Using Tweets to Help Sentence Compression for News Highlights Generation

Zhongyu Wei¹, Yang Liu¹, Chen Li¹, Wei Gao²

¹Computer Science Department, The University of Texas at Dallas
Richardson, Texas 75080, USA
²Qatar Computing Research Institute, Hamad Bin Khalifa University, Doha, Qatar
{zywei,yangl,chenli}@hltdallas.edu¹
wgao@qf.org.qa²

Abstract

We explore using relevant tweets of a given news article to help sentence compression for generating compressive news highlights. We extend an unsupervised dependency-tree based sentence compression approach by incorporating tweet information to weight the tree edge in terms of informativeness and syntactic importance. The experimental results on a public corpus that contains both news articles and relevant tweets show that our proposed tweets guided sentence compression method can improve the summarization performance significantly compared to the baseline generic sentence compression method.

1 Introduction

“Story highlights” of news articles are provided by only a few news websites such as CNN.com. The highlights typically consist of three or four succinct itemized sentences for readers to quickly capture the gist of the document, and can dramatically reduce reader’s information load. A highlight sentence is usually much shorter than its original corresponding news sentence; therefore applying extractive summarization methods directly to sentences in a news article is not enough to generate high quality highlights.

Sentence compression aims to retain the most important information of an original sentence in a shorter form while being grammatical at the same time. Previous research has shown the effectiveness of sentence compression for automatic document summarization (Knight and Marcu, 2000; Lin, 2003; Galanis and Androutsopoulos, 2010; Chali and Hasan, 2012; Wang et al., 2013; Li et al., 2013; Qian and Liu, 2013; Li et al., 2014). The compressed summaries can be generated through a pipeline approach that combines a generic sentence compression model with a summary sentence pre-selection or post-selection step. Prior studies have mostly used the generic sentence compression approaches, however, a generic compression system may not be the best fit for the summarization purpose because it does not take into account the summarization task in the compression module. Li et al. (2013) thus proposed a summary guided compression method to address this problem and showed the effectiveness of their method. But this approach relied heavily on the training data, thus has the limitation of domain generalization.

Instead of using a manually generated corpus, we investigate using existing external sources to guide sentence compression for the purpose of compressive news highlights generation. Nowadays it becomes more and more common that users share interesting news content via Twitter together with their comments. The availability of cross-media information provides new opportunities for traditional tasks of Natural Language Processing (Zhao et al., 2011; Subašić and Berendt, 2011; Gao et al., 2012; Kothari et al., 2013; Štajner et al., 2013). In this paper, we propose to use relevant tweets of a news article to guide the sentence compression process in a pipeline framework for generating compressive news highlights. This is a pioneer study for using such parallel data to guide sentence compression for document summarization.

Our work shares some similar ideas with (Wei and Gao, 2014; Wei and Gao, 2015). They also attempted to use tweets to help news highlights generation. Wei and Gao (2014) derived external features based on the relevant tweet collection to assist the ranking of the original sentences for extractive summarization in a fashion of supervised machine learning. Wei and Gao (2015) proposed a graph-based approach to simultaneously rank the
original news sentences and relevant tweets in an unsupervised way. Both of them focused on using tweets to help sentence extraction while we leverage tweet information to guide sentence compression for compressive summary generation.

We extend an unsupervised dependency-tree based sentence compression approach to incorporate tweet information from the aspects of both informativeness and syntactic importance to weight the tree edge. We evaluate our method on a public corpus that contains both news articles and relevant tweets. The result shows that generic compression hurts the performance of highlights generation, while sentence compression guided by relevant tweets of the news article can improve the performance.

2 Framework

We adopt a pipeline approach for compressive news highlights generation. The framework integrates a sentence extraction component and a post-sentence compression component. Each is described below.

2.1 Tweets Involved Sentence Extraction

We use LexRank (Erkan and Radev, 2004) as the baseline to select the salient sentences in a news article. This baseline is an unsupervised extractive summarization approach and has been proved to be effective for the summarization task.

Besides LexRank, we also use Heterogeneous Graph Random Walk (HGRW) (Wei and Gao, 2015) to incorporate relevant tweet information to extract news sentences. In this model, an undirected similarity graph is created, similar to LexRank. However, the graph is heterogeneous, with two types of nodes for the news sentences and tweets respectively.

Suppose we have a sentence set $S$ and a tweet set $T$. By considering the similarity between the same type of nodes and cross types, the score of a news sentence $s$ is computed as follows:

$$p(s) = \frac{d}{N + M} + (1 - d) \left[ \epsilon \sum_{m \in T} \frac{\text{sim}(s, m)}{\sum_{v \in T} \text{sim}(s, v)} p(m) \right] + (1 - \epsilon) \left[ \sum_{n \in T \setminus \{s\}} \frac{\text{sim}(s, n)}{\sum_{m \in T \setminus \{s\}} \text{sim}(s, m)} p(n) \right]$$

(1)

where $N$ and $M$ are the size of $S$ and $T$, respectively, $d$ is a damping factor, $\text{sim}(x, y)$ is the similarity function, and the parameter $\epsilon$ is used to control the contribution of relevant tweets. For a tweet node $t$, its score can be computed similarly. Both $d$ and $\text{sim}(x, y)$ are computed following the setup of LexRank, where $\text{sim}(x, y)$ is computed as cosine similarity:

$$\text{sim}(x, y) = \frac{\sum_{w \in x,y} t_{fw,w} \times t_{fw,w} \times idf_w}{\sqrt{\sum_{w \in x,y} (t_{fw,w} \times idf_w)^2} \times \sqrt{\sum_{w \in x,y} (t_{fw,w} \times idf_w)^2}}$$

(2)

where $t_{fw,w}$ is the number of occurrences of word $w$ in instance $x$, $idf_w$ is the inverse document frequency of word $w$ in the dataset. In our task, each sentence or tweet is treated as a document to compute the IDF value.

Although both types of nodes can be ranked in this framework, we only output the top news sentences as the highlights, and the input to the subsequent compression component.

2.2 Dependency Tree Based Sentence Compression

We use an unsupervised dependency tree based compression framework (Filippova and Strube, 2008) as our baseline. This method achieved a higher F-score (Riezler et al., 2003) than other systems on the Edinburgh corpus (Clarke and Lapata, 2006). We will introduce the baseline in this part and describe our extended model that leverages tweet information in the next subsection.

The sentence compression task can be defined as follows: given a sentence $s$, consisting of words $w_1, w_2, ..., w_m$, identify a subset of the words of $s$, such that it is grammatical and preserves essential information of $s$. In the baseline framework, a dependency graph for an original sentence is first generated and then the compression is done by deleting edges of the dependency graph. The goal is to find a subtree with the highest score:

$$f(X) = \sum_{e \in E} x_e \times w_{inf}(e) \times w_{syn}(e)$$

(3)

where $x_e$ is a binary variable, indicating whether a directed dependency edge $e$ is kept ($x_e = 1$) or removed ($x_e = 0$), and $E$ is the set of edges in the dependency graph. The weighting of edge $e$ considers both its syntactic importance ($w_{syn}(e)$) as well as the informativeness ($w_{inf}(e)$). Suppose edge $e$ is pointed from head $h$ to node $n$ with dependency label $I$, both weights can be computed from a background news corpus as:

$$w_{inf}(e) = \frac{P_{\text{summary}(n)}}{P_{\text{article}(n)}}$$

(4)
\[ w_{\text{syn}}(e) = P(l|h) \]

where \( P_{\text{summary}}(n) \) and \( P_{\text{article}}(n) \) are the unigram probabilities of word \( n \) in the two language models trained on human generated summaries and the original articles respectively. \( P(l|h) \) is the conditional probability of label \( l \) given head \( h \). Note that here we use the formula in (Filippova and Altun, 2013) for \( w_{\text{info}}(e) \), which was shown to be more effective for sentence compression than the original formula in (Filippova and Strube, 2008).

The optimization problem can be solved under the tree structure and length constraints by integer linear programming\(^1\). Given that \( L \) is the maximum number of words permitted for the compression, the length constraint is simply represented as:

\[ \sum_{e \in E} x_e \leq L \quad (6) \]

The surface realizations is standard: the words in the compression subtree are put in the same order they are found in the source sentence. Due to space limit, we refer readers to (Filippova and Strube, 2008) for a detailed description of the baseline method.

2.3 Leverage Tweets for Edge Weighting

We then extend the dependency-tree based compression framework by incorporating tweet information for dependency edge weighting. We introduce two new factors, \( w_{\text{info}}^T(e) \) and \( w_{\text{syn}}^T(e) \), for informativeness and syntactic importance respectively, computed from relevant tweets of the news. These are combined with the weights obtained from the background news corpus defined in Section 2.2, as shown below:

\[ w_{\text{info}}(e) = (1 - \alpha) \cdot w_{\text{info}}^N(e) + \alpha \cdot w_{\text{info}}^T(e) \quad (7) \]

\[ w_{\text{syn}}(e) = (1 - \beta) \cdot w_{\text{syn}}^N(e) + \beta \cdot w_{\text{syn}}^T(e) \quad (8) \]

where \( \alpha \) and \( \beta \) are used to balance the contribution of the two sources, and \( w_{\text{info}}^N(e) \) and \( w_{\text{syn}}^N(e) \) are based on Equation 4 and 5.

The new informative weight \( w_{\text{info}}^T(e) \) is calculated as:

\[ w_{\text{info}}^T(e) = \frac{P_{\text{relevant}}(n)}{P_{\text{background}}(n)} \quad (9) \]

\( P_{\text{relevant}}(n) \) and \( P_{\text{background}}(n) \) are the unigram probabilities of word \( n \) in two language models trained on the relevant tweet dataset and a background tweet dataset respectively.

The new syntactic importance score is:

\[ w_{\text{syn}}^T(e) = \frac{NT(h, n)}{NT} \quad (10) \]

\( NT(h, n) \) is the number of tweets where \( n \) and head \( h \) appear together within a window frame of \( K \), and \( NT \) is the total number of tweets in the relevant tweet collection. Since tweets are always noisy and informal, traditional parsers are not reliable to extract dependency trees. Therefore, we use co-occurrence as pseudo syntactic information here. Note \( w_{\text{info}}^N(e), w_{\text{info}}^T(e), w_{\text{syn}}^N(e) \) and \( w_{\text{syn}}^T(e) \) are normalized before combination.

3 Experiment

3.1 Setup

We evaluate our pipeline news highlights generation framework on a public corpus based on CNN/USAToday news (Wei and Gao, 2014). This corpus was constructed via an event-oriented strategy following four steps: 1) 17 salient news events taking place in 2013 and 2014 were manually identified. 2) For each event, relevant tweets were retrieved via Topsy\(^2\) search API using a set of manually generated core queries. 3) News articles explicitly linked by URLs embedded in the tweets were collected. 4) News articles from CNN/USAToday that have more than 100 explicitly linked tweets were kept. The resulting corpus contains 121 documents, 455 highlights and 78,419 linking tweets.

We used tweets explicitly linked to a news article to help extract salience sentences in HGRW and to generate the language model for computing \( w_{\text{info}}^T(e) \). The co-occurrence information computed from the set of explicitly linked tweets is very sparse because the size of the tweet set is small. Therefore, we used all the tweets retrieved for the event related to the target news article to compute the co-occurrence information for \( w_{\text{syn}}^T(e) \). Tweets retrieved for events were not published in (Wei and Gao, 2014). We make it available here\(^3\). The statistics of the dataset can be found in Table. 1.

---

\(^1\)In our implementation we use GNU Linear Programming Kit (GULP) (https://www.gnu.org/software/glpk/)

\(^2\)http://topsy.com

\(^3\)http://www.hlt.utdallas.edu/~zywei/data/CNNUSATodayEvent.zip
Table 1: Distribution of documents, highlights and tweets with respect to different events

<table>
<thead>
<tr>
<th>Event</th>
<th>Doc #</th>
<th>HLight #</th>
<th>Linked Tweet #</th>
<th>Retained Tweet #</th>
<th>Event</th>
<th>Doc #</th>
<th>HLight #</th>
<th>Linked Tweet #</th>
<th>Retained Tweet #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aurora shooting</td>
<td>14</td>
<td>54</td>
<td>12.463</td>
<td>588.140</td>
<td>African runner murder</td>
<td>8</td>
<td>29</td>
<td>9.461</td>
<td>303.335</td>
</tr>
<tr>
<td>Boston bombing</td>
<td>13</td>
<td>47</td>
<td>3.021</td>
<td>213.864</td>
<td>US military in Syria</td>
<td>2</td>
<td>7</td>
<td>719</td>
<td>619.22</td>
</tr>
<tr>
<td>Connecticut shooting</td>
<td>5</td>
<td>17</td>
<td>1.955</td>
<td>379.349</td>
<td>DPRK Nuclear Test</td>
<td>2</td>
<td>8</td>
<td>3.329</td>
<td>103.964</td>
</tr>
<tr>
<td>Egypt balloon crash</td>
<td>4</td>
<td>15</td>
<td>6.07</td>
<td>189.082</td>
<td>Moore Tornado</td>
<td>5</td>
<td>10</td>
<td>1.259</td>
<td>1,154.656</td>
</tr>
<tr>
<td>Hurricane Sandy</td>
<td>2</td>
<td>6</td>
<td>6.841</td>
<td>239.281</td>
<td>Chinese Computer Attacks</td>
<td>2</td>
<td>8</td>
<td>507</td>
<td>28.988</td>
</tr>
<tr>
<td>Russian meteor</td>
<td>3</td>
<td>11</td>
<td>6.304</td>
<td>1,042.169</td>
<td>Williams Olefins Explosion</td>
<td>1</td>
<td>4</td>
<td>268</td>
<td>14.196</td>
</tr>
<tr>
<td>US Flu Season</td>
<td>7</td>
<td>23</td>
<td>6.304</td>
<td>1,042.169</td>
<td>Total</td>
<td>121</td>
<td>455</td>
<td>78.419</td>
<td>6,890.987</td>
</tr>
<tr>
<td>Super Bowl blackout</td>
<td>2</td>
<td>4</td>
<td>4.82</td>
<td>214.775</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Overall Performance. Bold: the best value in each group in terms of different metrics.

<table>
<thead>
<tr>
<th>Method</th>
<th>ROUGE-1</th>
<th>Compute</th>
<th>Rate(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LexRank</td>
<td>25.2</td>
<td>22.4</td>
<td>29.6</td>
</tr>
<tr>
<td>LexRank + SC</td>
<td>26.2</td>
<td>23.5</td>
<td>30.4</td>
</tr>
<tr>
<td>LexRank + SC + w^T_{info}</td>
<td>26.4</td>
<td>24.9</td>
<td>29.5</td>
</tr>
<tr>
<td>LexRank + SC + w^T_{syn}</td>
<td>27.5</td>
<td>25.0</td>
<td>31.4</td>
</tr>
<tr>
<td>LexRank + SC + both</td>
<td>28.1</td>
<td>22.9</td>
<td>29.5</td>
</tr>
<tr>
<td>HGRW</td>
<td>26.4</td>
<td>24.9</td>
<td>29.5</td>
</tr>
<tr>
<td>HGRW + SC</td>
<td>27.5</td>
<td>25.3</td>
<td>30.8</td>
</tr>
<tr>
<td>HGRW + SC + w^T_{info}</td>
<td>27.5</td>
<td>25.3</td>
<td>30.2</td>
</tr>
<tr>
<td>HGRW + SC + w^T_{syn}</td>
<td>28.4</td>
<td>26.9</td>
<td>31.2</td>
</tr>
<tr>
<td>HGRW + SC + both</td>
<td>28.4</td>
<td>26.9</td>
<td>31.2</td>
</tr>
</tbody>
</table>

3.2 Results
Table 2 shows the overall performance. For summaries generated by both LexRank and HGRW, “+SC” means generic sentence compression base-

3http://nlp.stanford.edu/software/lex-parser.shtml

4The performance of HGRW reported here is different from (Wei and Gao, 2015) because the setup is different. We use all the explicitly linked tweets in the ranking process here without considering redundancy while a redundancy filtering process was applied in (Wei and Gao, 2015).

The improvement obtained by LexRank + SC + both compared to LexRank is more promising than that obtained by HGRW + SC + both compared to HGRW. This may be because HGRW has used tweet information already, and leaves limited room for improvement for the sentence compression model when using the same source of information.

6Significance throughout the paper is computed by two tailed t-test and reported when p < 0.05.
Figure 1: The influence of $\alpha$ and $\beta$. Solid lines are used for approaches based on LexRank; Dotted lines are used for HGRW based approaches.

<table>
<thead>
<tr>
<th>Method</th>
<th>Example 1</th>
<th>Example 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LexRank</td>
<td>Boston bombing suspect Tamerlan Tsarnaev, killed in a shootout with police days after the blast, has been buried at an undisclosed location, police in Worcester, Mass., said.</td>
<td>Three people were hospitalized in critical condition, according to information provided by hospitals who reported receiving patients from the blast.</td>
</tr>
<tr>
<td>LexRank+SC</td>
<td>suspect Tamerlan Tsarnaev, killed in a shootout after the blast, has been buried at an location, police in Worcester Mass. said.</td>
<td>Three people were hospitalized, according to information provided by hospitals who reported receiving from the blast.</td>
</tr>
<tr>
<td>LexRank+SC+both</td>
<td><strong>Boston bombing</strong> suspect Tamerlan Tsarnaev, killed in a shootout after the blast, has been buried at an location police said.</td>
<td>Three people were hospitalized in <strong>critical condition</strong>, according to information provided by hospitals.</td>
</tr>
<tr>
<td>Ground Truth</td>
<td><strong>Boston bombing</strong> suspect Tamerlan Tsarnaev has been buried at an undisclosed location</td>
<td>Hospitals report three people in <strong>critical condition</strong></td>
</tr>
</tbody>
</table>

Table 3: Example highlight sentences from different systems

- By incorporating tweet information for both sentence selection and compression, the performance of $HGRW+SC+both$ outperforms $LexRank$ significantly.

Table 3 shows some examples. As we can see in Example 1, with the help of tweet information, our compression model keeps the valuable part “Boston bombing” for summarization while the generic one abandons it.

We also investigate the influence of $\alpha$ and $\beta$. To study the impact of $\alpha$, we fix $\beta$ to 0.8, and vice versa. As shown in Figure 1, it is clear that larger $\alpha$ or $\beta$, i.e., giving higher weights to tweets related information, is generally helpful.

4 Conclusion and Future Work

In this paper, we showed that the relevant tweet collection of a news article can guide the process of sentence compression to generate better story highlights. We extended a dependency-tree based sentence compression model to incorporate tweet information. The experiment results on a public corpus that contains both news articles and rele-
vant tweets showed the effectiveness of our approach. With the popularity of Twitter and increasing interaction between social media and news media, such parallel data containing news and related tweets is easily available, making our approach feasible to be used in a real system.

There are some interesting future directions. For example, we can explore more effective ways to incorporate tweets for sentence compression; we can study joint models to combine both sentence extraction and compression with the help of relevant tweets; it will also be interesting to use the parallel dataset of the news articles and the tweets for timeline generation for a specific event.

Acknowledgments

We thank the anonymous reviewers for their detailed and insightful comments on earlier drafts of this paper. The work is partially supported by NSF award IIS-0845484 and DARPA Contract No. FA8750-13-2-0041. Any opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the funding agencies.
References


