Automatic News Source Detection in Twitter Based on Text Segmentation

Takashi Inui **Masaki Saito*** **Mikio Yamamoto**
Graduate School of Systems and Information Engineering
University of Tsukuba
1-1-1 Tenoudai, Tsukuba, Ibaraki 305-8573, JAPAN
{inui@,masaki@mibel.,myama@}cs.tsukuba.ac.jp

Abstract

In this paper, we discuss news source detection (NSD), which involves finding additional information of a message generated in social media to understand the original message more deeply. We propose an NSD method based on the text segmentation and two extension models using web content and post times. Through the experiments using the real-world data, the proposed methods outperformed the baseline methods and exhibited an F-measure of 34.9.

1 Introduction

Recently, with the advent of social-media, it has become easy to express opinions or comment about experiences. In particular, *Twitter*[1] is a popular service used worldwide, and extremely large number of messages (*tweets*) is generated every day on it. It has been widely recognized that Twitter can potentially contain much useful information. Therefore, many researchers have conducted content analysis on Twitter (Java et al., 2006; Krishnamurthy et al., 2008; Pennacchiotti and Gurumurthy, 2011; Mehrotra et al., 2013).

Twitter can be regarded as a news feeder (Zhao et al., 2011). News content distributed by other media are often re-distributed and diffused to more people through Twitter. For example, a user *X* posted a tweet as follows.

t_{ex}: *Goal! Mario! http://example.football.com*

Many people have a chance to know the details of Mario's fantastic goal[2] through t_{ex}. Web content included in the URL *http://example.football.com* functions as an information source on t_{ex}. It can be said that tweets, such as t_{ex}, contain suitable information for news feeders. However, such cases are rare. Almost all tweets on Twitter are unsuitable due to a variety of reasons, e.g. (i) *X* did not write the information source in her stream of tweets, (ii) a tweet message and its information source (URL) were written in separate tweets, or (iii) *X* included a URL that was not related to the tweet message. In these cases, tweets do not function as the news feeders and people cannot obtain any additional information from them.

We discuss news source detection (NSD), which involves finding additional information of a message generated on social media to understand the original message more deeply. In Twitter, given a tweet t_i, the goal with NSD is to find another tweet t_j ($\neq t_i$) that includes a reference to its information source on t_i. The details of NSD are described in Section 2. We propose an NSD method based on the text segmentation. It is difficult to straightforwardly resolve NSD because a search space of tweet pair combinations is exponentially large. Therefore, we simplify the NSD problem from the viewpoint of the text segmentation and provide an approximate solution. We also discuss two extension models of the proposed method using web content and post times.

*Currently, Fujitsu Limited.
[1]Twitter. https://twitter.com/

[2]Mario Götze is a German footballer who scored a goal at the final game at the FIFA Brazil World Cup.

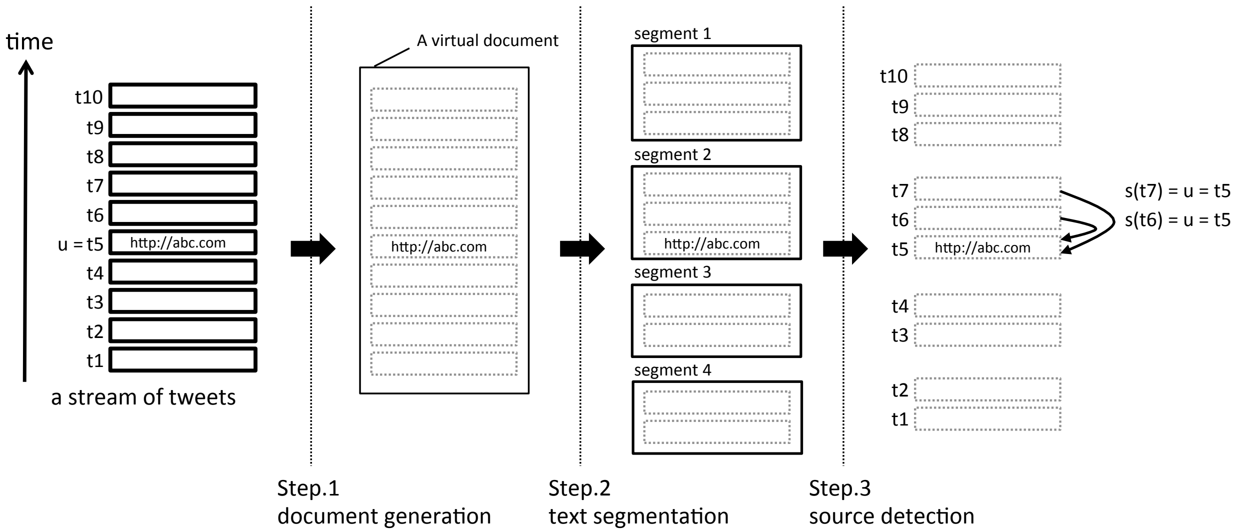

Figure 1: News source detection based on text segmentation

The rest of the paper is organized as follows. First, we define NSD and introduce some concepts and their notations for a formal description of NSD in Section 2. We then propose an NSD method that is based on the text segmentation and also discuss two extensions of the proposed method in Section 3. In Section 4, we introduce related work and discuss the differences between them. In Section 5, we describe the details of the experiments using real-world data and argue that the proposed method performs better than the baseline methods. We summarize the paper in Section 6.

2 News Source Detection

First, we introduce some concepts and their notations for a formal description of NSD.

- *target tweet* (t): a tweet for finding the information source. We call the information source especially, *news source*, hereafter.
- *source tweet* ($s(t)$): a tweet that includes a reference to the news source on t. In this paper, we only consider URL strings included in tweets as references.
- *URL tweet* (u): a tweet including a URL string.

Given a stream of tweets $T = \langle t_1, t_2, ..., t_{|T|}\rangle$ that includes at least one u, the task of NSD is to detect whether u is a source tweet on t_i for each t_i except u.

3 Proposed Methods

3.1 NSD based on Text Segmentation

We found two valuable findings in our preliminary analysis.

- A u adjacent to a t tends to be a $s(t)$ on t ($u = s(t)$).
- Two target tweets, t_i and t_j, adjacent to each other tend to have the same source tweet ($s(t_i) = s(t_j)$).

From these findings, we use *text segmentation*, which is one of the fundamental tasks in the NLP research domain. The goal with the text segmentation problem is to divide an input document into parts based on subtopics held in the input document.

We designed an algorithm to solve NSD as follows and illustrated in Figure 1.

Step.1 document generation. A stream of tweets is regarded as a virtual document.

Step.2 text segmentation. The document is divided into some segments by using a text segmentation method.

Step.3 source detection. The u is detected as a source tweet on t ($u = s(t)$) if and only if a u and t in the document belong to the same segment.

From a technical viewpoint, the text segmentation problem in Step.2 is the core part of this algorithm. We explain the details of Step.2 in the next section.

3.2 Applying TextTiling

3.2.1 TextTiling

We used a modified version of the text segmentation algorithm called TextTiling (Hearst, 1997), which is a well-known and standard text segmentation method, and is focused on adjacent sentence pairs. Suppose that s_i and s_j is an adjacent sentence pair in the input document, then, TextTiling determines whether s_i and s_j belong to the same segment or not according to a boundary score[3]. If the sentence boundary sb_{ij} between s_i and s_j has a lower boundary score than the threshold d_{th}, the sentence pair is detected as belonging to the same segment; otherwise, it is not. As a result, text segmentation in the input document is naturally done when all sentence boundaries are determined.

A boundary score d_{ij} held on the sentence boundary sb_{ij} is defined as follows:

$$d_{ij} = (ss_l - ss_{ij}) + (ss_r - ss_{ij}) \tag{1}$$

where ss_{ij} indicates a similarity score at sb_{ij} and ss_l (ss_r) indicates a similarity score at a local maximum point on the left(right)-hand side of sb_{ij}. Each similarity score ss_{ij} is defined as follows:

$$\sum_{w \in L} \frac{f(w, c_i^f) f(w, c_j^b)}{\sqrt{\sum_{w \in L} f(w, c_i^f)^2 \sum_{w \in L} f(w, c_j^b)^2}} \tag{2}$$

where c_i^f and c_j^b indicate context windows, where c_i^f indicates a forward window and c_j^b indicates a backward window (see Figure 2). The symbol L indicates a lexicon set.

The function $f(w, c_i^f)$ returns the number of occurrences of a word w in the context window c_i^f and $f(w, c_j^b)$ likewise. Intuitively, this score represents a topical coherence between c_i^f and c_j^b. The higher the ss_{ij}, the stronger the coherence.

[3]This is called the "depth" score in (Hearst, 1997).

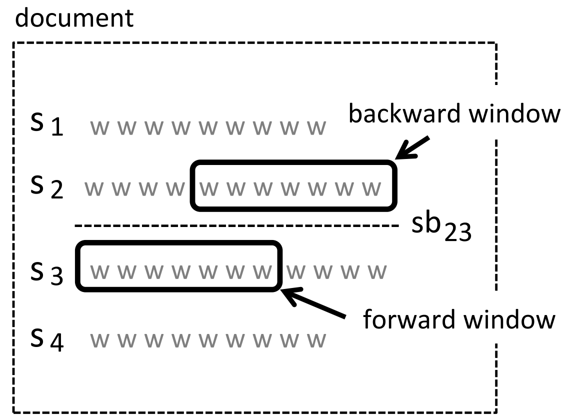

Figure 2: Backward and forward windows

Actually, d_{ij} is only measured at each local minimum point of ss_{ij} and compared with d_{th}. The d_{th} to the boundary score is defined as $d_{th} = \overline{S} - \frac{\sigma}{2}$. Here, $\overline{S}$ indicates an average value of all boundary scores and σ indicates their standard deviation.

3.2.2 Modifications

We introduce three modifications to the original TextTiling algorithm to appropriately apply it to a virtual document composed of a stream of tweets.

First, we focus on tweet boundaries instead of sentence boundaries because we want to make segments in units of tweets.

Second, we add another type of context window. The word-based window is only defined in the original algorithm. Figure 2 shows an example of the word-based window of size 7. We also use the post-based window. With the post-based window, the number of words to be included in the window varies with the length of each tweet. Therefore, we can include more meaningful context into the boundary scores.

The third is a normalization of the similarity scores. Our stream data are much shorter than those assumed in the original TextTiling algorithm. Therefore, it was frequently observed that the number of words is less than the window size at the end of the stream when using the word-based window.

We therefore prepared a normalized similarity score function to resolve this problem. The normalized score function is defined as follows.

$$\sum_{w \in L} \frac{\frac{f(w,c_i^f)}{|c_i^f|}\frac{f(w,c_j^b)}{|c_j^b|}}{\sqrt{\sum_{w\in L}\left(\frac{f(w,c_i^f)}{|c_i^f|}\right)^2 \sum_{w\in L}\left(\frac{f(w,c_j^b)}{|c_j^b|}\right)^2}} \quad (3)$$

Here, each $|c_i^f|$ and $|c_j^b|$ indicates the real number of words existing in c_i^f and c_j^b.

We call the modified algorithm described in this section **Basic** for comparing it to the extensions described in the next section.

3.3 Extension1: Web Content Concatenation (WCC)

It was found that there are many URL tweets with insufficient information to detect source tweets because they are composed of very few words. Therefore, we consider enriching URL tweets with web content referred by the URL written in them.

Suppose that $web(u)$ is web content referred by a URL written in a u. Then, we simply concatenate $web(u)$ with u and use both strings $web(u)$ and u in **Basic**. Web pages are generally composed of logical constituents such as *title*, *head*, and *body*. Some might contribute to the source detection, and some might not. We selected content in *title* and *body* as $web(u)$ in the experiments. A specific pattern rule based on HTML tags was used for extracting the main document parts from *body* in the Web pages.

We call this extension technique web content concatenation (**WCC**).

3.4 Extension2: Using Post Time (PT)

Intuitively, it seems that arbitrary tweet pairs have semantic relationships each other when they are sequentially posted in a very short span. On the other hand, it seems that they have no semantic relationships when posted in a longer span. Based on this insight, we introduce a weighted frequency function by using time span information between two tweets. Equation (4) represents the alternative weighted frequency function $f'(w, c_i^f)$, which is used in Equation (2) and Equation (3) instead of $f(w, c_i^f)$.

$$f'(w, c_i^f) = \sum_{e \in \mathcal{W}} \max\{0, 1 - \delta(e, c_i^f)\} \quad (4)$$

The set $\mathcal{W}$ indicates an instance set of w existing in c_i^f, and the symbol e indicates an element in $\mathcal{W}$. That is, $f(w, c_i^f) = |\mathcal{W}|$. The $\delta(e, c_i^f)$ is a penalty term and defined as follows:

$$\delta(e, c_i^f) = log(T(t_f^e) - T(t_b^0)). \quad (5)$$

Here, $T(t_*^*)$ indicates the time at which t_*^* was posted. The tweet t_f^e indicates a tweet in which a word instance e exists in the forward window. The tweet t_b^0 indicates a tweet in the backward window and adjacent to a tweet in the forward window. For example, when t_b^0 was posted at 09:15 and t_f^e was posted at 09:18, $\delta(e, c_i^f) = log(3) = 0.477$ because t_f^e was posted 3 minutes later from t_b^0. The $f'(w, c_j^b)$ is defined, likewise.

We call this extension technique post time (**PT**).

4 Related work

In this section, we discuss two NLP tasks related to NSD; first story detection (FSD) and document alignment (DA), then, discuss the differences between them. Figure 3 shows the outlines of the three tasks. Note that the only central phenomena are drawn in this figure. One can return to the original papers referred to the explanation below to understand the strict definition for each task.

First story detection is a subtask defined within Topic Detection and Tracking[4](Allen, 2002). The aim with FSD is detecting a news manuscript reporting a given topic for the first time from a stream of news stories. The topics given in FSD are worldwide events or disasters such as the Oklahoma City bombing and the earthquake in Kobe. Traditional techniques used in FSD are similarity-based methods. A news manuscript is detected as the *first story* when it is not similar to all past news. Petrovic et al. (2010) investigated the FSD task on Twitter. They modified the traditional FSD technique to tackle the speed and volume problems due to the tremendous updates of data generated on Twitter. They used a streaming technique based on locality sensitive hashing (Indyk and Motwani, 1998) which makes high-speed approximate calculations of similarities possible and achieves good performance.

[4]For more details, see http://www.itl.nist.gov/iad/mig//tests/tdt/.

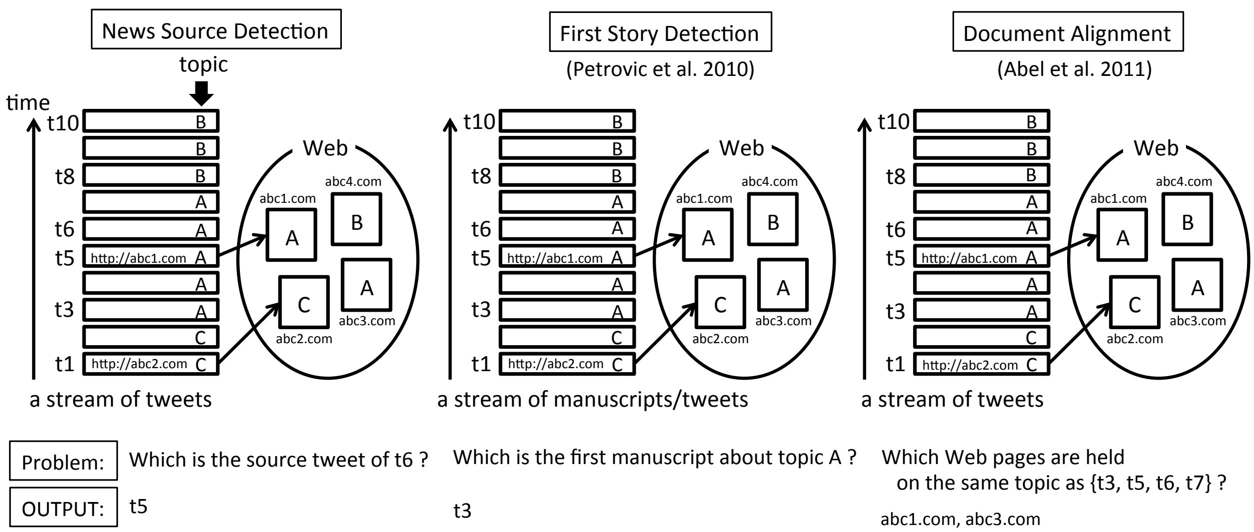

Figure 3: Differences in task definitions

Abel et al. (2011) proposed a DA method for automatically acquiring Twitter-user profiles. The goal of the user profile acquisition for a user A is to create a set of semantic entities composing text content indicating entities in the real world, such as persons and events[5], from text context A generated. For example, suppose that A's hobby is tennis and she posts something about tennis such as "French open (*event*)" and Italian tennis player "Francesca Schiavone (*person*)" on Twitter. Then A's user profile could be composed of "French open" and "Francesca Schiavone". Abel et al.(2011) adapted DA between tweets and web pages to enrich user profiles to be acquired. The aim with DA is to find all web pages aligned with the input tweets in terms of topics. In DA, all web pages are aligned with input tweets that have the same topic as the web pages. To resolve DA, they used explicit URL linkages and implicit linkages estimated using TFIDF-based similarity between tweets and web pages.

The above-mentioned research has an affinity to NSD. However, the definition of the problem(input)/output relation slightly differs in each study as shown in Figure 3. Moreover, the interest for our study was to investigate the effectiveness of the two aspects, web content and posting time of tweets, to improve NSD performance, which garnered no interest in the previous studies.

In Twitter, the *hashtag* "#" symbol is used to mark keywords or topics in a tweet. Users can mark categories of content written in tweet messages by using hashtags such as #Fashion, #Food, and #WorldCup2014. Unfortunately, they are unsuitable for NSD because categories obtained through hashtags are usually very coarse. In fact, to use hashtags for NSD, we conducted an experiment that involved the same conditions as those described in the next section and achieved a very low F-measure of 8.0.

5 Experiments

5.1 Data

We selected *SportsNavi* (`http://sports.yahoo.co.jp/`) as a news source in the experiments and crawled web pages belonging to *SportsNavi*. This site is a popular Japanese sports news sites provided by Yahoo!.

We collected 317 streams of tweets by using the TwitterAPI[6]. All tweets collected were written in Japanese. Furthermore, we required that at least one u be included for each stream of tweets. Such a tweet has a URL string referring to a web page belonging to *SportsNavi*. Of these collected stream

[5]For semantic entities, see also the OpenCalais project `http://www.opencalais.com/`.

[6]`https://dev.twitter.com/docs`

data, we focused on a set of tweet pairs $\langle u, t\rangle$ in which t exist within five tweets from u in the stream then used 3,170 $\langle u, t\rangle$ pairs as our evaluation data. The problem to be solved in the experiments was detecting whether u is the source tweet on t for each $\langle u, t\rangle$ in the evaluation data.

We asked two annotators to create a gold standard dataset. The annotators were required to independently judge whether u into $\langle u, t\rangle$ in the evaluation data is regarded as a source tweet on t. We measured the κ statistics (Cohen, 1960) to assess the reliability of the gold standard dataset. The result is that $\kappa = 0.782$. This value indicates that the data substantially agree.

5.2 Baseline methods

We adopted two baseline methods for comparison with the proposed methods. **Naive** is the most naive method and **SIM** is a customized version of the method (Abel et al., 2011) proposed to resolve DA described in Section 4.

Naive For all tweet pairs in the evaluation data, the u in $\langle u, t\rangle$ is always detected as $s(t)$ on t.

SIM This is a similarity-based method originally proposed by (Abel et al., 2011). Suppose that $\mathcal{U}$ indicates a set of URL tweets in the evaluation data and $web(u)$ indicates a web page referred from a URL written in u ($\in \mathcal{U}$). SIM focuses on each similarity between t and a web page $web(u')$ ($u' \in \mathcal{U}$) to detect whether $u = s(t)$, that is, the u in $\langle u, t\rangle$ is the $s(t)$ on t. First, given t in $\langle u, t\rangle$, u_o is selected using Equation (6).

$$u_o = \arg\max_{u' \in \mathcal{U}} sim(t, web(u')) \tag{6}$$

After that, u is detected as a source tweet on t only when $u_o = u$; otherwise, it is not. We used Equation (7) as the similarity function $sim(t, web(u'))$, which is the same setting as (Abel et al., 2011).

$$\sum_{i \in \mathcal{T}} TF(i, web(u')) * IDF(i) \tag{7}$$

where $\mathcal{T}$ is a set of words included in t, $TF(i, web(u'))$ indicates the term frequency of i in $web(u')$, and $IDF(i)$ indicates the inverse document frequency in terms of web pages in the evaluation data.

5.3 Other settings

We used the Japanese morphological analyzer *MeCab*[7] for word recognition. It is observed that each tweet in the evaluation data is composed of an average of six words.

We conducted our experiments by changing the size of the context window used in the text segmentation phase. We set up sizes from 1 to 15 for the word-based window and from 1 to 2 for the post-based window. We used only nouns as a lexicon set L.

We used Precision and Recall as evaluation measures, which are defined as

$$Precision = \frac{|X \cap Y|}{|X|} * 100,$$
$$Recall = \frac{|X \cap Y|}{|Y|} * 100.$$

The symbol X indicates a set of $\langle u, t\rangle$ instances in which the u in $\langle u, t\rangle$ is detected using a method as the source tweet on t and Y indicates a set of $\langle u, t\rangle$ instances in which the u in $\langle u, t\rangle$ is actually source tweet on t. We also used F-measure index $\frac{2*Precision*Recall}{Precision+Recall}$ as a summary of the above measures.

5.4 Experimental Results

5.4.1 Results of proposed method: Basic

We start by discussing the results of the simplest method proposed in Section 3, which we call **Basic**. We discuss the results obtained using the extended models of **Basic** in the next section.

Table 1 lists the results of **Basic**. The results from which the word-based window was used in the text segmentation are shown in the upper part of Table 1 and those from the post-based window are shown in the lower part. With the word-based window, Precision dropped when the window size was larger. Recall, on the other hand, tended to increase when the window size was larger. Similar phenomena were observed with the post-based window. The best F-measure value was 29.5, obtained when the size of

[7] `https://code.google.com/p/mecab/`

Table 1: Results of proposed method (**Basic**)

word-based window			
window size	Precision	Recall	F-measure
1	**100.0**	0.3	0.6
2	35.3	1.9	3.6
3	40.5	10.7	17.0
4	31.7	18.3	23.2
5	23.8	23.0	23.4
6	25.8	34.4	**29.5**
7	20.2	33.4	25.2
8	18.8	37.2	25.0
9	18.2	38.2	24.6
10	17.1	**38.8**	23.7
11	8.9	21.5	12.6
12	7.7	18.6	10.9
13	8.8	21.1	12.5
14	8.1	19.9	11.5
15	8.8	21.1	12.5
post-based window			
window size	Precision	Recall	F-measure
1	**35.2**	21.8	**26.9**
2	19.5	**29.7**	23.5

Table 2: Comparison with baseline methods

	Precision	Recall	F-measure
Naive	9.1	100.0	16.6
SIM	76.5	8.2	14.8
Basic (6)	25.8	34.4	29.5

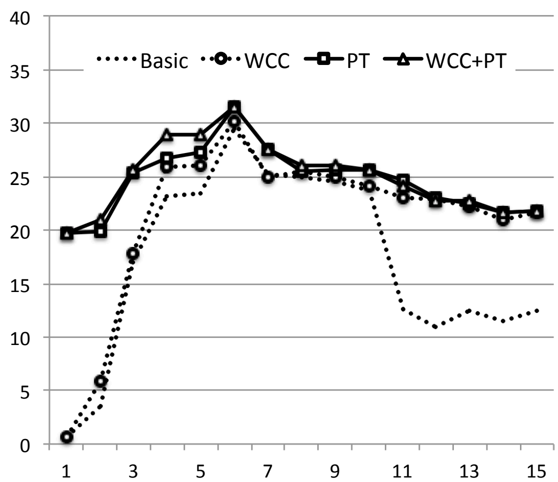

Figure 4: F-measure values from proposed methods

word-based window was 6, and 26.9, obtained when the size of the post-based window was 1.

Next, we compare **Basic** with the baseline methods. Table 2 lists the results obtained from the baseline methods. The best result obtained from **Basic** with the word-based window of size 6 is also shown in the bottom of Table 2. **Naive** naturally achieved 100% Recall while Precision was very low (9.1%). **SIM** had a contrary phenomenon to **Naive**, low Recall (8.2%) and high Precision (76.5%), since it would induce conservative decision-making by Equation (6). One can see that **Basic** achieved a well-balanced performance and higher F-measure than the baseline methods.

5.4.2 Effectiveness of extensions

We investigated the effectiveness of the two extensions, **WCC** discussed in Section 3.3 and **TP** discussed in Section 3.4. First, we discuss the results of **WCC** and then discuss those of **PT**.

Table 3 lists the results obtained from **WCC**. **WCC** outperformed **Basic** when larger windows were used. This is because **WCC** was able to make good use of word information included in both tweets and web pages. This is especially evident in the cases in which the post-based window was used. The best F-measure value was 34.7 obtained with WCC with a post-based window of size 2.

Next, Table 4 lists the results obtained from **PT**. **PT** almost totally outperformed **Basic** and also outperformed **WCC** when small windows were used. It exhibited an F-measure of 34.9 with a post-based window of size 1. This is the best performance of all experimental conditions.

5.4.3 Sensitivity to window size

We investigated the sensitivity of the proposed methods to the context window size. Figure 4 shows F-measure values obtained from the proposed methods with the word-based window. The horizontal axis indicates the size of the window and the vertical axis indicates F-measure. Each line corresponds to the result of each method. In the figure, **WCC+PT**

Table 3: Results of proposed method (**WCC**)

word-based window			
window size	Precision	Recall	F-measure
1	**100.0**	0.3	0.6
2	43.5	3.2	5.9
3	37.4	11.7	17.8
4	32.1	21.8	25.9
5	25.8	26.5	26.1
6	25.7	36.6	**30.2**
7	20.1	33.1	25.0
8	19.2	37.9	25.5
9	18.4	39.1	25.0
10	17.4	**39.7**	24.2
11	16.4	39.4	23.1
12	16.2	39.1	22.9
13	15.6	38.2	22.2
14	14.8	36.3	21.0
15	15.3	36.9	21.6
post-based window			
window size	Precision	Recall	F-measure
1	**33.1**	31.9	32.5
2	28.7	**43.8**	**34.7**

Table 4: Results of proposed method (**PT**)

word-based window			
window size	Precision	Recall	F-measure
1	**35.8**	13.6	19.7
2	31.7	14.5	19.9
3	33.2	20.5	25.3
4	29.0	24.9	26.8
5	25.0	30.0	27.3
6	26.2	39.7	**31.6**
7	21.3	39.1	27.6
8	19.0	38.8	25.5
9	18.8	40.4	25.7
10	18.4	**42.6**	25.7
11	17.4	42.3	24.7
12	16.3	39.7	23.1
13	15.9	38.5	22.5
14	15.2	37.2	21.6
15	15.5	36.9	21.8
post-based window			
window size	Precision	Recall	F-measure
1	**31.0**	**40.1**	**34.9**
2	20.2	38.8	26.6

indicates the results obtained from the method with both extension models.

One can see that all models exhibited the best performance when the window size = 6. This is intuitively supported since each tweet in the evaluation data was composed of an average of six words. One can see from Figure 5 that Precision and Recall were balanced when the window size was around 6. There seemed to be a semantic boundary seemly for NSD around 6.

It is less sensitive in the case of the **PT** extension model and the **WCC+PT** combination model. These models exhibited almost the same F-measure values. It would be reasonable and sufficient to use the **PT** extension model when it is difficult to crawl web pages.

6 Conclusion

We proposed an NSD method based on text segmentation and two extension models using web content and post times. Using the TextTiling algorithm, we achieved an F-measure of 34.9. The following issues will need to be addressed to refine our models.

- The proposed methods can provide a lightweight, approximate solution to NSD by using text segmentation. This means that it is only applicable to continuous conditions. Methods applicable to non-continuous conditions should be developed to improve performance.

- We only considered web pages referred from tweets as news sources in this paper. It would be valuable to enlarge the target of news sources to other media such as TV and radio.

References

Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. 2011. Semantic enrichment of twitter posts for user profile construction on the social web. In *Proceedings*

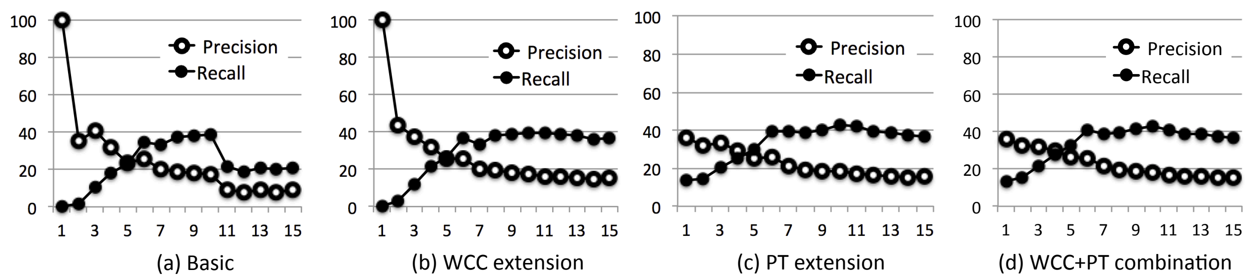

Figure 5: Precision and Recall values obtained from proposed methods

of the 8th extended semantic web conference on The semanic web, pages 375–389.

James Allen. 2002. *Topic detection and tracking: event-based information organization*. Kluwer Academic Publishers.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Education and Psychological Measurement*, 43(6):37–46.

Marti A. Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33–64.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the 13th annual ACM symposium on Theory of computing*, pages 604–613.

Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2006. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD workshop on Web mining and social network analysis*, pages 56–65.

Balachander Krishnamurthy, Phillipa Gill, and Martin Arlitt. 2008. A few chirps about twitter. In *Proceedings of the rirst workhop on Lnline social networks*, pages 19–24.

Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. 2013. Improving LDA topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 889–892.

Marco Pennacchiotti and Siva Gurumurthy. 2011. Investigating topic models for social media user recommendation. In *Proceedings of the 20th International World Wide Web Conference*, pages 101–102.

Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to twitte. In *Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 181–189.

Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *Proceedings of the 33rd European conference on Advances in information retrieval*, pages 338–349.