Category-Level Transfer Learning from Knowledge Base to Microblog Stream for Accurate Event Detection

Weijing Huang[1], Wei Chen[1] ⋆, Yazhou Wang[1], and Tengjiao Wang[1]

Key Laboratory of High Confidence Software Technologies (Ministry of Education), EECS, Peking University, Beijing, China
huangwaleking@gmail.com,{pekingchenwei,tjwang}@pku.edu.cn

Abstract. Many web applications need the accurate event detection technique on microblog stream. But the accuracy of existing methods are still challenged by microblog's short length and high noise. We develop a novel category-pivot transfer learning method TRANSDETECTOR to deal with the task. TRANSDETECTOR bases on two facts, that microblog is short but can be enriched by knowledge base semantically with transfer learning; and events can be detected more accurately on microblogs with richer semantics. The following contributions are made in TRANSDETECTOR. (1) We propose a structure-guided category-pivot topics extraction method, which exploits the knowledge base's hierarchical structure to extract categories' highly correlated topics. (2) We develop a probabilistic model CTrans-LDA for category-pivot transfer learning, which utilizes the word co-occurrences and transfers the knowledge base's category-pivot topics into microblogs. (3) Events are detected accurately on category-pivot word time series, due to richer semantics and less noise. (4) Experiment verifies the quality of category-pivot topics extracted from knowledge base, and the further study on the benchmark *Edinburgh twitter corpus* validates the effectiveness of our proposed transfer learning method for event detection. TRANSDETECTOR achieves high accuracy, promoting the precision by 9% without sacrificing the recall rate.

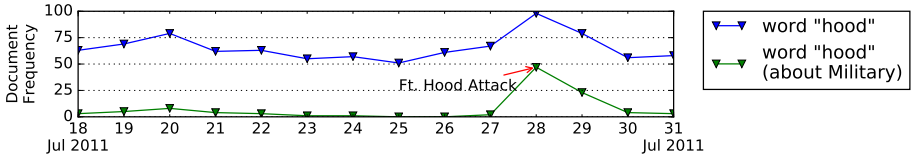
1 Introduction

Many web applications need the accurate event detection technique on microblog stream, such as public opinion analysis[1], public security[2], and disaster response[3], etc. Although event detection has been a research topic for a long while[4], event detection in microblog stream is still challenging[5]. According to [6], the characteristics of microblog, which is fast changing, high noise, and short length, raise the challenge.

Knowledge base can be a good supplementary for event detection on microblog stream. Different from the not-well-organized microblog stream, knowledge base (e.g. Wikipedia) is constructed elaborately and contains rich information. For example, the microblog message "Possible Ft. Hood Attack Thwarted (2011-07-28)" is short, but still comprehensible because the words "*Ft. Hood*" is included in the wiki page "*Ft. Hood*", and belongs to the category "*Military*". By reading these two wiki pages, the model easily understands the example tweet is talking about something related to *Military*. In

⋆ Corresponding author.

other words, knowledge base enriches the linkages between words, and provides more comprehensive context for microblogs. Since the transfer learning[7] aims at utilizing the extra information restored in the source dataset to benefit the target dataset, it provides a feasible way to enhance the event detection in microblog stream.



But it's non-trivial to conduct transfer learning from knowledge base to microblog stream directly. The existing RDF model[8] lacks an efficient quick mechanism to transfer the knowledge, since it's mainly designed for managing knowledge as tuples on graph. And the query on large graph is also very expensive[9], which is not suitable for the scene of quickly and acuaretely detecting events. What most meet the efficiency demand is Twevent[10], but it's limited to filtering out meaningless microblogs by looking up Wikipedia. Some event, which may contains the segments not restored in Wikipedia, may be dropped incorrectly.

In our paper, to balance the performance and the cost of transfer learning, new data structure *KB states* is proposed. It bases on the following facts. The knowledge base's three fold structure includes the taxonomy graph (*class* $\rightarrow$ *subclass*), the category-page bipartite graph (*class* $\rightarrow$ *instance*), and the page-content map (*instance* $\rightarrow$ *content*) relations in the knowledge base. In terms of concept level, the latter part is finer than the former. And the last page-content map goes into the detail at the word level. By considering these three parts together, we can extract a class or category's highly related words (evaluated by chi-square score in Section 3.1), and restore them in the newly proposed data structure *history states*. In transfer learning, *history states* play an important role at speeding up building the linkage between the word in microblog and the high level concept in knowledge base.

More than that, we propose a new quick and lightweight probabilistic model KB-Prior-LDA for transfer learning. After deriving KB-Prior-LDA (detailed in Section 3.2), it reveals that whether a word in microblog links to a high-level category in knowledge base depends on three factors, the word's score in the category's *KB state*, the linkages of microblogs' other words, the linkages of the word types in the same time window. All these statistics are easy to gain, which leads the model efficient. What KB-Prior-LDA learned is summarized as *recent states*, shown in Figure 1(a). Since the recent states have filtered out meaningless words, the detected events on recent states is much more accurate than other methods which don't do the transfer learning. KB-PRIOR-LDA, *history states*, and *recent states* forms up our proposed event detection method LTDETECTOR. Our experiment on the real dataset *Edinburgh twitter corpus (30 million tweets)* validates the effectiveness of our proposed method LTDETECTOR.

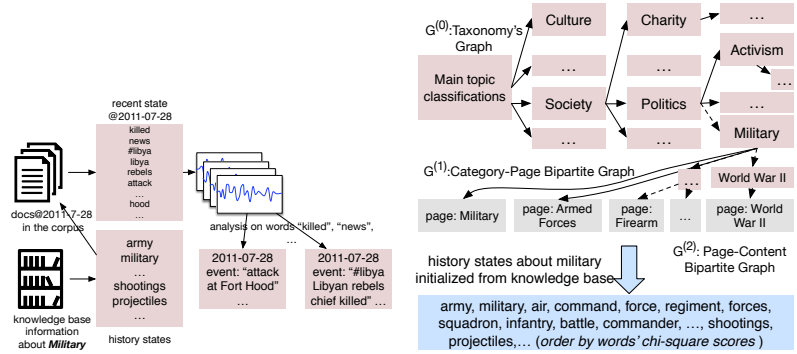


Fig. 1. (a)LTDETECTOR's process flow, taking *Military* related events in cyberspace as an example. LTDETECTOR initializes the normal states about military from knowledge base, learns the topical words further within the time windows on the target corpus, detects events.;(b)Illustration on how to initialize history states from Wikipedia, taking *Military* as an example.

2 Related Works

Event Detection. According to how much data are used, the approaches are divided into two groups: the detection *without extra information*, and the detection by *leveraging extra information*.

Most existing methods are implemented *without extra information*. They utilize the bursty pattern in the text stream to detect events, and are carried out by clustering articles[11,12,13], analyzing word frequencies[14,15], or finding bursty topics[16,17]. (1) By clustering articles, UMass[11], LSH[12], k-Term-FSD[13] and other similar methods model the occurred events as clusters of articles, and link the incoming article with an already existed event cluster or label it as the first story of a new event. The decision is based on whether the dissimilarity between the incoming article and existed event clusters is over the user-specified threshold. Taking LSH as an example, as [12] discussed, the threshold controls the result: the big one leads to very few and broad clusters, while the small one results in very small and specific clusters. Even after carefully set the appropriate threshold, the clusters generated by this kind method are still mixed of events and non-events due to large portion of non-event microblogs, which requires some post-processing methods as the supplement. (2) By analyzing word frequencies, TwitterMonitor[14] models the bursty words by queueing theory, and EDCoW[15] exploits wavelet transformation, which converts signal from time domain to time-scale domain, to detect the change point of word signal. However they treat the word as the most basic unit in analysis, without regarding how many concepts are represented by the word. (3) By finding bursty topics via topic modelling, TimeUserLDA[16], BurstyBTM[17] generates detected events. TimeUserLDA distinguishes user's long term interests and short term bursty events, and BurstyBTM utilizes the burstiness of biterms as prior knowledge. They both need to set the appropriate number of topics to run the topic modelling, and detect the "large" events that associate with many articles but ignore the "small" ones. To summarize, due to microblog's short

length and high noise, it's not easy to set the directly applicable parameter for existing methods to achieve high precision and high recall simultaneously.

The second group methods detect events by *leveraging extra information*. [18] incorporates user's and the followees' profiles in microblog platform as the extra information for modeling user's long term interests, and further detects the bursty events which draws the attention of users with different interests. However many event detection scenes only provide the microblog stream without user profile information and the following network. [19] compares two time series generated by event-related tweets and corresponding Wikipedia article's page views, and further filter out spurious events of microblogs. Twevent [10] divides the tweet into segments according to the Microsoft Web N-Gram service and Wikpedia, then detect the bursty segments and cluster these segments into candidate events for necessary post processings. But Twevent is still hampered by the performance of simply clustering the bursty segments shown by [17]. If not limited to microblog stream, many systems also use the extra information for improving event detection. [20] utilizes the concurrent wikipedia edit spikes for event detection. And [21] leverages the ACE corpus[22] and the structure of FrameNet[23] to improve the event detection within FrameNet, but has not shown how to apply the knowledge base for more general unstructured corpus. In a nutshell, existing works are different from ours, as we transfer the category information of knowledge base into microblog's words and get the finer processing objects in event detection, rather than treating the knowledge base as a lookup table or a comparison base.

Knowledge Base Recently, many general knowledge bases and customized knowledge bases are constructed and utilized for different text mining tasks. For example, Probase[24] constructs a large general probabilistic *IsA* taxomomy from webpages, and is used for semantic web search, and text classification[25]. Such kind efforts are also made by DBPedia[26], Yago[27], and Freebase[28]. These works manages knowledge as tuples on graph. However the query on large graph is very expensive[9], which is not very suitable for the scene of quickly and accurately detecting events.

The customized knowledge bases such as EVIN[29], Event Registry[30], and Storybase[31], are designed for managing events. These knowledge bases are mainly built on news articles. EVIN maps the existing event-related news articles into semantic classes. Event Registry collects the articles by the API of News Feed Service[32], then detects events, and provides the structural information of the events, such as the related Wikipedia article, timestamp, and location etc. Storybase introduces Wikipedia current events[1] as the resources for constructing event-and-storyline knowledge base on news articles, which are provided by GDELT project[33]. As the news articles may lag behind the microblogs when the emergency of events, it's not enough to detect events from microblog stream by directly applying these customized knowledge bases.

3 Proposed Method

Knowledge base often contains much information about what happened. But the existing RDF model[8] on knowledge base lacks an efficient mechanism to transfer the

[1] `https://en.wikipedia.org/wiki/Portal:Current_events`

information from knowledge base to text stream. With the help of knowledge base's structure (detailed in sec 3.1), we provide a lightweight model to transfer the information in knowledge base to text stream.

History state is the proposed data structure to restore the information about what happened. Initially, *history state* extracts category-related history data into a set of tuples, which weighs the importance of given words to the specific category, defined in Definition 1 formally. Taking the *military* category in Figure 1(a) as an example, the history state of *military* contain the words *army*, *military*, and *shootings* etc.

Recent state is the proposed data structure to maintain the information about what's happening. It extracts category-related words from incoming text stream, for each time window. Still taking the *military* category in Figure 1(a) as an example, based on extracted history states, our proposed probabilistic continuous learning model HS-PRIOR-LDA can recognize the words *libyan* and *rebel* in the time window 2011-07-28 in the text stream are related to the category *military*. Furthermore the time series analysis on the neighbouring recent states can recognize the set of events' candidate words, and finally detects the events in the text stream. In the example of Figure 1(a), the words *libyan* and *rebel* are related to the *military* event "#libya Libyan rebels chief is killed (2011-07-28)".

After processing each time window's data, the words in microblogs can be enriched semantically with knowledge base. In this way, the transfer learning model provides richer semantics and less noise for microblog stream, which promotes the accuracy of event detection significantly. The following definitions are the concepts used by the transfer learning model TRANSDETECTOR.

Definition 1 (History State) *The history state of the specific category is defined by a set of tuples, in which the first element is the word $w_i^{(c)}$ related to the category c, and the second element is the chi-square score $chi(c, w_i^{(c)})$ under the category c. And we denote the history states of category c as $\boldsymbol{h}_c = \{< w_i^{(c)}, chi(c, w_i^{(c)}) >\}_{i=1,\ldots,N_c}$.*

Definition 2 (Recent State) *The recent state at time t of the specific category c is defined by a set of tuples, and denoted as $\boldsymbol{r}_{c,t} = \{< w_i^{(c)}, n(c,t,w_i^{(c)}) >\}_{i=1,\ldots,N_c}$, in which word $w_i^{(c)}$ is related to the category c, and $n(c,t,w_i^{(c)})$ is its document frequency in the time window t.*

Definition 3 (The Set of Events' Candidate Words) *The set of events' candidate words $\mathcal{B}_{c,t}$ are defined by the bursty words in recent state $\boldsymbol{r}_{c,t}$.*

Definition 4 (Event Phrase) *Event phrase $\mathcal{C}_{c,t,i}$ is the i-th combination of words which occurred in the set of events' candidate words $\mathcal{B}_{c,t}$, and represents the i-th event happened in time t under the category c.*

Definition 5 (Event Related Microblogs) *Event related microblogs $\mathcal{D}_{c,t,i}$ are articles that relate to the i-th event in time t under the category c, and correspond to the event phase $\mathcal{C}_{c,t,i}$.*

Section 3.1 explains the details of how to initialize history states from knowledge base. Section 3.2 shows how to perform transfer learning for recent states. And section 3.3 interprets the high accuracy event detection based on the processed recent states.

3.1 History States' Extraction

In this part, we discuss in detail how to initialize the history states from the given knowledge base. The knowledge base such as Wikipedia has the structure of classes, subclasses, instances, and the edges between them. This structure usually can be represented as triples in RDF graph[8], which is adopted to build DBPedia[26] and YAGO[34] from Wikipedia. But the reasoning and maintenance of knowledge on the graph is usually expensive[35,36]. To make a trade-off between cost and performance, we use the lightweight data structure *history state* to represent the knowledge about what happened, which is extracted from the knowledge base. And the knowledge base's threefold structure $G^{(0)}$, $G^{(1)}$, $G^{(2)}$ benefits the extraction of history states.

Taxonomy Graph $G^{(0)}$. The directed edges in $G^{(0)}$ represent the *class→subclass* relations in the knowledge base. Taking Wikipedia for example (Figure 1(b)), the node *Main topic classifications*[2] has the subclass *Society*, further contains the subclass *Politics*, which is the ancestor of the subclass *Military*. As $G^{(0)}$ is not a Directed Acyclic Graph originally[37], we remove the cycles according to nodes' PageRank-HITS[38] score. Specifically, the edges *class→subclass* are preserved only when the node *class* has the higher PageRank-HITS score than the node *subclass*, which is shown in the line 2 of Algorithm 1. After removing cycles, the taxonomy structure on the knowledge base is better represented by the directed acyclic graph $G^{(0)'}$. As shown in line 3 of Algorithm 1, by visiting the category *Military* in the DAG $G^{(0)'}$, the breadth-first traverse can reach its successor sub-category nodes such as *Firearms*, *The World Wars*, and *World War II*, etc.

Category-Page Bipartite Graph $G^{(1)}$. The directed edges in $G^{(1)}$ represent the *class→instance* relations in the knowledge base. In Wikipedia, by considering $G^{(0)'}$ and $G^{(1)}$ together as shown in line 5 of Algorithm 1, we can get all the pages related to the given category.

Page-Content Map $G^{(2)}$. For a specific Wikipedia dumps version, the edges *page →content* in $G^{(2)}$ define a one-to-one mapping. There are a bulk of history information restored in the wiki text content. In order to extract the key words in wiki page's content, which distinguish it from other pages, we use the chi-square statistics[39][40] to measure the importance of each word for the specific page.

For a given category in $G^{(0)'}$, chi-square statistics also can evaluate the importance of each word appeared in text contents. For example, the chi-square statistic of the term *shooting* under the category *military* is 2888.7 under chi-square test with 1 degree of freedom. That means the term *shooting* is highly related to the category *military*. Finally, we can get the category's history state, which are composed by the category related words and their corresponding importance.

Considering the full Wikipedia's contents, history state evaluates the importance of word to the concerned category accurately. For example, in *Military*'s history state, the top words are *army*, *military*, *air*, *command*, *force*, and *regiment*, etc. The document which contains these words is related to the category *Military* with high probability. We further discuss how to apply history states to the text stream on the following subsection 3.2.

[2] `https://en.wikipedia.org/wiki/Category:Main_topic_classifications`

Algorithm 1: History State Initialization from Knowledge Base

Input: Taxonomy's Graph $G^{(0)}$, Category-Page Bipartite Graph $G^{(1)}$, Page-Content Bipartite Graph $G^{(2)}$, topic related category node c
Output: History state $\boldsymbol{h}_c$ on category c

$Pages(c) \leftarrow \varnothing, \boldsymbol{h}_c \leftarrow \varnothing$
DAG $G^{(0)'} \leftarrow$ Remove Cycles of $G^{(0)}$ by nodes' HITS-PageRank scores.
$SuccessorNodes(c) \leftarrow$ Breadth-first-traverse($G^{(0)'}$, c)
for $node \in SuccessorNodes(c)$ **do**
    $Pages(c) \leftarrow Pages(c) \cup G^{(1)}.neighbours(node)$
Word frequency table $n(c, .) \leftarrow$ do word count on the text contents of $Pages(c)$
Word frequency table $n(All, .) \leftarrow$ do word count on the text contents of all pages in $G^{(2)}$.
for *word w in WordFrequencyTable(All).keys()* **do**
    $chi(c, w) \leftarrow w$'s chi-square statistics on $WordFrequencyTable(c)$ and $WordFrequencyTable(All)$.
    $h_{c,w} \leftarrow chi(c, w)$
return $\boldsymbol{h}_c$

3.2 Recent States' Maintenance

In this subsection, we describe how the proposed probabilistic model HS-PRIOR-LDA utilizes the history states to learn the recent states from text stream.

There are two facts inspiring HS-PRIOR-LDA. (1) The topics in the document may contain history-state-like topics. As an example, the tweet "Libyan rebel chief gunned down in Benghazi (2011-07-28)" contains the topic that is similar to the *Military*'s history state. (2) The history-state-like topics can reuse the information stored in the history states. The word *libyan* in the aforementioned example tweet ranks much higher in the *Military* and the *Middle East* history states than the other categories' history states. After considering the relatedness between the remaining context and the history states, the learned topics of the example tweet include *Military* and *Middle East*.

The generative process of HS-PRIOR-LDA can be described as follows.

1. Draw corpus prior distribution $\boldsymbol{m} \sim Dir(\alpha \boldsymbol{u})$, where $\boldsymbol{u}$ is the uniform distribution.
2. For each topic $k \in \{1, \cdots, K\}$,
   (a) word distribution on the topic $\boldsymbol{\phi_k} \sim Dir(\boldsymbol{\beta} + \boldsymbol{\tau_k})$.
3. For each document index $d \in \{1, \cdots, D\}$,
   (a) topic distribution on the document $\theta_d \sim Dir(\boldsymbol{m})$,
   (b) for each word index $n \in \{1, \cdots, N_d\}$,
      i. word's topic assignment $z_{dn} \sim Multinomial(\theta_d)$,
      ii. word $w_{dn} \sim Multinomial(\phi_{z_{dn}})$.

In the above generative process, the line 2(a) is the key point to distinguish HS-PRIOR-LDA from LDA[41], where $\boldsymbol{\tau}_k$ is defined by Equation(1). K_{KB} is the number of pre-defined history-state-like topics, and S_k is the set of words appeared in the k-th topic's history state. As [42] mentioned that the asymmetric prior distribution can significantly improve the quality of topic modelling, $\boldsymbol{\phi_k} \sim Dir(\boldsymbol{\beta} + \boldsymbol{\tau_k})$ incorporates history state into the asymmetric prior of the word distribution on topic. The effect of $\boldsymbol{\tau_k}$ is obvious, e.g., $\tau_{Military,army}/\tau_{Military,basketball} = 203$ leads to that topic *Military* prefers to contain the word *army* other than *basketball*. The parameter λ controls how

much the learned topics are similar to the history state, which can be chosen by cross-validated grid-search.

$$\tau_{kv} = \begin{cases} \lambda \frac{h_{kv}}{\sum_{v \in S_k} h_{kv}}, v \in S_k \; and \; k \leq K_{\boldsymbol{KB}} \\ 0, v \notin S_k \; or \; k > K_{\boldsymbol{KB}} \end{cases} \tag{1}$$

To solve HS-PRIOR-LDA, the gibbs sampling is adopted to determine the hidden variable z_{dn} and the model parameter ϕ_k. In the initialization phase of Gibbs sampling for HS-PRIOR-LDA, the hidden variable z_{dn} is initialized to topic k with probability $\hat{q}_{k|v}$ as Equation (2). For the word v that belongs to any history state, $\hat{q}_{k|v}$ is proportional to its importance in the history state τ_{kv} as Eq2(a). For the new word v in text stream, $\hat{q}_{k|v}$ is set uniformly $1/(K - K_{\boldsymbol{KB}})$ on the other topics as Eq2(b,c). The initialization makes sure that the learned topics are aligned to the pre-defined history states.

$$\hat{q}_{k|v} = \begin{cases} \frac{\tau_{kv}}{\sum_{k=1}^{K} \tau_{kv}}, \sum_k \tau_{kv} > 0 & (a) \\ 0, \sum_k \tau_{kv} = 0 \; and \; k \leq K_{\boldsymbol{KB}} & (b) \\ 1/(K - K_{\boldsymbol{KB}}), \sum_k \tau_{kv} = 0 \; and \; k > K_{\boldsymbol{KB}} & (c) \end{cases} \tag{2}$$

The sampling process uses the conditional probability $p(z_{dn} = k|.) \propto (n_{dk}^{(d)} + \alpha m_k)(n_{kv}^{(w)} + \tau_{kv} + \beta)/(n_{k,\cdot}^{(w)} + \tau_{k,\cdot} + V\beta)$, where $n_{dk}^{(d)}$ is the number of words in document d assigned to topic k, and $n_{kv}^{(w)}$ is the times of word v assigned to topic k. We also optimize $\alpha \boldsymbol{m}$ for promoting HS-PRIOR-LDA's fitting to documents according to [42][3].

After Gibbs sampling on discrete time windows, HS-PRIOR-LDA learned all hidden topics of words in microblog stream. And for a specific word type w and its history-state-like topic assignment c in time window t, its document frequency is counted as $n(c, t, w^{(c)})$, which is the element of category c's t-th recent state. Event detection on recent states is discussed in the following subsection.

3.3 Detecting Events from Recent States

After transfer learning, events are detected accurately because the word time series is refined to finer grain as multiple corresponding (word, category) time series, and the bursty pattern is better exposed. Taking the word *hood* as an example, it belongs to the category *Military* when it appeared in *Ft Hood*, a US military establishment located in Texas. As shown in Figure 1, the bursty pattern of *hood* is clearly exposed after distinguishing its *Military* time series from the raw one.

Given (word, category) time series and microblogs in a time window, we take the detection in three sub-phases: (1) detecting events' candidate words; (2) generating event phrases; (3) retrieving event related microblogs. Note that these sub-phases are also available for the common text stream, but they performs better when words are enriched by category concepts of knowledge base.

[3] `https://github.com/mimno/Mallet/blob/master/src/cc/mallet/types/Dirichlet.java`

Detecting events' candidate words. For a word w and a category c, on its time series $\{n(c,t,w^{(c)})\}_{t=1}^{T}$, many bursty detection methods can be applied to check if the word w is bursty in category c by given time t. We assume the document frequency of $w^{(c)}$ follows a poisson distribution, which is also used in [16]. And the document frequency of nonbursting $w^{(c)}$ has the possison distribution with the parameter $\lambda = \mu_0$; while the bursting one with the parameter $4\mu_0$ means the document frequency $n(c,t,w^{(c)})$ is much higher when bursting. We empirically set μ_0 as the moving average $\mu_0^{(t)} = \frac{1}{T}\sum_{\tau=t-T}^{t-1} n(c,\tau,w^{(c)})$, and set T=10. Hence, at time t, bursty or not is determined by comparing the probabilities of above poisson distributions. After bursty detection on the time series, we add each time t's bursty word $w^{(c)}$ into the set of event's candidate words $\mathcal{B}_{c,t}$.

Generating event phrases. In the set of event's candidate words $\mathcal{B}_{c,t}$, some words appear together and should be grouped into event phrases. For example, there are six bursty words in $\mathcal{B}_{military,2011-07-28}$ belonging to two event phrases *"ft, hood, attack"* and *"libyan, rebel, gunned"* in the time window 2011-07-28 to represent the events happened to the US military establishment and the Libyan rebel respectively.

In order to group the bursty words together, we construct the directed weighted graph $\mathcal{G}_{c,t} = (\mathcal{B}_{c,t}, \mathcal{E}_{c,t}, \mathcal{W}_{c,t})$, where $\mathcal{E}_{c,t} = \{(a,b)|a \in \mathcal{B}_{c,t}, b \in \mathcal{B}_{c,t}, PMI(a,b) > 0\}$, and $\mathcal{W}_{c,t}$ gives the PMI scores on the edges. The graph $\mathcal{G}_{c,t}$ means the words in $\mathcal{B}_{c,t}$ are connected if and only if their PMI score in the given time window's microblogs is over 0.

Given the graph $\mathcal{G}_{c,t}$, spectral clustering[43] is utilized for exploring the best partition of words. To get the optimum cluster number, we use the graph density as the criteria. The graph density is the ratio of the number of edges to that of complete graph (the graph with all possible edges), and in practice all the sub-graphs in $\mathcal{G}_{c,t}$ should have the densities over an empirical threshold. And the empirical threshold is set to 0.6 in our setting. We run the spectral clustering with cluster number from 1 to $|\mathcal{B}_{c,t}|$, and stop when all the resulting sub-graphs satisfy the criteria that the density is over the given threshold. In this way, the generated event phrase combines the co-occurred bursty words and excludes the unrelated.

Retrieving event related microblogs. To better understand the event, we retrieve the event microblogs $\mathcal{D}_{c,t,i}$ by using the event phrase $\mathcal{C}_{c,t,i}$. Generally, according to the number of bursting words in the event phrase $\mathcal{C}_{c,t,i}$, there are two situations to be addressed. (1) $|\mathcal{C}_{c,t,i}| = 1$, we directly add the microblog in time window t, which contains the bursting category-word $w^{(c)}$, into the set $\mathcal{D}_{c,t,i}$. (2) $|\mathcal{C}_{c,t,i}| \geq 2$, it's not necessary that all the bursting words are included in the event related microblog. For example, the tweet "*Soldier wanted to attack Fort Hood troops*" contains the bursting words *attack* and *hood*, not all the event phrase "*ft, hood, attack*". To tackle this problem, we consider the microblog, that contains any pair of category-words in the event phrase, as the event related microblog.

Finally, TRANSDETECTOR gets the detected events $\{(\mathcal{C}_{c,t,i}, \mathcal{D}_{c,t,i})\}_{i=1}^{|\mathcal{C}_{c,t}|}$, containing the event phrase and the corresponding event related microblogs for the given category and the time window.

4 Experiments

4.1 Effects of History States and Recent States

In this subsection, we demonstrate the effectiveness of *history states* initialized from knowledge base and *recent states* learned by transfer learning.

Knowledge Base. We construct the taxonomy graph $G^{(0)}$, the category-page bipartite graph $G^{(1)}$ from the latest dump of category links[4] and the page-content map $G^{(2)}$ from Wikipedia pages[5]. We set $K_{KB} = 100$, which means 100 categories are selected manually to cover the topics of Wikipedia and the target corpus as widely as possible. There are two kinds of categories are considered. The mid-high categories in the taxonomy graph $G^{(0)'}$, which are representative, are likely to be selected, such as *Aviation*, *Military*, and *Middle East*, etc. And the mid categories, which reflect the main interests of the target corpus, are also taken into consideration, such as *American Football*, *Basketball*, and *Baseball*, etc.

Microblog Stream Dataset. We conduct the empirical analysis on a text stream benchmark *Edinburgh twitter corpus* which is constructed by [44] and widely used by previous event detection researches [45] [13]. Due to the developer policy of Twitter, [44] only redistributes tweets' IDs[6]. We collected the tweets' contents according to the IDs with the help of Twitter API. Though we cannot get the whole dataset due to the limit of Twitter API, after necessary pre-processing, our rebuilt dataset still contains 36,627,434 tweets, which spans identically from 2011/06/30 to 2011/09/15. More details of the original dataset are described in [46].

History States and Recent States. The history states are initialized on the predefined categories on the knowledge base as Algorithm 1. For HS-PRIOR-LDA, K is set to be 200, which means HS-PRIOR-LDA learned 100 history-state-like topics and 100 other topics. After cross-validated grid-search, λ is set to be 12.8. The other parameters are set as $\alpha = 0.1, \beta = 0.005$. We run HS-PRIOR-LDA window by window on text stream, and learn specific categories' recent states.

We compare the topic coherence of history states with the topics learned from Wikipedia by LDA in terms of NPMI[47]. Different from traditional experiments that only compute the topic coherence of top words, we want to check whether it can hold for more words. Due to the limit of NPMI computing module[7], which computes the coherence of up to 10 words each time, we compute NPMI on the combination of top five words with each next five words as Table 1. Observe that even for the combination of 96th to 100th words with the top five words in *Aviation*'s history state, NPMI=0.131 shows that the topic coherence still holds well without drifting. More generally, Figure 2 illustrates that the history states are much more stable than the topics learned by LDA. Taking the group 10 (the combination of 56th to 60th words with the top five words) as

[4] https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-categorylinks.sql.gz
[5] https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2
[6] http://demeter.inf.ed.ac.uk/cross/docs/fsd_corpus.tar.gz
[7] https://github.com/AKSW/Palmetto

an example, the median one of history states performs better than all those learned by LDA.

Since the history states extracted from knowledge base is stable and topic coherence,

Table 1. Top words' topic coherence of *Aviation*'s history state. * means the group also contains the five top words *aircraft*, *air*, *airport*, *flight*, and *airline*, but we don't show them in table to save space. NPMI is computed on ten words (a combination of words in each row with the five top words).

Category-Level Topics extracted from Wikipedia by TransDetector				Topics Learned from Wikipedia by LightLDA			
Group ID	#words*	words	NPMI	#group*	Group ID	words	NPMI
-1	1-5	aircraft air airport flight airline		-1	1-5	engine aircraft car air power	
0	6-10	airlines aviation flying pilot squadron	0.113	0	6-10	design flight model production speed	0.112
1	11-15	flights pilots raf airways fighter	0.155	1	11-15	system vehicle cars engines mm	0.062
2	16-20	boeing runway force crashed flew	0.092	2	16-20	fuel vehicles designed models type	0.072
3	21-25	airfield landing passengers plane aerial	0.179	3	21-25	version front produced rear electric	0.035
4	26-30	bomber radar wing bombers crash	0.137	4	26-30	space control motor standard development	0.085
5	31-35	airbus airports operations jet helicopter	0.189	5	31-35	film range light using available	-0.002
6	36-40	squadrons base flown havilland crew	0.088	6	36-40	wing powered wheel weight launch	0.087
7	41-45	combat luftwaffe aerodrome carrier fokker	0.159	7	41-45	developed low test ford cylinder	0.007
8	46-50	planes fly engine takeoff fleet	0.186	8	46-50	equipment side pilot hp aviation	0.091
9	51-55	fuselage helicopters aviator naval aero	0.157	9	51-55	systems us sold body drive	-0.051
10	56-60	glider command training balloon faa	0.166	10	56-60	gear introduced class safety seat	0.069
...	...	...	...	...	...	...	...
18	96-100	scheduled carriers military curtiss biplane	0.131	18	96-100	transmission special replaced limited different	0.059
19	101-105	accident engines iaf albatross rcaf	0.068	19	101-105	features machine nuclear even unit	0.011

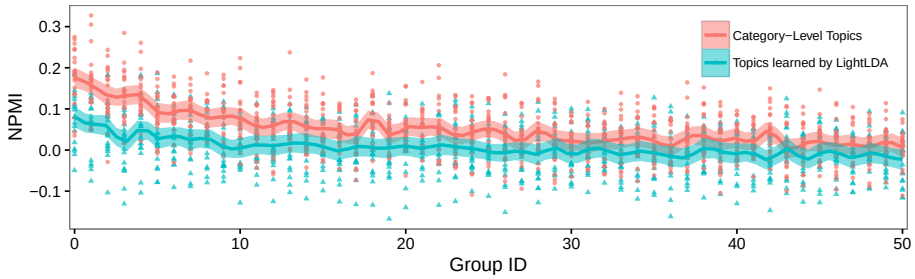


Fig. 2. Effectiveness of history state in terms of NPMI

TODO: add the top words of the category-level topic in each window of microblog stream AFTER Table 2.

TODO: add the effect of transfer learning by using text classification, compared with fastText and SVM.

4.2 Effects of Event Detection

Baselines Methods. We compare our proposed method against the following methods, Twevent[10], BurstyBTM[17], LSH[12], EDCoW[15], TimeUserLDA[16], and UMass System[11], which are mentioned in the Section 2. We implement these competing methods based on the open source community versions, e.g. EDCoW[8], or the authors' releases, e.g. BurstyBTM[9].

[8] https://github.com/Falitokiniaina/EDCoW
[9] https://github.com/xiaohuiyan/BurstyBTM

Table 2. Category-Level Topics extracted from knowledge base and the corresponding topics on microblog stream learned from CTrans-LDA, taking the categories *Aviation*, *Health*, *Middle East*, *Military*, and *Mobile Phones* as examples. The words in ***bold italic*** font are newly learned on the microblog stream by the transfer learning, which semantic meanings are verified consistent with the categories; while the words in normal font play the role as the bridge in transfer learning, and appear in the both category-level topics in two domains.

Aviation		*Health*		*Middle East*		*Military*		*Mobile Phones*	
Knowledge Base	Microblog Stream	Knowledge Base	Microblog Stream	Knowledge Base	Microblog Stream	Knowledge Base	Microblog Stream	Knowledge Base	Microblog Stream
aircraft	air	health	weight	al	***#syria***	army	killed	android	iphone
air	plane	patients	loss	israel	***#bahrain***	military	news	mobile	apple
airport	flight	medical	diet	iran	people	air	***#libya***	nokia	android
flight	time	disease	health	arab	israel	command	libya	ios	app
airline	airlines	treatment	cancer	israeli	police	force	rebels	phone	ipad
airlines	news	hospital	lose	egypt	***#libya***	regiment	people	samsung	samsung
aviation	boat	patient	fat	egyptian	#egypt	forces	police	game	mobile
flying	airport	clinical	tips	ibn	news	squadron	war	app	blackberry
pilot	force	symptoms	treatment	jerusalem	***#israel***	infantry	libyan	iphone	tablet
squadron	fly	cancer	body	syria	world	battle	attack	htc	apps

Evaluation Metrics. The first benchmark on *Edinburgh twitter corpus* contains 27 manually labeled events[45][10], which all exist in our rebuilt dataset on the *Edinburgh twitter*'s IDs. These labeled events focus on the events that are both mentioned in twitter and newswire,e.g. *"Oslo Attacks"* and *"US Increasing Debt Ceiling"*, but still miss many important events such as *"Hurricane Irene"*, *"Al-Qaida's No. 2 Leader Being Killed"*, and popular events such as *"Harry Potter and the Deathly Hallows (Part 2)"*. To enlarge the ground truth of realistic events, we manually evaluate the candidate events detected by LTDetector, Twevent, EDCoW, BurstyBTM, and TimeUserLDA, and use the labeled events as the second benchmark. We use precision and recall to evaluate each method on both benchmarks.

TODO: check the number of detected events.

Table 3. Overall Performance on Event Detection

Method	Recall@ Benchmark1	Precision@ Benchmark2	Recall@ Benchmark2	F@ Benchmark2	DERate (Duplicate Event Rate)
LSH	0.824	0.095	0.803	-	0.302
TimeUserLDA	0.353	0.536	0.071	-	0.054
Twevent	0.824	0.697	0.641	-	0.113
EDCoW	0.412	0.756	0.119	-	0.290
BurstyBTM	0.647	0.809	0.170	-	0.045
TRANSDETECTOR	1.000	0.894	0.950	-	0.042

Overall Performances. 中文范例！超级长的一句中文中文范例！超级长的一句中文中文范例！超级长的一句中文中文范例！超级长的一句中文

Discussions.

5 Conclusions & Future Work

Knowledge base is constructed elaborately and contains rich information, which can benefit the not-well-organized text stream. As a part of our future work, we will explore

[10] http://demeter.inf.ed.ac.uk/cross/docs/Newswire_Events.tar.gz

Table 4. Events about *military* detected by systems between 2011-07-22 and 2011-07-28

Date	Event key words	Representative event tweet	Number of event tweet	Methods[a]					
				L	TU	TW	E	B	TD
7/22/11	Norway, Oslo, attacks, bombing	Terror Attacks Devastate Norway: A bomb ripped through government offices in Oslo and a gunman... http://dlvr.it/cLbk8	557	✓					
7/23/11	Gunman, rink	Gunman Kills Self, 5 Others at Texas Roller Rink http://dlvr.it/cLcTH	43	✓					
7/26/11	Kandahar, mayor, suicide, attack	TELEGRAPH]: Kandahar mayor killed by Afghan suicide bomber: The mayor of Kandahar, the biggest city in south _	47	✓					
7/28/11	Ft., Hood, attack	Possible Ft. Hood Attack Thwarted http://t.co/BSJ33hk	52	✓					
7/28/11	Libyan, rebel, gunned	Libyan rebel chief gunned down in Benghazi http://sns.mx/prfvy1	44	✓					

[a] L=LSH[12], TU=TimeUserLDA[16], TW=Twevent[10], E=EDCoW[15], B=BurstyBTM[17], TD=TRANSDETECTOR.

the effects of transfer learning from knowledge base to text stream for more tasks, such as text classification and key words extraction, especially for short texts.

References

1. Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. Sentiment in twitter events. *Journal of the American Society for Information Science and Technology*, 62(2):406–418, 2011.
2. Rui Li, Kin Hou Lei, Ravi Khadiwala, and Kevin Chen-Chuan Chang. TEDAS: A Twitter-based Event Detection and Analysis System. *ICDE*, 2012.
3. Jie Yin, Sarvnaz Karimi, Bella Robinson, and Mark A Cameron. ESA: emergency situation awareness via microbloggers. *CIKM*, 2012.
4. James Allan, Jaime G Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic detection and tracking pilot study final report. 1998.
5. Farzindar Atefeh and Wael Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164, 2015.
6. Jiajia Huang, Min Peng, Hua Wang, Jinli Cao, Wang Gao, and Xiuzhen Zhang. A probabilistic method for emerging topic tracking in microblog stream. *World Wide Web*, 2016.
7. Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE*, 22(10), 2010.
8. Graham Klyne and Jeremy J Carroll. Resource description framework (rdf): Concepts and abstract syntax. 2006.
9. Jiewen Huang, Daniel J Abadi, and Kun Ren. Scalable sparql querying of large rdf graphs. *VLDB*, 2011.
10. Chenliang Li, Aixin Sun, and Anwitaman Datta. Twevent: segment-based event detection from tweets. *CIKM*, 2012.
11. James Allan, Victor Lavrenko, Daniella Malin, and Russell Swan. Detections, bounds, and timelines: Umass and tdt-3. In *Proceedings of topic detection and tracking workshop*, 2000.
12. Sasa Petrovic, Miles Osborne, and Victor Lavrenko. Streaming First Story Detection with application to Twitter. *HLT-NAACL*, 2010.
13. Dominik Wurzer, Victor Lavrenko, and Miles Osborne. Twitter-scale New Event Detection via K-term Hashing. In *EMNLP*, 2015.
14. Michael Mathioudakis and Nick Koudas. TwitterMonitor: trend detection over the twitter stream. *SIGMOD*, 2010.
15. Jianshu Weng, Yuxia Yao, Erwin Leonardi, and Francis Lee. Event Detection in Twitter. *ICWSM*, 2011.
16. Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim. Finding Bursty Topics from Microblogs. *ACL*, 2012.
17. Xiaohui Yan, Jiafeng Guo, Yanyan Lan, Jun Xu, and Xueqi Cheng. A Probabilistic Model for Bursty Topic Discovery in Microblogs. *AAAI*, 2015.
18. Weijing Huang, Wei Chen, Lamei Zhang, and Tengjiao Wang. An efficient online event detection method for microblogs via user modeling. In *Asia-Pacific Web Conference*, pages 329–341. Springer, 2016.
19. Miles Osborne, Saša Petrovic, Richard McCreadie, Craig Macdonald, and Iadh Ounis. Bieber no more: First story detection using twitter and wikipedia. In *SIGIR 2012 Workshop on Time-aware Information Access*, 2012.
20. Thomas Steiner, Seth Van Hooland, and Ed Summers. Mj no more: using concurrent wikipedia edit spikes with social network plausibility checks for breaking news detection. In *Proceedings of the 22nd International Conference on World Wide Web*, 2013.
21. Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. Leveraging framenet to improve automatic event detection. In *ACL*, 2016.
22. George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. The automatic content extraction (ace) program-tasks, data, and evaluation. In *LREC*, volume 2, page 1, 2004.

23. Collin F Baker, Charles J Fillmore, and John B Lowe. The berkeley framenet project. In *ACL*, 1998.
24. Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: a probabilistic taxonomy for text understanding. In *SIGMOD*, 2012.
25. Fang Wang, Zhongyuan Wang, Zhoujun Li, and Ji-Rong Wen. Concept-based short text classification and ranking. In *CIKM*, 2014.
26. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.
27. MS Fabian, K Gjergji, and W Gerhard. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *WWW*, 2007.
28. Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.
29. Erdal Kuzey and Gerhard Weikum. Evin: building a knowledge base of events. In *WWW*, 2014.
30. Gregor Leban, Blaz Fortuna, Janez Brank, and Marko Grobelnik. Event registry: learning about world events from news. In *WWW*, 2014.
31. Zhaohui Wu, Chen Liang, and C Lee Giles. Storybase: Towards building a knowledge base for news events. *ACL-IJCNLP*, 2015.
32. Mitja Trampuš and Blaž Novak. Internals of an aggregated web news feed. In *Proceedings of 15th Multiconference on Information Society*, pages 431–434, 2012.
33. Kalev Leetaru and Philip A Schrodt. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA Annual Convention*, volume 2. Citeseer, 2013.
34. Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *WWW*, 2007.
35. Jeen Broekstra and Arjohn Kampman. Inferencing and truth maintenance in rdf schema. *PSSS*, 89, 2003.
36. Damian Bursztyn, François Goasdoué, Ioana Manolescu, and Alexandra Roatiş. Reasoning on web data: Algorithms and performance. In *ICDE*, 2015.
37. Stefano Faralli, Giovanni Stilo, and Paola Velardi. Large scale homophily analysis in twitter using a twixonomy. In *IJCAI*, 2015.
38. Rui Yan, Yiping Song, Cheng-Te Li, Ming Zhang, and Xiaohua Hu. Opportunities or Risks to Reduce Labor in Crowdsourcing Translation? Characterizing Cost versus Quality via a PageRank-HITS Hybrid Model. *IJCAI*, 2015.
39. Yiming Yang and Jan O Pedersen. A comparative study on feature selection in text categorization. In *ICML*, 1997.
40. Ying Liu, Han Tong Loh, and Aixin Sun. Imbalanced text classification: A term weighting approach. *Expert systems with Applications*, 36(1):690–701, 2009.
41. David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *JMLR*, 2003.
42. Hanna Megan Wallach. *Structured topic models for language*. PhD thesis, University of Cambridge, 2008.
43. Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.
44. Saša Petrović, Miles Osborne, and Victor Lavrenko. Using paraphrases for improving first story detection in news and twitter. In *NAACL-HLT*, 2012.
45. Saša Petrović, Miles Osborne, Richard McCreadie, Craig Macdonald, and Iadh Ounis. Can twitter replace newswire for breaking news? In *ICWSM*, 2013.
46. Saša Petrović, Miles Osborne, and Victor Lavrenko. The edinburgh twitter corpus. In *NAACL-HLT*, 2010.
47. Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *WSDM*, 2015.