IITP: Hybrid Approach for Text Normalization in Twitter

Md Shad Akhtar, Utpal Kumar Sikdar and Asif Ekbal
Dept of Computer Science and Engineering
IIT Patna
Patna, India
(shad.pcs15,utpal.sikdar,asif)@iitp.ac.in

Abstract

In this paper we report our work for normalization of noisy text in Twitter data. The method we propose is hybrid in nature that combines machine learning with rules. In the first step, supervised approach based on conditional random field is developed, and in the second step a set of heuristics rules is applied to the candidate wordforms for the normalization. The classifier is trained with a set of features which were are derived without the use of any domain-specific feature and/or resource. The overall system yields the precision, recall and F-measure values of 90.26%, 71.91% and 80.05% respectively for the test dataset.

1 Introduction

Twitter has seen a phenomenal growth in the number of users during the last few years. Over 500 million user accounts have been registered with it with approx 302 million active users [1]. Amount of user generated contents over the web would be unarguably enormous i.e. almost 500 million tweets per day [2]. The fact that Twitter data (*or tweets*) are typically noisy and unstructured in nature are due to several grammatical & spelling mistakes it contain. The size limitation (constitute upto 140 characters only) is the another prominent reason. It confines a user to devise different short forms (*e.g. 'c u ltr.'* for *'see you later.'*) of a valid word. Interpreting such forms may be an easier task for a human being but, is very difficult to build an accurate system for solving any problem related to natural language processing (NLP). At times, user puts extra emphasis by stretching/elongating a valid word to express their feelings. For example, they often use word like '*yeeessss*' to show their happiness, which is a stretched form of '*yes*'.

Normalization of noisy text is an important and necessary pre-processing task for building different applications related to text processing. It is pretty obvious from various studies (Liu et al., 2011; Foster et al., 2011) that presence of noisy texts makes any natural language processing (NLP) task very tedious to achieve good accuracy levels. The goal of normalization is twofold, i.e. a) identification of candidates for normalization and b) converting the candidate wordforms to the normalized form. Unlike the general well-formatted corpus, like newswire, it does not always contain noisy text. Its main sources are normally those platforms on which users have complete freedom to express themselves. Therefore, user generated tweets are one of the major sources of noisy texts. In the last couples of years researchers across worldwide are actively working for the normalization of noisy contents of twitter (Han and Baldwin, 2011; Liu et al., 2012; Wang and Ng, 2013; Porta and Sancho, 2013; Chrupala, 2014). In (Han and Baldwin, 2011), a linear Support Vector Machine (SVM) classifier was trained for detecting ill-formed words, and then performed normalization based on morphophonemic similarity. Application of edit operations and recurrent neural embedding can be found in (Chrupala, 2014) for text normalization. Their method learns sequence of edit operations using conditional random field (CRF). In another work, (Liu et al., 2012) investigated the human perspectives of enhanced letter transformation, visual priming and the phonetic similarity for the text normalization. The use of beam search decoder and finite-state transducers can be seen in (Wang and Ng, 2013; Porta and Sancho, 2013) for the word normalization. These existing works are based on different setups and datasets.

[1] http://en.wikipedia.org/wiki/Twitter

[2] http://www.cnet.com/news/report-twitter-hits-half-a-billion-tweets-a-day/

Proceedings of the ACL 2015 Workshop on Noisy User-generated Text, pages 106–110,
Beijing, China, July 31, 2015.

For further advancement of research on text normalization and to provide a common benchmark setup for evaluation, a shared task "ACL2015 W-NUT: Normalization of Noisy Text in Twitter"[3] was organized. The shared task had two variants: constrained mode and unconstrained mode. We participated only for the constrained mode which did not permit us to use any external resources and/or tools except few that were recommended by the organizers. In this paper we report our work for normalization. We implemented a hybrid system where machine learning along with rules are utilized to perform the task. We have exploited lexical and syntactic properties of a tweet as discussed in section 3.1 to derive a feature set for identification of noisy text in the first step. We train Conditional Random Field (CRF) (Lafferty et al., 2001) as a machine learning algorithm to identify the candidate wordforms that need to be normalized. In second step, we apply some rule based methods (as defined in section 3.2) in order to normalize the wordforms which were identified in first step.

The organization of the paper is as follows. A brief theoretical discussion on CRF is presented in section 2. Section 3 discuss about the feature set and methodology used in the proposed work. Experimental result and analysis can be found in section 4. We conclude the paper in section 5.

2 Conditional Random Field (CRF)

Conditional Random Field, introduced by (Lafferty et al., 2001), is a robust sequence learning algorithm based on the conditional probability. Let an observation sequence $O =< o_1, o_2, ..., o_T >$ is given, then the conditional probability of a state sequence $S =< s_1, s_2, ..., s_T >$ can be formulated as:

$$P(S|O) = \frac{1}{Z_0} exp(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k \times f_k(s_{t-1}, s_t, o, t)) \quad (1)$$

where λ_k is the weight of the feature function $f_k(s_{t-1}, s_t, o, t)$, that is to be learned via training. In general, feature functions takes binary value but at times it may range between $-\infty$ to $+\infty$. The output of this function relies on certain state sequence i.e. s_{t-1}, s_t and observation properties. The normalization factor Z_o, define in equation 2, is used to make all conditional probabilities sum up to unity and can be calculated efficiently using dynamic programming.

$$Z_0 = \sum_{s} exp(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k \times f_k(s_{t-1}, s_t, o, t)) \quad (2)$$

3 Methods

After discussing theoretical aspect of CRF, we now describe our methodology that we use to perform text normalization. It comprises of two steps. First step consists of training a supervised machine learning model for the identification of noisy text. We implement a set of features that were mostly derived without using any deep domain-specific resources and/or tools. We perform 3-fold cross validation on the training data to determine the best feature combination. In the second step, potential candidates identified to be noisy were analysed and subsequently processed using various heuristic based rules for normalization. Figure 1 depicts schematic diagram of the proposed system.

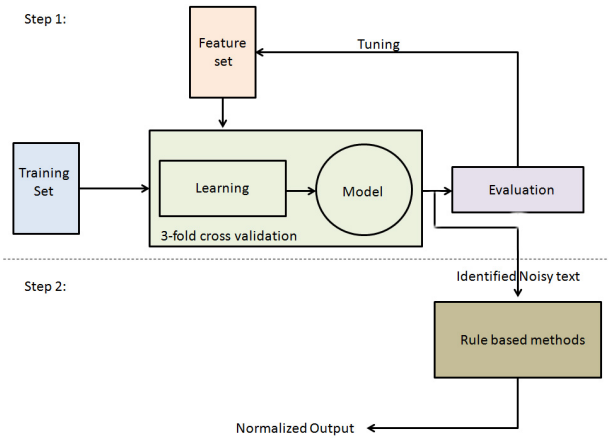

Figure 1: Proposed methodology. Dotted horizontal line separates two steps.

3.1 Feature Set

This section describes the feature set that was implemented for identifying the potential candidates that need to be normalized. All the features defined are domain-independent in nature. No other external resources and/or tools, with the exception of vocabulary of words [4], were used in the proposed work. Following are the brief descriptions of the implemented features.

[3] http://noisy-text.github.io/

[4] http://noisy-text.github.io/files/scowl.american.70

1. **Local context:** Local contextual information in the forms of surrounding words are used as the feature.

2. **Vocabulary word:** Noisy word can not be a part of valid vocabulary. Therefore, all out-of-vocabulary (OOV) are the potential candidates that should be normalized. We define a feature that fires if the current is OOV.

3. **Part-of-Speech (PoS) information:** We use CMU-Tweet PoS tagger [5] for extracting the PoS information. This is used as a feature of CRF.

4. **Word length:** From the given training data we observed that noisy texts are generally shorter in lengths. We define a binary valued feature that is set to high if the length of the candidate token exceeds a predetermined threshold. In our case we assume the token to be a noisy text if its length is less than 4 characters.

5. **Suffix and Prefix:** Suffixes and prefixes of length upto 4 characters of the current word are used as the features.

6. **Only digit:** This feature checks whether the current token is consisting of only digits or not. The word has a low probability of being noisy if it contains only the digits. Few exceptions are *2*(to), *4*(for) etc.

7. **AlphaDigit:** An alphanumeric token have a high probability of being a noisy text. A binary valued feature is thus defined in the proposed work which fires when the token is alphanumeric.

8. **Consecutive characters:** This feature fires when a token consists of more than 2 consecutive characters is found. This feature helps in identifying the stretched/elongated words.

9. **Compact word form:** Apostrophe mark (') is used to indicate the omission of one or more letters from a word (e.g. *i'm, you're* etc.). A binary feature is defined which identifies the missing apostrophe mark in a word.

10. **Present participle (*a.k.a* ing-form) of a verb:** From the analysis of training data we observed that people tends to skip 'i' or 'g' from the present participle, i.e. *ing* form, of a verb. For example, they use *goin* in place of *going*. Thus a feature is defined and set to 'on' if a token is found with the above pattern.

11. **Single character:** This feature fires when the token consists of a single character only with the exception of two characters i.e. 'I' and 'a'.

12. **Hash tag & Username:** Hash tags and usernames in tweets, which starts with # & @ respectively, are not considered as noisy text in the training data. Therefore this feature is set to false if a token starts with # or @.

3.2 Heuristic rules for normalization

Once the noisy text was identified in the first step, we devise a set of rules for normalization. These rules are heuristic in nature and based on the facts & analysis on the training data. Below is the list of rules implemented according to their application in the proposed work.

1. **Frequent abbreviation:** This is the first rule that we apply on the noisy text. We make use of a list of frequent abbreviations used in twitter and its normal form. The list was compiled from the Web [6,7] and training data. If the token identified as a potential candidate in the first step is present in the list we simply replace it with the normal text, otherwise, we move onto the next rule.

2. **Present participle of a verb:** A rule is defined for a misspelled present participle verb as discussed in section 3.1. We identify and cross check its PoS tag (i.e. *VERB*) in order to retrieve its valid equivalent form.

3. **Missing apostrophe('):** Twitter users normally drops apostrophe mark in tweets. We define a rule to identify and insert a apostrophe mark at proper place. This rule was employed for handling following variants: *'m*, *'ll*, *'ve*, *'re*, *n't*, *'s* etc.

[5] http://www.ark.cs.cmu.edu/TweetNLP/

[6] http://www.webopedia.com/quick_ref/Twitter_Dictionary_Guide.asp

[7] http://marketing.wtwhmedia.com/30-must-know-twitter-abbreviations-and-acronyms/

4. **Elongated form:** Noisy word in its elongated form (i.e. *yeeeeesss* for *yes*) are identified and translated into valid word by iteratively stripping off consecutive characters.

5. **Split two merged words:** This rule splits a noisy word, if it is a concatenation of two valid words. For example 'thankyou' is concatenation of two separate words i.e. 'thank' and 'you'. We find out word pair at each split point and applied this rule for the pair that has both valid word. Token 'thankyou' has following word pair for 7 spit point: i.e. (1: 't', 'hankyou'; 2: 'th', 'ankyou'; 3: 'tha', 'nkyou'; 4: 'than', 'kyou'; 5: 'thank, 'you'; 6: 'thanky', 'ou'; and 7: 'thankyo', 'u';). Word pair ('thank, 'you') at split point 5 is chosen for the normalization. Before applying this rule a threshold for word length was heuristically set to 6 characters.

6. **British to American standard:** American standard was preferred as an official English language standard for the shared task. We define a rule which identifies British standard word and convert it to corresponding American standard counterpart. Notable differences between the two standards that we have incorporated in the work are 'our' to 'or' (e.g. *labour* to *labor*), 'ise' to 'ize' (e.g. *realise* to *realize*), 're' to 'er' (e.g. *centre* to *center*) etc.

4 Datasets and Experiments

In subsequent subsections we discuss the dataset used in the system and evaluation results, respectively.

4.1 Data Set

Objective of the shared task was to identify and normalize the noisy text in tweets. Only training dataset was provided by the shared task organizers. The training dataset comprise of 2,950 tweets and a total of 3,942 noisy tokens were present in the dataset. In absence of the development dataset, we use 3-fold cross validation for training the model. Gold standard test datasets contains 1,967 tweets. Table 1 list the statistics of the datasets.

Dataset	# Tweets	# Tokens	# Noisy
$train$	2950	44385	3942
$test$	1967	29421	2776

Table 1: Statistics of the dataset

4.2 Experimental Results

Conditional Random Field (CRF)(Lafferty et al., 2001) was used as a base learning algorithm in the proposed work. We use the CRF++ [8] based package for training and testing. To evaluate the performance of the system, an evaluation script along with the dataset was provided by the organizers. We perform 3-fold cross-validation technique to fine-tune the system, and identify the best fitting feature combination. The performance of 3-fold cross validation experiment yields the F-measure of 92.21% for identification problem (i.e. denoting only the candidates for normalization). For the test set it shows the F-measure of 86.63%. After identifying the candidates of normalization we apply heuristics to perform normalization. Rules were applied according to their appearance. We have tried various combination of rule sequences and found that the listed sequence is the one which gives us better performance. While we perform 3-fold cross validation we obtain the precision, recall and F-measure values of 88.59%, 74.92% and 81.19%, respectively. Finally we obtain the precision, recall and F-measure values of 90.26%, 71.91% and 80.05%, respectively. Results of these experiments are shown in Table 2.

We closely analyze the errors encountered by our system. We observed that many errors were due to the incorrect identification of the candidates that need to be normalized. The jumbled words, e.g. *'liek', 'whta'* etc. were not properly recognized. With more accurate identification system we would have achieved better result. For example in case of 100% noisy text identification, we obtained an increase of 3.75% in our final F-measure. For normalization error, our method arguably lags behind in two fronts: a) ambiguities in normalization and b) many-to-one mapping cases. Many of these may be reduced by careful design of the heuristic rules.

5 Conclusion

In this paper we have reported our works that we carried out as part of our participation in the Twitter text normalization shared task. We have developed a hybrid system where in the first step

[8] http://taku910.github.io/crfpp/

Task	Dataset	Precision	Recall	F-measure	Accuracy
Identification	*3-fold cv*	89.51	95.08	92.21	98.70
	test	93.08	81.01	86.63	97.64
Normalization	*3-fold cv*	88.59	74.92	81.19	-
	test	90.26	71.91	80.05	-

Table 2: Result of the proposed system. All values are in %.

we identify the candidates for normalization using a CRF based approach, and in the second step we employed several heuristics for converting the wordforms into the normalized form. We have implemented the features which are mostly domain-independent in the sense that we did not make use of any domain specific resources and/or tools for their extraction. Official evaluation shows that our system achieves the F-measure of 80.05%.

In future we would like to carry out more comprehensive analysis on the evaluation results. The features and rules that we used here are very general and straightforward in nature. In future we would like to modify the system into a fully machine learning based approach and put extra emphasis on errors.

References

Grzegorz Chrupala. 2014. Normalizing tweets with edit scripts and recurrent neural embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 680–686.

Jennifer Foster, Özlem Çetinoglu, Joachim Wagner, Joseph Le Roux, Stephen Hogan, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. 2011. #hardtoparse: POS tagging and parsing the twitterverse. In *Analyzing Microtext, Papers from the 2011 AAAI Workshop, San Francisco, California, USA, August 8, 2011*.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 368–378, Stroudsburg, PA, USA. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, pages 282–289.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 359–367, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 1035–1044, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jordi Porta and José-Luis Sancho. 2013. Word normalization in twitter using finite-state transducers. In *Proceedings of the Tweet Normalization Workshop co-located with 29th Conference of the Spanish Society for Natural Language Processing (SEPLN 2013), Madrid, Spain, September 20th, 2013.*, pages 49–53.

Pidong Wang and Hwee Tou Ng. 2013. A beam-search decoder for normalization of social media text with application to machine translation. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 471–481.