Rule-Based Normalization of German Twitter Messages

Uladzimir Sidarenka, Tatjana Scheffler, and Manfred Stede

University of Potsdam,
Karl-Liebknecht Str. 24-25
14476 Potsdam
{uladzimir.sidarenka,tatjana.scheffler,manfred.stede}@uni-potsdam.de

Abstract. In this article, we conduct quantitative and qualitative analyses of unknown words in German Twitter messages, and propose a normalization method which prepares German tweets for standard text processing tools. In the first part, the prevalence of different types of out-of-vocabulary (OOV) tokens and non-standard language in German Twitter data is determined. In a second step, we present a set of ad-hoc techniques which can tackle some of the most prominent effects found during the analysis. We show how this set of techniques helps us lower the average rate of out-of-vocabulary tokens in Twitter messages and how this lower OOV-rate in turn helps improve the quality of automatic part-of-speech tagging.

Keywords: twitter, social media, text normalization, spelling correction

1 Introduction

When Jack Dorsey, the present CEO of Twitter Inc., was sending the very first tweet on March 21, 2006 (Dorsey, 2006), he probably did not realize that his message – "just setting up my twttr" – already contained a word which was unknown to the majority of NLP applications existing at that time and that social media text analysis would be dominated by many unknown words in the future. Even though the problem of out-of-vocabulary words and textual normalization has been extensively studied in computational linguistics since as early as the late 1950s (cf. Petersen, 1980) and was certainly anything but new at the time when online communication emerged, it were short messages that revived interest in this field in the past two decades.

In the next section, we give an overview of existing approaches to the problem of tackling noise in non-standard texts. In Section 3, we analyze which types of phenomena are characteristic for German Twitter messages. Section 4 subsequently describes an automatic procedure for mitigating some of the most prominent of those effects. Finally, we perform an evaluation of the results of this procedure and give further suggestions for future research.

2 Related Work

At the very onset of the works on noisy text normalization (NTN), the two major sources of noisy data were optical character recognition and automatic speech recognition applications. The relatively low recognition accuracy of those tools at the beginning of the 1990-s forced researchers to think about how words distorted during the automatic processing could be restored to their respective standard language forms. A method which seemed to be most suitable for these purposes at that time was a technique called "noisy channel model" (NCM) first introduced by Shannon (1948).

NCM was also one of the first methods which were applied to normalization of mobile messages and Internet-based communication (IBC) texts. In 2003, Clark proposed a unified NCM system which jointly performed tokenization, sentence splitting, and word normalization of Usenet forum posts. Choudhury et al. (2007) extended the NCM-approach proposed by Toutanova and Moore (2002) by converting it to a Hidden Markov Model and then used this enhanced model for normalization of short text messages (SMS). Beaufort et al. (2010) tried to normalize French SMSes by using an NCM method which was combined with a finite state transducer (FST) technique.

Starting from the early 2000-s, the increasing quality of statistical machine-translation (SMT) applications and gradual realization of the shortcomings of NCM methods which operated solely on the character level of words spurred the researchers on the development of NTN methods which were more similar to the established SMT approaches. In the scope of this framework, the normalization task was defined as mapping an unnormalized word or expression to its normalized equivalent. This equivalent could either be specified manually as was done by Clark and Araki (2011) or derived automatically during the alignment of normalized and unnormalized training data as suggested by Aw et al. (2006). The latter approach however presupposed that a sufficiently large collection of such data was available. One of the first attempts to apply an SMT-like technique to normalization of Twitter messages was made by Kaufmann (2010), who used a corpus of SMS messages as his training set.

Finally, it was noticed by Kobus et al. (2008) that NTN methods relying on either NCM or SMT techniques usually revealed complementary strengths and weaknesses. This notion led to the idea that incorporating these two normalization approaches into one system would improve the overall performance, as different sources of information would benefit from each other. Kobus et al. (2008) proposed a combined method which first used a trained SMT module and then fed its output into a FST whose transitions represented phonetic or graphematic substitutions frequently occurring in unnormalized words.

It should however be noted that almost all of the above methods concentrated on English data only. A few exceptions were the approaches suggested by Beaufort et al. (2010) and Kobus (2008) for French, and Oliva et al. (2013) for Spanish. In order to better understand how relevant the text normalization task would be for German and to get a clearer picture of the difficulties that may arise when processing noisy text data in that language, we decided to have

a closer look at German Twitter messages and to analyze words occurring there which were regarded as unknown by standard NLP tools commonly used for processing German texts.

3 Analysis of Unknown Tokens

In order to estimate the percentage of unknown words in Twitter, we randomly selected 10,000 messages from a corpus of 24,179,871 German tweets gathered in April 2013. We developed a Twitter-aware sentence splitter.[1] For tokenization, we slightly adjusted the specialized Twitter tokenizer[2] developed by Christopher Potts to the peculiarities of German. After skipping all numbers, punctuation marks, and words consisting only of a single letter, we obtained a list of 129,146 tokens. As reference systems for dictionary lookup we used the open-source spell checking program `hunspell`[3] and the publicly available part-of-speech tagger `TreeTagger`[4] (Schmid, 1994).

Out of this token list, 26,018 tokens (20.15 %) were regarded as unknown by `hunspell` and 28,389 tokens (21.98 %) were considered as OOV by `TreeTagger`. We also performed these estimations for word types. The relative rate of unknown items there was as expected higher and ran up to 46.96 % for `hunspell` and 58.24 % for `TreeTagger`, out of a total of 32,538 types.

We classified the OOV tokens into the following three groups according to the reasons why these tokens could have been omitted from applications' dictionaries:

1. **Limitation of machine-readable dictionaries (MRD)**. Among this group, we counted valid words of basic vocabulary which had erroneously been omitted from an applications' MRD;
2. **Stylistic specifics of text genre**. This group comprised words which did not belong to the standard language but were perfectly valid terms in the domain of web discourse or more specifically in Twitter communication;
3. **Spelling deviations**. In the scope of this group, we considered non-standard spellings of words encountered in text.

To see how detected OOV words were distributed among these 3 major groups, we manually analyzed all OOV tokens which appeared in the text more than once and also looked at 1,000 randomly selected hapax legomena. The breakdown of these OOV tokens into the three major classes is shown in Table 1. We have considered OOV-distributions for both `hunspell` and `TreeTagger`. For each of them, we estimated the percentage of a particular class with regard to the total number of occurrences of all OOV tokens (column "% of OOV tokens") as well as with regard to the percentage rate in the list of OOV types (column "% of OOV types").

[1] On a test set of 759 messages, our sentence splitter correctly determined the sentence boundaries in 88.14 % of cases.

[2] `http://sentiment.christopherpotts.net/code-data/happyfuntokenizing.py`

[3] Ispell Version 3.2.06 (Hunspell Version 1.3.2); dictionary de_DE.

[4] Version 3.2 with German parameter file UTF-8.

Table 1. Distribution of OOV words over the three major classes

OOV subclass	`hunspell`		`TreeTagger`	
	% of OOV tokens	% of OOV types	% of OOV tokens	% of OOV types
Limitation of MRD	45.87	54.62	40.46	43.36
Stylistic specifics of genre	41.65	33.64	48.02	44.59
Spelling deviations	11.87	10.75	9.09	8.23

In order to better understand what particular kinds of linguistic phenomena contributed to each of the three major groups, we also performed a more fine-grained analysis and split each of the classes in smaller subcategories.

We subdivided the class 1, "Limitation of MRD", into the following subgroups:

1. regular German words, e.g. *Piraterie*, *losziehen*;
2. compounds, e.g. *Altwein*, *Amtsapothekerin*;
3. abbreviations, e.g. *NBG*, *OL*;
4. interjections, e.g. *aja*, *haha*;
5. named entities, with subclasses:
   (a) persons, e.g. *Ahmadinedschad*, *Schweiger*;
   (b) geographic locations, e.g. *Biel*, *Limmat*;
   (c) companies, e.g. *Apple*, *Facebook*;
   (d) product names, e.g. *iPhone*, *MacBook*;
6. neologisms, with subclasses:
   (a) newly coined German terms, e.g. *entfolgen*, *gegoogelt*;
   (b) loanwords, e.g. *Community*, *Stream*;
7. and, finally, foreign words like *is* or *now* which in contrast to 6b were not mentioned in any existing German lexica and did not comply with inflectional rules of German grammar.

The taxonomy reflects the fact that valid words could have been omitted from an MRD either due to the limitations of developers' capacities (group 1), active word formation processes or lexical productivity of the language itself (groups 2 through 6a) or also due to the language's openness to foreign language systems (groups 6b and 7). Percentage figures for each of the above subgroups are shown in Table 2.

Similarly to class 1, we divided the group "Stylistic specifics of text genre" into the following subclasses:

1. @-tokens, e.g. *@ZDFonline*, *@sechsdreinuller*;
2. hashtags, e.g. *#Kleinanzeigen*, *#wetter*;
3. links, e.g. *http://t.co*, *sueddeutsche.de*;
4. smileys, e.g. *:-P*, *xD*;
5. slang, e.g. *OMG*, *WTF* etc.

Table 2. Distribution of OOV words belonging to the class "Objective limitedness of MRD"

OOV subclass	`hunspell`		`TreeTagger`	
	% of OOV tokens	% of OOV types	% of OOV tokens	% of OOV types
regular German words	7.27	8.66	2.74	3.46
compounds	1.27	2.65	2.5	4.54
abbreviations	3.96	4.8	3.26	3.43
interjections	5.99	4.6	5.56	4.27
person names	4.77	6.49	2.31	3.46
geographic locations	1.53	2.6	1.16	1.87
company names	2.28	2.87	4.34	3
product names	2.16	2.65	2.45	3.22
newly coined terms	1.37	1.35	3.32	2.38
loanwords	3.7	4.06	3.28	2.86
foreign words	11.57	13.89	9.54	10.87
total	45.87	54.62	40.46	43.36

according to the formal or lexical class which tokens of this group belonged to. As slang, we considered colloquial and dialectal lexical expressions (e.g. *nö*, *bissl*), common phrases pertaining to the IBC-genre (e.g. *LOL*, *ava*), as well as spellings of words which resembled their colloquial pronunciation in everyday speech (e.g. *Tach* instead of *Tag*, *nen* instead of *einen*). The latter cases were assigned by us to two categories, namely, *slang* and *spelling deviations*. Detailed statistics on the afore-mentioned subgroups of class 2 are shown in Table 3.

Table 3. Distribution of OOV words belonging to the class "Stylistic specifics of text genre"

OOV subclass	`hunspell`		`TreeTagger`	
	% of OOV tokens	% of OOV types	% of OOV tokens	% of OOV types
@-tokens	13.12	20.49	16.14	21.84
hashtags	7.41	6.26	13.02	10.56
links	2.45	0.4	4.88	6.05
smileys	2.01	0.74	6.86	1.2
slang	16.66	5.75	7.12	4.94
total	41.65	33.64	48.02	44.59

A striking outlier of 16.66 % for slang tokens in column 1 of the table is explained by the fact that the word "RT" which occurred 1,235 times in our texts and was by far the most frequent OOV in the analyzed data set, was recognized as OOV by `hunspell` but was not deemed as such by `TreeTagger`.

The last major class, "Spelling deviations", included both intended (see above for "slang") and unintended spelling variations. Table 4 shows how the former and the latter group contributed to this class.

Table 4. Intended (slang) vs. unintended (typos) "Spelling deviations"

OOV subclass	`hunspell`		`TreeTagger`	
	% of OOV tokens	% of OOV types	% of OOV tokens	% of OOV types
intended deviations	8.06	5.09	5.97	3.7
unintended deviations	3.81	5.66	3.12	4.54
total	11.87	10.75	9.09	8.23

Additionally, we split this whole major class into the following groups:

1. insertions, e.g. *dennen* instead of *denen*;
2. deletions, e.g. *scho* instead of *schon*;
3. substitutions, e.g. *fur* instead of *für*;

according to the type of operation which led to a particular deviant spelling. In cases when multiple different operations were involved simultaneously on one word, we explicitly marked each of these operations in our data. Statistical distribution of these subclasses is shown in Table 5.

Table 5. Distribution of OOV words belonging to the class "Spelling deviations"

OOV subclass	`hunspell`		`TreeTagger`	
	% of OOV tokens	% of OOV types	% of OOV tokens	% of OOV types
insertions	1	1.66	0.79	1.08
deletions	8.3	6.28	6.55	5.33
substitutions	2.57	2.81	1.75	1.82
total	11.87	10.75	9.09	8.23

As is clear from the table, deletions were by far the most common type of operations which produced deviated spelling forms. This is partially explained by either deliberate or accidental omissions of characters made by users, but an even bigger part of deletions was due to the automatic truncations of too long messages which were performed by the Twitter service itself.[5]

Since the latter two major groups of OOVs (namely, Twitter-specific phenomena and spelling deviations) accounted for more than half of all unknown

[5] Since Twitter imposes a strict restriction of 140 characters on the length of posted messages, longer tweets get automatically truncated.

tokens found in Twitter and posed a significant problem for automatic analysis, we decided to address these classes by applying a set of normalization procedures which we describe in more detail in the next section.

4 Text Normalization Procedure

4.1 Replacement of Twitter-Specific Phenomena

Following the syntactic disambiguation approach proposed by Kaufmann (2010), we decided to replace Twitter-specific phenomena (TSP) which played a significant syntactic role in a sentence with artificial tokens representing the type of the TSP being replaced. In cases when a TSP was not a part of a phrase but rather a standalone element and if it was of little interest for further processing, we decided to strip such elements off from the message.

In order to determine in which cases a TSP formed a part of a broader syntactic construction and in which cases it did not, we created a set of regular expression rules which looked at the left and right context of the TSPs in question. These rules decided whether a particular TSP had to be replaced, deleted or kept as is.

One of such rules, for example, said that a @-mention which appeared at the beginning of a message had to be deleted from the tweet unless it was followed by a lowercased word which appeared to have the ending of a finite verb or was a preposition or conjunction. Another rule specified that @-mentions appearing in the middle of a message had to be replaced with the artificial token "%UserName". In cases when Twitter mentions were preceded by the words "RT" or "MT", they were always deleted from the text irrespectively of their positions.

Similar steps were done for e-mail addresses and hyperlinks. These elements were removed if they appeared at the end of a tweet and were not preceded by a preposition or conjunction. Otherwise, these tokens were replaced with the dummy words "%Link" and "%Mail", respectively.

Since our system was supposed to become a part of a larger sentiment analysis project, we did not remove emoticons from messages but instead always replaced them with one of three artificial tokens – "%PosSmiley", "%NegSmiley", or just "%Smiley" – depending on the primary sentiment polarity usually conveyed by a particular emoticon type. In cases when we were unsure about the polarity, we used the general replacement token "%Smiley".

In contrast to Kaufmann (2010), who suggested deleting contiguous runs of hashtags in the initial and final position of the tweets, we did not remove hashtags from messages since these words could be a valuable source of lexical information for further processing modules. We instead just stripped off the leading #-character from all hashtags leaving the alphabetical part of them.

Text fragments which were to be modified by our rules were marked as subgroups in our conditional regular expressions. In cases when multiple expressions matched the same context, we decided which rule to use by applying the leftmost

longest principle to the text spans matched by the subgroups. If these text spans were exactly the same, we looked which regular expression as a whole matched the leftmost longest fragment of the input text. If even that did not help prioritize a particular rule, the rule with the greater number of subgroups or (on equal number of subgroups) one which appeared first in our rule set was applied.

We also added all introduced artificial tokens to the custom dictionaries of `TreeTagger` and `hunspell`. Furthermore, positions and lengths of all performed replacements were remembered along with the original input words which were deleted or replaced, so that a restoration step could be made any time after the processing.

4.2 Restoration of spelling deviations

Since intended spelling deviations turned out to account for the majority of all tokens with deviant spelling found during the analysis (cf. Table 4), we decided to primarily address this subclass of OOVs.

A closer look at the words belonging to this subgroup revealed the fact that most of them were formed by some more or less regular rewriting processes. According to our data, the most productive of such processes were:

- Omissions of 'e' in unstressed positions, e.g. *würd*, *zuguckn* etc. In cases when 'e' was a part of the impersonal pronoun "es" following a verb, the remaining 's' of this pronoun was usually appended to the preceding verb form, e.g. *wirds* instead of *wird es*;
- Omissions or replacements of final consonants with their voiceless equivalents, e.g. *nich* instead of *nicht* or *Tach* instead of *Tag*;
- Multiple repetitions of characters as a means of expressing elongation of sounds, e.g. *Hilfeeee*, *süüüß*;
- Omissions of 'ei' from the indefinite article, e.g. *ne* instead of *eine* or *nem* in lieu of *einem*;
- Omissions of 'he' from the verbal prefixes *herauf-*, *heraus-*, *herum-* etc., e.g. *rauszukriegen*, *rumbasteln*.

Since the reverse of these operations could be easily formed with a handful of programmatic rules, handling them with stochastic techniques like NCM or SMT seemed to be unnecessary. We developed a rule-based method in which a set of transformation rules first searched for tokens with suspicious character sequences. It was then checked whether these matched tokens were not present in the dictionary and if their supposed normalized variants were valid in-vocabulary terms. If these conditions were satisfied, we applied the transformation associated with the given rule.

We tested our set of 11 rules on a held-out corpus of 184,331 tweets. It turned out, however, that dictionary checks alone were not sufficient, since they did not prevent us from making such incorrect changes like the one shown in the Example 1.

Example 1. Wulff tritt zurück, Georg **Schramme** wird neuer Bundespräsident

In this sentence, the surname "Schramm" was incorrectly replaced with the word "Schramme", since the former token was unknown to the dictionary whereas the latter was an in-vocabulary word. To prevent such erroneous replacements, we decided to incorporate statistical information into our system. To the rules producing errors, we added the restriction that for a given unknown input word w_i and its possible in-vocabulary suggestion w_i^*, the following inequality had to be satisfied:

$$\begin{aligned} log(P(w_{i-1}, w_i)) + log(P(w_i)) + log(P(w_i, w_{i+1})) < \\ log(P(w_{i-1}, w_i^*)) + log(P(w_i^*)) + log(P(w_i^*, w_{i+1})) \end{aligned} \tag{1}$$

This inequality means that the sum of log-probabilities of the preceding bigram, current unigram and the immediately following bigram for the word to be replaced had to be lesser than the corresponding sum of log-probabilities for its possible substitution.[6] Both unigram and bigram statistics were gathered from our held-out set of tweets and then smoothed with add-λ smoothing. This statistical information was later also used in the evaluation phase.

A different technique was used for tackling elongations of characters. For these, we first squeezed successive repetitions of more than three characters in a row to a maximum of three repetitive characters. After that, all possible combinations of consecutively squeezed repeated characters were generated. It means that for a word like "daaaaaassss", we first transformed it to "daaasss" and then generated all possible variants with triple, double and single repetitions of "a" and "s".

From this generated set, we removed all candidates which did not appear in the dictionary. If multiple candidates were left after that, e.g., "dass" and "das", these candidates were scored using the joint sum of bigram and unigram log-probabilities as described by the inequality (1). If no word in the generated set was found in the reference dictionary, we fell back to the method suggested by Brody and Diakopoulos (2011) and replaced each repeated letter with a single instance of that letter. For each such condensed form, there was a mapping to the most frequently elongated form occurring in a training corpus from which this dictionary was generated. If no condensed form was found, we simply returned the squeezed form of the word with maximum consecutive repetitions of three characters.

A much harder case for normalization represented spelling variants which were classified as unintended misspellings. In contrast to the intended spelling deviations, these variants did not show any regularities in the way they were formed. The only noticeable exception from this were the cases of incorrect spellings of umlauts. The German characters *ä*, *ö*, and *ü* were often rewritten as sequences *ae*, *oe*, and *ue*, respectively. Since these character sequences also often appear in common German words like, *Mauer*, *Feuer*, *virtuell*, we extracted all words having these character sequences from a corpus of newspaper articles

[6] One anonymous reviewer suggest to use a fixed threshold instead of only requiring the probability of the replacing token to be higher than the probability of the current token. We leave this interesting proposal for further research.

which were assumed to be typed with correctly spelled umlauts. After that, we replaced the sequences *ae*, *oe*, and *ue* with their respective umlaut forms, only if they did not appear as correctly spelled words in the newspaper corpus.

5 Evaluation

We performed both an intrinsic and extrinsic evaluation of the effectiveness of our normalization procedure. As a part of intrinsic evaluation, we first measured how the relative rate of OOV words in the input text changed after applying normalization. It turned out that this rate decreased by 5.6 % for `hunspell` and by 8.9 % for `TreeTagger`. Such significant decrease, however, was mainly explained by our replacement/removal of Twitter-specific terms and the addition of the artificial replacement tokens to the applications' dictionaries.

In order to separately measure the performance of our spelling normalization module, we decided to follow the evaluation approach that had been used by Han and Baldwin (2011). In this method, the BLEU (Papineni, 2002) and NIST scores between manually and automatically normalized tweets were measured. Additionally, for all tokens requiring normalization in the unnormalized input tweets, we calculated the number of tokens which actually were changed during normalization. This was called the recall of the system. Furthermore, for all items that were changed by the normalization procedure, the percentage of correct replacements was measured and considered as the system's precision. Finally, the F_1-score was calculated as the harmonic mean of the latter two metrics.

For our evaluation test set, we extracted all messages from our analyzed corpus of 10,000 tweets which contained at least one word that was marked as a deviant spelling in the annotated data. We constructed hand-corrected gold data from this as follows: We automatically replaced the deviant spellings with their correct variants which we had previously specified during the annotation. However, since we did not annotate all OOVs which occurred in the text only once, but only 1,000 randomly selected hapax legomena, we manually corrected unknown words which were not included in our annotation. After that, extracted messages were pre-processed by replacing Twitter-specific phenomena as described in Section 4.1. This gave us a test set of 1,492 messages in which 1,480 variant spellings were to be corrected. The results of our evaluation are shown in Table 6. In addition to the original, unmodified tweets ("before normalization") we report the scores by successively adding each sub-module (described in Section 4) to the normalization procedure.

Finally, as a first step towards an extrinsic evaluation, we measured how the accuracy of part-of-speech-tagging changed after normalization. For this, we randomly picked 200 tweets from our analyzed data and PoS-tagged them before and after normalization using `TreeTagger`. After manual annotation of incorrectly predicted tags, we measured what percentage of tags was assigned correctly. It turned out, that the overall accuracy increased by 6.41 % from 80.56 % to 86.97 % after performing normalization.

Table 6. Evaluation results for the spelling correction module. RBC = rule-based correction; DR = dictionary replacement of a fixed list of 20 frequently occurring colloquialisms on Twitter; ES = character elongation squeezing; UR = umlaut replacement

Input text	BLEU	NIST	Precision	Recall	F-score
before normalization	0.7929	12.55	–	–	–
RBC	0.8455	12.9873	0.84	0.29	0.4317
RBC + DR	0.8638	13.1474	0.875	0.383	0.5328
RBC + DR + ES	0.8687	13.1971	0.875	0.4162	0.5641
RBC + DR + ES + UR	0.8766	13.2638	0.8793	0.4584	0.6027

6 Conclusions and Future Work

With this article, we hope to have provided a better insight into the nature of ill-formed words in German Twitter messages. As was shown in Section 3, special markup elements and casual spellings accounted for more than half of the unknown words discovered in tweets. Furthermore, almost three quarters of non-standard spellings could be regarded as intended spelling variants rather than unintended slips of the finger. Such intended spellings also showed the tendency to be formed by well formalized processes and to be used frequently in text.

We suggested a rule-based text normalization approach which could serve as a baseline comparison measure for future normalization methods which may be suggested for German tweets. As was shown in the previous sections, our approach can effectively address some of the most frequent phenomena which contribute to a higher rate of out-of-vocabulary words in Twitter texts.

Still, our approach has significant potential for improvement. It only partially addressed the group of unintended spelling mistakes. And though this class of spelling deviations was a minority class in our analyzed data, it should be taken into account that we only analyzed 1,000 hapax legomena, while they were much more common in the data. Further, most of the unintended typos could be found in this group of hapax legomena. Another noticeable fact about unintended misspellings which became apparent during the analysis is that they can hardly be tackled with rule-based techniques due to their irregular stochastic nature. Instead, statistical methods like NCM or SMT would be more appropriate for these kinds of OOVs.

Acknowledgement.

This work was funded by the Federal Ministry of Education and Research as a part of the collaborative project "Analysis of Discourse in Social Media" (project number 01UG1232D). The authors would like to thank two anonymous reviewers for their helpful comments. We are also grateful to Martin Schwietzke for his help with the annotation of data.

References

Aw, A., Zhang, M., Xiao, J., Su, J.: A Phrase-based Statistical Model for SMS Text Normalization. COLING/ACL (2006) 33–40

Beaufort, R., Roekhaut, S., Cougnon, L. A., Fairon, C.: A Hybrid Rule/Model-Based Finite-State Framework for Normalizing SMS Messages. ACL (2010) 770–779

Brody, S., Diakopoulos, N.: Cooooooooooooooollllllllllllll!!!!!!!!!!!!!!: using word lengthening to detect sentiment in microblogs In Proceedings of the Conference on Empirical Methods in Natural Language Processing. (2011) 562–570

Choudhury, M., Saraf, R., Jain, V., Mukherjee, A., Sarkar, S., Basu A.: Investigation and Modeling of the Structure of Texting Language. International Journal of Document Analysis and Retrieval: Special Issue on Analytics of Noisy Text. 10 (2007) 157–174

Clark, A.: Pre-processing very noisy text. In Proceedings of Workshop on Shallow Processing of Large Corpora. Corpus Linguistics. Lancaster. pp. 12–22 (2003)

Clark, E., Araki, K.: Text Normalization in Social Media: Progress, Problems and Applications for a Pre-processing System of Casual English. PACLING. Procedia - Social and Behavioral Sciences 27 (2011) 2–11

Dorsey, J.: "just setting up my twttr". `https://twitter.com/jack/status/20` Accessed February 26, 2013. (2006)

Han, B., Baldwin, T.: Lexical Normalization of Short Text Messages: Makn Sens a #twitter. ACL HLT. (2011) 368–378

Kaufmann, M.: Syntactic normalization of twitter messages. The 8-th International Conference on Natural Language Processing. (2010)

Kobus, C., Yvon, F., Damnati, G.: Normalizing SMS: are Two Metaphors Better than One? COLING (2008) 441–448

Oliva, J., Serrano, J. I., and Del Castillo, M. D., and Igesias, A.: A SMS normalization system integrating multiple grammatical resources. Natural Language Engineering. (2013) 121–141

Papineni, K., Roukos, S., Ward, T., Zhu, W.-J.: Bleu: a Method for Automatic Evaluation of Machine Translation. ACL (2002) 311–318

Petersen, L. J.: Computer Programs for Detecting and Correcting Spelling Errors. Communications of the ACM 23/ 12 (1980) 676–687

Schmid, H.: Probabilistic Part-of-Speech Tagging Using Decision Trees. In Proceedings of International Conference on New Methods in Language Processing. (1994)

Shannon, C. E.: A mathematical theory of communication. Bell system techical journal, 27:379–423 (1948) 623–656

Toutanova, K., Moore, R. C.: Pronunciation Modeling for Improved Spelling Correction. ACL (2002) 144–151