Multi-task Pairwise Neural Ranking for Hashtag Segmentation

Mounica Maddela[1], Wei Xu[1], Daniel Preoţiuc-Pietro[2]
[1] Department of Computer Science and Engineering, The Ohio State University
[2] Bloomberg LP
{maddela.4, xu.1265}@osu.edu dpreotiucpie@bloomberg.net

Abstract

Hashtags are often employed on social media and beyond to add metadata to a textual utterance with the goal of increasing discoverability, aiding search, or providing additional semantics. However, the semantic content of hashtags is not straightforward to infer as these represent ad-hoc conventions which frequently include multiple words joined together and can include abbreviations and unorthodox spellings. We build a dataset of 12,594 hashtags split into individual segments and propose a set of approaches for hashtag segmentation by framing it as a pairwise ranking problem between candidate segmentations.[1] Our novel neural approaches demonstrate 24.6% error reduction in hashtag segmentation accuracy compared to the current state-of-the-art method. Finally, we demonstrate that a deeper understanding of hashtag semantics obtained through segmentation is useful for downstream applications such as sentiment analysis, for which we achieved a 2.6% increase in average recall on the SemEval 2017 sentiment analysis dataset.

1 Introduction

A hashtag is a keyphrase represented as a sequence of alphanumeric characters plus underscore, preceded by the # symbol. Hashtags play a central role in online communication by providing a tool to categorize the millions of posts generated daily on Twitter, Instagram, etc. They are useful in searches, tracking content about a certain topic (Berardi et al., 2011; Ozdikis et al., 2012), or discovering emerging trends (Sampson et al., 2016).

Hashtags often carry very important information, such as emotion (Abdul-Mageed and Ungar, 2017), sentiment (Mohammad et al., 2013), sarcasm (Bamman and Smith, 2015), and named entities (Finin et al., 2010; Ritter et al., 2011). However, inferring the semantics of hashtags is non-trivial since many hashtags contain multiple tokens joined together, which frequently leads to multiple potential interpretations (e.g., *lionhead* vs. *lion head*). Table 1 shows few examples of single- and multi-token hashtags. While most hashtags represent a mix of standard tokens, named entities and event names are prevalent and pose challenges to both human and automatic comprehension, as these are more likely to be rare tokens. Hashtags also tend to be shorter to allow fast typing, to attract attention or to satisfy length limitations imposed by some social media platforms (e.g., Twitter). Thus, they tend to contain a large number of abbreviations or non-standard spelling variations (e.g., *#iloveu4eva*) (Han and Baldwin, 2011; Eisenstein, 2013), which hinders their understanding.

Type	Single-token	Multi-token
Named-entity (33.0%)	*#lionhead*	*#toyotaprius*
Events (14.8%)	*#oscars*	*#ipv6summit*
Standard (43.6%)	*#snowfall*	*#epicfall*
Non-standard (11.2%)	*#nlproc*	*#iloveu4eva*

Table 1: Examples of single- (47.1%) and multi-word hashtags (52.9%) and their categorizations based on a sample of our data.

The goal of our study is to build efficient methods for automatically splitting an English hashtag into a meaningful word sequence. Our contributions are:

- A larger and better curated dataset for this task;
- Framing the problem as pairwise ranking using novel neural approaches, in contrast to previous work which ignored the relative order;
- A multi-task learning method that uses different sets of features to handle different types of hashtags;

[1]Our toolkit along with the code and data are publicly available at https://github.com/mounicam/hashtag_master

- Experiments demonstrating that hashtag segmentation improves sentiment analysis on a benchmark dataset.

Our new dataset includes segmentation for 12,594 unique hashtags and their associated tweets annotated in a multi-step process for higher quality than the previous dataset of 1,108 hashtags (Bansal et al., 2015). We frame the segmentation task as a pairwise ranking problem, given a set of candidate segmentations. We build several neural architectures using this problem formulation which use corpus-based, linguistic and thesaurus based features. We further propose a multi-task learning approach which jointly learns segment ranking and single- vs. multi-token hashtag classification. The latter leads to an error reduction of 24.6% over the current state-of-the-art. Finally, we demonstrate the utility of our method by using hashtag segmentation in the downstream task of sentiment analysis. Feeding the automatically segmented hashtags to a state-of-the-art sentiment analysis method on the SemEval 2017 benchmark dataset results in a 2.6% increase in the official metric for the task.

2 Background and Preliminaries

Current approaches for hashtag segmentation can be broadly divided into three categories: (a) gazeteer and rule based (Maynard and Greenwood, 2014; Declerck and Lendvai, 2015; Billal et al., 2016), (b) word boundary detection (Çelebi and Özgür, 2017, 2016), and (c) ranking with language model and other features (Wang et al., 2011; Bansal et al., 2015; Berardi et al., 2011; Reuter et al., 2016; Simeon et al., 2016). Hashtag segmentation approaches draw upon work on compound splitting for languages such as German or Finnish (Koehn and Knight, 2003) and word segmentation (Peng and Schuurmans, 2001) for languages with no spaces between words such as Chinese (Sproat and Shih, 1990; Xue and Shen, 2003). Similar to our work, Bansal et al. (2015) extract an initial set of candidate segmentations using a sliding window, then rerank them using a linear regression model trained on lexical, bigram and other corpus-based features. The current state-of-the-art approach (Çelebi and Özgür, 2017, 2016) uses maximum entropy and CRF models with a combination of language model and hand-crafted features to predict if each character in the hashtag is the beginning of a new word.

Microsoft Word Breaker (Wang et al., 2011) is, among the existing methods, a strong baseline for hashtag segmentation, as reported in Çelebi and Özgür (2017) and Bansal et al. (2015). It employs a beam search algorithm to extract k best segmentations as ranked by the n-gram language model probability:

$$Score^{LM}(s) = \sum_{i=1}^{n} \log P(w_i | w_{i-N+1} \dots w_{i-1})$$

where $[w_1, w_2 \dots w_n]$ is the word sequence of segmentation s and N is the window size. More sophisticated ranking strategies, such as Binomial and word length distribution based ranking, did not lead to a further improvement in performance (Wang et al., 2011). The original Word Breaker was designed for segmenting URLs using language models trained on web data. In this paper, we reimplemented[2] and tailored this approach to segmenting hashtags by using a language model specifically trained on Twitter data (implementation details in §3.6). The performance of this method itself is competitive with state-of-the-art methods (evaluation results in §5.3). Our proposed pairwise ranking method can effectively take the top k segmentations generated by this baseline as candidates.

However, in prior work, the ranking scores of each segmentation were calculated independently, ignoring the relative order among the top k candidate segmentations. To address this limitation, we utilize a novel pairwise ranking strategy and propose the first neural model for this task.

3 Multi-task Pairwise Neural Ranking

We propose a multi-task pairwise neural ranking approach to better incorporate and distinguish the relative order between the candidate segmentations of a given hashtag. Our model adapts to address single and multi-token hashtags differently via a multi-task learning strategy without requiring additional annotations. In this section, we describe three variants of our pairwise neural ranking models (Figure 1).

3.1 Segmentation as Pairwise Ranking

The goal of hashtag segmentation is to divide a given hashtag h into a sequence of meaningful words $s^* = [w_1, w_2, \dots, w_n]$. For a hashtag of

[2]To the best of our knowledge, Microsoft discontinued its Word Breaker and Web Ngram API services in early 2018.

hashtag (h)	*#songsonghaddafisitunes*
segmentation (s^*)	*songs on ghaddafi s itunes* (i.e. *songs on Ghaddafi's iTunes*)
candidate segmentations ($s \in S$)	*songs on ghaddafis itunes*
	songs on ghaddafisi tunes
	songs on ghaddaf is itunes
	song song haddafis i tunes
	songsong haddafisitunes
	(and . . .)

Table 2: Example hashtag along with its gold and possible candidate segmentations.

r characters, there are a total of 2^{r-1} possible segmentations but only one, or occasionally two, of them (s^*) are considered correct (Table 2).

We transform this task into a pairwise ranking problem: given k candidate segmentations $\{s_1, s_2, \ldots, s_k\}$, we rank them by comparing each with the rest in a pairwise manner. More specifically, we train a model to predict a real number $g(s_a, s_b)$ for any two candidate segmentations s_a and s_b of hashtag h, which indicates s_a is a better segmentation than s_b if positive, and vice versa. To quantify the quality of a segmentation in training, we define a *gold* scoring function g^* based on the similarities with the ground-truth segmentation s^*:

$$g^*(s_a, s_b) = sim(s_a, s^*) - sim(s_b, s^*).$$

We use the Levenshtein distance (minimum number of single-character edits) in this paper, although it is possible to use other similarity measurements as alternatives. We use the top k segmentations generated by Microsoft Word Breaker (§2) as initial candidates.

3.2 Pairwise Neural Ranking Model

For an input candidate segmentation pair $\langle s_a, s_b \rangle$, we concatenate their feature vectors $\mathbf{s}_a$ and $\mathbf{s}_b$, and feed them into a feedforward network which emits $g(s_a, s_b)$. The feature vector $\mathbf{s}_a$ or $\mathbf{s}_b$ consists of language model probabilities using Good-Turing (Good, 1953) and modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1999), lexical and linguistic features (more details in §3.5). For training, we use all the possible pairs $\langle s_a, s_b \rangle$ of the k candidates as the input and their *gold* scores $g^*(s_a, s_b)$ as the target. The training objective is to minimize the Mean Squared Error (MSE):

$$L_{MSE} = \frac{1}{m} \sum_{i=1}^{m} (g^{*(i)}(s_a, s_b) - \hat{g}^{(i)}(s_a, s_b))^2 \quad (1)$$

where m is the number of training examples.

To aggregate the pairwise comparisons, we follow a greedy algorithm proposed by Cohen et al. (1998) and used for preference ranking (Parakhin and Haluptzok, 2009). For each segmentation s in the candidate set $S = \{s_1, s_2, \ldots, s_k\}$, we calculate a single score $Score^{PNR}(s) = \sum_{s \neq s_j \in S} g(s, s_j)$, and find the segmentation s_{max} corresponding to the highest score. We repeat the same procedure after removing s_{max} from S, and continue until S reduces to an empty set. Figure 1(a) shows the architecture of this model.

3.3 Margin Ranking (MR) Loss

As an alternative to the pairwise ranker (§3.2), we propose a pairwise model which learns from candidate pairs $\langle s_a, s_b \rangle$ but ranks each individual candidate directly rather than relatively. We define a new scoring function g' which assigns a higher score to the better candidate, i.e., $g'(s_a) > g'(s_b)$, if s_a is a better candidate than s_b and vice-versa. Instead of concatenating the features vectors $\mathbf{s}_a$ and $\mathbf{s}_b$, we feed them separately into two identical feedforward networks with shared parameters. During testing, we use only one of the networks to rank the candidates based on the g' scores. For training, we add a ranking layer on top of the networks to measure the violations in the ranking order and minimize the Margin Ranking Loss (MR):

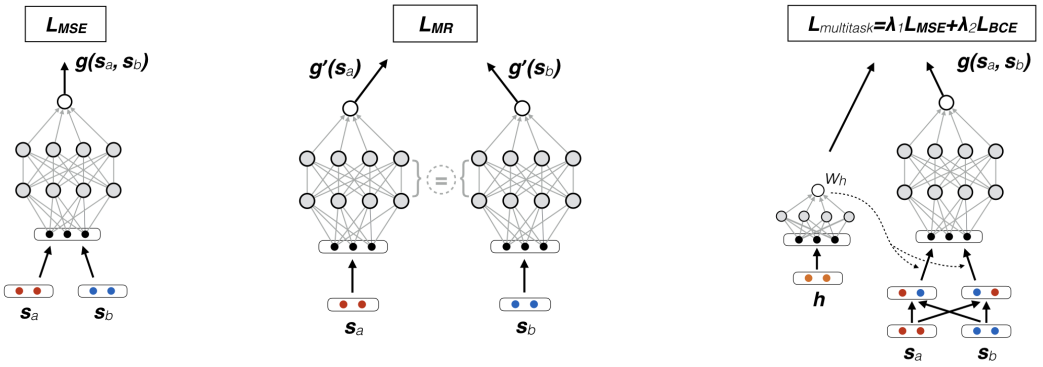
$$L_{MR} = \frac{1}{m} \sum_{i=1}^{m} \max(0, 1 - l_{ab}^{(i)}(\hat{g}'^{(i)}(s_a) - \hat{g}'^{(i)}(s_b))$$

$$l_{ab} = \begin{cases} 1 & g^*(s_a, s_b) > 0 \\ -1 & g^*(s_a, s_b) < 0 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where m is the number of training samples. The architecture of the model is presented in Figure 1(b).

3.4 Adaptive Multi-task Learning

Both models in §3.2 and §3.3 treat all the hashtags uniformly. However, different features address different types of hashtags. By design, the linguistic features capture named entities and multi-word hashtags that exhibit word shape patterns, such as camel case. The ngram probabilities with Good-Turing smoothing gravitate towards multi-word segmentations with known words, as its estimate for unseen ngrams depends on the fraction of ngrams seen once which can be very



(a) Pairwise Ranking Model (**MSE** §3.2)

(b) Margin Ranking Loss w/ shared parameters (**MR** §3.3)

(c) Adaptive Multi-task Learning for Pairwise ranking (**MSE+Multitask** §3.4)

Figure 1: Pairwise neural ranking models for hashtag segmentation. Given two candidate segmentations s_a and s_b of hashtag h, the goal is to predict the segmentation's *goodness* relative score (g) or absolute (g') score.

low (Heafield, 2013). The modified Kneser-Ney smoothing is more likely to favor segmentations that contain rare words, and single-word segmentations in particular. See §5.3 for a more detailed quantitative and qualitative analysis.

To leverage this intuition, we introduce a binary classification task to help the model differentiate single-word from multi-word hashtags. The binary classifier takes hashtag features $\mathbf{h}$ as the input and outputs w_h, which represents the probability of h being a multi-word hashtag. w_h is used as an adaptive gating value in our multi-task learning setup. The gold labels for this task are obtained with no extra cost by simply verifying whether the ground-truth segmentation has multiple words. We train the pairwise segmentation ranker and the binary single- vs. multi-token hashtag classifier jointly, by minimizing L_{MSE} for the pairwise ranker and the Binary Cross Entropy Error (L_{BCE}) for the classifier:

$$\begin{aligned} L_{multitask} &= \lambda_1 L_{MSE} + \lambda_2 L_{BCE} \\ L_{BCE} &= -\frac{1}{m}\sum_{i=1}^{m}\big[l^{(i)} * log(w_h^{(i)}) + \\ &\quad (1-l^{(i)}) * log(1-w_h^{(i)})\big] \end{aligned} \quad (3)$$

where w_h is the adaptive gating value, $l \in \{0, 1\}$ indicates if h is actually a multi-word hashtag and m is the number of training examples. λ_1 and λ_2 are the weights for each loss. For our experiments, we apply equal weights.

More specifically, we divide the segmentation feature vector $\mathbf{s}_a$ into two subsets: (a) $\mathbf{s}_a^{KN}$ with modified Kneser-Ney smoothing features, and (b) $\mathbf{s}_a^{GL}$ with Good-Turing smoothing and linguistic features. For an input candidate segmentation pair $\langle s_a, s_b \rangle$, we construct two pairwise vectors $\mathbf{s}_{ab}^{KN} = [\mathbf{s}_a^{KN}; \mathbf{s}_b^{KN}]$ and $\mathbf{s}_{ab}^{GL} = [\mathbf{s}_a^{GL}; \mathbf{s}_b^{GL}]$ by concatenation, combine them based on the adaptive gating value w_h before feeding them into the feedforward network G for pairwise ranking:

$$\hat{g}(s_a, s_b) = G\left(w_h \mathbf{s}_{ab}^{GL} + (1 - w_h)\mathbf{s}_{ab}^{KN}\right). \quad (4)$$

We use summation with padding, as it has been shown to work similarly to more complex strategies such as multi-column networks (Ciresan et al., 2012). Figure 1(c) shows the architecture of this model. An analogue multi-task formulation can also be built for the Margin Ranking loss as:

$$L_{multitask} = \lambda_1 L_{MR} + \lambda_2 L_{BCE}. \quad (5)$$

3.5 Features

We use a combination of corpus-based and linguistic features to rank the segmentations. For a candidate segmentation s, its feature vector $\mathbf{s}$ includes the number of words in the candidate, the length of each word, the proportion of words in an English dictionary[3] or Urban Dictionary[4] (Nguyen et al., 2018), ngram counts from Google Web 1TB corpus (Brants and Franz, 2006), and ngram probabilities from trigram language models trained on the Gigaword corpus (Graff and Cieri, 2003) and 1.1 billion English tweets from 2010, respectively. We train two language models on each corpus: one with Good-Turing smoothing using SRILM (Stolcke, 2002) and the other with modified Kneser-Ney smoothing using KenLM (Heafield, 2011).

[3] https://pypi.org/project/pyenchant
[4] https://www.urbandictionary.com

We also add boolean features, such as if the candidate is a named-entity present in the list of Wikipedia titles, and if the candidate segmentation s and its corresponding hashtag h satisfy certain word-shapes (§A.1).

Similarly, for hashtag h, we extract the feature vector $\mathbf{h}$ consisting of hashtag length, ngram count of the hashtag in Google 1TB corpus (Brants and Franz, 2006), and boolean features indicating if the hashtag is in an English dictionary or Urban Dictionary, is a named-entity, is in camel case, ends with a number, and has all the letters as consonants. We also include features of the best-ranked candidate by the Word Breaker model.

3.6 Implementation Details

We use the PyTorch framework to implement our multi-task pairwise ranking model. The pairwise ranker consists of an input layer, three hidden layers with eight nodes in each layer and hyperbolic tangent ($tanh$) activation, and a single linear output node. The auxiliary classifier consists of an input layer, one hidden layer with eight nodes and one output node with sigmoid activation. We use the Adam algorithm (Kingma and Ba, 2014) for optimization and apply a dropout of 0.5 to prevent overfitting. We set the learning rate to 0.01 and 0.05 for the pairwise ranker and auxiliary classifier respectively. For each experiment, we report results obtained after 100 epochs.

For the baseline model used to extract the k initial candidates, we reimplemented the Word Breaker (Wang et al., 2011) as described in §2 and adapted it to use a language model trained on 1.1 billion tweets with Good-Turing smoothing using SRILM (Stolcke, 2002) to give a better performance in segmenting hashtags (§5.3). For all our experiments, we set $k = 10$.

4 Hashtag Segmentation Data

We use two datasets for experiments (Table 3): *STAN*$_{small}$, created by Bansal et al. (2015), which consists of 1,108 unique hashtags from 1,268 randomly selected tweets in the Stanford Sentiment Analysis Dataset (Go and Huang, 2009) along with their crowdsourced segmentations; and *STAN*$_{large}$, our new expert curated dataset, which includes all 12,594 unique English hashtags and their associated tweets from the same Stanford dataset.

	Data	num. of Hashtags (multi-token%)	avg. #char	avg. #word
STAN$_{large}$	Train	2518 (51.9%)	8.5	1.8
	Dev	629 (52.3%)	8.4	1.7
	Test	9447 (53.0%)	8.6	1.8
STAN$_{small}$	Test	1108 (60.5%)	9.0	1.9

Table 3: Statistics of the *STAN*$_{small}$ and *STAN*$_{large}$ datasets – number of unique hashtags, percentage of multi-word hashtags, average length of hashtags in characters and words.

Dataset analysis. *STAN*$_{small}$ is the most commonly used dataset in previous work. However, after re-annotation, we found annotation errors in 6.8%[5] of the hashtags in this dataset, which is significant given that the error rate of the state-of-the-art models is only around 10%. Most of the errors were related to named entities. For example, *#lionhead*, which refers to the "Lionhead" video game company, was labeled as "*lion head*".

Our dataset. We therefore constructed the *STAN*$_{large}$ dataset of 12,594 hashtags with additional quality control for human annotations. We displayed a tweet with one highlighted hashtag on the Figure-Eight[6] (previously known as CrowdFlower) crowdsourcing platform and asked two workers to list all the possible segmentations. For quality control on the platform, we displayed a test hashtag in every page along with the other hashtags. If any annotator missed more than 20% of the test hashtags, then they were not allowed to work on the task. For 93.1% of the hashtags, the crowdsourced workers agreed on the same segmentation. We asked three in-house annotators (not authors) to cross-check the crowdsourced annotations using a two-step procedure: first, verify if the hashtag was a named entity based on the context of the tweet; then search on Google to find the correct segmentation(s). We also asked the same annotators to fix the errors in *STAN*$_{small}$. The human upperbound of the task is estimated at ∼98% accuracy, where we consider the crowdsourced segmentations (two workers merged) as correct if at least one of them matches with our expert annotator's segmentations.

5 Experiments

In this section, we present experimental results that compare our proposed method with the other

[5]More specifically, 4.8% hashtags is missing one of the two acceptable segmentations and another 2.0% is incorrect segmentation.

[6]https://figure-eight.com

	All Hashtags				Multi-token				Single-token		
	A@1	**F_1@1**	**A@2**	**MRR**	**A@1**	**F_1@1**	**A@2**	**MRR**	**A@1**	**A@2**	**MRR**
Original hashtag	51.0	51.0	–	–	19.1	19.1	–	–	100.0	–	–
Rule-based (Billal et al., 2016)	58.1	63.5	–	–	57.6	66.5	–	–	58.8	–	–
GATE Hashtag Tokenizer (M&G, 2014)	73.2	77.2	–	–	71.4	78.0	–	–	76.0	–	–
Viterbi (Berardi et al., 2011)	73.4	78.5	–	–	74.5	83.1	–	–	71.6	–	–
MaxEnt (Çelebi and Özgür, 2017)	92.4	93.4	–	–	91.9	93.6	–	–	93.1	–	–
Word Breaker w/ Twitter LM	90.8	91.7	97.4	94.5	88.5	90.0	97.8	93.7	94.3	96.8	95.7
Pairwise linear ranker	88.1	89.9	97.2	93.1	83.8	86.8	97.3	91.3	94.7	**97.0**	95.9
Pairwise neural ranker (MR)	92.3	93.5	98.2	95.4	90.9	92.8	99.0	95.2	94.5	96.9	95.8
Pairwise neural ranker (MSE)	92.5	93.7	98.2	95.5	91.2	93.1	99.0	95.4	94.5	**97.0**	95.8
Pairwise neural ranker (MR+multitask)	93.0	94.3	97.8	95.7	91.5	93.7	98.7	95.4	95.2	96.6	96.0
Pairwise neural ranker (MSE+multitask)	**94.5**	**95.2**	**98.4**	**96.6**	**93.9**	**95.1**	**99.4**	**96.8**	**95.4**	96.8	**96.2**
Human Upperbound	98.0	98.3	–	–	97.8	98.2	–	–	98.4	–	–

Table 4: Evaluation results on the corrected version of *STAN$_{small}$*. For reference, with the original version of *STAN$_{small}$*, the official Microsoft Word Breaker API reports an 84.6% F_1 score and an 83.6% accuracy for the top one output (Çelebi and Özgür, 2017), while our best model (MSE+multitask) achieves 89.8% F_1 and 91.0% accuracy.

	All Hashtags				Multi-token				Single-token		
	A@1	**F_1@1**	**A@2**	**MRR**	**A@1**	**F_1@1**	**A@2**	**MRR**	**A@1**	**A@2**	**MRR**
Original hashtag	55.5	55.5	–	–	16.2	16.2	–	–	100.0	–	–
Rule-based (Billal et al., 2016)	56.1	61.5	–	–	56.0	65.8	–	–	56.3	–	–
Viterbi (Berardi et al., 2011)	68.4	73.8	–	–	71.2	81.5	–	–	65.0	–	–
GATE Hashtag Tokenizer (M&G, 2014)	72.4	76.1	–	–	70.0	76.8	–	–	75.3	–	–
MaxEnt (Çelebi and Özgür, 2017)	91.2	92.3	–	–	90.2	92.4	–	–	92.3	–	–
Word Breaker w/ Twitter LM	90.1	91.0	96.6	93.9	88.5	90.0	97.0	93.4	91.9	96.2	94.4
Pairwise linear ranker	89.2	91.1	96.3	93.3	84.2	87.8	95.6	91.0	**94.8**	**97.0**	**95.9**
Pairwise neural ranker (MR)	91.3	92.6	97.2	94.6	89.9	92.4	97.5	94.3	92.8	96.8	94.9
Pairwise neural ranker (MSE)	91.3	92.6	97.0	94.5	91.0	93.6	97.7	94.9	91.5	96.2	94.1
Pairwise neural ranker (MR+multitask)	91.4	92.7	97.2	94.6	90.0	92.6	97.7	94.4	92.9	96.6	94.9
Pairwise neural ranker (MSE+multitask)	**92.4**	**93.6**	**97.3**	**95.2**	**91.9**	**94.1**	**98.0**	**95.4**	93.0	96.5	94.9
Human Upperbound	98.6	98.8	–	–	98.0	98.4	–	–	99.2	–	–

Table 5: Evaluation results on our *STAN$_{large}$ test* dataset. For single-token hashtags, the token-level F_1@1 is the same as the segmentation-level A@1. For multi-token cases, A@1 and F_1@1 for the Original hashtag baseline are non-zero because 11.4% of the test hashtags had more than one acceptable segmentations. Our best model (MSE+multitask) shows a statistically significant improvement ($p < 0.05$) over the state-of-the-art approach (Çelebi and Özgür, 2017) based on the paired bootstrap test (Berg-Kirkpatrick et al., 2012).

state-of-the-art approaches on hashtag segmentation datasets. The next section will show experiments of applying hashtag segmentation to the popular task of sentiment analysis.

5.1 Existing Methods

We compare our pairwise neural ranker with the following baseline and state-of-the-art approaches:

(a) The **original hashtag** as a single token;

(b) A **rule-based** segmenter, which employs a set of word-shape rules with an English dictionary (Billal et al., 2016);

(c) A **Viterbi** model which uses word frequencies from a book corpus[7] (Berardi et al., 2011);

(d) The specially developed **GATE Hashtag Tokenizer** from the open source toolkit,[8] which combines dictionaries and gazetteers in a Viterbi-like algorithm (Maynard and Greenwood, 2014);

(e) A maximum entropy classifier (**MaxEnt**) trained on the *STAN$_{large}$* training dataset. It predicts whether a space should be inserted at each position in the hashtag and is the current state-of-the-art (Çelebi and Özgür, 2017);

(f) Our reimplementation of the **Word Breaker** algorithm which uses beam search and a Twitter ngram language model (Wang et al., 2011);

(g) A **Pairwise linear ranker** which is a basically a linear perceptron learnt using pairwise ranking optimization (PRO) algorithm (Hopkins and May, 2011) by minimizing the hinge loss between g^* and a scoring function similar to g'. It uses all the features in §3.5 and is trained on the *STAN$_{large}$* training dataset.

[7] Project Gutenberg http://norvig.com/big.txt

[8] https://gate.ac.uk/

	Single		Multi		All	
	A	**MRR**	**A**	**MRR**	**A**	**MRR**
Kneser-Ney	**95.4**	**95.7**	56.0	75.3	74.9	85.1
Good-Turing (GT)	91.4	93.5	85.9	91.8	88.6	92.6
Linguistic (Ling)	89.4	91.7	71.6	82.6	80.1	87.0
GT + Ling	92.4	93.9	**86.2**	**92.3**	**88.9**	**92.7**
All Features	91.1	93.1	89.0	93.7	90.0	93.4

Table 6: Evaluation of automatic hashtag segmentation (MSE) with different features on the $STAN_{large}$ *dev* set. **A** denotes accuracy@1. While Kneser-Ney features perform well on single-token hashtags, GT + Ling features perform well on multi-token hashtags.

5.2 Evaluation Metrics

We evaluate the performance by the top k ($k = 1, 2$) accuracy, average token-level F_1 score, and mean reciprocal rank (MRR). In particular, the accuracy and MRR are calculated at the segmentation-level, which means that an output segmentation is considered correct if and only if it fully matches the human segmentation. We also report the token-level F_1 score, which is a micro-average across all the hashtags and accounts for partially correct segmentation in the multi-word hashtag cases. This is only computed on the top candidates (F_1@1).

5.3 Results

Tables 4 and 5 show the results on the $STAN_{small}$ and $STAN_{large}$ datasets, respectively. All of our pairwise neural rankers are trained on the 2,518 manually segmented hashtags in the training set of $STAN_{large}$ and perform favorably against other state-of-the-art approaches. Our best model (MSE+multitask) that utilizes different features adaptively via a multi-task learning procedure is shown to perform better than simply combining all the features together (MR and MSE). We highlight the 24.6% error reduction on $STAN_{small}$ and 16.5% on $STAN_{large}$ of our approach over the previous SOTA (Çelebi and Özgür, 2017) on the **Multi-token** hashtags, and the importance of having a separate evaluation of multi-word cases as it is trivial to obtain 100% accuracy for **Single-token** hashtags. While it is possible to achieve a very high accuracy@2 to make our hashtag segmentation tool practically useful, it remains a challenge to get the top one predication close to perfect accuracy.

The improved Word Breaker with our addition of a Twitter-specific language model is a very strong baseline, which echos the findings of the original Word Breaker paper (Wang et al., 2011) that having a large in-domain language model is extremely helpful for word segmentation tasks. It is worth noting that the other state-of-the-art system (Çelebi and Özgür, 2017) also utilized a 4-gram language model trained on 476 million tweets from 2009.

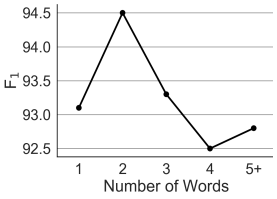
Kneser-Ney	Good-Turing	Linguistic	count	Example Hashtags
○	○	○	31	*#omnomnom #BTVSMB*
●	○	○	13	*#commbank #mamapedia*
○	●	○	38	*#wewantmcfly #winebarsf*
○	○	●	24	#cfp09 *#TechLunchSouth*
●	●	○	44	*#twittographers #bringback*
●	○	●	16	*#iccw #ecom09*
○	●	●	53	*#LetsGoPens #epicwin*
●	●	●	420	*#prototype #newyork*

Table 7: Error (○) and correct (●) segmentation analysis of three pairwise ranking models (MSE) trained with different feature sets Each row corresponds to one area in the Venn diagram; for example, ○○○ is the set of hashtags that all three models failed in the $STAN_{large}$ *dev* data and ●○○ is the set of hashtags that only the model with Kneser-Ney language model features (but not the other two models) segmented correctly.

5.4 Analysis and Discussion

Feature Analysis. To empirically illustrate the effectiveness of different features on different types of hashtags and our motivation for multi-task learning, we show the results for models using individual feature sets in pairwise ranking models (MSE) in Table 6. Language models with modified Kneser-Ney smoothing perform best on single-token hashtags, while Good-Turing and Linguistic features work best on multi-token hashtags, confirming our intuition about their usefulness in a multi-task learning approach. Table 7 shows a qualitative analysis with the first column (○○○) indicating which features lead to correct or wrong segmentations, their count in our data and illustrative examples with human segmentation. Some hashtags are very challenging e.g., *#BTVSMB* refers to the Social Media Breakfast (SMB) in Burlington, Vermont (BTV).

Length of Hashtags. As expected, longer hashtags with >3 words pose greater challenges and the accuracy of our best model (MSE+multitask) drops to 82.1%. For many error cases, our model predicts a close-to-correct segmentation, e.g., *#youknowyouupttooearly*,



Type	num. of Hashtags
single	4426 (47.1%)
2 tokens	3436 (36.2%)
3 tokens	1085 (11.2%)
4 tokens	279 (2.9%)
5+ tokens	221 (2.6%)

Figure 2: Token-level F_1 scores (MSE+multitask) on hashtags of different lengths in the $STAN_{large}$ *test* set.

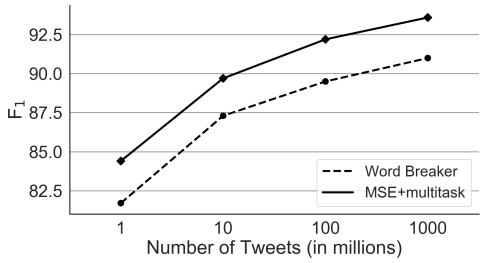


Figure 3: Token-level F_1 scores of hashtags in the $STAN_{large}$ *test* set with respect to the size of the language model used to train our pairwise ranker (MSE+multitask) and Word Breaker.

#iseelondoniseefrance, which is also reflected by the higher token-level F_1 scores across hashtags with different lengths in Figure 2.

Size of the Language Model. Since our approach heavily relies on Twitter language model, we experimented with its size. We observed that the drop in F_1 score for our pairwise neural ranker is only 1.4% and 3.9% when trained on language models containing 10% and 1% of the original 1.1 billion tweets respectively (Figure 3). Therefore, our approach performs well even with limited number of tweets.

Time Sensitivity. Language used in Twitter evolves over time (Eisenstein, 2013) but our pairwise ranker uses a language model trained on the tweets from the year 2010. To demonstrate that our approach is time-independent, we show the results on 500 random English hashtags from the year 2019 (Table 8). With an accuracy of 94.6%, the performance is very similar to the $STAN_{small}$ and $STAN_{large}$ datasets.

	A@1	F_1	MRR
Word Breaker w/ Twitter LM	92.1	93.9	94.7
Pairwise neural ranker (MSE+multitask)	**94.6**	**95.6**	**96.7**

Table 8: Evaluation results on 500 random hashtags from the year 2019. The results are very similar to the $STAN_{large}$ dataset from the year 2009.

6 Extrinsic Evaluation: Twitter Sentiment Analysis

We demonstrate the effectiveness of our hashtag segmentation system towards improving sentiment analysis in Twitter (Pang et al., 2002; Nakov et al., 2016; Rosenthal et al., 2017). We use our best model (MSE+multitask), under the name HashtagMaster, in the following experiments.

6.1 Experiment Setup

We compare the performance of the **BiLSTM+Lex** (Teng et al., 2016) sentiment analysis model under three configurations: (a) tweets with hashtags removed, (b) tweets with hashtags as single tokens excluding the # symbol, and (c) tweets with segmented hashtags by our system, HashtagMaster. BiLSTM+Lex is a state-of-the-art open source system for predicting tweet-level sentiment. It learns a context-sensitive sentiment intensity score by leveraging a Twitter-based sentiment lexicon (Tang et al., 2014). We use the same settings as described by Teng et al. (2016) to train the model.

We use the dataset from the *Sentiment Analysis in Twitter* shared task (subtask A) at SemEval 2017 (Rosenthal et al., 2017). [9] Given a tweet, the goal is to predict whether it expresses *POSITIVE*, *NEGATIVE* or *NEUTRAL* sentiment. The training and development sets consist of 49,669 tweets and we use 40,000 for training and the rest for development. There are a total of 4,840 tweets containing 12,128 hashtags in the SemEval 2017 test set, and our hashtag segmenter ended up splitting 6,975 of those hashtags present in 3,384 tweets.

6.2 Results and Analysis

In Table 9, we report the results based on the 3,384 tweets where HashtagMaster predicted a split, as for the rest of tweets in the test set, the hashtag segmenter would neither improve nor worsen the sentiment prediction. Our hashtag segmenter successfully improved the sentiment analysis performance by 2% on average recall and F_1^{PN} comparing to having hashtags unsegmented. This improvement is seemingly small but decidedly important for tweets where sentiment-related information is embedded in multi-word hashtags and sentiment prediction would be incorrect based

[9] We did not use the Stanford Sentiment Analysis Dataset (Go and Huang, 2009) because of its noisy sentiment labels obtained using distant supervision.

	AvgR	F_1^{PN}	Acc
Original tweets	**61.7**	60.0	58.7
− No Hashtags	**60.2**	58.8	54.2
+ Single-word	**62.3**	60.3	58.6
+ HashtagMaster	**64.3**	62.4	58.6

Table 9: Sentiment analysis evaluation on the 3384 tweets from SemEval 2017 test set using the BiLSTM+Lex method (Tang et al., 2014). Average recall (AvgR) is the official metric of the SemEval task and is more reliable than accuracy (Acc). F_1^{PN} is the average F_1 of positive and negative classes. Having the hashtags segmented by our system **HashtagMaster = (MSE+multitask)** significantly improves the sentiment prediction than not ($p < 0.05$ for AvgR and F_1^{PN} against the single-word setup).

Ofcourse** #**clown**shoes #alt**right** #Illinois**Nazis
#Finally**At**peace**With people calling me "Kim **Fatty** the Third"
*Leslie Odom Jr. sang that. #**Thank**YouObama*
*After some 4 months of vegetarianism .. it's all the same industry. #**cut**outthe**crap***

Table 10: Sentiment analysis examples where our hashtag segmentation tool helped. Red words and blue words are negative and positive entries in the Twitter sentiment lexicon (Tang et al., 2014), respectively.

only on the text (see Table 10 for examples). In fact, 2,605 out of the 3,384 tweets have multi-word hashtags that contain words in the Twitter-based sentiment lexicon (Tang et al., 2014) and 125 tweets contain sentiment words only in the hashtags but not in the rest of the tweet.

7 Other Related Work

Automatic hashtag segmentation can improve the performance of many applications besides sentiment analysis, such as text classification (Bilal et al., 2016), named entity linking (Bansal et al., 2015) and to model user interests for recommendations (Chen et al., 2016). It can also help in collecting data of higher volume and quality by providing a more nuanced interpretation of its content, as shown for emotion analysis (Qadir and Riloff, 2014) and sarcasm or irony detection (Maynard and Greenwood, 2014; Huang et al., 2018). Better semantic analysis can also potentially be applied in using hashtags as distant supervision labels in training classifiers for tasks such as sarcasm (Bamman and Smith, 2015), sentiment (Mohammad et al., 2013), and emotions (Abdul-Mageed and Ungar, 2017) and, more generally, as labels for pre-training representations of words (Weston et al., 2014), sentences (Dhingra et al., 2016), or images (Mahajan et al., 2018).

8 Conclusion

We proposed a new pairwise neural ranking model for hashtag segmention and showed significant performance improvement over the state-of-the-art. We also constructed a large and more accurate dataset for analyzing and benchmarking hashtag segmentation methods. We demonstrated that hashtag segmentation helps with downstream tasks like sentiment analysis. Although we focused on English hashtags, our pairwise ranking approach is language-independent and we intend to extend our toolkit to languages other than English as future work.

Acknowledgments

We thank Ohio Supercomputer Center (Center, 2012) for computing resources and the NVIDIA for providing GPU hardware. We thank Alan Ritter, Quanze Chen, Wang Ling, Pravar Mahajan, and Dushyanta Dhyani for valuable discussions. We also thank the annotators: Sarah Flanagan, Kaushik Mani, and Aswathnarayan Radhakrishnan. This material is based in part on research sponsored by the NSF under grants IIS-1822754 and IIS-1755898, DARPA through the ARO under agreement number W911NF-17-C-0095, through a Figure-Eight (CrowdFlower) AI for Everyone Award and a Criteo Faculty Research Award to Wei Xu. The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements of the U.S. Government.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. Emonet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, ACL, pages 718–728.

David Bamman and Noah A Smith. 2015. Contextualized Sarcasm Detection on Twitter. In *Ninth International AAAI Conference on Web and Social Media*, ICWSM, pages 574–577.

Piyush Bansal, Romil Bansal, and Vasudeva Varma. 2015. Towards Deep Semantic Analysis of Hashtags. In *Proceedings of the 37th European Conference on Information Retrieval*, ECIR, pages 453–464.

Giacomo Berardi, Andrea Esuli, Diego Marcheggiani, and Fabrizio Sebastiani. 2011. ISTI@TREC Mi-

croblog Track 2011: Exploring the Use of Hashtag Segmentation and Text Quality Ranking. In *Text REtrieval Conference (TREC)*.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An Empirical Investigation of Statistical Significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL, pages 995–1005.

Belainine Billal, Alexsandro Fonseca, and Fatiha Sadat. 2016. Named Entity Recognition and Hashtag Decomposition to Improve the Classification of Tweets. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, COLING, pages 102–111.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1. *Linguistic Data Consortium (LDC)*.

Arda Çelebi and Arzucan Özgür. 2016. Segmenting Hashtags using Automatically Created Training Data. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, LREC, pages 2981–2985.

Arda Çelebi and Arzucan Özgür. 2017. Segmenting Hashtags and Analyzing Their Grammatical Structure. *Journal of Association For Information Science and Technology (JASIST)*, 69(5):675–686.

Ohio Supercomputer Center. 2012. Oakley supercomputer. http://osc.edu/ark:/19495/hpc0cvqn.

Stanley F Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394.

Tao Chen, Xiangnan He, and Min-Yen Kan. 2016. Context-aware Image Tweet Modelling and Recommendation. In *Proceedings of the 24th ACM International Conference on Multimedia*, MM, pages 1018–1027.

Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. 2012. Multi-column Deep Neural Networks for Image Classification. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, pages 3642–3649.

William W Cohen, Robert E Schapire, and Yoram Singer. 1998. Learning to Order Things. In *Advances in Neural Information Processing Systems*, NIPS, pages 451–457.

Thierry Declerck and Piroska Lendvai. 2015. Processing and normalizing hashtags. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, RANLP, pages 104–109.

Bhuwan Dhingra, Zhong Zhou, Dylan Fitzpatrick, Michael Muehl, and William Cohen. 2016. Tweet2Vec: Character-Based Distributed Representations for Social Media. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, ACL, pages 269–274.

Jacob Eisenstein. 2013. What to do about bad language on the Internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL, pages 359–369.

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in Twitter data with crowdsourcing. In *Proceedings of the Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, NAACL, pages 80–88.

Bhayani R. Go, A. and L. Huang. 2009. Twitter Sentiment Classification using Distant Supervision. *CS224N Project Report, Stanford*.

Irving J Good. 1953. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3-4):237–264.

David Graff and Christopher Cieri. 2003. English Gigaword LDC2003T05. *Linguistic Data Consortium (LDC)*.

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*, ACL, pages 368–378.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT, pages 187–197.

Kenneth Heafield. 2013. *Efficient Language Modeling Algorithms with Applications to Statistical Machine Translation*. Ph.D. thesis, Carnegie Mellon University.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP.

Hen-Hsen Huang, Chiao-Chen Chen, and Hsin-Hsi Chen. 2018. Disambiguating false-alarm hashtag usages in tweets for irony detection. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, ACL, pages 771–777.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference for Learning Representations*, ICLR.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the 1995 International Conference on Acoustics, Speech, and Signal Processing*, ICASSP, pages 181–184.

Philipp Koehn and Kevin Knight. 2003. Empirical methods for compound splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, EACL, pages 187–194.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. 2018. Exploring the Limits of Weakly Supervised Pretraining. In *Tech Report*.

Diana Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, LREC, pages 4238–4243.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the Seventh International Workshop on Semantic Evaluation*, SemEval, pages 321–327.

Preslav Nakov, Sara Rosenthal, Svetlana Kiritchenko, Saif M. Mohammad, Zornitsa Kozareva, Alan Ritter, Veselin Stoyanov, and Xiaodan Zhu. 2016. Developing a successful SemEval task in sentiment analysis of Twitter and other social media texts. *Language Resources and Evaluation*, 50(1):35–65.

Dong Nguyen, Barbara McGillivray, and Taha Yasseri. 2018. Emo, love and god: making sense of urban dictionary, a crowd-sourced online dictionary. *Royal Society Open Science*, 5(5):172320.

Ozer Ozdikis, Pinar Senkul, and Halit Oguztuzun. 2012. Semantic Expansion of Hashtags for Enhanced Event Detection in Twitter. In *Proceedings of the 1st international Workshop on Online Social Systems*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP, pages 79–86.

M. Parakhin and P. Haluptzok. 2009. Finding the Most Probable Rranking of Objects with Probabilistic Pairwise Preferences. In *Proceedings of the 10th International Conference on Document Analysis and Recognition*, ICDAR, pages 616–620.

Fuchun Peng and Dale Schuurmans. 2001. A hierarchical em approach to word segmentation. In *NLPRS*, pages 475–480.

Ashequl Qadir and Ellen Riloff. 2014. Learning emotion indicators from tweets: Hashtags, hashtag patterns, and phrases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP, pages 1203–1209.

Jack Reuter, Jhonata Pereira-Martins, and Jugal Kalita. 2016. Segmenting twitter hashtags. *International Journal on Natural Language Computing*, 5:23–36.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named Entity Recognition in Tweets: An Experimental Study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP, pages 1524–1534.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment Analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, SemEval, pages 502–518.

Justin Sampson, Fred Morstatter, Liang Wu, and Huan Liu. 2016. Leveraging the implicit structure within social media for emergent rumor detection. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM, pages 2377–2382.

C. Simeon, H. J. Hamilton, and R. J. Hilderman. 2016. Word segmentation algorithms with lexical resources for hashtag classification. In *Proceedings of the 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 743–751.

Richard Sproat and Chilin Shih. 1990. A statistical method for finding word boundaries in chinese text. *Computer Processing of Chinese and Oriental Languages*, 4(4):336–351.

Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, ICSLP, pages 901–904.

Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. 2014. Building Large-Scale Twitter-Specific Sentiment Lexicon : A Representation Learning Approach. In *Proceedings of the 25th International Conference on Computational Linguistics*, COLING, pages 172–182.

Zhiyang Teng, Duy Tin Vo, and Yue Zhang. 2016. Context-Sensitive Lexicon Features for Neural Sentiment Analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, EMNLP, pages 1629–1638.

Kuansan Wang, Christopher Thrasher, and Bo-June Paul Hsu. 2011. Web Scale NLP: A Case Study on URL Word Breaking. In *Proceedings of the 20th International Conference on World Wide Web*, WWW, pages 357–366.

Jason Weston, Sumit Chopra, and Keith Adams. 2014. # tagspace: Semantic embeddings from hashtags. In *Proceedings of the 2014 Conference on Empirical*

Methods in Natural Language Processing, EMNLP, pages 1822–1827.

Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as LMR tagging. In *Proceedings of the second SIGHAN workshop on Chinese Language Processing*, SIGHAN, pages 176–179.

A Appendix

A.1 Word-shape rules

Our model uses the following word shape rules as boolean features. If the candidate segmentation s and its corresponding hashtag h satisfies a word shape rule, then the boolean feature is set to True.

Rule	**Hashtag → Segmentation**
Camel Case	XxxXxx → Xxx+Xxx
Consonants	cccc → cccc
Digits as prefix	ddwwww → dd+wwww
Digits as suffix	wwwwdd → wwww+dd
Underscore	www_www → www + _ + www

Table 11: Word-shape rule features used to identify good segmentations. Here, X and x represent capitalized and non-capitalized alphabetic characters respectively, c denotes consonant, d denotes number and w denotes any alphabet or number.