challenge in processing social media text is how to deal with the frequently occurring non-standard words, such as bday (meaning birthday), snd (meaning sound) and gl (meaning girl) . Normalizing informal text (changing non-standard words to standard ones) will ease subsequent language processing modules.

Text normalization has been an important topic for the text-to-speech field. See (Sproat et al., 2001) for a good report of this problem. Recently, much research on normalization has been done for social text domain, which has many abbreviations or non-standard tokens. A simple approach for normalization would be applying traditional spell checking model, which is usually based on edit distance (Damerau, 1964; Levenshtein, 1966). However, this model can not well handle the non-standard words in social media text due to the large variation in generating them.

Another line of work in normalization adopts a noisy channel model. For a non-standard token A, this method finds the most possible standard word $\hat{S}$ based on the Bayes rule: $\hat{S} = argmax P(S|A) = argmax P(A|S) * P(S)$. Different methods have been used to compute $P(A|S)$. Pennell and Liu (2010) used a CRF sequence modeling approach for deletion-based abbreviations. Liu et al. (2011) further extended this work by considering more types of non-standard words without explicit pre-categorization for non-standard tokens.

In addition, the noisy channel model has also been utilized on the sentence level. Choudhury et al. (2007) used a hidden Markov model to simulate SMS message generation, considering the non-standard tokens in the input sentence as emission states in HMM and labeling results as possible candidates. Cook and Stevenson (2009) extended work by adding several more subsystems in this error model according to the most common non-standard token's formation process.

Machine translation (MT) is another commonly chosen method for text normalization. It is also used on both the token and the sentence level. Aw et al. (2006) treated SMS as another language, and used MT methods to translate this 'foreign language' to regular English. Contractor et al. (2010) used an MT model as well but the focus of their work is to utilize an unsupervised method to clean noisy text. Pennell and Liu (2011) firstly introduced an MT method at the token level which translates an unnormalized token to a possible cor-

Proceedings of the ACL 2014 Student Research Workshop, pages 86–93,
Baltimore, Maryland USA, June 22-27 2014.

rect word.

Recently, a new line of work surges relying on the analysis of huge amount of twitter data, often in an unsupervised fashion. By using context information from a large corpus, Han et al. (2012) generated possible variant and normalization pairs, and constructed a dictionary of lexical variants of known words, which are further ranked by string similarity. This dictionary can facilitate lexical normalization via simple string substitution. Hassan and Menezes (2013) proposed an approach based on the random walk algorithm on a contextual similarity bipartite graph, constructed from n-gram sequences on a large unlabeled text corpus. Yang and Eisenstein (2013) presented a unified unsupervised statistical model for text normalization.

2 Previous Normalization Methods Used in Reranking

In this work we adopt several normalization methods developed in previous studies. The following briefly describes these previous approaches. Next section will introduce our proposed methods using unsupervised learning and discriminative reranking for system combination.

2.1 Character-block level MT

Pennell and Liu (2011) proposed to use a character-level MT model for text normalization. The idea is similar to traditional translation, except that the translation unit is characters, not words. Formally, for a non-standard word $A = a_1a_2...a_n$, the MT method finds the most likely standard word $S = s_1s_2...s_m$ (a_i and s_i are the characters in the words): $S = argmaxP(S|A) = argmaxP(A|S)P(S) = argmaxP(a_1a_2...a_n|s_1s_2...s_m)P(s_1s_2...s_m)$ where $P(a_1a_2...a_n|s_1s_2...s_m)$ is from a character-level translation model, and $P(s_1s_2...s_m)$ is from a character-level language model. (Li and Liu, 2012a) modified this approach to perform the translation at the character-block level in order to generate better alignment between characters (analogous to the word vs. phrase based alignment in traditional MT). This system generates one ranked list of word candidates.

2.2 Character-level Two-step MT

Li and Liu (2012b) extended the character-level MT model by incorporating the pronunciation information. They first translate non-standard words to possible pronunciations, which are then translated to standard words in the second step. This method has been shown to yield high coverage (high accuracy in its n-best hypotheses). There are two candidate lists generated by this two-step MT method. The first one is based on the pronunciation list produced in the first step (some phonetic sequences directly correspond to standard words). The second list is generated from the second translation step.

2.3 Character-Block level Sequence Labeling

Pennell and Liu (2010) used sequence labeling model (CRF) for normalizing deletion-based abbreviation at the character-level. The model labels every character in a standard word as 'Y' or 'N' to represent whether it appears or not in a possible abbreviation token. The features used for the classification task represent the character's position, pronunciation and context information. Using the sequence labeling model, a standard word can generate many possible non-standard words. A reverse look-up table is used to store the corresponding possible standard words for the non-standard words for reverse lookup during testing. Liu et al. (2011) extended the above model to handle other types of non-standard words. (Li and Liu, 2012a) used character-blocks (same ones as that in the character-block MT method above) as the units in this sequence labeling framework. There is one list of word candidates from this method.

2.4 Spell Checker

The forth normalization subsystem is the Jazzy Spell Checker[1], which is based on edit distance and integrates a phonetic matching algorithm as well. This provides one list of hypotheses.

3 Proposed Method

All the above models except the Spell Checker are supervised methods that need labeled data consisting of pairs of non-standard words and proper words. In this paper we propose an unsupervised method to create the lookup table of the non-standard words and their corresponding proper words offline. We further propose to use different discriminative reranking approaches to combine multiple individual systems.

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

3.1 Unsupervised Corpus-based Similarity for Normalization

Previous work has shown that unlabeled text can be used to induce unsupervised word clusters that can improve performance of many supervised NLP tasks (Koo et al., 2008; Turian et al., 2010; Täckström et al., 2012). We investigate using a large unlabeled Twitter corpus to automatically identify pairs of non-standard words and their corresponding standard words.

We use the Edinburgh Twitter corpus (Petrovic et al., 2010), and a dictionary obtained from http://ciba.iciba.com/ to identify all the invocabulary and out-of-vocabulary (OOV) words in the corpus. The task is then to automatically find the corresponding OOV words (if any) for each dictionary word, and the likelihood of each pair. The key question is how to compute this likelihood or similarity.

We propose to use an unsupervised method based on the large corpus to induce dense real-valued low-dimension word embedding and then use the inner product as a measure of semantic similarity. We use the continuous bag-of-words model that is similar to the feedforward neural network language model to compute vector representations of words. This model was first introduced by (Mikolov et al., 2013). We use the tool word2vec[2] to implement this model. Two constraints are used in order to eliminate unlikely word pairs: (I) OOV words need to begin with the same letter as the dictionary standard word; (II) OOV words can only consist of English letter and digits.

In addition to considering the above semantic similarity, for the normalization task, we use other information including the surface character level similarity based on longest common sequence between the two tokens, and the frequency of the token. The final score between a dictionary word w and an OOV word t is:

$$\begin{aligned} sim(w,t) &= \frac{longest\_common\_string(w,t)}{length(t)} \\ &* log(TermFreq(t)) \\ &* inner\_product(vec(w), vec(t)) \\ &* \frac{longest\_common\_seq(w,t)}{length(t)} \end{aligned} \quad (1)$$

The first and second term share the same property of visual prime value used in (Liu et al., 2012a). The third term is the vector-based semantic similarity of the two words, calculated by our proposed model. The last term is the length of longest common sequence between the two words divided by the length of the OOV word.

Using this method, we can identify all the possible OOV words for each dictionary word based on an unlabeled large corpus. Each pair has a similarity score. Then a reverse lookup table is created to store the corresponding possible standard words for each non-standard word, which is used during testing. This framework is similar to the sequence labeling method described in Section 2.3 in the sense of creating the mapping table between the OOV and dictionary words. However, the difference is that this is an unsupervised method whereas the sequence labeling uses supervised learning to generate possible candidates.

3.2 Reranking for System Combination

3.2.1 Word Level Reranking

Each of the above systems has its own strength and weakness. The MT model and the sequence labeling models have better precision, the two-step MT model has a broader coverage of candidates, and the spell checker has a high confidence for simple non-standard words. Therefore combining these systems is expected to yield better overall results. We propose to use a supervised maximum entropy reranking model to combine our proposed unsupervised method with those described in Section 2 (4 systems that have 5 candidate lists). The features we used in the normalization reranking model are shown in Table 1. This maxent reranking method has shown success in many previous work such as (Charniak and Johnson, 2005; Ji et al., 2006).

Features:
1.Boolean value to indicate whether a candidate is on the list of each system. There are 6 lists and thus 6 such features.
2.A concatenation of the 6 boolean features above.
3.The position of this candidate in each candidate list. If this candidate is not on a list, the value of this feature is -1 for that list.
4.The unigram language model probability of the candidate.
5.Boolean value to indicate whether the first character of the candidate and non-standard word is the same.
6.Boolean value to indicate whether the last character of the candidate and non-standard word is the same.

Table 1: Features for Reranking.

The first three features are related to the indi-

[2]https://code.google.com/p/word2vec/

vidual systems, and the last three features compare the candidate with the non-standard word. It is computationally expensive to include information represented in the last three features in the individual systems since they need to consider more candidates in the normalization step; whereas in reranking, only a small set of word candidates are evaluated, thus it is more feasible to use such global features in the reranking model. We also tried some other lexical features such as the length difference of the non-standard word and the candidate, whether non-standard word contains numbers, etc. But they did not obtain performance gain. Another advantage of the reranker is that we can use information about multiple systems, such as the first three features.

3.2.2 Sentence Level Reranking and Decoding

In the above reranking method, we only use information about the individual words. When contextual words are available (in sentences or Tweets), we can use that information. If a sentence containing OOV words is given during testing, we can perform standard sentence level Viterbi decoding to combine information from the normalization candidates and language model scores.

Furthermore, if sentences are available during training (not just isolated word pairs as used in all the previous supervised individual systems and the Maxent reranking above), we can also use contextual information for training the reranker. This can be achieved in two different ways. First, we add the Language Model score from context words as features in the reranker. In this work, in addition to the features in Table 1, we add a trigram probability to represent the context information. For every candidate of a non-standard word, we use trigram probability from the language model. The trigram consists of this candidate, and the previous and the following token of the non-standard word. If the previous/following word is also a non-standard token, then we calculate the trigram using all of their candidates and then take the average. After adding the additional LM probability feature, the same Maxent reranking method as above is used, which optimizes the word level accuracy.

The second method is to change the training objective and perform the optimization at the sentence level. The feature set can be the same as the word level reranker, or with the additional contextual LM score features. To train the model (feature weights), we perform sentence level Viterbi decoding on the training set to find the best hypothesis for each non-standard word. If the hypothesis is incorrect, we update the feature weight using structured perceptron strategy (Collins, 2002). We will explore these different feature and training configurations for reranking in the following experiments.

4 Experiments

4.1 Experimental Setup

The following data sets are used in our experiments. We use Data 1 and Data 2 as test data, and Data 3 as training data for all the supervised models.

- Data 1: 558 pairs of non-standard tokens and standard words collected from 549 tweets in 2010 by (Han and Baldwin, 2011).
- Data 2: 3,962 pairs of non-standard tokens and standard words collected from 6,160 tweets between 2009 and 2010 by (Liu et al., 2011).
- Data 3: 2,333 unique pairs of non-standard tokens and standard words, collected from 2,577 Twitter messages (selected from the Edinburgh Twitter corpus) used in (Pennell and Liu, 2011). We made some changes on this data, removing the pairs that have more than one proper words, and sentences that only contain such pairs.[3]
- Data 4: About 10 million twitter messages selected from the the Edinburgh Twitter corpus mentioned above, consisting of 3 million unique tokens. This data is used by the unsupervised method to create the mapping table, and also for building the word-based language model needed in sentence level normalization.

The dictionary we used is obtained from http://ciba.iciba.com/, which includes 75,262 English word entries and their corresponding phonetic symbols (IPA symbols). This is used in various modules in the normalization systems. The number of the final standard words used to create the look-up table is 10,105 because we only use the words that have the same number of character-block segments and phones. These 10,105 words

[3] http://www.hlt.utdallas.edu/∼chenli/normalization

cover 90.77% and 93.74% standard words in Data set 1 and Data set 2 respectively. For the non-standard words created in the CRF model, they cover 80.47% and 86.47% non-standard words in Data set1 and Data set 2. This coverage using the non-standard words identified by the new unsupervised model is 91.99% and 92.32% for the two data sets, higher than that by the CRF model.

During experiments, we use CRF++ toolkit [4] for our sequence labeling model, SRILM toolkit (Stolcke, 2002) to build all the language models, Giza++ (Och and Ney, 2003) for automatic word alignment, and Moses (Koehn et al., 2007) for translation decoding in three MT systems.

4.2 Isolated Word Normalization Experiments

Table 2 shows the isolated word normalization results on the two test data sets for various systems. The performance metrics include the accuracy for the top-1 candidate and other top-N candidates. Coverage means how many test cases correct answers can be obtained in the final list regardless of its positions. The top part presents the results on Data Set 1 and the bottom shows the results on Data Set 2. We can see that our proposed unsupervised corpus similarity model achieves better top-1 accuracy than the other individual systems described in Section 2. Its top-n coverage is not always the best – the 2-step MT method has advantages in its coverage. The results in the table also show that reranking improves system performance over any of the used individual systems, which is expected. After reranking, on Data set 1, our system yields better performance than previously reported ones. On Data set 2, it has better top-1 accuracy than (Liu et al., 2012a), but slightly worse top-N coverage. However, the method in (Liu et al., 2012a) has higher computational cost because of the calculation of the prime visual values for each non-standard word on the fly during testing. In addition, they also used more training data than ours.

System	Accuracy %				
	Top1	Top3	Top10	Top20	Cover
Data 1					
MT	61.81	73.53	78.50	79.57	80.00
MT21	39.61	52.93	63.59	65.36	65.72
MT22	53.64	68.56	77.44	80.46	88.10
SL	53.29	61.99	69.09	71.92	75.85
SC	50.27	56.31	56.84	57.02	57.02
UCS	61.81	69.98	74.60	76.55	82.17
Rerank	**77.14**	**86.96**	**93.04**	**94.82**	**95.90**
Sys1	75.69	n/a	n/a	n/a	n/a
Sys2	73	81.9	86.7	89.2	94.2
Data 2					
MT	55.02	63.3	66.99	67.77	68.00
MT21	35.64	47.65	54.67	56.01	56.4
MT22	49.02	62.49	70.99	74.86	80.07
SL	46.52	55.05	61.21	62.97	66.21
SC	51.16	55.48	55.88	55.88	55.88
UCS	57.29	65.75	70.55	72.64	80.84
Rerank	**74.44**	**84.57**	90.25	92.37	93.5
Sys1	69.81	82.51	**92.24**	**93.79**	**95.71**
Sys2	62.6	75.1	84	87.5	90.7
Sys3	73.04	n/a	n/a	n/a	n/a

Table 2: MT: Character-block Level MT; MT21&MT22: First&Second step in Character-level Two-step MT; SL: Sequence Labeling system; SC: Spell Checker; UCS: Unsupervised Corpus Similarity Model; Sys1 is from (Liu et al., 2012a); Sys2 is from (Li and Liu, 2012a); Sys3 is from (Yang and Eisenstein, 2013).

4.3 Sentence Level Normalization Results

We have already seen that after reranking we obtain better word-level normalization performance, for both top-1 and other top-N candidates. One follow-up question is whether this improved performance carries over to sentence level normalization when context information is used via the incorporation of a language model. Since detecting which tokens need normalization in the first place is a hard task itself in social media text and is an open question currently, similar to some previous work, we assume that we already know the non-standard words that need to be normalized for a given sentence. Then the sentence-level normalization task is just to find which candidate from the n-best lists for each of those already 'detected' non-standard words is the best one. We use the tweets in the Data set 1 described above because Data set 2 only has token pairs but not sentences.

Table 3 shows the sentence level normalization results using different reranking configurations with respect to the features used in the reranker and the training process. Regarding features, reranker 1 and 3 use the features described

[4] http://crfpp.googlecode.com/

in Section 3.2.1, i.e., features based on the words only, without the additional trigram LM probability feature; reranker 2 and 4 use the additional LM probability feature. About training, reranker 1 and 2 use the Maxent reranking that is trained and optimized for the word level; reranker 3 and 4 use structure perceptron training at the sentence level. Note that all of the systems perform Viterbi decoding during testing to determine the final top one candidate for each non-standard word in the sentence. The scores from the reranked normalization output and the LM probabilities are combined in decoding. From the results, we can see that adding contextual information (LM probabilities) as features in the reranker is useful. When this feature is not used, using sentence-level training objective benefits (reranker 3 outperforms 1); however, when this feature is used, performing sentence-level training via structure perceptron is not useful (reranker 2 outperforms 4), partly because the contextual information is incorporated in the features already and using it in sentence-level decoding for training is redundant and does not bring additional gain. Finally compared to the previously report results, our system performs the best.

System	Acc %	System	Acc %
Reranker1	84.30	Reranker2	**86.91**
Reranker3	85.03	Reranker4	85.37
Sys1	84.13	Sys2	82.23

Table 3: Sentence level normalization results on Data Set 1 using different reranking setups. Sys1 is from (Liu et al., 2012a); Sys2 is from (Yang and Eisenstein, 2013). Acc % is the top one accuracy.

4.4 Impact of Unsupervised Corpus Similarity Model

Our last question is regarding unsupervised model importance in the reranking system and contributions of its different similarity measure components. We conduct the following two experiments: First, we removed the new model and just use the other remaining models in reranking (five candidate lists). Second, we kept this new model but changed the corpus similarity measure (removed the third item in Eq(1) that represents the semantic similarity). This way we can evaluate the impact of the semantic similarity measure based on the continuous word vector representation.

Table 4 shows the word level and sentence results on Data set 1 and 2 using these different setups. Because of space limit, we only present the top one accuracy. The other top-n results have similar patterns. Sentence level normalization uses the Reranker 2 described above. We can see that there is a degradation in both of the new setups, suggesting that the unsupervised method itself is beneficial, and in particular the word vector based semantic similarity component is crucial to the system performance.

System	Word Level		Sent Level
	Data1	Data2	Data1
system-A	73.75	70.33	84.51
system-B	74.77	70.83	86.22
system-C	**77.14**	**74.44**	**86.91**

Table 4: Word level and Sentence level normalization results (top-1 accuracy in %) after reranking on Data Set 1 and 2. System-A is without using the unsupervised model, system-B is without its semantic similarity measure, and system-C is our proposed system.

5 Conclusions

In this paper, we proposed a novel normalization system by using unsupervised methods in a large corpus to identify non-standard words and their corresponding proper words. We further combine it with several previously developed normalization systems by a reranking strategy. In addition, we explored different sentence level reranking methods to evaluate the impact of context information. Our experiments show that the reranking system not only significantly improves the word level normalization accuracy, but also helps the sentence level decoding. In the future work, we plan to explore more useful features and also leverage pairwise and link reranking strategy.

Acknowledgments

We thank the NSF for travel and conference support for this paper. The work is also partially supported by DARPA Contract No. FA8750-13-2-0041. Any opinions, findings, and conclusions or recommendations expressed are those of the author and do not necessarily reflect the views of the funding agencies.

References

Aiti Aw, Min Zhang, Juan Xiao, Jian Su, and Jian Su. 2006. A phrase-based statistical model for sms text normalization. In *Processing of COLING/ACL*.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd ACL*.

Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *IJDAR*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*.

Danish Contractor, Tanveer A. Faruquie, L. Venkata Subramaniam, and L. Venkata Subramaniam. 2010. Unsupervised cleansing of noisy text. In *Proceedings of COLING*.

Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text message normalization. In *Proceedings of NAACL*.

Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceeding of 49th ACL*.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 EMNLP*.

Hany Hassan and Arul Menezes. 2013. Social text normalization using contextual graph random walks. In *Proceedings of ACL*.

Heng Ji, Cynthia Rudin, and Ralph Grishman. 2006. Re-ranking algorithms for name tagging. In *Proceedings of the Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL*.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, page 707.

Chen Li and Yang Liu. 2012a. Improving text normalization using character-blocks based models and system combination. In *Proceedings of COLING 2012*.

Chen Li and Yang Liu. 2012b. Normalization of text messages using character- and phone-based machine translation approaches. In *Proceedings of 13th Interspeech*.

Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution?: normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th ACL: short papers*.

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012a. A broad-coverage normalization system for social media language. In *Proceedings of the 50th ACL*.

Xiaohua Liu, Ming Zhou, Xiangyang Zhou, Zhongyang Fu, and Furu Wei. 2012b. Joint inference of named entity recognition and normalization for tweets. In *Proceedings of ACL*.

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

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL*.

Deana Pennell and Yang Liu. 2010. Normalization of text messages for text-to-speech. In *ICASSP*.

Deana Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of sms abbreviations. In *Proceedings of 5th IJCNLP*.

Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2010. The edinburgh twitter corpus. In *Proceedings of NAACL*.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of EMNLP*.

Richard Sproat, Alan W. Black, Stanley F. Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of NAACL*.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*.

Yi Yang and Jacob Eisenstein. 2013. A log-linear model for unsupervised text normalization. In *Proceedings of EMNLP*.