Exploiting Hybrid Contexts for Tweet Segmentation

Chenliang Li[†], Aixin Sun[†], Jianshu Weng[‡], Qi He[§]
[†]School of Computer Engineering, Nanyang Technological University, Singapore
{lich0020,axsun}@ntu.edu.sg
[‡]Independent Researcher, Singapore
jianshu@acm.org
[§]IBM Almaden Research Center, USA
heq@us.ibm.com

ABSTRACT

Twitter has attracted hundred millions of users to share and disseminate most up-to-date information. However, the noisy and short nature of tweets makes many applications in information retrieval (IR) and natural language processing (NLP) challenging. Recently, segment-based tweet representation has demonstrated effectiveness in named entity recognition (NER) and event detection from tweet streams. To split tweets into meaningful phrases or segments, the previous work is purely based on external knowledge bases, which ignores the rich local context information embedded in the tweets. In this paper, we propose a novel framework for tweet segmentation in a batch mode, called *HybridSeg*. *HybridSeg* incorporates local context knowledge with global knowledge bases for better tweet segmentation. *HybridSeg* consists of two steps: learning from off-the-shelf weak NERs and learning from pseudo feedback. In the first step, the existing NER tools are applied to a batch of tweets. The named entities recognized by these NERs are then employed to guide the tweet segmentation process. In the second step, *HybridSeg* adjusts the tweet segmentation results iteratively by exploiting all segments in the batch of tweets in a collective manner. Experiments on two tweet datasets show that *HybridSeg* significantly improves tweet segmentation quality compared with the state-of-the-art algorithm. We also conduct a case study by using tweet segments for the task of named entity recognition from tweets. The experimental results demonstrate that *HybridSeg* significantly benefits the downstream applications.

Categories and Subject Descriptors

H.3.1 [**Information Systems**]: Content Analysis and Indexing—*Linguistic processing*

Keywords

Twitter, Tweet, Tweet segmentation, Named Entity Recognition

1. INTRODUCTION

Twitter has become one of the most important channels for people to find, share, and disseminate timely information. As of March 2012, there are more than 140 million active Twitter users with over 340 million tweets posted in a day[1]. Due to its large volume of timely information generated by its millions of users, it is imperative to understand tweets' language for the tremendous downstream applications like named entity recognition (NER) [12,14,21], event detection and summarization [5, 15, 22], opinion mining [16, 17], sentiment analysis [4, 13, 25], etc.

Given the limited length (*i.e.,* 140 characters) of a tweet and no restrictions on its writing styles, tweets often contain grammatical errors, misspellings, and informal abbreviations. The error-prone and short nature of tweets makes the word-level language models for tweets not reliable. For example, given a tweet "I call her, no answer. Her phone in the bag, she dancin.", there is no clue to guess its true theme with the word independent assumption. Very recently, a segment-based tweet representation model has been proposed to partially overcome the adverse features of tweets [12]. After splitting a tweet into a sequence of consecutive n-grams ($n \geq 1$), each of which is called a segment, the latent topics of the tweet can be better captured. For example, the segment "she dancin" in the previous example tweet is actually a key concept – it classifies this tweet into the family of tweets talking about the song "She Dancin", a trend topic in Bay Area in Jan, 2013.

A segment can be a named entity (*e.g.,* a movie title "finding nemo"), a semantically meaningful information unit (*e.g.,* "officially released"), or any other type of phrase which appears "more than chance" [12]. Because segments preserve more semantics of tweets than words do, they have been used as the language units for the tasks of NER and event detection [11, 12]. Segment-based representation has shown its effectiveness over word-based representation particularly in the event detection task.

However, developing a high quality tweet segmentation is not trivial, because the prevalence of grammatical errors and unreliable linguistic features makes classic Natural Language Processing (NLP) techniques including part-of-speech (POS) tagging [7] and NER less applicable to tweets. For example, classic POS taggers may fail in recognizing "she dancin" as a noun phrase in the previous example tweet. The seminal tweet segmentation algorithm in [12] mainly relies on external knowledge bases (*e.g.,* Wikipedia and Microsoft Web N-Gram corpus). The main assumption is that the core semantic information is well preserved in tweets in the form of named entities or semantic phrases for information sharing, despite the noise nature of tweets, and those named entities and semantic phrases can be largely found in aforementioned external knowledge bases. We categorize this method into *global context* based solution, since it utilizes the language-generic information contained in the external knowledge bases.

[1] http://blog.twitter.com/2012/03/twitter-turns-six.html

SIGIR'13, July 28–August 1, 2013, Dublin, Ireland.

Purely relying on external knowledge bases for tweet segmentation has two major weaknesses. First, tweets are highly time-sensitive so that many recently appeared terms like "She Dancin" can not be found in external knowledge bases. Another example is "Tin Pei Ling", a new politician who gained her reputation in Singapore General Election 2011[2]. No entry about her had been created in Wikipedia until 4th, April 2011, about a week later than the time her name appeared in tweets. Second, by defining other tweets published within the same short time window (i.e., an hour/a day) as the *local context*, a tween can only be well understood within its local context. For example, "Dancin" could be a typo if it only appears in a single tweet. But it may refer to some named entity if it appears in lots of tweets in a short term.

In this paper, we propose a hybrid tweet segmentation framework incorporating local contexts into the existing external knowledge bases, and name our method *HybridSeg*. *HybridSeg* conducts tweet segmentation in batch mode. Following the same scope of [12], we only segment tweets from a targeted Twitter stream. A targeted Twitter stream receives tweets based on user defined criteria (*e.g.,* tweets containing some predefined hashtags or keywords, tweets published by a predefined list of users, or tweets published by users from a specific geographical region). Tweets from a targeted Twitter stream are grouped into batches by their publication time using a fixed time interval (*e.g.,* an hour or a day). Each batch of tweets are then segmented by *HybridSeg* collectively.

HybridSeg conducts tweet segmentation in an iterative manner. At the first iteration, *HybridSeg* segments the tweet by utilizing the local linguistic features of the tweet itself. To avoid implementation from scratch, we simply apply a set of existing NER tools trained over general English languages on tweets. These existing NER tools provide an initial collection of confident segments by voting. Initializing *HybridSeg* with a set of off-the-shelf NER tools is based on the observation that some tweets from official accounts of news agencies, organizations, advertisers, and celebrities are likely well written. A small set of named entities extracted from these tweets based on voting of classic NER tools can be a high precise yet low recall solution of tweet segmentation. For example, although "Tin Pei Ling" is a new figure, it can be recognized as people name by classic NERs with high confidence because there are many well-written tweets about her generated by news agencies.

After that, *HybridSeg* learns better segmentation iteratively based on the confident segments extracted from the batch of tweets in the last iteration. The iterative process stops when no more changes are observed from the resulted segments. The philosophy is the confident segments of one tweet can affect the segmentation of other tweets in the same batch. External knowledge bases are also integrated here to better measuring the cohesiveness of segments. To the best of our knowledge, this paper proposes a novel framework by leveraging both global and local contexts in tweet segmentation for targeted Twitter streams.

We conduct extensive experimental analysis on two tweet datasets and compare *HybridSeg* with the approach purely relying on global knowledge bases [12], known as *GlobalSeg* in this paper. Our experimental results show that *HybridSeg* achieved significant improvement on tweet segmentation quality by comparing to the set of manually labeled tweets. We further split named entities from valid segments and only compared these named entities to human labels. Again, *HybridSeg* improved the NER quality compared to *GlobalSeg*.

The rest of the paper is organized as follows. Section 2 surveys the related works such as NLP, collocation measures, noun phrase extraction, and text segmentation. Section 3 gives an overview of the global knowledge guided segmentation process. The proposed hybrid-context based segmentation algorithm is presented in Section 4. In Section 5, we conduct experimental evaluation of the proposed *HybridSeg* framework and provide detailed algorithm analysis. Section 6 concludes this paper.

[2] http://en.wikipedia.org/wiki/Tin_Pei_Ling

2. RELATED WORK

Many conventional NLP techniques are designed for formal text. Many of these techniques are supervision based and heavily rely on the local linguistic features, such as POS tags, word capitalization, trigger words (*e.g.,* Mr., Dr.), and dictionary lookup like gazetteers, etc. These linguistic features, together with effective supervised learning algorithms (*e.g.,* hidden Markov model (HMM) and conditional random field (CRF)), achieve state-of-the-art performance on formal text corpus [6, 19, 26]. However, these techniques cannot be directly applied to tweets because of the noisy and short nature of the latter. The error-prone and short nature of tweets (and other user-generated short text) has attracted renewed interests in conventional tasks in NLP including POS tagging [7], named entity recognition (NER) [12, 14, 21], etc.

To improve POS tagging on tweets, Gimple *et al.* incorporate tweet-specific features including at-mentions, hashtags, URLs, and emotions [7]. In their approach, they measure the confidence of capitalized words and apply phonetic normalization for ill-formed words to address possible peculiar writings in tweets. Normalization of ill-formed words in tweets has established itself as an important research problem. A supervised approach is employed in [8] to first identify the ill-formed words. Then, the correct normalization of the ill-formed word is selected based on a number of lexical similarity measures.

Both supervised and unsupervised approaches have been proposed for named entity recognition in tweets. T-NER is a part of the tweet-specific NLP framework in [21]. T-NER first segments named entities using a CRF model with orthographic, contextual, and dictionary features, and then labels the named entities using a LDA (Latent Dirichlet allocation) model. The NER solution proposed in [14] is also based on a CRF model. The unsupervised approach named *TwiNER* recognizes named entities with two steps: tweet segmentation and segment ranking [12]. The tweets from a Twitter stream are processed in batch mode. Member tweets in a batch are segmented and each tweet segment is a candidate named entity (see Section 3). Collocation measure between words is utilized in tweet segmentation together with the global knowledge bases. Experimental results show that the collocation measure *Symmetric Conditional Probability* (SCP) is much more effective than *Pointwise Mutual Information* (PMI). To recognize named entities from the candidate segments, a segment graph is built based on segment co-occurrences. The segments are then ranked by applying random walk on the segment graph (see Section 5.4.1). Chua *et al.* [3] propose to extract noun phrases from tweets using an unsupervised approach which is mainly based on POS tagging. Each extracted noun phrase is a candidate named entity.

Tweet segmentation is conceptually similar to the task of text segmentation despite the unit for segmentation (words vs sentences) is significantly different. Text segmentation divides a given text document into consecutive non-overlapping topically cohesive segments [1]. Each segment consists of several consecutive sentences. Most existing works follow a similar idea that the boundaries of consecutive segments correspond to a change in word usage. The boundaries between segments are often detected by monitoring the drop of the lexical similarity. The proposed solutions differ widely in the way of calculating the sentence-pair similarity (*i.e.,* topical

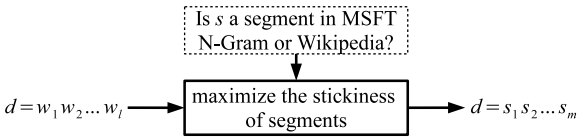


Figure 1: The framework of global knowledge bases guided tweet segmentation [12], named *GlobalSeg*.

cohesiveness). Measures based on word co-occurrence [2, 9, 10] and generative models [1, 18, 20, 23] have been extensively studied. The determination of the segment boundaries may not only be purely based on the local sentence-pair similarities but also be based on the global information derived from the distribution of the lexical similarities of the far neighboring sentences [2,10]. Conceptually, the idea of incorporating the global information for boundary determination is similar to our idea of exploiting local context in tweets for tweet segmentation. However, the two problems (text segmentation and tweet segmentation) are different by definition and the techniques for text segmentation cannot be directly applied in tweet segmentation.

3. GLOBAL KNOWLEDGE GUIDED TWEET SEGMENTATION

Global knowledge bases like Wikipedia and Microsoft Web N-Gram corpus [24] have been successfully utilized to guide the tweet segmentation, which is named *GlobalSeg* by this paper. In this section, we briefly review its process, as illustrated by Figure 1.

The input of *GlobalSeg* is a tweet $d = w_1 w_2 \dots w_\ell$, and its output is m ($m \leq \ell$) non-overlapped segments, $d = s_1 s_2 \dots s_m$, where a segment s_i is a n-gram ($n \geq 1$). The optimal segmentation is to maximize the sum of *stickiness* scores of m segments:

$$\arg\max_{s_1,\dots,s_m} \sum_{i=1}^{m} C(s_i), \tag{1}$$

where $C(s)$ denotes a *stickiness* function, defined as:

$$C(s) = \mathcal{L}(s) \cdot e^{Q(s)} \cdot \frac{2}{1 + e^{-SCP(s)}}. \tag{2}$$

Eq. 2 encodes three factors for measuring the stickiness of each segment. They are: 1) the normalized segment length $\mathcal{L}(s) = 1$ if $|s| = 1$ and $\mathcal{L}(s) = \frac{|s|-1}{|s|}$ if $|s| > 1$ which moderately alleviates the penalty on long segments; 2) the probability that s is inside an anchor text in Wikipedia $Q(s)$; 3) the *Symmetric Conditional Probability* (SCP) measure defined by

$$SCP(s) = \log \frac{\Pr(s)^2}{\frac{1}{|s|-1}\sum_{i=1}^{|s|-1} \Pr(w_1 \dots w_i) \Pr(w_{i+1} \dots w_{|s|})}, \tag{3}$$

$Pr(s)$ is the n-gram probability provided by Microsoft Web N-Gram service. If s is a single word w, $SCP(s) = 2 \log Pr(w)$. If s contains more words, SCP tends to keep a cohesive s while all its possible binary partitions are not cohesive.

4. HYBRID-CONTEXT BASED TWEET SEGMENTATION

GlobalSeg defines a tweet segmentation method solely based on external knowledge bases (*e.g.,* Wikipedia and Microsoft Web N-Gram corpus). Its main assumption is that the external knowledge bases provide more robust segmentation features for tweets than their original natural languages which tend to be error-prone [12].

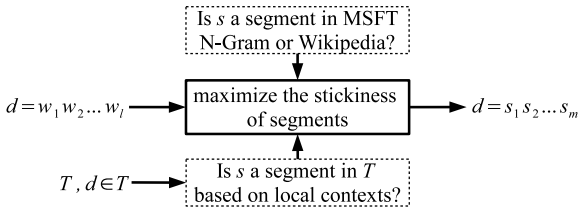


Figure 2: The framework of Hybrid-context based tweet segmentation, named *HybridSeg*.

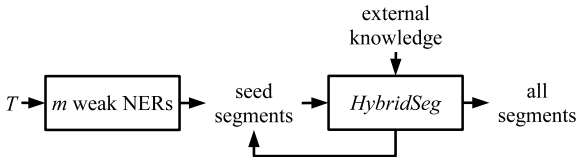


Figure 3: The iterative process of *HybridSeg*.

Despite its successful application in named entity recognition, it has two inherent limitations based on the below observations.

Observation 1. *Tweets are not topically independent to each other within a time window.*

In *GlobalSeg*, all tweets are segmented independently. This assumption is too strong as tweets published closely in time often talk about the same theme. These similar tweets largely share the same segments. For example, similar tweets have been grouped together to collectively detect events from tweets, and the event is usually represented by the common discriminative segments across tweets [11]. However, there is no mechanism in *GlobalSeg* to force similar tweets to be segmented consistently.

Observation 2. *Linguistic features of the tweets are not always useless.*

In *GlobalSeg*, the local linguistic features of a segment in its tweet has been abandoned completely. This assumption is mostly true especially when tweets contain many unreliable linguistic features like misspellings, informal abbreviations and unreliable capitalizations [21]. However, there indeed exist tweets composed in proper English. For example, some tweets are published by official accounts of news agencies, organizations, advertisers, and celebrities. For these tweets, the local linguistic features can be an informative source for tweet segmentation.

The two observations essentially reveal the same phenomenon: except for the external knowledge bases, local information like the other similar tweets within a time window or the linguistic features in the current tweet can also help in segmenting tweets. As both similar tweets and linguistic features refer to the *local contexts* of tweets, they lead to our novel idea of incorporating *local contexts* into the existing *external knowledge bases* for better tweet segmentation. Accordingly, we name our new method *HybridSeg*, short for a hybrid-context based segmentation for tweets.

4.1 Framework of *HybridSeg*

The general framework of *HybridSeg* is given in Figure 2. Compared to Figure 1, the major novelty is that we would examine the segment likelihood in a batch of tweets (including the tweet to be segmented) based on local contexts. That is to say, given a batch of tweets $\mathcal{T}$ within a certain time window, we are going to utilize

Algorithm 1: Tweet Segmentation by *HybridSeg*

input :
A tweet batch: $\mathcal{T} = \{t_1, t_2, ...\}$;
Off-the-shelf NERs: $r_i, r_2, ..., r_m$;
$\mu_{NER}(\lambda)$: function for learning λ from weak NERs;
$\mu_{Iter}(\lambda)$: function for learning λ from pseudo feedback;
$\mathcal{R}_\lambda$: the search space of λ;
output:
An optimal segmentation for the batch $\mathcal{T}$:
$\mathcal{S} = \{t_i = s_{i,1}s_{i,2}...s_{i,m}\}$;
/* iteration 0: learning from weak NERs */
1 **foreach** r_i **do**
/* apply NER r_i to the tweet batch $\mathcal{T}$ */
2 $\mathcal{S}_{r_i} \leftarrow$ recognize named entities by using NER r_i;
3 $\mathcal{M}_0 \leftarrow$ calculate $\hat{P}_{NE}(s)$ based on $\{\mathcal{S}_{r_i} | i = 1, 2, ...m\}$;
4 $\mathcal{N}_\cap \leftarrow$ extract the named entities that are recognized by all m NERs.
/* apply λ learning with $\mu_{NER}(\lambda)$ */
5 $\mathcal{S}_0 \leftarrow$ segmentation with the optimal λ by using $\mu_{NER}(\lambda)$ and $\mathcal{R}_\lambda$;
/* iteration $1, 2, ...$: iterative learning */
6 i=1;
7 **while** *true* **do**
8 $\mathcal{M}_i \leftarrow$ calculate $\hat{Pr}^i(s)$ based on $\mathcal{S}_{i-1}$;
/* apply λ learning with $\mu_{Iter}(\lambda)$ */
9 $\mathcal{S}_i \leftarrow$ segmentation with the optimal λ by using $\mu_{Iter}(\lambda)$ and $\mathcal{R}_\lambda$;
/* calculate Jensen-Shannon divergence between the segments in the last and current iterations */
10 $jsd \leftarrow JSD(\mathcal{S}_{i-1}, \mathcal{S}_i)$
11 **if** $jsd < \varepsilon$ **then**
/* the stop criterion is met */
12 break;
13 $i++$;
14 $\mathcal{S} \leftarrow \mathcal{S}_i$;
15 return $\mathcal{S}$ as the optimal segmentation;

their local linguistic features collectively for each member tweet's segmentation.

However, how to separate valid segments from the chaffs in a batch of tweets only based on local contexts is a challenging task itself. To tackle this problem, we design an iterative process in our framework, as shown in Figure 3.

Starting from a set of off-the-shelf NERs trained on classic English texts, we generate a collection of seed segments from $\mathcal{T}$ only based on local linguistic features by voting. The seed segment collection can be small yet highly confident. Then, we combine the seed segment collection with the external knowledge bases in *HybridSeg* to segment each tweet member. After that, only those segments ranked high by *HybridSeg* are used to replace the seed segment collection. The same process is then repeated until the segmentation results of *HybridSeg* do not change any more.

There are a couple of advantages in our design. First, existing NERs have already captured most known local linguistic features so that we do not need to implement from scratch. Second, most existing NERs are trained on formal texts. Applying them directly on tweets will easily cause the overfitting error. A voting algorithm can partially alleviate the error of training. Third, during each iteration, only top segments with high confidence scores are fed into *HybridSeg* in a pseudo feedback manner, further alleviating the errors caused by overfitting.

The tweet segmentation process of *HybridSeg* is outlined in Algorithm 1. In the following, we elaborate *HybridSeg* step by step.

4.2 Learning from weak NERs

In this section, we apply m weak NERs on $\mathcal{T}$ to generate the initial collection of seed segments. Based on the truth that a named entity is always a valid segment, we directly select the seed segments from the output of NERs.

Given a segment s, let f_s be its total frequency in $\mathcal{T}$. One NER r_i may recognize s as a named entity $f_{r_i,s}$ times. Note that $f_{r_i,s} \leq f_s$ since a NER may only recognize some of s's occurrences as named entity. Assuming there are multiple off-the-shelf NERs $r_1, r_2, \ldots, r_m$, we further denote f_s^R to be the number of NERs that have detected at least one occurrence of s as named entity, $f_s^R = \sum_i^m I(f_{r_i,s})$: $I(f_{r_i,s}) = 1$ if $f_{r_i,s} > 0$; $I(f_{r_i,s}) = 0$ otherwise.

We approximate the probability of s being a seed segment using a voting algorithm defined by Eq. 4:

$$\hat{Pr}(s) = w(s,m) \cdot \frac{1}{m} \sum_i^m \hat{Pr}_{r_i}(s), \quad (4)$$

$$w(s,m) = 1/\left(1 + e^{-\beta(f_s^R - m/2)}\right), \quad (5)$$

$$\hat{Pr}_{r_i}(s) = \left(1 + \frac{\alpha}{f_{r_i,s} + \varepsilon}\right)^{-\frac{f_s}{f_{r_i,s}+\varepsilon}}. \quad (6)$$

Our approximation contains two parts. The right part of Eq. 4 (rf. Eq. 6) is the average confidence that one weak NER recognizes s as named entity. A biased estimation of the right part is simply $1/m \cdot \sum_{i=1}^m f_{r_i,s}/f_s$ as each $f_{r_i,s}/f_s$ is a noisy version of the true probability. However, such simple average ignores the absolute value of $f_{r_i,s}$ which can also play an important role here. For example, for a party name in an election event, it can appear hundreds of times within a tweet batch. However, due to the free writing styles of tweets, only tens of the party name's occurrences can be recognized by weak NERs as named entity. In this case, $f_{r_i,s}/f_s$ is relatively small yet $f_{r_i,s}$ is relatively high. Thus, we design a function that can favor both $f_{r_i,s}/f_s$ and $f_{r_i,s}$. The favor scale is controlled by a factor α. When α is large, our function is more sensitive to the change of $f_{r_i,s}/f_s$; when α is small, a reasonably large $f_{r_i,s}$ can lead $\hat{Pr}_{r_i}(s)$ to be close to 1 despite of a relatively small value of $f_{r_i,s}/f_s$. In this paper we empirically set $\alpha = 0.2$ in experiments. A small constant ε is set to avoid dividing by zero.

The left part of Eq. 4, $w(s,m)$ (rf. Eq. 5) uses a sigmoid function to control the impact of the majority degree of m weak NERs on seed segments, which is tuned by a factor β. For example, in our paper we set $\beta = 10$ so that as long as more than half of weak NERs recognize s as named entity, $w(s,m)$ is close to 1. With a small β, $w(s,m)$ is getting closer to 1 only when more and more weak NERs recognize s as named entity.

The learning from weak NERs is summarized in lines 1 - 5 in Algorithm 1. Next, we discuss the combination of global knowledge bases and seed segments. The learning of λ will be detailed in Section 4.5.

4.3 Combination of knowledge bases and seed segments

After computing the probability of s being a valid segment, we can select top confident segments to be included in the seed segment collection. The next question is: how can we combine seed segments with knowledge bases for better tweet segmentation?

Recall that *GlobalSeg* encodes three factors: segment length, Wikipedia knowledge, and SCP measure based on Microsoft Web N-gram corpus, among which the text partition mainly relies on SCP measure. Now, the seed segment collection naturally provides another information source to guide the text partition. The idea can be as simple as assigning a high stickiness score to a partition in the seed segment collection. Accordingly, we modify the partition likelihood $Pr(s)$ in SCP measure using a linear combination:

$$Pr(s) = (1 - \lambda)Pr_{MS}(s) + \lambda \hat{Pr}(s), \quad (7)$$

where $\hat{Pr}(s)$ is defined by Eq. 4 with a coupling factor $\lambda \in [0, 1)$, and $Pr_{MS}(\cdot)$ is the n-gram probability provided by Microsoft Web N-Gram service.

4.4 Pseudo feedback on seed segments

The last step of *HybridSeg* is to iteratively improve the seed segments using the output of *HybridSeg*, with the final goal of improving its output in the end. The whole process will terminate until no more changes are observed in the output of *HybridSeg*.

Suppose at iteration i, *HybridSeg* outputs a set of segments $\{\langle s, f_s^i \rangle\}$, where f_s^i is the number of times s is a segment at iteration i. Then, f_s^i / f_s relatively records the segmentation confidence of *HybridSeg* on s at iteration i (recall that f_s denotes the frequency of s in current batch $\mathcal{T}$). Similar to Eq. 6, we denote

$$\hat{Pr}^i(s) = \left(1 + \frac{\alpha}{f_s^i + \varepsilon}\right)^{-\frac{f_s}{f_s^i + \varepsilon}}.$$

Following the same combination strategy defined by Eq. 7, we have the following iterative updating function:

$$Pr^{i+1}(s) = (1 - \lambda) Pr_{MS}(s) + \lambda \hat{Pr}^i(s), \tag{8}$$

Interestingly, we can treat learning weak NERs as the 0^{th} iteration and change Eq. 7 to:

$$Pr^1(s) = (1 - \lambda) Pr_{MS}(s) + \lambda \hat{Pr}^0(s),$$

where $\hat{Pr}^0(s)$ is the voting result on m weak NERs only. The iteration process will stop when predefined criterion is met. Here, we define the stop criterion based on Jensen-Shannon divergence (JSD) of the frequency distributions of tweet segments in two consecutive iterations. Note that, in the 0^{th} iteration, some segments may be wrongly segmented because of the errors introduced by the weak NERs. These errors could be further propagated at later iterations. For this reason, we do not consider the segments detected by weak NER whose frequency is smaller than 3 in the 0^{th} iteration.

The learning from pseudo feedback is summarized in lines 7 - 13 in Algorithm 1.

4.5 Learning the parameter λ

Eq. 8 defines the iterative learning process of *HybridSeg*. The coupling factor λ is crucial to control the convergence of learning and the segment re-ranking performance of the next iteration. To guarantee the convergence and quality of our algorithm, we need a systematic way to learn λ.

Our idea is: a good λ must ensure that top confident segments from the previous iteration should be detected more times in the next iteration. Note that at each iteration we do not classify text partitions into binary classes (*i.e.,* segment and non-segment), but simply assign a stickiness score to each text partition and then treat segmentation as a ranking problem. Therefore, our idea is equivalent to maximizing the sum of detected frequency of top confident segments (weighted by their stickiness scores, rf. Eq. 2) extracted from the previous iteration. Accordingly, learning the parameter λ is converted to an optimization problem as follows:

$$\begin{aligned} \hat{\lambda} &= \arg\max_{\lambda} \mu_{Iter}(\lambda) \\ &= \arg\max_{\lambda} \sum_{s \in \text{top-k at iteration } i} C^i(s) \cdot f^{i+1}(s). \end{aligned} \tag{9}$$

$C^i(s)$ is the stickiness score of s output by *HybridSeg* and used for segment ranking at the previous iteration. Based on it, top-k segments can be retrieved. $f^{i+1}(s)$ is the detected frequency of s at the current iteration, which is an unknown function of the variable λ.

Table 1: Statistics of SIN and SGE datasets

Dataset	Collection period	#Tweets	#Annotations
SIN	Jun 01 - Jun 30, 2010	4,331,937	4,422
SGE	Apr 13 - May 13, 2011	226,744	3,328

Therefore, the optimal λ is intractable. In our experiments, we used brute-force search strategy to find the optimal λ for each iteration and each tweet batch. Fortunately, as the size of each tweet batch is controllable, the efficiency is not our concern in the current work.

Note that for the 0^{th} iteration, λ must be learned differently because there do not exist top confident segments from the previous iteration. Since we use the voting results of m weak NERs as the seed segments in the 0^{th} iteration, a good λ must ensure that the confident segments voted by m weak NERs can be detected more times by *HybridSeg*.

Let $\mathcal{N}_\cap$ be the segments that are recognized by all m NER systems (*i.e.,* $\mathcal{N}_\cap = \{s | f_s^R = m\}$). For each segment $s \in \mathcal{N}_\cap$, we consider its confident frequency to be the minimum number of times that s is recognized by a NER as named entity among all m NERs. Let the confident frequency of s be $f_{c,s}$. $f_{c,s} = min_i^m f_{r_i,s}$. Then λ is learnt as follows in the 0^{th} iteration:

$$\hat{\lambda} = \arg\max_{\lambda} \mu_{NER}(\lambda) = \arg\max_{\lambda} \sum_{s \in \mathcal{N}_\cap} \hat{Pr}^0(s) \cdot f_{c,s} \cdot f_s^0 \tag{10}$$

In this equation, $\hat{Pr}^0(s)$ is the value computed using Eq. 4; $\hat{Pr}^0(s) \cdot f_{c,s}$ serves as a weighting factor to adjust the importance of f_s^0 in learning λ. If segment s is very likely to be a named entity (*i.e.,* $\hat{Pr}^0(s)$ is high) and it has been detected many times by all NERs (*i.e.,* $f_{c,s}$ is large), then the number of times s is successfully segmented f_s^0 has a big impact to the selection of λ. On the other hand, if $\hat{Pr}^0(s)$ is low, or $f_{c,s}$ is small, or both conditions hold, then f_s^0 is less important to λ setting.

By defining $f_{c,s} = min_i^m f_{r_i,s}$, Eq. 10 conservatively considers segments recognized by all weak NERs because of the noisy nature of tweets. This helps to reduce the possible oscillations resulted from different λ settings, since λ is a global factor (*i.e.,* not per-tweet dependent). On the other hand, we also assume that all the off-the-shelf NERs are reasonably good, *e.g.,* when they are applied on formal text. If there is a large number of NERs, then the definition of $f_{c,s}$ could be relaxed to avoid having too small values for all segments, due to one or two poor-performing NERs among them.

5. EVALUATION

In this section, we evaluate *HybridSeg* against *GlobalSeg* as the baseline. We conducted experiments on two datasets, *i.e.,* the *SIN* and *SGE* datasets that were used to evaluate *GlobalSeg* in [12]. Three weak NERs, namely, LBJ-NER, Standford-NER, and T-NER were used as input in *HybridSeg* for learning local context. We compare segmentation accuracy of *HybridSeg* against *GlobalSeg*. Because *GlobalSeg* was used to detect named entities in [12], we also report the accuracy of named entity recognition using *HybridSeg* and *GlobalSeg* respectively.

5.1 Dataset and Setting

Tweet Datasets. The *SIN* and *SGE* datasets were originally collected for simulating two targeted Twitter streams in [12]. The former was a stream consisting of tweets from users in a specific geographical region (*i.e.,* Singapore in this case), and the latter was a stream consisting of tweets matching some predefined keywords

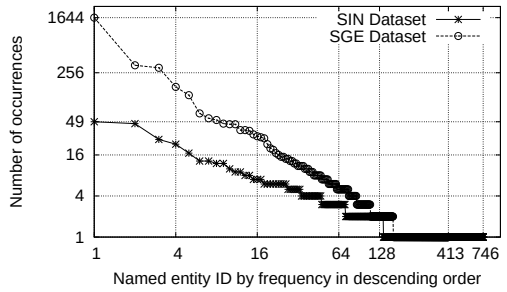


Figure 4: Frequency distribution of the annotated NEs

and hashtags for a major event (*i.e.,* Singapore General Election 2011 in this case).

Reported in Table 1, named entities in 4,422 tweets from *SIN* and 3,328 tweets from *SGE* have been manually annotated to evaluate tweet segmentation and named entity recognition performance in [12]. Table 2 reports the statistics of the annotated NEs in the two datasets where f_s^g denotes the number of occurrences (or frequency) of segment s in the annotated ground truth $\mathcal{G}$. The frequency distributions of the NEs are plotted in Figure 4.

All the annotated tweets in each dataset were published on the same day, making it possible to evaluate batch-mode tweet segmentation using a day as the time interval. To fairly compare with the results reported in [12], we conduct our experiments using all the annotated tweets in each dataset as a batch so as not to take additional context from other unlabeled tweets published in the same day.

We also used the same Wikipedia dump (released on 30 Jan, 2010[3]) as in [12]. This dump contains 3,246,821 articles and there are 4,342,732 distinct entities appeared as anchor texts in these articles.

Evaluation Metric. Recall that the task of tweet segmentation is to split a tweet into semantically meaningful segments. Ideally, a tweet segmentation method shall be evaluated by comparing its segmentation result against manually segmented tweets. However, manual segmentation of a reasonably sized data collection is extremely expensive and requires good understanding of the tweets. We therefore choose to follow [12] to evaluate a tweet segmentation method based on whether the manually annotated named entities are correctly segmented (*i.e.,* split as segments). Because each named entity is a valid segment, the annotated named entities in this case serve as partial ground truth for the evaluation. We use two measures, namely *Recall* and *Frequency-biased Recall*.

- *Recall*, denoted by *Re*, quantifies the percentage of the manually annotated named entities that are correctly split as segments. Because a segmentation method outputs *exactly one* possible segmentation for each tweet, using annotated named entities as partial ground truth, recall is the same as precision in this setting. In fact, the same measure is named as precision in [12]. Note that each named entity appearing in a specific position of a tweet is considered as one distinct instance. For example, the phrase "pap pap pap!", commonly used to express a user's strong support to the PAP party[4] in Singapore general election, consists of three distinct named entity instances "pap".

[3] http://dumps.wikimedia.org/enwiki/

[4] http://en.wikipedia.org/wiki/People's_Action_Party

Table 2: The annotated named entities in SIN and SGE datasets

Dataset	#NEs	min f_s^g	max f_s^g	$\sum f_s^g$	#NEs *s.t.* $f_s^g > 1$
SIN	746	1	49	1234	136
SGE	413	1	1644	4073	161

- *Frequency-biased Recall*, denoted by Re_F, factors in the frequency of the named entities in a batch of tweets. Because the frequent named entities are often related to hot topics or emerging events in the targeted Twitter stream, correctly identifying these named entities as segments is critical for downstream applications, like user opinion mining, topic discovery, and event detection.

  Let f_s^g be the frequency of segment s in the ground truth $\mathcal{G}$. Let $Re(s)$ be the recall of segment $s \in \mathcal{G}$ which is the percentage of segment s being correctly extracted from all occurrences of s in the annotated tweets. The frequency-biased recall is defined as follows:

$$Re_F = \frac{1}{Z} \sum_{s \in \mathcal{G}} \log(f_s^g + \varepsilon) \cdot Re(s) \tag{11}$$

  In this equation, Z is a normalization factor assuming all named entities are corrected segmented (*i.e.,* $Z = \sum_{s \in \mathcal{G}} \log(f_s^g + \varepsilon)$). The logarithm function is applied to avoid domination of few extremely frequent named entities, because of the power-law like distribution of the NEs (see Figure 4). ε is a small constant to avoid zero values for the named entities that have only single appearance ($\varepsilon = 0.001$ in our experiments).

Methods and Parameter Setting. We compare our *HybridSeg* method with *GlobalSeg* method in [12] as the baseline. Note that the method proposed in [12] is named *TwiNER* for named entity recognition and segmentation was one of its intermediate step. In this paper, *GlobalSeg* refers to the segmentation step in *TwiNER*.

The following three weak NERs are used in *HybridSeg* to derive the local context knowledge (*i.e.,* $m = 3$). Note that, we used the version downloaded from their corresponding websites. These NERs are not trained using our data.

- LBJ-NER is based on the regularized averaged perceptron approach for learning and inference. It uses gazetteers extracted from Wikipedia, word class models derived from unlabeled text corpus, and expressive non-local features [19].

- Stanford-NER is a NER system based on CRF model for learning and inference. Besides the conventional linguistic features, it also incorporates long-distance information [6].

- T-NER is a NER system designed for tweet corpus. It employs some tweet-specific features as well as the widely used conventional features for formal text [21].

For parameter settings, α in Eq. 6 is set to $\alpha = 0.2$; the top-K segments in Eq. 9 for λ adaption is set to $K = 50$. The search space for λ is set to be $[0, 0.95]$ with a step 0.05.

5.2 Segmentation Accuracy

In this section we first report the accuracy of the three weak NERs in detecting named entities. We then compare *HybridSeg* and *GlobalSeg* on segmentation accuracy.

Accuracy of three Weak NERs. The accuracy of the three weak NERs in recognizing named entities is evaluated by three standard

Table 3: Accuracy of the three weak NERs; $\mathcal{N}_\cap$ denotes the set of NEs detected by all three weak NERs. The best results are in boldface.

Method	*SIN*			*SGE*		
	Pr	*Re*	F_1	*Pr*	*Re*	F_1
LBJ-NER	0.335	0.357	0.346	0.674	0.402	0.504
T-NER	0.273	**0.523**	0.359	0.696	0.341	0.458
Stanford-NER	0.447	0.448	**0.447**	0.685	**0.623**	**0.653**
All NERs $\mathcal{N}_\cap$	**0.578**	0.233	0.332	**0.876**	0.192	0.315

metrics: Precision (*Pr*), Recall (*Re*), and F_1. *Pr* measures the percentage of the recognized named entities that are true named entities; *Re* measures the percentage of the true named entities that are correctly recognized; and $F_1 = 2 \cdot Pr \cdot Re/(Pr + Re)$ is the harmonic mean of *Pr* and *Re*. The type of the named entity (*e.g.,* person, location, and organization) is ignored as in [12]. Similar to the segmentation recall measure, each occurrence of a named entity in a specific position of a tweet is considered as one instance for the three measures *Pr*, *Re*, and F_1.

Table 3 reports the performance of the three weak NERs on the two datasets *SIN* and *SGE* respectively. Observed from the table, all three weak NERs perform poorly on the two tweets collections. The results are consistent with that reported in [12]. However, if consider the set of NEs that are recognized by all three NERs $\mathcal{N}_\cap$, the precision is much higher than any of the individual weak NER. In particular, the precision on *SGE* dataset is 0.876. That is, the set of NEs detected by all the three NERs are of good precision and can be used as local context knowledge in improving tweet segmentation. On the other hand, the recall of $\mathcal{N}_\cap$ is much lower than any of the three NERs as expected.

Segmentation Accuracy. We report two sets of results for *HybridSeg* and compare them with the baseline *GlobalSeg*. Specifically, in Table 4, we report the results of $HybridSeg_{NER}$ after learning from weak NERs, and the results of $HybridSeg_{NER+Iter}$ after the iterative learning converges in *HybridSeg*. We make three observations from the results:

1. Learning from the local context brought in by weak NERs, $HybridSeg_{NER}$ significantly improves tweet segmentation accuracy on both datasets;

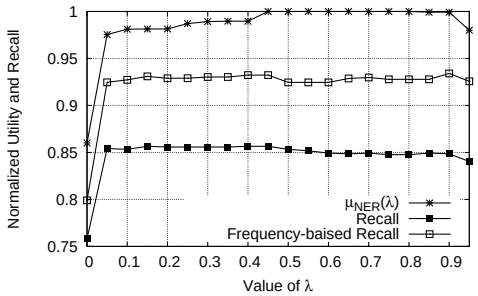
2. $HybridSeg_{NER+Iter}$ further improves *Re* and Re_F on both datasets. However, the amount of improvement on top of $HybridSeg_{NER}$ is relatively small on both datasets; and

3. Compared with the baseline method *GlobalSeg*, the proposed $HybridSeg_{NER+Iter}$ improves *Re* by 13.2% and 8.2% respectively on *SIN* and *SGE* datasets. Considering segment frequency in the evaluation using Re_F measure, much larger improvements are achieved, 17.6% on *SIN* and 17.1% on *SGE*.

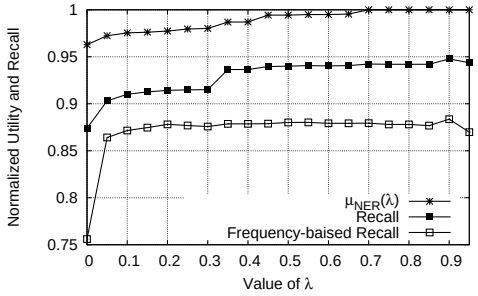
Our experimental results show that, incorporating local context knowledge in a batch of tweets greatly improves tweet segmentation accuracy. More importantly, the local context knowledge can be obtained with almost no cost with off-the-shelf NERs and locally derived statistics. On the other hand, the improvement made by learning from pseudo feedback is relatively small compared to the improvement made by learning from weak NERs. In the next section, we conduct a detailed analysis of *HybridSeg* for possible reasons.

Table 4: Performance of tweet segmentation algorithms, where $*$ indicates the difference against the baseline is statistically significant by paired t-test.

Method	SIN		SGE	
	Re	Re_F	*Re*	Re_F
GlobalSeg	0.758	0.799	0.874	0.756
$HybridSeg_{NER}$	0.857*	0.932*	0.942*	0.879*
$HybridSeg_{NER+Iter}$	**0.858***	**0.940***	**0.946***	**0.885***



(a) *SIN* dataset



(b) *SGE* dataset

Figure 5: *Re*, Re_F and $\mu_{NER}(\lambda)$ (%) values of $HybridSeg_{NER}$ with varying λ in the range of $[0, 0.95]$.

5.3 Algorithm Analysis

Impact of λ Adaption. We exploit the local context by using linear combination in the calculation of SCP score (rf. Eq. 7 and 8). The choice of λ largely affects the performance of the tweet segmentation process. While a small λ may not sufficiently exploit the local context, a very large λ could make the local context dominate the segmentation process which may adversely affect the segments with weak local context because of their limited number of occurrences.

Figure 5 demonstrates the impact of varying λ on $HybridSeg_{NER}$ in terms of *Re* and Re_F in the 0^{th} iteration (rf Eq. 10). For easy demonstration, we plot the normalized score obtained by Eq. 10 with different λ, denoted by $\mu_{NER}(\lambda)$ in the figure. Observe that $\mu_{NER}(\lambda)$ is positively correlated with the performance metrics *Re* and Re_F on both datasets. In our experiments, we set the parameter λ to be the smallest value leading to the best $\mu_{NER}(\lambda)$, *i.e.,* $\lambda = 0.5$ on *SIN* and $\lambda = 0.7$ on *SGE*. Because λ is a global factor for all member tweets in the batch and $\mu_{NER}(\lambda)$ is computed based on a small set of seed segments. A larger λ may not affect the segmentation of the seed segments because of their confident local context, but may cause some other segments to be wrongly split due to their noisy local context. Observe there is minor degradation for both *Re* and Re_F on *SIN* dataset when $\lambda > 0.45$ although $\mu_{NER}(\lambda)$ remains

Table 5: The performance of *HybridSeg*$_{NER+Iter}$ up to 4 iterations.

Iteration	SIN			SGE		
	Re	Re_F	*JSD*	*Re*	Re_F	*JSD*
0	0.857	0.932	–	0.942	0.879	–
1	0.857	0.940	0.0059	0.946	0.885	0.0183
2	0.858	0.940	0.0001	0.946	0.885	0.0003
3	0.858	0.940	0	0.946	0.885	0

the maximum. The impact of varying λ in the following iterations is not plotted for the interests of space. Similar observations hold.

Analysis of the Iterative Learning. *HybridSeg* employs an iterative strategy to learn from the segments produced in the previous iteration. Table 5 reports the performance of *HybridSeg* and *JSD* between the frequency distributions of the segments from two iterations.

It is observed that *HybridSeg* converges after the second iteration. To understand the possible reasons for the quick convergence, we analyze the segments detected in each iteration. There are three categories of them:

- Fully detected segments (FS): all occurrences of the segments are detected from the batch of tweets. Their $Pr(s)$ is further increased by considering their local context. No more occurrences can be detected on this category of segments in the next iteration.
- Missed segments (MS): not a single occurrence of the segment is detected from the previous iteration. In this case, no local context information can be derived for them to increase their $Pr(s)$. They will be missed in the next iteration.
- Partially detected segments (PS): some but not all occurrences of the segments are detected. For this category of segments, local context can be derived from the detected occurrences. Depending on the local context, $Pr(s)$ will be adjusted. More occurrences may be detected or missed in the next iteration.

Table 6 reports the number of segments and their number of occurrences in each of the three sets (FS, MS, and PS). For partially detected segments, the number of detected and missed occurrences are reported.

As shown in the table, very few segments are partially detected after learning from weak NERs in 0^{th} iteration (19 for *SIN* and 24 for *SGE*). The possible improvement can be made in the next iteration is to further detect the total 25 missed occurrences in *SIN* (resp. 67 in *SGE*), which accounts for 2.03% (resp. 1.64%) of all annotated NEs in the dataset. That is, the room for further performance improvement by iterative learning is marginal.

Consider the *SIN* dataset, on average, there are about 6 detected occurrences to provide local context for each of the 19 partially detected segments. With the local context, *HybridSeg* manages to reduce the number of partially detected segments from 19 to 11 in the next iteration and the total number of their missed instances are reduced from 25 to 14. No changes are observed for the remaining 11 partially detected segments in iteration 2. Interestingly, the number of fully detected instances increased by 2 in the last iteration. The best segmentation of a tweet is the one maximizes the stickness of its member segments (rf Eq. 1). The change in the stickness $C(s)$ of other segments leads to the detection of these two new segments in the fully detected category, each occurs once in the dataset.

In *SGE* dataset, the 24 partially detected segments reduce to 12 in the next iteration by learning from pseudo feedback. No change to these 12 partially detected segments are observed in the next iteration, despite some of these segments have very strong local context based on their detected occurrences. A manual investigation show that the missed occurrences are wrongly detected as part of some other longer segments. For example, "NSP"[5] becomes part of "NSP Election Rally" and the latter is not annotated as a named entity. A further investigation shows that, probably based on its capitalization, "NSP Election Rally" is detected by weak NERs with strong confidence (*i.e.,* all its occurrences are detected). Because of its strong confidence, "NSP" cannot be separated from this longer segment in the next iteration regardless the setting of λ. We note that although "NSP Election Rally" is not annotated in the ground truth as named entity, it is indeed a semantically meaningful phrase. On the other hand, a large portion of the occurrences for these 12 partially detected segments have been successfully detected from other tweets.

Compared to the baseline *GlobalSeg* which does not take local context, *HybridSeg* significantly reduces the number of missed segments from 195 to 152, which is 22% reduction on *SIN* dataset. On *SGE* dataset, the reduction is 20% from 140 to 112. Many of these segments are fully detected in *HybridSeg*.

5.4 Application: Named Entity Recognition

In this section, we use Named Entity Recognition as a downstream application to evaluate the impact of tweet segmentation. We first briefly review the *TwiNER* method for NER [12] and then propose our own NER method taking tweet segment as input.

5.4.1 NER by Random Walk

TwiNER recognizes named entities by applying Random Walk on the segments from a batch of tweets. The main assumption is that a named entity often co-occurs with other named entities in a batch of tweets while non-entity segments rarely co-occurs with named entities. The weight of the co-occurrence between two segments is measured by Jaccard Coefficient. A random walk model is then applied to the segment graph. Let ρ_s be the stationary probability of segment s by applying random walk, the segment is then weighted by:

$$y(s) = e^{Q(s)} \cdot \rho_s \tag{12}$$

In this equation, $e^{Q(s)}$ carry the same semantic as in Eq. 2, indicating that a segment that frequently appears in Wikipedia as an anchor text is more likely to be a named entity. With the weighting $y(s)$, the top K segments are considered as named entities.

5.4.2 NER by POS Tagger

Due to the short nature of tweets, the *gregarious* property could be very weak in tweets. As shown in Table 2, 82% of the annotated named entities appear only once in *SIN* (and 61% in *SGE*). We choose to explore the part-of-speech tags in tweets for named entity recognition by considering noun phrases as named entities.

We estimate the likelihood of a segment being a noun phrase (NP) by considering the POS tags of its constituent words. Table 7 lists three POS tags that are considered as the indicators of a segment being a noun phrase. Let $w^s_{i,j}$ be the j^{th} word of segment s in its i-th occurrence, we calculate the probability of segment s being an noun phrase as follow:

$$\hat{P}_{NP}(s) = \frac{\sum_i \sum_j [w^s_{i,j}]}{|s| \cdot f_s} \cdot \frac{1}{1 + e^{-5\frac{(f_s - \bar{f}_s)}{\sigma(f_s)}}} \tag{13}$$

[5] http://en.wikipedia.org/wiki/National_Solidarity_Party_(Singapore)

Table 6: Fully detected, missed, and partially detected segments for *GlobalSeg* and *HybridSeg* (3 iterations). #NE: number of distinct segments, #Occur: number of occurrences, #Detect: number of detected occurrences, #Miss: number of missed occurrences.

Dataset	*SIN* dataset							*SGE* dataset						
Method/	Fully detected		Missed		Partially detected			Fully detected		Missed		Partially detected		
Iteration	#NE	#Occur	#NE	#Occur	#NE	#Detect	#Miss	#NE	#Appr	#NE	#Occur	#NE	#Detect	#Miss
0	581	944	146	152	19	113	25	295	1464	94	168	24	2374	67
1	581	959	154	163	11	98	14	291	1858	110	191	12	1996	28
2	583	961	152	161	11	98	14	289	1856	112	193	12	1996	28
GlobalSeg	504	647	195	214	47	113	85	234	708	140	336	40	2850	179

Table 7: Three POS tags as the indicator of a segment being a noun phrase, reproduced from [7]

Tag	Definition	Examples
N	common noun (NN, NNS)	books; someone
^	proper noun (NNP, NNPS)	lebron; usa; iPad
$	numeral (CD)	2010; four; 9:30

This equation considers two factors. The first factor estimates the probability as the percentage of the constituent words being labeled with an NP tag for all the occurrences of segment s, where $[w]$ is 1 if w is labeled as one of the three POS tags in Table 7, and 0 otherwise; For example, "chiam see tong", the name of a Singaporean politician and lawyer[6], is labeled as ^^^ (66.67%), NVV (3.70%), ^V^ (7.41%) and ^VN (22.22%)[7]. By considering the types of all words in a segment, we can obtain a high probability of 0.877 for "chiam see tong". The second factor of the equation introduces a scaling factor to give more preference to frequent segments, where $\bar{f}_s$ and $\sigma(f_s)$ are the mean and standard deviation of segment frequency.

The segments are then ranked by $y(s) = e^{Q(s)} \cdot \hat{P}_{NP}(s)$, *i.e.,* replacing ρ_s in Eq 12 by $\hat{P}_{NP}(s)$.

With two named entity recognition approaches, namely Random Walk (RW) and POS tagging respectively, and two segmentation methods, namely *GlobalSeg* and *HybridSeg*, we have four combinations to evaluate: $GlobalSeg_{RW}$, $HybridSeg_{RW}$, $HybridSeg_{POS}$, and $GlobalSeg_{POS}$. Note that, $GlobalSeg_{RW}$ is the same method named *TwiNER* in [12].

We introduce another baseline method $Unigram_{POS}$, which use POS Tagging without segmentation. Similar to the work in [3], we extract noun phrases from the batch of tweets by using the following regular expression.

$$NP := [\hat{}N][\hat{}N\$]*$$

The regular expression states that a noun phrase can be a combination of common noun, proper noun and numeral, which begins with common or proper noun. The confidence of a noun phrase is computed using a modified version of Eq. 13 by removing its first component.

In summary, we have five methods to evaluate: $GlobalSeg_{RW}$, $HybridSeg_{RW}$, $HybridSeg_{POS}$, $GlobalSeg_{POS}$, and $Unigram_{POS}$.

5.4.3 Experimental Results

Table 8 reports the performance of the 5 methods. The results reported is the highest F_1 of each method achieved for varying $K > 50$ following the same setting in [12]. Three observations are made.

1. Tweet segmentation greatly improves NER. $Unigram_{POS}$ is the worst performer on all three evaluation metric Pr, Re, F_1 among all methods.

2. For a specific NER approach, either Random Walk or POS based, better segmentation results lead to better NER accuracy. That is, $HybridSeg_{RW}$ performed better than $GlobalSeg_{RW}$ and $HybridSeg_{POS}$ performed better than $GlobalSeg_{POS}$ on all evaluation metrics.

3. Without local context in segmentation $GlobalSeg_{POS}$ is slightly worse than $GlobalSeg_{RW}$ by F_1. However, with much better segmentation results, $HybridSeg_{POS}$ is much better than $HybridSeg_{RW}$. By F_1 measure, $HybridSeg_{POS}$ achieves the best NER result.

Table 8: The performance of *GlobalSeg* and *HybridSeg* with Random Walk and POS for NER

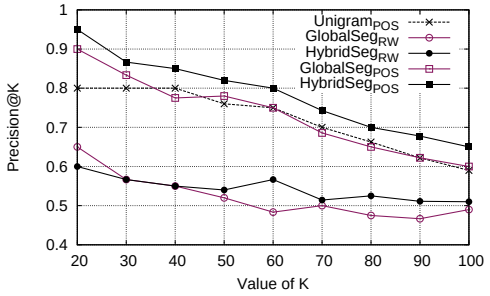
Method	*SIN* dataset			*SGE* dataset		
	Pr	*Re*	F_1	*Pr*	*Re*	F_1
$Unigram_{POS}$	0.516	0.190	0.278	0.845	0.333	0.478
$GlobalSeg_{RW}$	0.576	0.335	0.423	**0.929**	0.646	0.762
$HybridSeg_{RW}$	0.618	0.343	0.441	0.907	0.683	0.779
$GlobalSeg_{POS}$	0.647	0.306	0.415	0.903	0.657	0.760
$HybridSeg_{POS}$	**0.685**	**0.352**	**0.465**	0.911	**0.686**	**0.783**

Figure 6 plots the *Precision@K* for the 5 methods on the two datasets with varying K from 20 to 100. The *Precision@K* reports the ratio of named entities among the top-K ranked segments by each method. Note that, *Precision@K* measures the ranking of the segments detected from a batch of tweets; the occurrences of each segment in the ranking are not considered. This is different from the measures (*e.g., Pr*) reported in Table 8 where the occurrences of the named entities are considered (*i.e.,* whether a named entity is correctly detected at a specific position in a given tweet). We made the following observations from Figure 6.

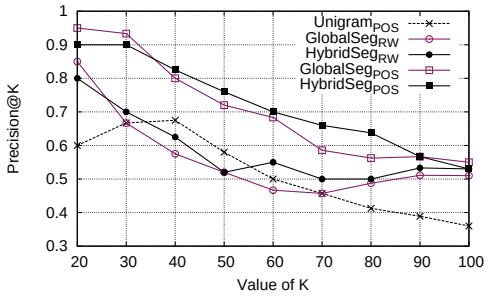
On *SIN* dataset, it is clear that all methods using POS tagging for NER enjoy much better precision. RW based methods deliver much poorer precisions due to the lack of co-occurrences in the tweets. As reported in Table 2, 82% of the annotated named entities appear only once in *SIN*. Among the three POS based methods, $HybridSeg_{POS}$ dominates the best precisions on all K values from 20 to 100. On *SGE* dataset, the differences in precisions between POS based methods and RW based methods become smaller compared to those on *SIN* dataset. The reason is that in *SGE* dataset, about 39% of named entities appear more than once, which makes a much larger chance of co-occurrences. Between the two best performing methods $HybridSeg_{POS}$ and $GlobalSeg_{POS}$, the former outperformed the latter on 6 K values plotted between 40 and 90. Without segmentation $Unigram_{POS}$ performed poorly in terms of *Precison@K* measure on *SGE* dataset.

[6] http://en.wikipedia.org/wiki/Chiam_See_Tong

[7] V:verb including copula, auxiliaries; for example, might, gonna, ought, is, eats.



(a) SIN



(b) SGE

Figure 6: Precision@K on two datasets

6. CONCLUSION

Recently, tweet segmentation has been proven to be effective in the tasks of NER, event detection, and summarization. In this paper, we present a novel framework *HybridSeg*, which aggregates both the local context knowledge and the global knowledge bases in the process of tweet segmentation. First, *HybridSeg* exploits the local linguistic features in a collective manner by using the existing NER tools. The recognized named entities with high confidence positively enhance the performance of tweet segmentation. Moreover, information that has not been well captured in the global knowledge bases is derived based on all segments from the tweets. Then, *HybridSeg* iteratively learns a better segmentation by considering confident seed segments in the previous iteration. Through extensive experiments, we show that *HybridSeg* significantly outperforms the existing state-of-the-art algorithm on tweet segmentation. We also show that the better segmentation benefits the named entity recognition application in tweets. In future, we plan to take more local factors into consideration for tweet segmentation, such as local word dependency.

7. REFERENCES

[1] D. Beeferman, A. Berger, and J. Lafferty. Statistical models for text segmentation. *Mach. Learn.*, 34(1-3):177–210, Feb. 1999.

[2] F. Y. Y. Choi. Advances in domain independent linear text segmentation. In *NAACL*, pages 26–33, 2000.

[3] F. C. T. Chua, W. W. Cohen, J. Betteridge, and E.-P. Lim. Community-based classification of noun phrases in twitter. In *CIKM*, pages 1702–1706, 2012.

[4] J. E. Chung and E. Mustafaraj. Can collective sentiment expressed on twitter predict political elections? In *AAAI*, 2011.

[5] A. Cui, M. Zhang, Y. Liu, S. Ma, and K. Zhang. Discover breaking events with popular hashtags in twitter. In *CIKM*, pages 1794–1798, 2012.

[6] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370, 2005.

[7] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. Part-of-speech tagging for twitter: annotation, features, and experiments. In *ACL-HLT*, pages 42–47, 2011.

[8] B. Han and T. Baldwin. Lexical normalisation of short text messages: Makn sens a #twitter. In *ACL*, pages 368–378, 2011.

[9] M. A. Hearst. Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64, Mar. 1997.

[10] A. Kazantseva and S. Szpakowicz. Linear text segmentation using affinity propagation. In *EMNLP*, pages 284–293, 2011.

[11] C. Li, A. Sun, and A. Datta. Twevent: segment-based event detection from tweets. In *CIKM*, pages 155–164, 2012.

[12] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. Twiner: Named entity recognition in targeted twitter stream. In *SIGIR*, pages 721–730, 2012.

[13] K.-L. Liu, W.-J. Li, and M. Guo. Emoticon smoothed language models for twitter sentiment analysis. In *AAAI*, 2012.

[14] X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. In *ACL*, pages 359–367, 2011.

[15] X. Liu, X. Zhou, Z. Fu, F. Wei, and M. Zhou. Exacting social events for tweets using a factor graph. In *AAAI*, 2012.

[16] Z. Luo, M. Osborne, and T. Wang. Opinion retrieval in twitter. In *ICWSM*, 2012.

[17] X. Meng, F. Wei, X. Liu, M. Zhou, S. Li, and H. Wang. Entity-centric topic-oriented opinion summarization in twitter. In *KDD*, pages 379–387, 2012.

[18] H. Misra, F. Yvon, J. M. Jose, and O. Cappe. Text segmentation via topic modeling: an analytical study. In *CIKM*, pages 1553–1556, 2009.

[19] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155, 2009.

[20] M. Riedl and C. Biemann. Topictiling: a text segmentation algorithm based on lda. In *ACL 2012 Student Research Workshop*, pages 37–42, 2012.

[21] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *EMNLP*, pages 1524–1534, 2011.

[22] A. Ritter, Mausam, O. Etzioni, and S. Clark. Open domain event extraction from twitter. In *KDD*, pages 1104–1112, 2012.

[23] M. Utiyama and H. Isahara. A statistical model for domain-independent text segmentation. In *ACL*, pages 499–506, 2001.

[24] K. Wang, C. Thrasher, E. Viegas, X. Li, and P. Hsu. An overview of microsoft web n-gram corpus and applications. In *HLT-NAACL*, pages 45–48, 2010.

[25] X. Wang, F. Wei, X. Liu, M. Zhou, and M. Zhang. Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *CIKM*, pages 1031–1040, 2011.

[26] G. Zhou and J. Su. Named entity recognition using an hmm-based chunk tagger. In *ACL*, pages 473–480, 2002.