ABSTRACT

The rapid proliferation of microblogs such as Twitter has resulted in a vast quantity of written text becoming available that contains interesting information for NLP tasks. However, the noise level in tweets is so high that standard NLP tools perform poorly. In this paper, we present a statistical truecaser for tweets using a 3-gram language model built with truecased newswire texts and tweets. Our truecasing method shows an improvement in named entity recognition and part-of-speech tagging tasks.

1. INTRODUCTION

The growing popularity of Twitter is resulting in the creation of a large number of short messages (tweets) that contain interesting information for several NLP tasks. However, traditional NLP tools have been shown to perform poorly on tweets due to their lack of standardization [7, 8, 9, 19]. The language of social media text is colloquial, and contains a high proportion of mispellings, insertions, neologisms, jargon and non standard capitalization [1].

Table 1: Examples of noisy capitalization in tweets.

<table>
<thead>
<tr>
<th>Tweets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 the one and only MARK MEISMER is teaching TONIGHT At EDGe at 830-1030 !</td>
</tr>
<tr>
<td>2 ThAnK gOd ItS fRiDaY !</td>
</tr>
<tr>
<td>3 HATE IT WHEN THEY KNOW I'M PAGAN AND INVITE ME . ANGER .</td>
</tr>
</tbody>
</table>

Twitter users frequently use capitalization as emphasis. For example, in table 1 the first tweet shows that the user has entirely capitalized the word TONIGHT and the first letter of a series of words including At. Tweet 2 shows that the user has randomly capitalized letters in words such as ThAnK or fRiDaY. In the last example, every word of the tweet is in uppercase.

In this paper, we show the overall impact of tweet truecasing in Named Entity Recognition (NER) and part-of-speech (PoS) tagging. We demonstrate that our statistical truecaser, which uses a 3-gram language model, can improve NER and PoS tagging accuracy.

The paper is divided as follows. In section 2 we describe work related to case restoration. Then, we present our approach in section 3. Next, in section 4, we show the performance of our 3-gram capitalizer and the benefit of truecasing in NER. Finally, we summarize the paper.

2. RELATED WORK

The case restoration task, also known as truecasing [15], is the process of recovering case information for texts in lowercase. In addition to improving the readability of texts, truecasing is useful for several NLP tasks.

Recently, truecasing has been explored with various methods for statistical machine translation or speech recognition, but has received much less attention in social media normalization.

[15] view truecasing as a lexical ambiguity resolution problem, where several versions of a word happen to have different surface forms. They use an approach based on 3-gram language models estimated from a corpus with case information. [22] exploits bilingual information to create a probabilistic capitalization model using Conditional Random Fields.

[13] presents a purely text-based n-gram language model for punctuation and capitalization restoration in order to improve automatic speech transcripts. [1] uses machine-learning techniques to restore punctuation and case in English text. They achieved the best results using a variety of lexical and contextual features, as well as iterative retagging.

[14] creates a normalization pipeline for tweets including a truecaser based on a 3-gram language model. However, this approach has not yet been evaluated. Thus, the impact of recasing in NLP tasks such as NER for social media was not shown. [19] create a capitalization classifier which predicts whether or not a tweet is informatively capitalized. This feature was then used to improve their statistical NER model for microblogs.
3. APPROACH

3.1 Tweet casing classification

Our truecasing strategy is to determine if the tweet capitalization is informative or uninformative. To achieve this, we build a classifier that can effectively predict whether or not a tweet is informatively capitalized.

For learning, we train an SVM classifier using these features:

- the fraction of words in the tweet which are capitalized,
- the fraction which appear in a dictionary of frequently lowercase/capitalized words but are not lowercase/capitalized in the tweet,
- the fraction of lowercase/capitalized proper nouns,
- the number of times the word ‘I’ appears in lowercase and whether or not the first word in the tweet is capitalized.

We have used 800 tweets manually labelled as having either informative or uninformative capitalization. We train the classifier using these 800 tweets and we compute the average 4-fold cross-validation results.

<table>
<thead>
<tr>
<th></th>
<th>NB</th>
<th>MaxEnt</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>71.25</td>
<td>91.71</td>
<td>94.91</td>
</tr>
</tbody>
</table>

Table 2: Average four-fold cross-validation accuracies in percent.

Table 2 shows that the SVM algorithm performs better than the Naïve Bayes and Maximum Entropy approaches, and the accuracy achieved is around 95%.

3.2 Casing distribution

Using our capitalization classifier, we have analysed the casing distribution in 3 Twitter NER datasets: Ritter [19], Finin [9] and MSM [3]; the NUS-SMS corpus [4] and the CoNLL 2003 testing data [21].

Figure 1 shows that the Ritter and MSM datasets contain around 10 to 12% noisy capitalization. The Finin corpus seems to be a better quality corpus with 5% uninformative capitalization. NUS-SMS corpus contains less than 9% noisy capitalization. Finally, the CoNLL 2003 test data contains less than 2% wrong capitalization.

3.3 Data set

To build our truecaser, we train a 3-gram language model using a portion of the English Gigaword Corpus (Fifth Edition), provided by the Linguistic Data Consortium [18]. It consists of newswire texts covering the 24-month period of January 2009 through December 2010 from the following news agencies:

- Agence France-Presse, English Service
- New York Times Newswire Service
- Washington Post/Bloomberg Newswire Service

Additionally, we have used our capitalization classifier to build a corpus of 200,000 tweets containing informative capitalization. We have included these tweets in order to improve our language model. Altogether, the corpus comprises about 2 billion words.

3.4 Architecture

Our truecasing approach used the HMM-based tagger included in the disambig tool from the SRILM toolkit [20]. Figure 2 describes the tweet truecasing pipeline. In the first step, we build a recasing model by training a language model on truecased English text (see subsection 3.3). Next, we create a Map that contains a list of all the alternative ways that each word can be capitalized. This map lists a lowercased word as the key and associates it with all of the alternate capitalization of that word. This Map contains 1.1 million of entries and will be used in our truecaser.

The truecaser is built on GATE [6] and consists of a set of processing resources executed in a pipeline over a corpus of tweets. The pipeline consists of 3 parts:

- Statistical capitalization classifier (categorizes each well and badly capitalized tweet);
- LowerCase sentence step (each badly capitalized sentence is lowercased);
- HMM-based tagger that uses 3-gram language model to compute the probabilities of the most likely case.

1The wrong capitalization in the CoNLL 2003 test data is essentially due to the headlines.
4. EXPERIMENTS

4.1 Truecaser evaluation

We include the lowercase (lc) version as a baseline and we compare our tool to the Stanford Truecaser [16], which is implemented with a discriminative model using CRF sequence tagger [10].

We use BLEU score [17] to measure truecasing performance. BLEU computes the similarity between two sentences using n-gram statistics, and is widely-used in machine translation evaluation. A set of parallel corpora, consisting of 55,000 sentences from the Agence France-Presse (AFP) corpus and 2,400 English tweets [19] and their normalized references is used as a gold standard.

The performance comparisons between our n-gram truecaser and the lowercase baseline are shown in Table 3. The BLEU score shows that the capitalization n-gram model performs substantially better than the baseline. The baseline achieved a BLEU score of 73.11% on the AFP corpus and 73.14% on the Ritter dataset. Our 3-gram language model improves truecasing effectiveness; BLEU score rises to 94.13% on the AFP corpus and 78.36% on Ritter.

The Stanford Truecaser performs better than the baseline with a BLEU score of 89.15% on the AFP dataset, but when applied on tweets it achieved worse results with a BLEU score of 72.17%.

Our 3-gram language model without tweets obtains 92.07% on the AFP corpus and 74.45% on the Ritter dataset. Adding tweets to the language model improves BLEU score; 94.13% compared with 78.36%. Our 4-gram language model achieved 94.19% on the AFP English corpus and 78.38% on the Ritter dataset. The 5-gram language model achieved 94.20% on the AFP English corpus and 78.39% on Ritter. Therefore, increasing the n-gram order does not help nearly as much so we choose to use the 3-gram language model as it is faster.

The 3-gram approach resolves ambiguous cases. For example, with a unigram model “new york” will be recapitalized “New York” because new is almost always lowercased. The 3-gram approach takes into account the context and when new is followed by York, it is almost always capitalized.

4.2 Truecaser output

We have analyzed the differences between results produced by the Stanford Truecaser and our LM Truecaser. Examples (1) and (2) show original microblog messages, including a variety of mis-capitalization, and the truecased text (by Stanford Truecaser and our LM Truecaser).

(1) Original Tweet: Time Warner Cable Boycotting Epix Movie Channel Because It Did A Deal With Netflix

Stanford Truecaser output: Time Warner cable boycotting EPIX movie channel because it did a deal with Netflix

LM Truecaser output: Time Warner Cable boycotting Epix movie channel because it did a deal with Netflix

(2) Original Tweet: Fooooootball game toooonight then auburn game tomorrow war damn eagle

Stanford Truecaser output: fooooooootball game toooonight then Auburn Game Tomorrow War Damn Eagle

LM Truecaser output: fooooooootball game toooonight then Auburn game tomorrow war damn Eagle

As illustrated, not all errors are corrected, and our LM Truecaser performs better than the Stanford Truecaser. It also appears that the Stanford Truecaser introduces noisy capitalization.

4.3 Application: Named Entity Recognition

In our experiment on NER for tweets, we have integrated our truecaser in TwitIE \(^2\) and Stanford NER [10].

We evaluate our system on the Ritter and MSM datasets, which contain around 10% to 12% uninformative capitalization.

In Table 4, TwitIE without truecasing achieves a traditional F-Measure of 47.65% on the Ritter corpus and 66.71% on the MSM dataset. Adding truecasing does not improve accuracy; the system obtains an F-Measure of 47.63% on Ritter and 66.64% on MSM. This result is explained by the fact that TwitIE doesn’t actually use case information when detecting named entities.

Stanford NER without truecasing achieves a traditional F-Measure of 47.34% on Ritter and 73.25% on MSM. Adding truecasing improves extraction effectiveness, and the system obtains an F-Measure of 48.94% on Ritter and 74.64%.

4.4 Application: PoS Tagging

Part-of-speech tagging is necessary for many tasks such as named entity recognition and linking. However, microblog content is difficult to part-of-speech tag as it is noisy and contains linguistic er-

\(^2\)TwitIE is a customisation of ANNIE [5] for microblog content.
Table 4: Named Entity Recognition performance with truecasing and without on Ritter and MSM datasets. Experiments using TwitIE and Stanford NER (StanNER) systems.

<table>
<thead>
<tr>
<th>System</th>
<th>Ritter P</th>
<th>Ritter R</th>
<th>Ritter F1</th>
<th>MSM P</th>
<th>MSM R</th>
<th>MSM F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwitIE</td>
<td>46.94</td>
<td>48.38</td>
<td>47.65</td>
<td>68.87</td>
<td>64.69</td>
<td>66.71</td>
</tr>
<tr>
<td>TwitIE_LM_TC</td>
<td>46.90</td>
<td>48.38</td>
<td>47.63</td>
<td>68.80</td>
<td>64.62</td>
<td>66.64</td>
</tr>
<tr>
<td>StanNER</td>
<td>52.31</td>
<td>43.23</td>
<td>47.34</td>
<td>76.08</td>
<td>70.62</td>
<td>73.25</td>
</tr>
<tr>
<td>StanNER_LM_TC</td>
<td>51.81</td>
<td>46.37</td>
<td>48.94</td>
<td>76.62</td>
<td>72.76</td>
<td>74.64</td>
</tr>
</tbody>
</table>

Table 5: Performance (in %) of PoS tagging with truecasing and without on T-Pos and DCU datasets.

<table>
<thead>
<tr>
<th>Tagger</th>
<th>T-Pos Token</th>
<th>T-Pos Sentence</th>
<th>DCU Token</th>
<th>DCU Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derczynski et al., 2013</td>
<td>88.69</td>
<td>20.34</td>
<td>89.37</td>
<td>36.80</td>
</tr>
<tr>
<td>Derczynski_TC</td>
<td>89.73</td>
<td>21.54</td>
<td>90.27</td>
<td>37.60</td>
</tr>
</tbody>
</table>

5. CONCLUSION

In this paper, we have discussed tweet truecasing, the process of restoring case information in social media messages. We have presented a truecasing method using an n-gram language model built with truecased newswire texts and tweets. Our results suggest that our capitalization 3-gram model performs substantially better than the Stanford Truecaser. Our evaluation shows an improvement of restoring case information in social media messages. We have integrated our truecaser in Derczynski et al’s tagger [8].

A natural direction in which to continue this research is to adapt this process to languages other than English and also to apply our truecaser to other tasks such as named entity linking.

6. ACKNOWLEDGMENTS

This work was carried out during the first author’s stay at Dept. Computer Science, University of Sheffield, funded by the Swiss National Science Foundation (grant P1GEPI_151615).


