Event-Driven Headline Generation

Rui Sun[†], **Yue Zhang**[‡], **Meishan Zhang**[‡] **and Donghong Ji**[†]
[†] Computer School, Wuhan University, China
[‡] Singapore University of Technology and Design
{ruisun, dhji}@whu.edu.cn
{yue_zhang, meishan_zhang}@sutd.edu.sg

Abstract

We propose an event-driven model for headline generation. Given an input document, the system identifies a key event chain by extracting a set of structural events that describe them. Then a novel multi-sentence compression algorithm is used to fuse the extracted events, generating a headline for the document. Our model can be viewed as a novel combination of extractive and abstractive headline generation, combining the advantages of both methods using event structures. Standard evaluation shows that our model achieves the best performance compared with previous state-of-the-art systems.

1 Introduction

Headline generation (HG) is a text summarization task, which aims to describe an article (or a set of related paragraphs) using a single short sentence. The task is useful in a number of practical scenarios, such as compressing text for mobile device users (Corston-Oliver, 2001), generating table of contents (Erbs et al., 2013), and email summarization (Wan and McKeown, 2004). This task is challenging in not only informativeness and readability, which are challenges to common summarization tasks, but also the length reduction, which is unique for headline generation.

Previous headline generation models fall into two main categories, namely extractive HG and abstractive HG (Woodsend et al., 2010; Alfonseca et al., 2013). Both consist of two steps: candidate extraction and headline generation. Extractive models choose a set of salient sentences in candidate extraction, and then exploit sentence compression techniques to achieve headline generation (Dorr et al., 2003; Zajic et al., 2005). Abstractive models choose a set of informative phrases for candidate extraction, and then exploit sentence synthesis techniques for headline generation (Soricut and Marcu, 2007; Woodsend et al., 2010; Xu et al., 2010).

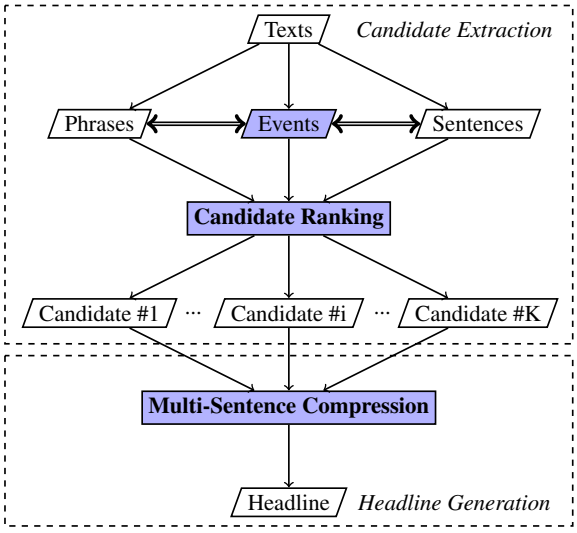

Figure 1: System framework.

Extractive HG and abstractive HG have their respective advantages and disadvantages. Extractive models can generate more readable headlines, because the final title is derived by tailoring human-written sentences. However, extractive models give less informative titles (Alfonseca et al., 2013), because sentences are very sparse, making high-recall candidate extraction difficult. In contrast, abstractive models use phrases as the basic processing units, which are much less sparse. However, it is more difficult for abstractive HG to ensure the grammaticality of the generated titles, given that sentence synthesis is still very inaccurate based on a set of phrases with little grammatical information (Zhang, 2013).

In this paper, we propose an event-driven model for headline generation, which alleviates the

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, pages 462–472, Beijing, China, July 26-31, 2015.

disadvantages of both extractive and abstractive HG. The framework of the proposed model is shown in Figure 1. In particular, we use *events* as the basic processing units for candidate extraction. We use structured tuples to represent the subject, predicate and object of an event. This form of event representation is widely used in open information extraction (Fader et al., 2011; Qiu and Zhang, 2014). Intuitively, events can be regarded as a trade-off between sentences and phrases. Events are meaningful structures, containing necessary grammatical information, and yet are much less sparse than sentences. We use salience measures of both sentences and phrases for event extraction, and thus our model can be regarded as a combination of extractive and abstractive HG.

During the headline generation step, A graph-based multi-sentence compression (MSC) model is proposed to generate a final title, given multiple events. First a directed acyclic word graph is constructed based on the extracted events, and then a beam-search algorithm is used to find the best title based on path scoring.

We conduct experiments on standard datasets for headline generation. The results show that headline generation can benefit not only from exploiting events as the basic processing units, but also from the proposed graph-based MSC model. Both our candidate extraction and headline generation methods outperform competitive baseline methods, and our model achieves the best results compared with previous state-of-the-art systems.

2 Background

Previous extractive and abstractive models take two main steps, namely candidate extraction and headline generation. Here, we introduce these two types of models according to the two steps.

2.1 Extractive Headline Generation

Candidate Extraction. Extractive models exploit sentences as the basic processing units in this step. Sentences are ranked by their salience according to specific strategies (Dorr et al., 2003; Erkan and Radev, 2004; Zajic et al., 2005). One of the state-of-the-art approaches is the work of Erkan and Radev (2004), which exploits centroid, position and length features to compute sentence salience. We re-implemented this method as our baseline sentence ranking method. In this paper, we use `SentRank` to denote this method.

Headline Generation. Given a set of sentences, extractive models exploit sentence compression techniques to generate a final title. Most previous work exploits single-sentence compression (SSC) techniques. Dorr et al. (2003) proposed the Hedge Trimmer algorithm to compress a sentence by making use of handcrafted linguistically-based rules. Alfonseca et al. (2013) introduce a multi-sentence compression (MSC) model into headline generation, using it as a baseline in their work. They indicated that the most important information is distributed across several sentences in the text.

2.2 Abstractive Headline Generation

Candidate Extraction. Different from extractive models, abstractive models exploit phrases as the basic processing units. A set of salient phrases are selected according to specific principles during candidate extraction (Schwartz, 01; Soricut and Marcu, 2007; Xu et al., 2010; Woodsend et al., 2010). Xu et al. (2010) propose to rank phrases using background knowledge extracted from Wikipedia. Woodsend et al. (2010) use supervised models to learn the salience score of each phrase. Here, we use the work of Soricut and Marcu (2007) , namely `PhraseRank`, as our baseline phrase ranking method, which is an unsupervised model without external resources. The method exploits unsupervised topic discovery to find a set of salient phrases.

Headline Generation. In the headline generation step, abstractive models exploit sentence synthesis technologies to accomplish headline generation. Zajic et al. (2005) exploit unsupervised topic discovery to find key phrases, and use the Hedge Trimmer algorithm to compress candidate sentences. One or more key phrases are added into the compressed fragment according to the length of the headline. Soricut and Marcu (2007) employ WIDL-expressions to generate headlines. Xu et al. (2010) employ keyword clustering based on several bag-of-words models to construct a headline. Woodsend et al. (2010) use quasi-synchronous grammar (QG) to optimize phrase selection and surface realization preferences jointly.

3 Our Model

Similar to extractive and abstractive models, the proposed event-driven model consists of two steps, namely candidate extraction and headline generation.

3.1 Candidate Extraction

We exploit events as the basic units for candidate extraction. Here an event is a tuple (S, P, O), where S is the subject, P is the predicate and O is the object. For example, for the sentence *"Ukraine Delays Announcement of New Government"*, the event is *(Ukraine, Delays, Announcement)*. This type of event structures has been used in open information extraction (Fader et al., 2011), and has a range of NLP applications (Ding et al., 2014; Ng et al., 2014).

A sentence is a well-formed structure with complete syntactic information, but can contain redundant information for text summarization, which makes sentences very sparse. Phrases can be used to avoid the sparsity problem, but with little syntactic information between phrases, fluent headline generation is difficult. Events can be regarded as a trade-off between sentences and phrases. They are meaningful structures without redundant components, less sparse than sentences and containing more syntactic information than phrases.

In our system, candidate event extraction is performed on a bipartite graph, where the two types of nodes are lexical chains (Section 3.1.2) and events (Section 3.1.1), respectively. Mutual Reinforcement Principle (Zha, 2002) is applied to jointly learn chain and event salience on the bipartite graph for a given input. We obtain the top-*k* candidate events by their salience measures.

3.1.1 Extracting Events

We apply an open-domain event extraction approach. Different from traditional event extraction, for which types and arguments are predefined, open event extraction does not have a closed set of entities and relations (Fader et al., 2011). We follow Hu's work (Hu et al., 2013) to extract events.

Given a text, we first use the Stanford dependency parser[1] to obtain the Stanford typed dependency structures of the sentences (Marneffe and Manning, 2008). Then we focus on two relations, *nsubj* and *dobj*, for extracting event arguments. Event arguments that have the same predicate are merged into one event, represented by tuple (*Subject, Predicate, Object*). For example, given the sentence, *"the Keenans could demand the Aryan Nations' assets"*, Figure 2 present its partial parsing tree. Based on the parsing results, two event arguments are obtained: *nsubj(demand, Keenans)* and *dobj(demand, assets)*. The two event arguments are merged into one event: *(Keenans, demand, assets)*.

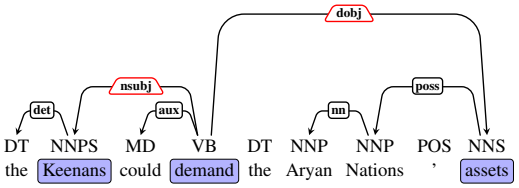

Figure 2: Dependency tree for the sentence *"the Keenans could demand the Aryan Nations' assets"*.

3.1.2 Extracting Lexical Chains

Lexical chains are used to link semantically-related words and phrases (Morris and Hirst, 1991; Barzilay and Elhadad, 1997). A lexical chain is analogous to a semantic synset. Compared with words, lexical chains are less sparse for event ranking.

Given a text, we follow Boudin and Morin (2013) to construct lexical chains based on the following principles:

1. All words that are identical after stemming are treated as one word;
2. All NPs with the same head word fall into one lexical chain;[2]
3. A pronoun is added to the corresponding lexical chain if it refers to a word in the chain (The coreference resolution is performed using the Stanford Coreference Resolution system);[3]
4. Lexical chains are merged if their main words are in the same synset of WordNet.[4]

[1] http://nlp.stanford.edu/software/lex-parser.shtml

[2] NPs are extracted according to the dependency relations *nn* and *amod*. As shown in Figure 2, we can extract the noun phrase *Aryan Nations* according to the dependency relation *nn(Nations, Aryan)*.

[3] http://nlp.stanford.edu/software/dcoref.shtml

[4] http://wordnet.princeton.edu/

At initialization, each word in the document is a lexical chain. We repeatedly merge existing chains by the four principles above until convergence. In particular, we focus on content words only, including verbs, nouns and adjective words. After the merging, each lexical chain represents a word cluster, and the first occuring word in it can be used as the main word of chain.

3.1.3 Learning Salient Events

Intuitively, one word should be more important if it occurs in more important events. Similarly, one event should be more important if it includes more important words. Inspired by this, we construct a bipartite graph between lexical chains and events, shown in Figure 3, and then exploit MRP to jointly learn the salience of lexical chains and events. MRP has been demonstrated effective for jointly learning the vertex weights of a bipartite graph (Zhang et al., 2008; Ventura et al., 2013).

Given a text, we construct bipartite graph between the lexical chains and events, with an edge being constructed between a lexical chain and an event if the event contains a word in the lexical chain. Suppose that there are n events $\{e_1, \cdots, e_n\}$ and m lexical chains: $\{l_1, \cdots, l_m\}$ in the bipartite graph G_{bi}. Their scores are represented by $sal(e) = \{sal(e_1), \cdots, sal(e_n)\}$ and $sal(l) = \{sal(l_1), \cdots, sal(l_m)\}$, respectively. We compute the final $sal(e)$ and $sal(l)$ iteratively by MRP. At each step, $sal(e_i)$ and $sal(l_j)$ are computed as follows:

$$
\begin{aligned}
sal(e_i) &\propto \sum_{j=1}^{m} r_{ij} \times sal(l_j) \\
sal(l_j) &\propto \sum_{i=1}^{n} r_{ij} \times sal(e_i) \\
r_{ij} &= \frac{\sum_{(l_j, e_i) \in G_{bi}} w(l_j) \cdot w(e_i)}{A}
\end{aligned}
\tag{1}
$$

where $r_{ij} \in R$ denotes the cohesion between lexicon chain l_i and event e_j, A is a normalization factor, $sal(\cdot)$ denotes the salience, and the initial values of $sal(e)$ and $sal(t)$ can be assigned randomly.

The remaining problem is how to define the salience score of a given lexicon chain l_i and a given event e_j. In this work, we use the guidance of abstractive and extractive models to compute

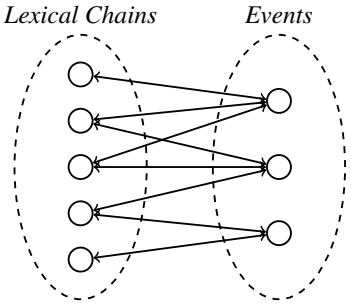

Figure 3: Bipartite graph where two vertex sets denote lexical chains and events, respectively.

$sal(l_j)$ and $sal(e_i)$, respectively, as shown below:

$$
\begin{aligned}
w(l_j) &= \sum_{w \in l_j} sal_{abs}(w) \\
w(e_i) &= \sum_{s \in Sen(e_i)} sal_{ext}(s)
\end{aligned}
\tag{2}
$$

where $sal_{abs}(\cdot)$ denotes the word salience score of an abstractive model, $sal_{ext}(\cdot)$ denotes the sentence salience score of an extractive model, and $Sen(e_i)$ denotes the sentence set where e_i is extracted from. We exploit our baseline sentence ranking method, `SentRank`, to obtain the sentence salience score, and use our baseline phrase ranking method, `PhraseRank`, to obtain the phrase salience score.

3.2 Headline Generation

We use a graph-based multi-sentence compression (MSC) model to generate the final title for the proposed event-driven model. The model is inspired by Filippova (2010). First, a weighted directed acyclic word graph is built, with a start node and an end node in the graph. A headline can be obtained by any path from the start node to the end node. We measure each candidate path by a scoring function. Based on the measurement, we exploit a beam-search algorithm to find the optimum path.

3.2.1 Word-Graph Construction

Given a set of candidate events CE, we extract all the sentences that contain the events. In particular, we add two artificial words, $\langle S \rangle$ and $\langle E \rangle$, to the start position and end position of all sentences, respectively. Following Filippova (2010), we extract all words in the sentences as graph vertexes, and then construct edges based on these words. Filippova (2010) adds edges

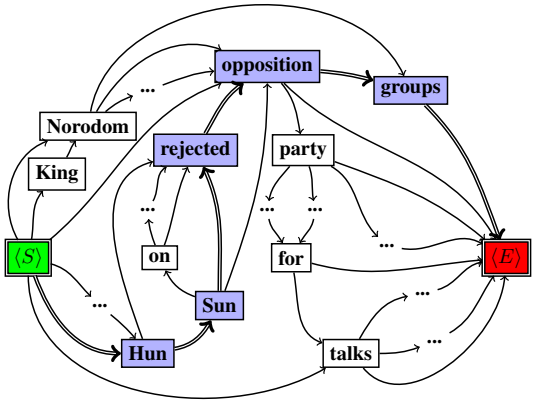

Figure 4: Word graph generated from candidates and a possible compression path.

for all the word pairs that are adjacent in one sentence. The title generated using this strategy can mistakenly contain common word bigrams(i.e. adjacent words) in different sentences. To address this, we change the strategy slightly, by adding edges for **a**ll word pairs of one sentence in the original order. In another words, if word w_j occurs after w_i in one sentence, then we add an edge $w_i \rightarrow w_j$ for the graph. Figure 4 gives an example of the word graph. The search space of the graph is larger compared with that of Filippova (2010) because of more added edges.

Different from Filippova (2010), salience information is introduced into the calculation of the weights of vertexes. One word that occurs in more salient candidate should have higher weight. Given a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{V_1, \cdots, V_n\}$ denotes the word nodes and $\mathcal{E} = \{E_{ij} \in \{0,1\}, i,j \in [1,n]\}$ denotes the edges. The vertex weight is computed as follows:

$$w(V_i) = \sum_{e \in CE} \mathit{sal}(e) \exp\{-\mathit{dist}(V_i.w, e)\} \quad (3)$$

where $\mathit{sal}(e)$ is the salience score of an event from the candidate extraction step, $V_i.w$ denotes the word of vertex V_i, and $\mathit{dist}(w, e)$ denotes the distance from the word w to the event e, which are defined by the minimum distance from w to all the related words of e in a sentence by the dependency path[5] between them. Intuitively, equation 3 demonstrates that a vertex is salient when its corresponding word is close to salient events. It is worth noting that the formula can adapt to extractive and abstractive models as well, by replacing events with sentences and phrases. We use them for the `SentRank` and `PhraseRank` baseline systems in Section 4.3, respectively.

The equation to compute the edge weight is adopted from Filippova (2010):

$$\begin{aligned} w'(E_{ij}) &= \sum_s \mathit{rdist}(V_i.w, V_j.w) \\ w(E_{ij}) &= \frac{w(V_i)w(V_j) \cdot w'(E_{ij})}{w(V_i) + w(V_j)} \end{aligned} \quad (4)$$

where $w'(E_{ij})$ refers to the sum of $\mathit{rdist}(V_i.w, V_j.w)$ over all sentences, and $rdist(\cdot)$ denotes the reciprocal distance of two words in a sentence by the dependency path. By the formula, an edge is salient when the corresponding vertex weights are large or the corresponding words are close.

3.2.2 Scoring Method

The key to our MSC model is the path scoring function. We measure a candidate path based on two aspects. Besides the sum edge score of the path, we exploit a trigram language model to compute a fluency score of the path. Language models have been commonly used to generate more readable titles.

The overall score of a path is compute by:

$$\begin{aligned} \mathit{score}(p) &= \mathit{edge}(p) + \lambda \times \mathit{flu}(p) \\ \mathit{edge}(p) &= \frac{\sum_{E_{ij} \in p} \ln\{w(E_{ij})\}}{n} \\ \mathit{flu}(p) &= \frac{\sum_i \ln\{p(w_i | w_{i-2} w_{i-1})\}}{n} \end{aligned} \quad (5)$$

where p is a candidate path and the corresponding word sequence of p is $w_1 \cdots w_n$. A trigram language model is trained using SRILM[6] on English Gigaword (LDC2011T07).

3.2.3 Beam Search

Beam search has been widely used aiming to find the sub optimum result (Collins and Roark, 2004; Zhang and Clark, 2011), when exact inference is extremely difficult. Assuming our word graph has a vertex size of n, the worst computation complexity is $O(n^4)$ when using a trigram language model, which is time consuming.

[5]The distance is $+\infty$ when e and w are not in one sentence.

[6]http://www.speech.sri.com/projects/srilm/

Input: $G \leftarrow (\mathcal{V}, \mathcal{E})$, *LM*, *B*
Output: *best*
candidates $\leftarrow$ { {$\langle S \rangle$} }
loop do
  beam $\leftarrow$ { }
  for each *candidate* **in** *candidates*
    if *candidate* **endwith** $\langle E \rangle$
      ADDTOBEAM(*beam*, *candidate*)
      continue
    for each V_i **in** $\mathcal{V}$
      candidate $\leftarrow$ ADDVERTEX(*candidate*, V_i)
      COMPUTESCORE(*candidate, LM*)
      ADDTOBEAM(*beam*, *candidate*)
    end for
  end for
  candidates $\leftarrow$ TOP-K(*beam, B*)
  if *candidates* **all endwith** $\langle E \rangle$: **break**
end loop
best $\leftarrow$ BEST(*candidates*)

Figure 5: The beam-search algorithm.

Using beam search, assuming the beam size is B, the time complexity decreases to $O(Bn^2)$.

Pseudo-code of our beam search algorithm is shown in Figure 5. During search, we use *candidates* to save a fixed size (B) of partial results. For each iteration, we generate a set of new candidates by adding one vertex from the graph, computing their scores, and maintaining the top B candidates for the next iteration. If one candidate reaches the end of the graph, we do not expand it, directly adding it into the new candidate set according to its current score. If all the candidates reach the end, the searching algorithm terminates and the result path is the candidate from *candidates* with the highest score.

4 Experiment

4.1 Settings

We use the standard HG test dataset to evaluate our model, which consists of 500 articles from DUC–04 task 1[7], where each article is provided with four reference headlines. In particular, we use the first 100 articles from DUC–07 as our development set. There are averaged 40 events per article in the two datasets. All the pre-processing steps, including POS tagging, lemma analysis, dependency parsing and anaphora resolution, are conducted using the Stanford NLP tools (Marneffe and Manning, 2008). The MRP iteration number is set to 10.

We use ROUGE (Lin, 2004) to automatically measure the model performance, which has been widely used in summarization tasks (Wang et al., 2013; Ng et al., 2014). We focus on Rouge1 and Rouge2 scores, following Xu et al. (2010). In addition, we conduct human evaluations, using the same method as Woodsend et al. (2010). Four participants are asked to rate the generated headlines by three criteria: informativeness (how much important information in the article does the headline describe?), fluency (is it fluent to read?) and coherence (does it capture the topic of article?). Each headline is given a subjective score from 0 to 5, with 0 being the worst and 5 being the best. The first 50 documents from the test set and their corresponding headlines are selected for human rating. We conduct significant tests using t-test.

4.2 Development Results

There are three important parameters in the proposed event-driven model, including the beam size B, the fluency weight λ and the number of candidate events N. We find the optimum parameters on development dataset in this section. For efficiency, the three parameters are optimized separately. The best performance is achieved with $B = 8$, $\lambda = 0.4$ and $N = 10$. We report the model results on the development dataset to study the influences of the three parameters, respectively, with the other two parameters being set with their best value.

4.2.1 Influence of Beam Size

We perform experiments with different beam widths. Figure 6 shows the results of the proposed model with beam sizes of 1, 2, 4, 8, 16, 32, 64. As can be seen, our model can achieve the best performances when the beam size is set to 8. Larger beam sizes do not bring better results.

4.2.2 Influence of Fluency Weight

The fluency score is used for generating readable titles, while the edge score is used for generating informative titles. The balance between them is important. By default, we set one to the weight of edge score, and find the best weight λ for the fluency score. We set λ ranging from 0 to 1 with and interval of 0.1, to investigate the influence of

[7] http://duc.nist.gov/duc2004/tasks.html

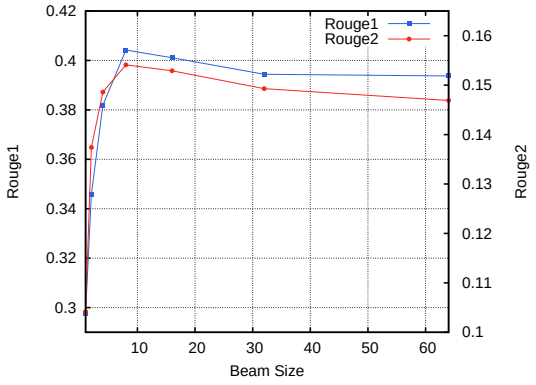

Figure 6: Results with different beam sizes.

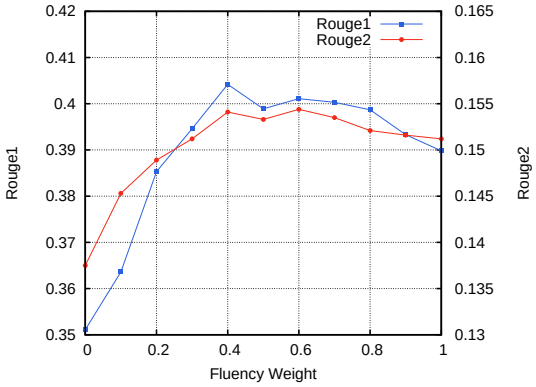

Figure 7: Results using different fluency weights.

this parameter[8]. Figure 7 shows the results. The best result is obtained when $\lambda = 0.4$.

4.2.3 Influence of Candidate Event Count

Ideally, all the sentences of an original text should be considered in multi-sentence compression. But an excess of sentences would bring more noise. We suppose that the number of candidate events N is important as well. To study its influence, we report the model results with different N, from 1 to 15 with an interval of 1. As shown in Figure 8, the performance increases significantly from 1 to 10, and no more gains when $N > 10$. The performance decreases drastically when M ranges from 12 to 15.

4.3 Final Results

Table 1 shows the final results on the test dataset. The performances of the proposed event-driven model are shown by `EventRank`. In addition, we use our graph-based MSC model to generate titles for `SentRank` and `PhraseRank`, respectively, as mentioned in Section 3.2.1. By comparison with the two models, we can examine the effectiveness of the event-driven model. As shown in Table 1, the event-driven model achieves the best scores on both Rouge1 and Rouge2, demonstrating events are more effective than sentences and phrases.

Further, we compare our proposed MSC method with the MSC proposed by Filippova (2010), to study the effectiveness of our novel MSC. We use `MSC`[10] and `SalMSC`[11] to

[8]Preliminary results show that λ is better below one.

[9]The mark $*$ denotes the results are inaccurate, which are guessed from the figures in the published paper.

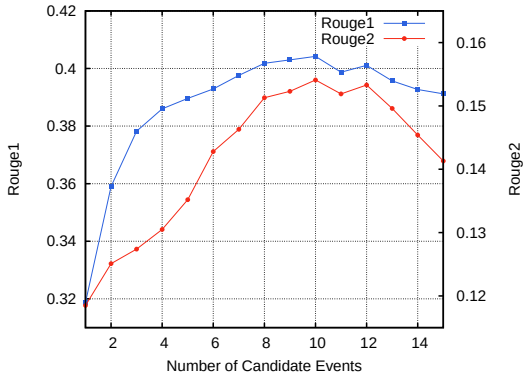

Figure 8: Results using different numbers of candidate events.

Method	Model Type	Rouge1	Rouge2
Our SalMSC			
SentRank	Extractive	0.3511	0.1375
PhraseRank	Abstractive	0.3706	0.1415
EventRank	Event-driven	**0.4247**‡	**0.1484**‡
Using MSC			
SentRank	Extractive	0.2773	0.0980
PhraseRank	Abstractive	0.3652	0.1299
EventRank	Event-driven	**0.3822**‡	**0.1380**‡
Other work			
SentRank+SSC	Extractive	0.2752	0.0855
Topiary	Abstractive	0.2835	0.0872
Woodsend	Abstractive	0.26*	0.06*[9]

Table 1: Performance comparison for automatic evaluation. The mark ‡ denotes that the result is significantly better with a p-value below 0.01.

[10]The MSC source code, published by Boudin and Morin (2013), is available at https://github.com/boudinfl/takahe.

[11]Our source code is available at https://github.com/dram218/WordGraphCompression.

Method	Info.	Infu.	Cohe.
SentRank	4.13	2.85	2.54
PhraseRank	4.21	3.25	2.62
EventRank	**4.35**‡	**3.41**‡	**3.22**‡

Table 2: Results from the manual evaluation. The mark ‡ denotes the result is significantly better with a p-value below 0.01.

SentRank, PhraseRank and EventRank to denote their MSC method and our proposed MSC, respectively, applying them, respectively. As shown in Table 1, better performance is achieved by our MSC, demonstrating the effectiveness of our proposed MSC. Similarly, the event-driven model can achieve the best results.

We report results of previous state-of-the-art systems as well. SentRank+SSC denotes the result of Erkan and Radev (2004), which uses our SentRank and SSC to obtain the final title. Topiary denotes the result of Zajic et al. (2005), which is an early abstractive model. Woodsend denotes the result of Woodsend et al. (2010), which is an abstractive model using a quasi-synchronous grammar to generate a title. As shown in Table 1, MSC is significantly better than SSC, and our event-driven model achieves the best performance, compared with state-of-the-art systems.

Following Alfonseca et al. (2013), we conduct human evaluation also. The results are shown in Table 2, by three aspects: informativeness, fluency and coherence. The overall tendency is similar to the results, and the event-driven model achieves the best results.

4.4 Example Outputs

We show several representative examples of the proposed event-driven model, in comparison with the extractive and abstractive models. The examples are shown in Table 3.

In the first example, the results of both SentRank and PhraseRank contain the redundant phrase "catastrophe Tuesday". The output of PhraseRank is less fluent compared with that of SentRank. The preposition "for" is not recovered by the headline generation system PhraseRank. In contrast, the output of EventRank is better, capturing the major event in the reference title.

Method	Generated Headlines
Reference	Honduras, other Caribbean countries brace for the wrath of Hurricane Mitch
SentRank	Honduras braced for potential catastrophe Tuesday as Hurricane Mitch roared through northwest Caribbean
PhraseRank	Honduras braced catastrophe Tuesday Hurricane Mitch roared northwest Caribbean
EventRank	Honduras braced for Hurricane Mitch roared through northwest Caribbean
Reference	At Ibero-American summit Castro protests arrest of Pinochet in London
SentRank	Castro disagreed with the arrest Augusto Pinochet calling international meddling
PhraseRank	Cuban President Fidel Castro disagreed arrest London Chilean dictator Augusto Pinochet
EventRank	Fidel Castro disagreed with arrest in London of Chilean dictator Augusto Pinochet
Reference	Cambodian leader Hun Sen rejects opposition demands for talks in Beijing
SentRank	Hun Sen accusing opposition parties of internationalize the political crisis
PhraseRank	opposition parties demands talks internationalize political crisis
EventRank	Cambodian leader Hun Sen rejected opposition parties demands for talks

Table 3: Comparison of headlines generated by the different methods.

In the second example, the outputs of three systems all lose the phrase "Ibero-American summit". SentRank gives different additional information compared with PhraseRank and EventRank. Overall, the three outputs can be regarded as comparable. PhraseRank also has a fluency problem by ignoring some function words.

In the third example, SentRank does not capture the information on "demands for talks". PhraseRank discards the preposition word "for". The output of EventRank is better, being both more fluent and more informative.

From the three examples, we can see that SentRank tends to generate more readable titles, but may lose some important information. PhraseRank tends to generate a title with more important words, but the fluency is relatively weak even with MSC. EventRank combines the advantages of both SentRank and PhraseRank, generating titles that contain more important events with complete structures. The observation verifies our hypothesis in the introduction — that extractive models have the problem of low information coverage, and

abstractive models have the problem of poor grammaticality. The event-driven mothod can alleviate both issues since event offer a trade-off between sentence and phrase.

5 Related Work

Our event-driven model is different from traditional extractive (Dorr et al., 2003; Erkan and Radev, 2004; Alfonseca et al., 2013) and abstractive models (Zajic et al., 2005; Soricut and Marcu, 2007; Woodsend et al., 2010; Xu et al., 2010) in that events are used as the basic processing units instead of sentences and phrases. As mentioned above, events are a trade-off between sentences and phrases, avoiding sparsity and structureless problems. In particular, our event-driven model can interact with sentences and phrases, thus is a light combination for two traditional models.

The event-driven model is mainly inspired by Alfonseca et al. (2013), who exploit events for multi-document headline generation. They leverage titles of sub-documents for supervised training. In contrast, we generate a title for a single document using an unsupervised model. We use novel approaches for event ranking and title generation.

In recent years, sentence compression (Galanis and Androutsopoulos, 2010; Yoshikawa and Iida, 2012; Wang et al., 2013; Li et al., 2014; Thadani, 2014) has received much attention. Some methods can be directly applied for multi-document summarization (Wang et al., 2013; Li et al., 2014). To our knowledge, few studies have been explored on applying them in headline generation.

Multi-sentence compression based on word graph was first proposed by Filippova (2010). Some subsequent work was presented recently. Boudin and Morin (2013) propose that the key phrase is helpful to sentence generation. The key phrases are extracted according to syntactic pattern and introduced to identify shortest path in their work. Mehdad et al. (2013; Mehdad et al. (2014) introduce the MSC based on word graph into meeting summarization. Tzouridis et al. (2014) cast multi-sentence compression as a structured predication problem. They use a large-margin approach to adapt parameterised edge weights to the data in order to acquire the shortest path. In their work, the sentences introduced to a word graph are treated equally, and the edges in the graph are constructed according to the adjacent order in original sentence.

Our MSC model is also inspired by Filippova (2010). Our approach is more aggressive than their approach, generating compressions with arbitrary length by using a different edge construction strategy. In addition, our search algorithm is also different from theirs. Our graph-based MSC model is also similar in spirit to sentence fusion, which has been used for multi-document summarization (Barzilay and McKeown, 2005; Elsner and Santhanam, 2011).

6 Conclusion and Future Work

We proposed an event-driven model headline generation, introducing a graph-based MSC model to generate the final title, based on a set of events. Our event-driven model can incorporate sentence and phrase salience, which has been used in extractive and abstractive HG models. The proposed graph-based MSC model is not limited to our event-driven model. It can be applied on extractive and abstractive models as well. Experimental results on DUC–04 demonstrate that event-driven model can achieve better results than extractive and abstractive models, and the proposed graph-based MSC model can bring improved performances compared with previous MSC techniques. Our final event-driven model obtains the best result on this dataset.

For future work, we plan to explore two directions. Firstly, we plan to introduce event relations to learning event salience. In addition, we plan to investigate other methods about multi-sentence compression and sentence fusion, such as supervised methods.

Acknowledgments

We thank all reviewers for their detailed comments. This work is supported by the State Key Program of National Natural Science Foundation of China (Grant No.61133012), the National Natural Science Foundation of China (Grant No.61373108, 61373056), the National Philosophy Social Science Major Bidding Project of China (Grant No.11&ZD189), and the Key Program of Natural Science Foundation of Hubei, China (Grant No.2012FFA088). The corresponding authors of this paper are Meishan Zhang and Donghong Ji.

References

Enrique Alfonseca, Daniele Pighin and Guillermo Garrido. 2013. HEADY: News headline abstraction through event pattern clustering. In *Proceedings of ACL 2013*,pages 1243–1253.

Regina Barzilay and Michael Elhadad. 1997. Using Lexical Chains for Text Summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop(ISTS'97), Madrid*.

Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3), pages 297–328.

Florian Boudin and Emmanuel Morin. 2013. Keyphrase Extraction for N-best Reranking in Multi-Sentence Compression. In *Proccedings of the NAACL HLT 2013 conference*, page 298–305.

James Clarke and Mirella Lapata. 2010. Discourse Constraints for Document Compression. *Computational Linguistics*, 36(3), pages 411–441.

Michael Collins and Brian Roark. 2004. Incremental Parsing with the Perceptron Algorithm. In *Proceedings of ACL 2004*, pages 111-118.

Corston-Oliver, Simon. 2001. Text compaction for display on very small screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*, Pittsburg, PA, 3 June 2001, pages 89–98.

Xiao Ding, Yue Zhang, Ting Liu, Junwen Duan. 2014. Using Structured Events to Predict Stock Price Movement : An Empirical Investigation. In *Proceedings of EMNLP 2014*, pages 1415–1425.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *proceedings of the HLT–NAACL 03 on Text summarization workshop*, volume 5, pages 1–8.

Micha Elsner and Deepak Santhanam. 2011. Learning to fuse disparate sentences. In *Proceedings of ACL 2011*, pages 54–63.

Nicolai Erbs, Iryna Gurevych and Torsten Zesch. 2013. Hierarchy Identification for Automatically Generating Table-of-Contents. In *Proceedings of Recent Advances in Natural Language Processing*, Hissar, Bulgaria, pages 252–260.

Gunes Erkan and Dragomir R Radev. 2004. LexRank : Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research 22*, 2004, pages 457–479.

Fader A, Soderland S, Etzioni O. 2011. Identifying relations for open information extraction. In *Proceedings of EMNLP 2011*, pages 1535–1545.

Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of Coling 2010*, pages 322–330.

Dimitrios Galanis and Ion Androutsopoulos. 2010. An extractive supervised two-stage method for sentence compression. In *Proceedings of NAACL 2010*, pages 885–893.

Barbara J. Grosz and Scott Weinstein and Aravind K. Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, volume 21, pages 203–225.

Zhichao Hu, Elahe Rahimtoroghi, Larissa Munishkina, Reid Swanson and Marilyn A.Walker. 2013. Unsupervised Induction of Contingent Event Pairs from Film Scenes. In *Proceedings of EMNLP 2013*, pages 369–379.

Chen Li,Yang Liu, Fei Liu, Lin Zhao, Fuliang Weng. 2014. Improving Multi-documents Summarization by Sentence Compression based on Expanded Constituent Parse Trees. In *Proceedings of EMNLP 2014*, pages 691–701.

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

Andre F.T. Martins and Noah A. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 1–9.

Yashar Mehdad, Giuseppe Carenini, Frank W.Tompa and Raymond T.Ng. 2013. Abstractive Meeting Summarization with Entailment and Fusion. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 136–146.

Yashar Mehdad, Giuseppe Carenini and Raymond T.Ng. 2014. Abstractive Summarization of Spoken and Written Conversations Based on Phrasal Queries. In *Proceedings of ACL 2014*, pages 1220–1230.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1), pages 21–48.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The stanford typed dependencies representation. In *COLING 2008 Workshop on Cross-framework and Cross-domain Parser Evaluation*.

Jun-Ping Ng, Yan Chen, Min-Yen Kan, Zhoujun Li. 2014. Exploiting Timelines to Enhance Multi-document Summarization. *Proceedings of ACL 2014*, pages 923–933.

Likun Qiu and Yue Zhang. 2014. ZORE: A Syntax-based System for Chinese Open Relation Extraction. *Proceedings of EMNLP 2014*, pages 1870–1880.

Robert G. Sargent. 1988. Polynomial Time Joint Structural Inference for Sentence Compression. *Management Science*, 34(10), pages 1231–1251.

Schwartz R. 1988. Unsupervised topic discovery. In *Proceedings of workshop on language modeling and information retrieval*, pages 72–77.

R. Soricut, and D. Marcu. 2007. Abstractive headline generation using WIDL-expressions. *Information Processing and Management*, 43(6), pages 1536–1548.

Kapil Thadani. 2014. Approximation Strategies for Multi-Structure Sentence Compression. *Proceedings of ACL 2014*, pages 1241–1251.

Emmanouil Tzouridis, Jamal Abdul Nasir and Ulf Brefeld. 2014. Learning to Summarise Related Sentences. *Proceedings of COLING 2014*,Dublin, Ireland, August 23-29 2014. pages 1636–1647.

Carles Ventura, Xavier Giro-i-Nieto, Veronica Vilaplana, Daniel Giribet, and Eusebio Carasusan. 2013. Automatic keyframe selection based on Mutual Reinforcement Algorithm. In *Proceedings of 11th international workshop on content-based multimedia indexing(CBMI)*, pages 29–34.

Stephen Wan and Kathleen McKeown. 2004. Generating overview summaries of ongoing email thread discussions. In *Proceedings of COLING 2004*, Geneva, Switzerland, 2004, pages 1384–1394.

Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, Claire Cardie. 2013. A sentence compression based framework to query-focused mutli-document summarization. In *Proceedings of ACL 2013*, Sofia, Bulgaria, August 4-9 2013, pages 1384–1394.

Kristian Woodsend, Yansong Feng and Mirella Lapata. 2010. Title generation with quasi-synchronous grammar. In *Proceedings of EMNLP 2010*, pages 513–523.

Songhua Xu, Shaohui Yang and Francis C.M. Lau. 2010. Keyword extraction and headline generation using novel work features. In *Proceedings of AAAI 2010*, pages 1461–1466.

Katsumasa Yoshikawa and Ryu Iida. 2012. Sentence Compression with Semantic Role Constraints. In *Proceedings of ACL 2012*, pages 349–353.

David Zajic, Bonnie Dorr and Richard Schwartz. 2005. Headline generation for written and broadcast news. *lamp-tr-120, cs-tr-4698*.

Hongyuan Zha. 2002. Generic summarization and keyphrase extraction using mutual reinforement principle and sentence clustering. In *Proceedings of SIGIR 2002*, pages 113–120.

Qi Zhang, Xipeng Qiu, Xuanjing Huang, Wu Lide. 2008. Learning semantic lexicons using graph mutual reinforcement based bootstrapping. *Acta Automatica Sinica*, 34(10), pages 1257–1261.

Yue Zhang, Stephen Clark. 2011. Syntactic Processing Using the Generalized Perceptron and Beam Search. *Computational Linguistics*, 37(1), pages 105–150.

Yue Zhang. 2013. Partial-Tree Linearization: Generalized Word Ordering for Text Synthesis. In *Proceedings of IJCAI 2013*, pages 2232–2238.