Segmenting Hashtags using Automatically Created Training Data

Arda Çelebi and Arzucan Özgür
Department of Computer Engineering
Boğaziçi University
Bebek, 34342 Istanbul, Turkey
{ arda.celebi, arzucan.ozgur } @boun.edu.tr

Abstract

Hashtags, which are commonly composed of multiple words, are increasingly used to convey the actual messages in tweets. Understanding what tweets are saying is getting more dependent on understanding hashtags. Therefore, identifying the individual words that constitute a hashtag is an important, yet a challenging task due to the abrupt nature of the language used in tweets. In this study, we introduce a feature-rich approach based on using supervised machine learning methods to segment hashtags. Our approach is unsupervised in the sense that instead of using manually segmented hashtags for training the machine learning classifiers, we automatically create our training data by using tweets as well as by automatically extracting hashtag segmentations from a large corpus. We achieve promising results with such automatically created noisy training data.

Keywords: Social Media Processing, Hashtags, Segmentation

1. Introduction

Word segmentation is a well-known problem in the field of Natural Language Processing (NLP) with a number of application fields including tokenization in languages such as Chinese and Japanese, speech recognition, and URL segmentation.

All these well-studied cases aside, the introduction of hashtags on social media platforms including Twitter and Instagram creates a new challenge. Originally, hashtags are introduced as labels to assign contexts to tweets. They start with the '#'sign, followed by one or more words concatenated one after another. While some of them consist of single words, people increasingly create hashtags out of complex phrases and even whole sentences. Since hashtags indicate the contexts of tweets and sometimes even act as the actual messages conveyed in the tweets, breaking them into their constituent words is essential for understanding tweets.

In this study, we develop a feature-rich supervised machine learning based approach for hashtag segmentation. Unlike the traditional supervised setting, where training is performed using manually annotated training data, we utilize the large amount of unlabeled data available in the social media and use two approaches to automatically create our training data. In this respect, our approach can be considered as unsupervised. First, we use normalized tweets to create synthetic hashtag segmentations. Second, we employ a method which automatically extracts hashtag segmentations from a large set of tweets. As a side-contribution of this paper, we also create and share[1] two data sets each one consisting of 1000 manually segmented hashtags.

2. Related Work

Word segmentation methods can be as simple as using a vocabulary to find words. One of the best known vocabulary-based methods is maximum matching (Wong and Chan, 1996) and its variations such as the greedy algorithm used in (Dale et al., 2010). Many state-of-the-art systems employ statistical approaches. In general, they are more successful at handling unknown words and picking the best possible alternative in case of ambiguity. Such methods treat the problem as tagging (Xue, 2003), where they assign a label to each character based on whether it indicates a word boundary. For Chinese word boundary detection, discriminative models such as the word-based perceptron algorithm (Zhang and Clark, 2008), as well as unsupervised methods (Magistry and Sagot, 2001; Chen et al., 2012) were explored. Neural networks (Rumelhart and McClelland, 1986) and lazy learning approaches (Daelamans et al., 1997) were also used in this domain. In a more generic boundary detection study, Islam et al. (2007) use corpus type frequency information along with maximum length frequency and entropy rate.

The problem of hashtag segmentation has only recently drawn the attention of the researchers. Srinivasan et al. (2012) used an unsupervised method to calculate weights from multiple corpora with a joint probability model. Their evaluation was based on the improvement obtained in recall in Twitter search. Berardi et al. (2011) applied the well-known Viterbi algorithm (Jr., 1973) for hashtag segmentation. Recently, Bansal et al. (2015) proposed a method where they first generate candidate hashtag segmentations and then, choose the best one by a re-ranking approach. Besides n-gram and character capitalization features, they make use of context similarity and Wikipedia relatedness features. On their manually annotated test set, they achieved 87.3% accuracy.

3. Methods

3.1. Learning Methods

We approach hashtag segmentation as a word boundary detection problem. We use the BI (BI; Beginning or Inside of a word) schema, which formulates the problem as a binary classification task. We consider two feature-based learning

[1] http://tabilab.cmpe.boun.edu.tr/projects/hashtag_segmentation

FEATURE DESCRIPTION	FEATURE TEMPLATE	TEMPLATE VALUES
frequency of most occ. bigram starting at cursor	%d	*bigramFreq*(*maxOccBigram*(W_s,W_{ss}))
word class bigram starting at cursor	%s-%s	*wordClass*(W_s) + *wordClass*(W_{ss})
length of words around cursor, as well as overpassing word	%d-%d-%d	*len*(W_e) + *len*(W_o) + *len*(W_s)
word overpassing the cursor	%s	W_o
lenght and log freq. of overpassing word	%d-%d	*logFreq*(W_o) + *len*(W_o)
length of overpassing word	%d	*len*(W_o)
longest word itself starting at cursor	%s	W_s
length of longest word starting at cursor	%d	*len*(W_s)
length and log freq. of longest word at cursor	%d-%d	*logFreq*(W_s) + *len*(W_s)
short ngram over cursor and length of surr. words	%s-%d-%d	*short-ngram-over*(*cursor*[0]) + *len*(W_{ee}) + *len*(W_{ss})
short word in middle and length of surr. words	%s-%d-%d	*short-word-at*(*cursor*[0]) + *len*(W_e) + *len*(W_{ss})
three character starting at cursor	%c-%c-%c	*cursor*[0] + *cursor*[1] + *cursor*[2]
orthogonal shape of current and prev. character	%c-%c	*orth*(*cursor*[-1]) + *orth*(*cursor*[0])
orthogonal shape of current and neighbour characters	%c-%c-%c	*orth*(*cursor*[-1]) + *orth*(*cursor*[0]) + *orth*(*cursor*[1])
orthogonal shape of current and prev. two characters	%c-%c-%c	*orth*(*cursor*[-2]) + *orth*(*cursor*[-1]) + *orth*(*cursor*[0])
length and log freq. of word ended just before cursor	%d-%d	*logFreq*(W_e) + *len*(W_e)
word class bigram around cursor	%s-%s	*wordClass*(W_e) + *wordClass*(W_s)
length and log freq. of two words before cursor	%d-%d-%d-%d	*logFreq*(W_{ee}) + *len*(W_{ee}) + *logFreq*(W_e) + *len*(W_e)
length of longest words around cursor	%d-%d	*len*(W_e) + *len*(W_s)
length and log freq. of of words around cursor	%d-%d-%d-%d	*logFreq*(W_e) + *len*(W_e) + *logFreq*(W_s) + *len*(W_s)
freq. of most occ. bigram around cursor and overpassing word length	%d-%d-%d	*biFreq*(*maxBigramOcc*(W_e,W_s)) + *logFreq*(W_o) + *len*(W_o)
log freq. of words around cursor	%d-%d	*logFreq*(W_e) + *logFreq*(W_s)

Table 1: List of the most effective features formed from characters and longest words around cursor position. Feature template is basic *printf* format string, where %s is to print a character sequence, %d for digits and %c for single character.

methods, Maximum Entropy[2] (MaxEnt) and Conditional Random Fields[3] (CRFs) (Lafferty et al., 2001). As a baseline, we consider two different approaches. The first one is Hidden Markov Models (HMM) due to its simplistic approach for sequence labeling problems like word segmentation. We built the HMM segmentor on character tri-grams. It considers both the current and previous two characters for boundary detection. The second baseline approach is Naive Bayes. We use Peter Norvig's implementation, which was trained on word bigrams.[4]

3.2. Automatically Generating Training Data

Instead of manually segmenting thousands of hashtags for training purposes, we considered two approaches. The first one involves acquiring automatically segmented hundreds of thousands of hashtags and using them for training. For this purpose, we used the SNAP Stanford Twitter data set (Yang and Leskovec, 2011). We extracted 2.6M distinct hashtags from 476M tweets and applied simple heuristics to automatically segment those hashtags. We searched for consecutive word sequences in the SNAP tweets such that their concatenation corresponds to one of those hashtags. For 1.25M hashtags, we detected at least one word sequence. We selected the most frequent word sequence for each hashtag, if the hashtag's total occurrence is higher than 10 and if the word sequence corresponds to 75% of the total occurrences of all word sequences that correspond to that hashtag. For example, in case of *#twittermarketing* hashtag, we detected 29892 occurrences of *"twitter marketing"* and 116 occurrences of *"twittermarketing"* in the SNAP set. We ended up with 734K hashtags and their automatic segmentation, which we call it the Hashtag set.

In our second approach, we generated synthetic hashtags by concatenating the words in tweets. Since the word boundaries are known from the tweets, we can use these for training purposes. We create two sets of tweets from different sources. The first set is selected from the Stanford Sentiment Data set. The second tweet set BOUN was collected with the Twitter Search API by using the names of popular people, movies, tv shows, sports teams etc. as query.

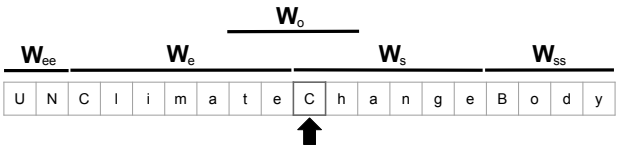

Figure 1: Longest words detected around cursor position for hashtag *#UNClimateChangeBody*

3.3. Features

Each character of the training data presents one training instance for the learning system. As we consider each char-

[2]http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html
[3]http://www.chokkan.org/software/crfsuite/
[4]http://norvig.com/ngrams/

acter, we create active features at that position and label this instance with either B or I depending on if that character is at the (B)eginning or (I)nside of the word. The most effective features are given in Table 1.

Vocabulary-based features look for known words in a given input character sequence. We create our vocabulary from the BOUN and Stanford tweet sets and hashtag set. As shown in the Figure 1, we look at five positions w.r.t. the cursor position for the longest matching words. While the existence of overlapping word (W_o) suppresses the boundary decision such as "tech" in Figure 1, word starting at cursor position (W_s) and end just before the cursor position (W_e) are positive indicators for a boundary at that cursor position. We also consider words coming before (W_{ee}) and after (W_{ss}) those words. In addition to words themselves, feature combinations are also created from their lengths (*len*($\cdot$)), log-frequencies[5] (*logFreq*($\cdot$)), their corresponding class code (*wordClass($\cdot$)*) from CMU's Twitter Word Clusters (Owoputi et al., 2012).

Bigram-based features are extension of word-based features. Word sequences like (W_e, W_s), (W_s, W_{ss}) are used to create such features. Difficulty of detecting less than 4-character length, or short words lead to features like *short-in-middle* which uses words before and after short words and *short-ngram* which considers two or more consecutive short words and longest words before and after them. We collect word bigram frequencies (*bigramFreq($\cdot$)*) from BOUN and Stanford tweets as well as Hashtag set.

Orthography-based features complement the vocabulary-based ones. Feature *orth*($\cdot$) converts characters to their orthographic shape[6] . A capitalized letter or a number around the cursor position can be a good indicator of a word boundary. Orthography-based features are especially effective when no word is detected at cursor position.

4. Experiments and Results

4.1. Test Sets and Evaluation

We use two development and test set pairs. In the first pair, Dev-BOUN and Test-BOUN (Celebi and Ozgur, 2016a) each includes 500 manually segmented hashtags randomly selected from the Tw-BOUN set. In case of the second pair, we use manually segmented hashtag set Test-Stanford which was created by Bansal et al. (2015). It includes 1268 hashtags along. We use this set as the test set. For the development, we randomly selected another set of 1000 hashtags (Dev-Stanford) from the Tw-Stanford (Celebi and Ozgur, 2016b) and manually segmented them.

Evaluation of word segmentation is usually done by measuring the percentage of the test instances that are segmented exactly as expected. We call this measure accuracy. We also argue that F_1-score can be a more appropriate evaluation metric depending on in which task the segmentor will be used. Even if not all words are detected correctly, detecting most of them might still be useful for certain end applications. The F_1-score, which takes the harmonic mean of precision and recall, awards partially correct segmentations as well. Precision is what percentage of the words outputted by the segmentor is correct, while recall measures what percentage of the actual words in the test set are identified.

4.2. Results

Varying in strengths, we consider three baseline methods, namely HMM, Norvig's Naive Bayes,and Microsoft's Word Breaker (Wang et al., 2011). In addition, adapted from our automatic training data generating approach, a fourth baseline method (SNAP n-grams) is developed as follows. Given a hashtag, we identify the sequences of words in the tweets in the SNAP data set that constitute the hashtag when concatenated. Among all possible such word sequences (n-grams), we select the one with the highest frequency as the segmentation of the hashtag. If there are no any n-grams that match with the hashtag when concatenated, the hashtag is left as it is. The results obtained by the baseline methods are shown in Table 2. While HMM, Naive Bayes, and SNAP n-grams turn out to be weak baselines, Word Breaker provides a strong baseline. Despite its state-of-the-art nature, Word Breaker was originally designed to segment URLs, not hashtags. As these results suggest, hashtags present a more challenging problem for Word Breaker. Hence, we consider it as our primary baseline.

	Test-BOUN		Test-Stanford	
System	F_1-score	Acc.	F_1-score	Acc.
HMM	74.0	69.3	63.0	64.3
Naive Bayes	63.6	57.8	70.1	68.2
SNAP n-grams	68.9	69.7	70.3	68.7
Word Breaker	**84.4**	**86.2**	**84.6**	**83.6**

Table 2: Baseline results on Test-BOUN and Test-Stanford sets.

We train the models on three different training sets, each set having different characteristics. We also experiment with five subsets of those training sets with increasing sizes. We consider 5K, 10K, 20K, 50K, and 100K tweets. In case of the Hashtag set, since it consists of hashtags instead of tweets, we match the size in terms of number of characters. We run various feature combinations on the development sets to find the best combination on each training and development set pair. Then, we run the best feature combinations on the test sets.

Table 3 lists the results for the Test-BOUN set. In general, higher F_1-score and accuracy are obtained by using the tweet training data sets with the MaxEnt classifier, compared too the hashtag training data set. The best F_1-score and accuracy are achieved with the Tw-Stanford tweet set. The results obtained with Tw-BOUN as the training set are slightly lower. Nevertheless, the best results from each training set still outperform the MS Word Breaker. Compared to Word Breaker, our best accuracy is 2 points higher. Also, the F_1-score of Word Breaker is significantly lower: 84.6% vs 92.4%. CRF performs poorly compared to Max-

[5]We use Microsoft's Web N-Gram Service.

[6]Upper-case letter is replaced with 'C', lower-case one with 'c', digit with 'd'; otherwise use the letter itself

Training Set	Method	Training Data Size									
		5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
		F_1-score					Accuracy				
Tw-BOUN	CRF	81.0	82.7	84.0	85.6	85.3	72.0	75.0	77.0	79.0	78.8
	MaxEnt	90.5	91.3	91.4	91.5	91.5	85.6	86.8	87.2	87.4	87.4
Hashtags	CRF	82.2	83.8	85.2	87.3	88.8	75.2	76.8	78.2	81.2	83.6
	MaxEnt	89.9	89.4	89.7	90.8	91.0	85.8	85.0	85.0	86.2	86.6
Tw-Stanford	CRF	82.3	84.2	83.6	83.9	84.7	75.9	77.2	76.4	77.4	78.4
	MaxEnt	92.0	92.1	92.1	**92.4**	91.8	88.0	**88.2**	87.8	**88.2**	87.6

Table 3: Best results on Test-BOUN set. Baseline (MS Word Breaker) F_1-score = 84.4%, Accuracy = 86.2%

Training Set	Method	Training Data Size									
		5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
		F_1-score					Accuracy				
Tw-BOUN	CRF	74.8	77.2	79.5	81.4	81.3	73.7	74.8	76.5	78.6	78.1
	MaxEnt	84.9	86.9	**87.0**	**87.0**	**87.0**	83.0	85.1	**85.4**	85.3	**85.4**
Hashtags	CRF	78.8	79.7	79.6	81.7	82.7	76.3	77.4	77.1	79.1	79.9
	MaxEnt	84.8	85.3	85.4	85.8	86.2	82.9	83.5	83.7	84.1	84.5
Tw-Stanford	CRF	78.1	78.4	78.3	79.3	79.2	74.4	75.1	74.7	75.7	75.8
	MaxEnt	84.4	86.4	85.8	85.9	86.7	83.9	84.2	83.5	84.8	84.8

Table 4: Best results on Test-Stanford set. Baseline (MS Word Breaker) F_1-score = 84.6%, Accuracy = 83.6%

Ent. Among the three training data sets, CRF performs better when the Hashtag data set is used.

In Table 4, results on the Test-Stanford test set are given. In this case, the best result is achieved by training on the Tw-BOUN tweet set. MaxEnt and CRF perform similarly as before. MaxEnt achieves 87% F_1-score and 85.4% accuracy. Compared to the Word Breaker's performance, our best case achieves 2.4 points higher in F_1-score and 1.8 points higher in accuracy. However, when we compare our accuracy with the 87.3% accuracy reported by (Bansal et al., 2015), ours is 1.9 points lower. This difference can be explained by the fact that context and Wikipedia relatedness based features were used in (Bansal et al., 2015), while our current approach does not use such external resources nor context-based features. In addition, Bansal et al. (2015) reported their results using cross-validation on the Test-Stanford data set. On the other hand, we used the Test-Stanford data set as a test set only and tuned our parameters using our manually created development set (Dev-Stanford). Therefore, our results are not directly comparable.

Our results show that training on the automatically extracted hashtag segmentations from the Hashtag set do not perform better than training on tweets (i.e. BOUN and Stanford tweet sets). These results suggest that hashtags may not include diverse training cases. Another reason might be that the features that we used may have not fully utilized the characteristics of the Hashtag set. These require further investigations.

5. Conclusion and Future Work

We presented a feature-rich machine learning based approach for hashtag segmentation. Instead of using manually segmented hashtags for training, we automatically generated training data sets from the hashtags and tweets in a large corpus. Our results show that promising results are obtained by using this approach. Two manually generated development and test sets each consisting of 1000 hashtags is another side contribution of this paper. As future work, instead of using only the hashtag itself, we will utilize the other tokens in the tweet that hosts that hashtag. Such context information might give clues for better segmentation.

6. Acknowledgements

This research is supported by Boğaziçi University Research Fund Grant Number 11170.

7. Bibliographical References

Bansal, P., Bansal, R., and Varma, V. (2015). Towards deep semantic analysis of hashtags. *To Appear in 37th European Conference on Information Retrieval*.

Berardi, G., Esuli, A., Marcheggiani, D., and Sebastian, F. (2011). Isti@trec microblog track 2011: exploring the use of hashtag segmentation and text quality ranking. In *Proceedings of Twentieth Text REtrieval Conference*.

Chen, S., Y., X., and Chang, H. (2012). A simple and effective unsupervised word segmentation approach. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Daelamans, W., van den Bosch, A., and Weijters, A. (1997). Igtree: Using trees for compression and classifi-

cation in lazy learning algorithms. *Artificial Intelligence Review, 11*, pages 407–423.

Dale, R., Moisl, H., and Somers, H. (2010). *Handbook of Natural Language Processing*. Marcel Dekker, Inc., New York, 2nd edition.

Islam, A., Inkpen, D., and Kiringa, I. (2007). A generalized approach to word segmentation using maximum length descending frequency and entropy rate. *Computational Linguistics and Intelligent Text Processing Lecture Notes in Computer Science Volume 4394*, pages 175–185.

Jr., G. F. (1973). The viterbi algorithm. In *Proceedings of the IEEE, 61(3)*, pages 268–278.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.

Magistry, P. and Sagot, B. (2001). Unsupervized word segmentation: the case for mandarin chinese. *In the Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 383–387.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., and Schneider, N. (2012). Part-of-speech tagging for twitter: Word clusters and other advances. Technical Report CMU-ML-12-107, Carnegie Mellon University.

Rumelhart, D. and McClelland, J. L. (1986). On learning the past tenses of english verbs. *Parallel distributed processing: explorations in the microstructure of cognition, Vol. 2, MIT Press Cambridge, MA, USA*, pages 216–271.

Srinivasan, S., Bhattacharya, S., and Chakraborty, R. (2012). Segmenting web-domains and hashtags using length specific models. *In the Proceeding CIKM 2012 Proceedings of the 21st ACM international conference on Information and knowledge, ACM New York, NY, USA*, pages 1113–1122.

Wang, K., Thrasher, C., and Hsu, B.-J. (2011). Web scale nlp: A case study on url word breaking. *In The International World Wide Web Conference*, pages 357–366.

Wong, P. K. and Chan, C. (1996). Chinese word segmentation based on maximum matching and word binding force. *In the proceedings of the 16th conference on Computational linguistics*.

Xue, N. (2003). Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing volume 8(1)*.

Yang, J. and Leskovec, J. (2011). Patterns of temporal variation in online media. *Proceedings of the fourth ACM international conference on Web Search and Data Mining*.

Zhang, Y. and Clark, S. (2008). Chinese segmentation with a word-based perceptron algorithm. *Annual Meeting of the Association of Computational Linguistics*, pages 888–896.

8. Language Resource References

Arda Celebi and Arzucan Ozgur. (2016a). *BOUN Hashtag Segmentation Evaluation Set*. TABILAB, Bogazici University, TABILAB Hashtag Segmentation Evaluation Sets, 1.0.

Arda Celebi and Arzucan Ozgur. (2016b). *STAN Hashtag Segmentation Evaluation Sets*. TABILAB, Bogazici University, TABILAB Hashtag Segmentation Evaluation Sets, 1.0.