Using Content-level Structures for Summarizing Microblog Repost Trees [*]

Jing Li[1,2], **Wei Gao**[3], **Zhongyu Wei**[4], **Baolin Peng**[1,2] and **Kam-Fai Wong**[1,2]
[1]The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[2]MoE Key Laboratory of High Confidence Software Technologies, China
[3]Qatar Computing Research Institute, Hamad Bin Khalifa University, Doha, Qatar
[4]The University of Texas at Dallas, Richardson, Texas, USA
{lijing,blpeng,kfwong}@se.cuhk.edu.hk[1,2]
wgao@qf.org.qa[3], zywei@@hlt.utdallas.edu[4]

Abstract

A microblog repost tree provides strong clues on how an event described therein develops. To help social media users capture the main clues of events on microblogging sites, we propose a novel repost tree summarization framework by effectively differentiating two kinds of messages on repost trees called leaders and followers, which are derived from content-level structure information, i.e., contents of messages and the reposting relations. To this end, Conditional Random Fields (CRF) model is used to detect leaders across repost tree paths. We then present a variant of random-walk-based summarization model to rank and select salient messages based on the result of leader detection. To reduce the error propagation cascaded from leader detection, we improve the framework by enhancing the random walk with adjustment steps for sampling from leader probabilities given all the reposting messages. For evaluation, we construct two annotated corpora, one for leader detection, and the other for repost tree summarization. Experimental results confirm the effectiveness of our method.

1 Introduction

Microblogging platforms have become the center for reporting, discussing, and disseminating real-life issues, on which users usually *repost* to share microblog messages with their following users. Also, users can *repost with commentary* for not only further broadcasting but also extending the original microblog post content. Because an individual post is generally too short to cover the main clues of an event, microblogging users cannot easily capture the key information from received posts due to the lack of context. And reposting messages, namely reposts, can provide valuable context information to the previous posts including their background, development, public opinions and so on. However, a popular post usually attracts a large number of reposts. It is impractical for users to read them all and fully understand their contents.

The task of microblog context summarization aims to produce succinct summaries to help users better understand the main clues by extracting salient information among massive reposts of the original posts. An intuitive approach is to directly apply existing extractive summarizers based on the unstructured, plain microblog contents. But such short and informal reposts render the lack of structures in each individual message, and it is difficult for conventional extractive summarizers to identify salient messages. Chang et al. (2013) proposed to summarize Twitter context trees by focusing on modeling user influence. However, the reposts of influential users might not be salient summary candidates necessarily. For instance, celebrities might simply repost with nothing important. Also, modeling user influence accurately needs tremendous historical user interaction data external to the tree being summarized while such kind of information cannot be utilized directly for summarizing the messages on the tree.

In this paper, we propose a novel microblog context summarization framework based on content-level structures, i.e., message contents and reposting relations, rather than user-level influence signals. The reposting relations connect the reposting messages and form a cohesive body as a tree structure named *repost tree*. The root represents the original post and the edges denote re-

[*] This work is partially supported by General Research Fund of Hong Kong (417112), RGC Direct Grant (417613), and Huawei Noah's Ark Lab, Hong Kong. We would like to thank anonymous reviewers for the useful comments.

posting relations. Our idea is to exploit the structure of repost tree together with content of messages to help distinguish two different *messages* on repost tree, i.e., leaders and followers. Specifically, **leader** is referred to as a *message* on repost tree covering salient new information, which can lead further comments or discussions in its descendant reposts; **follower** is referred to as a *message* that contains no comment, simply repeats or naively responds to its ancestor leader message, thus providing no important information. The example below illustrates a repost tree path, where we use $[O]$ and $[R_i]$ to indicate the original post and the i-th repost, respectively:

$[O]$ @MAS: Malaysia Airlines has lost contact of MH17 from Amsterdam. The last known position was over Ukrainian airspace.

$[R_1]$ @Hanna: OMG... Poor on MH17... Preying...

$[R_2]$ @Victoria: OMG that's horrible!!! I'm sorry to hear that. God will bless u poor guys. Wish world can be peaceful. And no one will get hurt.

$[R_3]$ @Dr.Dr: Six top HIV scientists are on MH17. They go for AIDS and would NEVER come back!!!

$[R_4]$ @TomyBlack: 6 experts died?! Terrible loss to HIV research :(

$[O]$ reports the news about MH17 missing, which brings about further comments in $[R_1]$ and $[R_2]$. $[R_3]$ does not continue commenting on that but offers some new information and triggers shocking reaction in $[R_4]$. So $[O]$ and $[R_3]$ act as leaders; $[R_1]$, $[R_2]$ and $[R_4]$ are followers.

Intuitively, leaders would be more important than followers from the summarization's perspective since leaders are supposed to capture the main clues or aspects of event evolvement. The first step of our summarization system is to distinguish leaders and followers effectively. Leaders are detected across repost tree paths which provide rich context information owing to the tree structure. We utilize sequence tagging model Conditional Random Fields (CRF) to infer how likely it is each repost being a leader or follower. Then we incorporate leader detection result into an unsupervised summarization model based on random walk. Our model uses content similarities between messages and consider their possibilities of being leaders to rank and select salient reposting messages that form summaries. Furthermore, we improve the framework by enhancing the random walk to reduce the impact of errors cascaded from the leader detection module. Compared to the state-of-the-art baselines, the experimental results confirm the effectiveness of our proposed framework.

Our contributions are given as follows:

- We propose a novel microblog context summarization framework, in which given reposting messages organized as a repost tree (obtaining repost tree is trivial using public microblogging toolkit (Ren et al., 2014)), we summarize the repost trees based on content information and reposting relations of messages.
- We identify a novel problem of leader detection for summarization, which aims to reduce noise on repost trees, and present a CRF-based method for effectively detecting leaders by utilizing the tree structure and message contents.
- We incorporate the leader detection result into an unsupervised summarization model based on random walk and substantially enhance the model to reduce the impact of leader detection errors on summarization.

2 Related Work

The goal of text summarization is to automatically produce a succinct summary for one or more documents that preserves important information (Radev et al., 2002). Generally, text summarization techniques can be categorized into extractive and abstractive methods (Das and Martins, 2007). Extractive approaches focus on how to identify and distill salient contents from original texts whereas abstractive approaches aim at producing grammatical summaries by text generation.

Recently, the development of social media has made microblog summarization a hot topic. Most prior works are on event-level or topic-level summarization. Typically, the first step is to cluster posts into sub-events (Chakrabarti and Punera, 2011; Duan et al., 2012; Shen et al., 2013) or subtopics (Long et al., 2011; Rosa et al., 2011; Meng et al., 2012), and then the second step generates summary for each cluster.

Some works tried to apply conventional extractive summarization models directly, e.g., LexRank (Erkan and Radev, 2004), MEAD (Radev et al., 2004), TF-IDF (Inouye and Kalita, 2011), Integer Linear Programming (Liu et al., 2011; Takamura et al., 2011), etc. Sharif et al. (2010) modeled the problem as optimal path finding on a phrase reinforcement graph. However, these general summarizers were found not suitable for microblog posts, which are informal and noisy (Chang et al., 2013). Researchers then considered social signals like user following relations and retweet count (Duan et al., 2012; Liu et al., 2012), and reported such features useful to help summarize microblog posts. Our work studies repost tree summarization by leveraging content-level structure to enrich context of messages, which is a different kind of signal.

Chang et al. (2013) proposed a task to summarize Twitter context trees consisting of an original tweet and all its reposts (i.e., replies and retweets). They combined user influence signals into a supervised summarization framework. Our work is different from theirs: 1) They simply treat a context tree as a tweets stream while we consider repost tree structures in summarization; 2) They rely on user interactions to calculate *user influence* for extracting salient messages while we focus on how to utilize contents and repost tree structures to differentiate *leader and follower messages* for summarization; 3) Our summarization module is unsupervised, thus no need of ground-truth summaries.

3 Leader Detection Model

This section deals with how to differentiate leader and follower messages on a repost tree. Intuitively, identifying leaders effectively makes one step closer to obtaining a good summary.

Figure 1 illustrates an example of a repost tree[1]. As shown in the figure, a leader message contains contents that brings essential information increase, such as a new clue about MH17 reported in $[R_6]$, and potentially triggers a new round of information propagation by attracting follower messages to focus on the raised clue, like $[R_7]$, $[R_8]$ and $[R_9]$. As the repost tree grows, it also happens that some new reposts join in, following the clue raised by one of their ancestors, but further extend it by mentioning something new, thus some of these messages may evolve into new leaders, such as $[R_{10}]$.

A simple way to detect leaders on repost tree is to directly apply a binary classifier like SVM on each individual message. However, these models assume reposts are independent without effectively leveraging abundant context along the repost tree paths, such as the reposting relations among different reposts on a path. For instance, $[R_2]$ covering rich content may be misclassified as a leader if not leveraging context information. But if we look into its context, we can find that $[R_2]$ talks about similar things as $[R_1]$, then $[R_1]$ classified as a follower indicates the higher chance of $[R_2]$ being a follower rather than a leader. Therefore, context information is important for indicating the messages being leaders or followers.

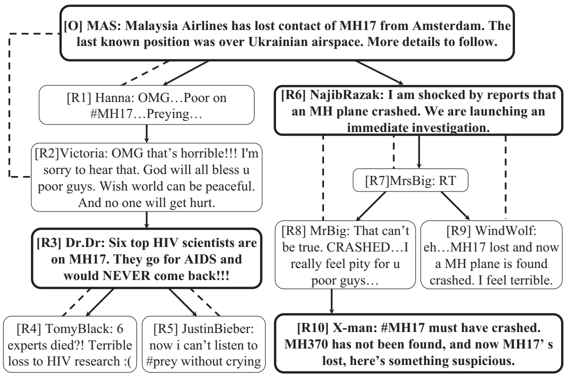
[O] MAS: Malaysia Airlines has lost contact of MH17 from Amsterdam. The last known position was over Ukrainian airspace. More details to follow.
[R1] Hanna: OMG…Poor on #MH17…Preying…
[R2]Victoria: OMG that's horrible!!! I'm sorry to hear that. God will all bless u poor guys. Wish world can be peaceful. And no one will get hurt.
[R3] Dr.Dr: Six top HIV scientists are on MH17. They go for AIDS and would NEVER come back!!!
[R4] TomyBlack: 6 experts died?! Terrible loss to HIV research :(
[R5] JustinBieber: now i can't listen to #prey without crying
[R6] NajibRazak: I am shocked by reports that an MH plane crashed. We are launching an immediate investigation.
[R7]MrsBig: RT
[R8] MrBig: That can't be true. CRASHED…I really feel pity for u poor guys…
[R9] WindWolf: eh…MH17 lost and now a MH plane is found crashed. I feel terrible.
[R10] X-man: #MH17 must have crashed. MH370 has not been found, and now MH17' s lost, here's something suspicious.

Figure 1: An example of repost tree. $[O]$: the original post; $[R_i]$: the i-th repost; Solid arrow lines: reposting relationship; Dotted lines: hidden leader-follower relationship; Dark boxes: leaders to be detected.

We extract all root-to-leaf paths within a repost tree structure and detect leaders across each path. We formulate leader detection on repost tree paths as a sequence tagging problem by utilizing a state-of-the-art sequence learning model CRF (Lafferty et al., 2001), and taking advantage of its power in maximizing the likelihood of global label sequences. We adopt CRF rather than other competitive context sensitive model like SVM^{hmm} (Altun et al., 2003) due to its probabilistic nature. The probability of prediction by CRF can provide critical chances for the following summarization procedure to reduce the impact of errors made by leader detection model on summarization (see Section 4.2).

We map a repost tree path with n microblogs $(m_1, m_2, \cdots, m_n)$ to a training instance (X, Y). Let $X = (x_1, x_2, \cdots, x_n)$ represents observed sequence, where x_i denotes the observed feature vector extracted from the i-th microblog m_i, and $Y = (y_1, y_2, \cdots, y_n)$ where y_i is the label indicating whether m_i is a leader or not. CRF defines the discriminative function as a joint distribution over Y given X as follows:

$$P(Y|X;\theta) \propto \exp\left(\sum_{i,j} \lambda_j f_j(y_i, y_{i-1}, X) + \sum_{i,k} \mu_k g_k(y_i, X)\right)$$

where f_j and g_k are the fixed feature functions,

[1] The example in Section 1 actually denotes the left-most path extracted from this tree

Table 1: Features used for leader detection

Feature category	Feature name	Feature description
Lexical	# of terms	The number of terms in m_i
	POS	The part-of-speech of each term in m_i
	Type of sentence	Whether m_i contains a question mark or an exclamation
Microblog-specific	# of emoticons	The number of emoticons in m_i
	# of hashtags	The number of hashtags in m_i
	# of urls	The number of URLs in m_i
	# of mentions	The number of mentions, or @UserName, in m_i
Path-specific	Similarity to neighbors	Cosine similarity between m_i and m_{i+d} where $d \in \{\pm 1, \pm 2, \pm 3\}$
	Similarity to root	Cosine similarity to the root microblog in repost tree path

$\theta = (\lambda_1, \lambda_2, ...; \mu_1, \mu_2, ...)$ are the parameters indicating the weights of features that can be estimated by maximum likelihood procedure in training process. The prediction is done based on dynamic programming. More details can be found in (Lafferty et al., 2001). Table 1 lists the features we use for leader detection.

CRF can utilize both historical and future information for prediction so as to maximize the likelihood of the global label sequences. But we would encounter the problem of label conflict, i.e., the predictions for the same repost in context of different paths might be different. For this reason, we determine a repost as a leader if its average marginal probabilities being a leader in context of different paths exceeds 50%.

4 LeadSum Summarization Model

Let $T = (V, E)$ represent a repost tree to be summarized, where V is a set of nodes corresponding to microblog messages, and $E = \{(u, v) | v \text{ is the repost of } u\}$ is the edge set denoting reposting relations. This section describes how to rank nodes in V to produce repost tree summaries. Enlightened by the general random-walk-based ranking algorithm DivRank (Mei et al., 2010), we propose an unsupervised summarization model called LeadSum that aims to select true and salient leaders into summaries utilizing a variant of random walk based on content similarities and reposting relations of messages. We first present a basic LeadSum model, which assumes leader detection is perfect. Then, we enhance it to become a soft LeadSum model that reduces the impact of leader detection errors on the summarization.

4.1 Basic-LeadSum Model

Due to the nature of leaders, they generally cover more important contents than follows do. Thus our first summarizer selects contents only from detected leaders. For the leaders detected in a repost tree T, we build a similarity graph among leaders denoted as $G_L = (V_L, E_L)$, where $V_L = \{v \in V | v \text{ is a detected leader}\}$ is the vertex set and $E_L = \{(u, v) | u \in V_L, v \in V_L, \text{ and } u \neq v\}$ is the edge set. The weight for any edge (u, v) represents the content similarity between u and v, for which we use cosine similarity.

DivRank (Mei et al., 2010) is a generic graph ranking model that aims to balance high information coverage and low redundancy in top ranking vertices, which are also two key requirements for choosing salient summarization sentences (Li et al., 2009; Liu et al., 2015). Based on that, we present a model to rank and select salient messages from leader set V_L to form a summary. Since this model simply assumes perfect leader detection, it is therefore named Basic-LeadSum.

Similar as DivRank (Mei et al., 2010), the transition probability at the t-th iteration of random walk is given as follows:

$$p_t(u \to v) = (1 - \mu) \cdot p_0(v) + \mu \cdot \frac{p_0(u \to v) N_{t-1}(v)}{Z(u)} \quad (1)$$

and $Z(u)$ is the normalizing factor:

$$Z(u) = \sum_{w \in V_L} p_0(u \to w) N_{t-1}(w) \quad (2)$$

where $p_0(u \to v)$ is the organic transition probability which represents the content similarity between u and v; $N_{t-1}(v)$ denotes the times vertex v is visited up to the $(t-1)$-th iteration; $p_0(v) = \frac{1}{|V_L|}$ denotes random jumping probability similar to that in PageRank; and μ is the damping weight set as 0.85 following most PageRank-based models. The probability of traveling to leader v can accumulate as its weight increases during random walk, and leaders already having high weight can "absorb" weights from other leaders with high similarity to it, thus avoids redundancy.

For any $v \in V_L$, the update function for its ranking score at the t-th iteration $R_t(v)$ is formulated as:

$$R_t(v) = \sum_{u \in V_L} p_t(u \to v) R_{t-1}(u) \tag{3}$$

It has been proved that the Markov chain is ergodic, thus can converge to a stationary distribution (Mei et al., 2010), which determines the final rankings for leaders.

4.2 Soft-LeadSum Model

As a two-step summarization system, the performance of LeadSum relies on the leader detection, which might be error-prone. Followers misidentified as leaders participating in leader ranking brings risks to extract real followers into summary. Also, leaders misclassified as followers may leave out strong summary candidates. To reduce such error propagation, we enhance Basic-LeadSum by using an even-length random walk with adjustment steps that sample from leader probabilities given all the reposting messages, which is referred to as Soft-LeadSum.

Different from Basic-LeadSum, every message on repost tree T, no matter detected as a leader or a follower, participates in ranking process of Soft-LeadSum. In other words, in the random walk, visitor wanders on a complete graph $G = (V, E')$ whose vertex set V is identical to repost tree T, and $E' = \{(u, v) | u \in V, v \in V, \text{ and } u \neq v\}$ represents the edge set. Therefore, this makes it possible to include true leaders misclassified as followers by leader detection module into summary.

However, allowing all messages to participate in ranking also increases the risk of selecting real followers. To avoid this problem, Soft-LeadSum is composed of two types of walks on G, namely WALK-1 and WALK-2. In WALK-1, visitor moves based on content similarities between messages, which follows transition probabilities similar to equation (1), but is specifically given as:

$$p_t(u \to v) = (1-\mu) \cdot \frac{1}{|V|} + \mu \cdot \frac{p_0(u \to v) N_{t-1}(v)}{Z(u)} \tag{4}$$

where $u, v \in V$, $p_0(u \to v)$ is proportional to content similarity between u and v similar to Basic-LeadSum, and $Z(u)$ is the normalizing factor.

WALK-2 attempts to avoid selecting true followers by adopting a sampling process, whose result determines the next vertex on G to be visited. Suppose the current vertex being visited is u, then we sample from $p_L(u)$, i.e., the probability of u being a leader. Practically, $p_L(u)$ can be estimated with the average of u's marginal probabilities as a leader over all root-to-leaf paths passing u on T output by the leader detection module. If u is sampled to be a leader, we claim that leader detection is correct and the visitor stays; otherwise, u is sampled as a follower, indicating that leader detection module misclassifies u, so the visitor should go to u's leader. Here we assume that a follower u's leader is its nearest ancestor leader on T as shown by the dotted lines in Figure 1. Based on such simplification, we let the visitor trace back one by one along the path on T from u to root and sample from their leader probabilities until a node v is sampled as a leader and then we determine v as u's leader.

So for any u's ancestor v, the probability of v being u's leader is:

$$\begin{aligned} &Pr\{v \text{ is } u\text{'s leader}\} \\ &= p_L(v)(1 - p_L(u) - \sum_{w \in \mathcal{P}(v,u)} Pr\{w \text{ is } u\text{'s leader}\}) \\ &= p_L(v) \prod_{w \in \mathcal{P}(v,u) \bigcup \{u\}} (1 - p_L(w)) \end{aligned} \tag{5}$$

where $\mathcal{P}(v, u)$ is the set of nodes between v and u on v-to-u path of repost tree, i.e., $\mathcal{P}(v, u) = \{w \in V | w \text{ is } v\text{'s descendant and } u\text{'s ancestor on } T\}$. In particular, we assume that $p_L(r) = 1$ so as to stop the sampling process when the visitor arrives at root r.

Therefore, WALK-2's transition probabilities can be calculated as follows:

$$q(u \to v) = \begin{cases} p_L(v) & \text{if } v = u; \\ Pr\{v \text{ is } u\text{'s leader}\} & \text{if } v \text{ is } u\text{'s ancestor;} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Algorithm 1 shows the ranking process of Soft-LeadSum, during which the visitor walks on G following WALK-1 and WALK-2 alternately. The fact that WALK-1 is ergodic ensures the ergodicity and convergency of the algorithm. In implementation, we set max iteration N=1000 empirically which is large enough to ensure convergence, or stop random walk process in advance when the condition of convergence is met, i.e., the change of Euclidean difference of ranking scores for three consecutive iterations are all less than 1e-6.

Soft-LeadSum can reduce the impact of errors made by leader detection on summarization due to the following two reasons: 1) It allows all messages to participate in ranking process, thus permits those leaders leaving out by leader detection module to be selected into summary; 2) With

Algorithm 1 Algorithm of Soft-LeadSum

Input: T, G, μ=0.85, max iteration N, length cut-off n
Output: Summary with n microblog messages

```
For all $v \in V$, initialize $R_0(v) = p_0(v) = \frac{1}{|V|}$
Initialize WALK-1's transition probabilities $p_0(u \to v)$
with normalized cosine similarity between $u$ and $v$.
Calculate WALK-2's transition probabilities $q(u \to v)$
by equation (5) and (6).
Initialize current_walk="WALK-1"
for $t = 1$ to $N$ and not converged do
    for all $v \in V$ do
        if current_walk=="WALK-1" then
            Update $p_t(u \to v)$ by equation (4)
            Update $R_t(v)$ as follows:
                $R_t(v) = \sum_{u \in V} R_{t-1}(u) \cdot p_t(u \to v)$
            Set current_walk="WALK-2"
        end if
        if current_walk=="WALK-2" then
            Update $R_v(v)$ as follows:
                $R_t(v) = \sum_{u \in V} R_{t-1}(u) \cdot q(u \to v)$
            Set current_walk="WALK-1"
        end if
    end for
end for
Sort all $v \in V$ by $R_N(v)$ in descending order
Pick the top-$n$ messages as summary
```

WALK-2 sampling from leader probabilities, it also reduces the risk of including real followers into summary.

5 Experiments and Results

To evaluate the two modules in our repost tree summarization system, i.e., CRF-based model for leader detection and LeadSum model for summarization, we conducted two sets of experiments based on microblog posts data collected from Sina Weibo, which has a similar market penetration as Twitter (Rapoza, 2011)[2]. Microblog messages on Sina Weibo are in Chinese and we use FudanNLP (Qiu et al., 2013) for text preprocessing including word segmentation and POS tagging.

5.1 Experiment for Leader Detection

In this experiment, we evaluated the performance of CRF model for leader detection task.

5.1.1 Data Collection and Setup

We first crawled 1,300 different repost trees using the public PKUVIS toolkit (Ren et al., 2014). Given an original microblog post, the toolkit can automatically crawl its complete repost tree. For each tree, we randomly selected one path and further formed a set with 1,300 repost tree paths, which ensures that paths have different roots and the dataset can cover a wide variety of context information.

Table 2: The performance of leader detection (%)

	Cross-validation			Held-out		
	Prec	Rec	F1	Prec	Rec	F1
Random	29.8	49.5	37.3	31.6	49.6	38.6
LR	70.5	66.3	68.4	70.4	66.2	68.2
SVM	70.9	66.9	68.8	68.9	66.2	67.5
SVM^{hmm}	74.8	65.5	69.8	69.3	70.1	69.7
CRF	**75.5**	**72.0**	**73.7**	**71.1**	**70.7**	**70.9**

Then three annotators were invited to label each repost as a leader or a follower in the context of its repost tree path independently. The average Cohen's Kappa of each two of the three annotators was 0.52, which is considered good agreement (Fleiss et al., 2013). Then, we used the labels agreed by at least two annotators as the ground truth. The training and test of the leader detection models were conducted on this corpus.

We compared the performance of CRF-based leader detection model with four baselines: Random Classifier (RC) as a weak baseline; two state-of-the-art point-wise supervised models Logistic Regression (LR) and Support Vector Machine (SVM); and an effective context sensitive model SVM^{hmm}. We applied LibLinear toolkit (Fan et al., 2008) to implement LR and SVM with linear kernel. SVM^{hmm} was implemented by SVM^{struct} toolkit (Joachims et al., 2009). And CRF's implementation was based on CRF++[3]. For all the baselines, we used features listed in Table 1. The hyper-parameters of all leader detection models were tuned to the same extent based on 5-fold cross validation (with 1 fold as development set). The evaluation metrics were precision, recall and F1 score for the detected leaders.

5.1.2 Results

Table 2 shows the comparison result of 5-fold cross validation on 1,000 repost tree paths and held-out experiment on 300 complete fresh paths.

Among all baselines, SVM^{hmm} performed the best, which indicates the effectiveness of incorporating structure information for leader detection. And among context-sensitive models, both SVM^{hmm} and CRF were competitive. CRF outperformed SVM^{hmm} slightly with 5.6% and 1.7% improved F1 score in cross validation and held-out

[2] The datasets are available at `http://www1.se.cuhk.edu.hk/~lijing/data/repost_tree_summ.zip`

[3] `http://taku910.github.io/crfpp/`

experiments, respectively. In spite of their comparable performance, our framework applies CRF instead of SVM^{hmm} for leader detection because of its probabilistic nature, which can be exploited by the sampling process in Soft-LeadSum to reduce the propagation of classification error to the summarization stage. Section 5.2.2 shows the relevant experiment.

5.2 Experiment for Summarization

In this experiment, we evaluated end-to-end performance of our basic and soft LeadSum summarization models by comparing them with state-of-the-art microblog summarizers.

5.2.1 Data Collection and Evaluation Metrics

There is no public editorial repost tree dataset. Therefore, we manually selected 10 hot events taking place during January 2nd – July 28th 2014, and then used the PKUVIS toolkit (Ren et al., 2014) to crawl the complete repost trees for all the events given the corresponding original posts. Table 3 shows the details about the repost tree corpus[4]. Note that this repost tree corpus has no overlap with the repost tree path dataset for learning leader detection models in Section 5.1.1.

After that, we invited three experienced editors to write summaries for each repost tree. To ensure the quality of reference summaries, we first extracted a list of frequent nouns from each repost tree and generalized 7 to 10 topics based on the nouns list, which provided a high-level overview of a repost tree to editors. Then, our guideline required editors to read all repost microblogs ordered sequentially on a repost tree. For every message, its entire repost tree path was also provided as supplementary context information. When finished reading, editors wrote down one or two sentences to summarize each topic in the list.

We utilized ROUGE-N metric (Lin, 2004) for benchmark, which is a standard for evaluating automatic summaries based on N-gram overlapping between a generated summary and a reference. Specifically, ROUGE-1 and ROUGE-2 F1-measure were used as our evaluation metrics. Lin et al. (2004) has demonstrated that ROUGE-2 correlates well with humans in summarizing formal texts. And ROUGE-1 is a better alternative in evaluating summaries for short and informal microblog messages (Inouye and Kalita, 2011; Chang et al., 2013).

In our human-generated summaries, the average inter-annotator-agreement by ROUGE-1 is 0.431, which means each pair of manual summaries have no more than 50% words overlap on average even written under topic constraints. This indicates that microblog repost tree summarization is generally a difficult task. The reason is that repost trees have complex structure, and editors could hardly reconstruct the repost trees even though they went through all the microblogs. Therefore, in evaluation for each tree, we computed the average ROUGE F1 score between the model-generated summary and the three human-generated summaries.

5.2.2 Results

In each automatic summarizer, we selected the top-10 ranked reposts to form a summary. We compared the end-to-end performance with the following baseline systems:

- **RandSum:** RandSum is a weak baseline that randomly selects reposts into summaries.
- **RepSum:** RepSum ranks and selects messages simply by their reposts count, i.e., the size of their subtrees, based on reposting relations.
- **UserRankSum:** UserRankSum ranks and selects reposts by their authors' follower count based on user following relations.
- **LeadProSum:** LeadProSum ranks and selects reposting messages by their marginal probabilities as leaders determined by our CRF-based leader detection model.
- **SVDSum:** SVDSum adopts the Singular Value Decomposition (SVD) to discover hidden sub-topics for summarization (Gong and Liu, 2001). Reposting messages are ranked according to latent semantic analysis with SVD on term-message matrix.
- **DivRankSum:** DivRankSum directly applies DivRank (Mei et al., 2010) algorithm to rank all messages unaware of leaders and followers. A similar model is also reported in Yan et al. (2011). Following their work, we set damping weight as 0.85.
- **UserInfSum:** Chang et al. (2013) ranks messages utilizing Gradient Boosted Decision Tree (GBDT) algorithm with text, popularity, temporal and user influence signals to summarize Twitter context tree. In particular, without the interaction data with external users, we utilize users' fol-

[4] All descriptions are English translations of the root microblogs originally in Chinese.

Table 3: Description of repost tree summarization corpus consisting of 10 hot events

Name	# of nodes	# of nodes with comments	Height	Description
Tree (I)	21,353	15,409	16	HKU dropping out student wins the college entrance exam again.
Tree (II)	9,616	6,073	11	German boy complains hard schoolwork in Chinese High School.
Tree (III)	13,087	9,583	8	Movie Tiny Times 1.0 wins high grossing in criticism.
Tree (IV)	12,865	7,083	8	"I am A Singer" states that singer G.E.M asking for resinging conforms to rules.
Tree (V)	10,666	7,129	8	Crystal Huang clarified the rumor of her derailment.
Tree (VI)	21,127	15,057	11	Germany routs Brazil 7:1 in World-Cup semi-final.
Tree (VII)	18,974	12,399	13	The pretty girl pregnant with a second baby graduated with her master degree.
Tree (VIII)	2,021	925	18	Girls appealed for equality between men and women in college admission
Tree (IX)	9,230	5,408	14	Violent terrorist attack in Kunming railway station.
Tree (X)	10,052	4,257	25	MH17 crash killed many top HIV researchers.

Table 4: Comparison of different summarizers

	ROUGE-1			ROUGE-2		
	F1	σ	SIG	F1	σ	SIG
RandSum	.159	.046	**‡	.037	.009	**‡
RepSum	.162	.071	**‡	.030	.016	**‡
UserRankSum	.292	.066	‡	.087	.028	†
LeadProSum	.270	.119	‡	.064	.038	‡
SVDSum	.222	.070	**‡	.048	.032	**‡
DivRankSum	.159	.079	**‡	.029	.018	**‡
UserInfSum	.272	.091	‡	.071	.028	‡
B-LS+SVM^{hmm}	.301	.031	‡	.085	.020	†
B-LS+CRF	.300	.029	‡	.082	.016	‡
S-LS+CRF	**.351**	.027	NA	**.105**	.018	NA

Remarks:
B-LS: Basic-LeadSum model; S-LS: Soft-LeadSum model
F1: F1-measure of ROUGE-1 or ROUGE-2
σ: Standard deviation of F1-measure over 10 repost trees
SIG: Significance indicator of F1-measure based on one-tailed pairwise t-test:
– Significantly different with B-LS+CRF: * ($p < 0.1$); ** ($p < 0.05$)
– Significantly different with S-LS+CRF: † ($p < 0.1$); ‡ ($p < 0.05$)

lower count to approximate user influence. GBDT implementation is based on RankLib[5], and as a supervised method, UserInfSum is evaluated with 10-fold cross validation.

In addition, we observed that SVM^{hmm} is a competitive baseline for leader detection (see Table 2). So we also study its impact on the Basic-LeadSum model. Note that SVM^{hmm} cannot be combined with Soft-LeadSum since it is not probabilistic.

Table 4 shows the result of overall comparisons. We have the following observations:

- RepSum utilized trivial structure information, i.e., the size of sub-tree, and its performance was poor, which was even worse than RandSum on ROUGE-2. This implies that messages with a lot of reposts may not be good candidates as other reasons may lead to their popularity, e.g., a good posting time or sense of humor.

- UserRankSum performed the best on ROUGE-1&2 among all baseline summarizers, which confirms that user following relations can indeed be a strong signal in microblog summarization. UserRankSum is even slightly better than Basic-LeadSum on ROUGE-2. But, it does not perform consistently well for all repost trees, evidenced as the large standard deviation on ROUGE-1&2. This suggests that the user following relations cannot always effectively indicate salient candidates. It may not work for repost trees where authors have similar number of following users, or reposts of influential users contain nothing salient.

- LeadProSum achieved the second best performance among all unsupervised baselines, which indicates that the marginal probabilities as leaders can signal good summary candidates. This also confirms that leaders contain salient contents and should be distinguished from followers in summarization.

- Utilizing either SVM^{hmm} or CRF as leader detection model to filter out followers, Basic-LeadSum almost doubled the ROUGE-1 and tripled the ROUGE-2 scores compared to DivRankSum's performance. This indicates that differentiating leaders and followers is very helpful to summarization.

- Basic-LeadSum performed better than all baselines on ROUGE-1&2 except for a marginal drop compared to UserRankSum on ROUGE-2. But the differences with UserRankSum, LeadProSum and UserInfSum are not statistically significant. This may be ascribed to the error propagated from leader detection module to summarization process.

- Soft-LeadSum outperformed all the baselines with a large margin on ROUGE-1&2, including supervised summarizer UserInfSum. The one-tailed pairwise t-test indicates that all the improvements over baselines are significant at the 95% confidence level except for UserRankSum with 90% confidence level on ROUGE-2. This confirms the effectiveness of our framework for producing high-quality repost tree summaries.

[5] http://sourceforge.net/p/lemur/wiki/RankLib/

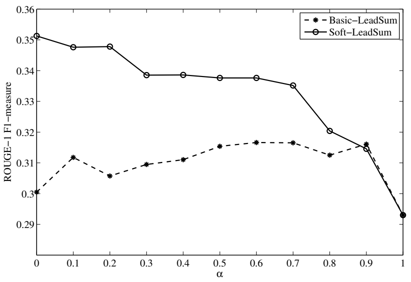

Figure 2: The impact of α on the ROUGE-1 F1-measure of combined models

• The supervised model UserInfSum did not perform quite well. The reason is that the model needs large amount of user interaction data external to the tree which are not readily available, and also it might be overfitting to the limited number of training instances.

• Basic-LeadSum with CRF and SVM^{hmm} had very close ROUGE-1&2 scores. Basic-LeadSum+SVM^{hmm} is even slightly better than Basic-LeadSum+CRF. Though SVM^{hmm} was marginally worse in leader detection experiment (Table 2), we can conclude that SVM^{hmm} is a comparable alternative as the leader detection module for Basic-LeadSum.

• Among our models, Soft-LeadSum significantly outperformed both Basic-LeadSum with CRF and that with SVM^{hmm}. This implies that sampling steps in the enhanced random walk of Soft-LeadSum is effective in reducing the impact of leader detection error on summarization.

5.3 Discussion

From Table 4, we observed that user following relations used by UserRankSum is a strong signal for microblog summarization. A natural question is: "Can the user following relations commonly used for modeling user influence be complementary to the content-level structure information used in our summariztaion models?"

We thus linearly combine the normalized ranking scores of LeadSum and UserRankSum using the formula $\alpha * \mathbf{u} + (1 - \alpha) * \mathbf{l}$, where $\mathbf{u}$ and $\mathbf{l}$ denote the UserRankSum and LeadSum ranking scores, respectively. Figure 2 demonstrates the impact of α on our basic and soft LeadSum model with CRF.

Clearly, Basic-LeadSum can benefit from user influence information by incorporating UserRankSum scores into it. From the incremental trend of summarization performance with the increase of α for $\alpha \in [0, 0.9]$, we can conclude that user influence is helpful to it. This is because Basic-LeadSum is not sufficiently robust to the errors cascaded from leader detection module, thus user-level structures can have the chance to compensate these errors for content-level structures.

Incorporating the same information into Soft-LeadSum cannot improve its performance regardless of the value of α. This implies that content-level structures, i.e., message content and reposting relations together, are better indicative of good summary candidates. When these features are appropriately modeled by Soft-LeadSum, user influence, a traditionally well-known strong signal, cannot provide extra benefit at all.

6 Conclusion and Future Work

This work presents a study for microblog repost tree summarization, whose output can provide important clues for event analysis on microblogging platforms. Conventional works considering only plain text streams is insufficient to summarize noisy repost trees. We propose a novel summarization system by effectively differentiating leader and follower messages on repost tree based on content-level structure information. Firstly, a leader detection model categorizes each repost on repost tree path as a leader or a follower. Then, a random-walk variant summarization model called LeadSum is proposed to rank and select salient microblog messages on the basis of leader detection result. To reduce errors cascaded from leader detection, we enhance LeadSum based on an even-length random walk by sampling from leader probabilities for improving summarization. Based on real-world microblog post dataset, the experimental results confirm that our proposed framework is effective for repost tree summarization by the end-to-end comparison with the state-of-the-art baselines.

Constrained by the amount of annotation, we adopt this two-step framework and an unsupervised summarization algorithm. With the development of our corpora, we plan to explore the usefulness of supervised structure learning approaches, such as tree-structured CRF (Tang et al., 2006; Mensink et al., 2013), to integrate leader detection and summarization into a unified framework, and make global inference for important leaders by capturing various non-linear dependencies.

References

Yasemin Altun, Ioannis Tsochantaridis, and Thomas Hofmann. 2003. Hidden markov support vector machines. In *Proceedings of the Twentieth International Conference on Machine Learning, ICML*, pages 3–10.

Deepayan Chakrabarti and Kunal Punera. 2011. Event summarization using tweets. In *Proceedings of the Fifth International Conference on Weblogs and Social Media*, pages 66–73.

Yi Chang, Xuanhui Wang, Qiaozhu Mei, and Yan Liu. 2013. Towards twitter context summarization with user influence models. In *Sixth ACM International Conference on Web Search and Data Mining, WSDM*, pages 527–536.

Dipanjan Das and André FT Martins. 2007. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195.

Yajuan Duan, Zhimin Chen, Furu Wei, Ming Zhou, and Heung-Yeung Shum. 2012. Twitter topic summarization by ranking tweets using social influence and content quality. In *Proceedings of the 24th International Conference on Computational Linguistics, COLING*, pages 763–780.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res. (JAIR)*, 22:457–479.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. 2013. *Statistical methods for rates and proportions*. John Wiley & Sons.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM Conference on Research and Development in Information Retrieval, SIGIR*, pages 19–25.

David Inouye and Jugal K. Kalita. 2011. Comparing twitter summarization algorithms for multiple post summaries. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom)*, pages 298–306.

Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. 2009. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML*, pages 282–289.

Liangda Li, Ke Zhou, Gui-Rong Xue, Hongyuan Zha, and Yong Yu. 2009. Enhancing diversity, coverage and balance for summarization through structure learning. In *Proceedings of the 18th International Conference on World Wide Web, WWW*, pages 71–80.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL-04 Workshop*, pages 74–81.

Fei Liu, Yang Liu, and Fuliang Weng. 2011. Why is sxsw trending?: exploring multiple text sources for twitter topic summarization. In *Proceedings of the Workshop on Languages in Social Media*, pages 66–75. Association for Computational Linguistics.

Xiaohua Liu, Yitong Li, Furu Wei, and Ming Zhou. 2012. Graph-based multi-tweet summarization using social signals. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1699–1714.

He Liu, Hongliang Yu, and Zhi-Hong Deng. 2015. Multi-document summarization based on two-level sparse representation model. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence, AAAI*, pages 196–202.

Rui Long, Haofen Wang, Yuqiang Chen, Ou Jin, and Yong Yu. 2011. Towards effective event detection, tracking and summarization on microblog data. In *Web-Age Information Management - 12th International Conference, WAIM*, pages 652–663.

Qiaozhu Mei, Jian Guo, and Dragomir R. Radev. 2010. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,KDD*, pages 1009–1018.

Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Sujian Li, and Houfeng Wang. 2012. Entity-centric topic-oriented opinion summarization in twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 379–387.

Thomas Mensink, Jakob J. Verbeek, and Gabriela Csurka. 2013. Tree-structured CRF models for interactive image labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):476–489.

Xipeng Qiu, Qi Zhang, and Xuanjing Huang. 2013. Fudannlp: A toolkit for chinese natural language processing. In *Proceedings of Annual Meeting of the Association for Computational Linguistics, ACL*, pages 49–54.

Dragomir R. Radev, Eduard H. Hovy, and Kathleen McKeown. 2002. Introduction to the special issue on summarization. *Computational Linguistics*, 28(4):399–408.

Dragomir R. Radev, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Çelebi, Stanko Dimitrov, Elliott Drábek, Ali Hakim, Wai Lam, Danyu Liu, Jahna Otterbacher, Hong Qi, Horacio Saggion, Simone Teufel, Michael Topper, Adam Winkel, and Zhu Zhang. 2004. MEAD - A platform for multidocument multilingual text summarization. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC*.

Kenneth Rapoza. 2011. China's weibos vs us's twitter: And the winner is.

Donghao Ren, Xin Zhang, Zhenhuang Wang, Jing Li, and Xiaoru Yuan. 2014. Weiboevents: A crowd sourcing weibo visual analytic system. In *IEEE Pacific Visualization Symposium, PacificVis*, pages 330–334.

Kevin Dela Rosa, Rushin Shah, Bo Lin, Anatole Gershman, and Robert Frederking. 2011. Topical clustering of tweets. *Proceedings of the ACM SIGIR: SWSM*.

Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. 2010. Automatic summarization of twitter topics. In *National Workshop on Design and Analysis of Algorithm*.

Chao Shen, Fei Liu, Fuliang Weng, and Tao Li. 2013. A participant-based approach for event summarization using twitter streams. In *Proceedings of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, HLT-NAACL*, pages 1152–1162.

Hiroya Takamura, Hikaru Yokono, and Manabu Okumura. 2011. Summarizing a document stream. In *Advances in Information Retrieval - 33rd European Conference on IR Research, ECIR*, pages 177–188.

Jie Tang, MingCai Hong, Juan-Zi Li, and Bangyong Liang. 2006. Tree-structured conditional random fields for semantic annotation. In *The Proceedings of 5th International Semantic Web Conference, ISWC*, pages 640–653.

Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. 2011. Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 433–443.