Revealing Semantic Structures of Texts: Multi-grained Framework for Automatic Mind-map Generation

Yang Wei[1] , **Honglei Guo**[2*] , **Jinmao Wei**[1] and **Zhong Su**[2]

[1]College of Computer Science, Nankai University

[2]IBM Research - China

weiyang_tj@outlook.com, guohl@cn.ibm.com, weijm@nankai.edu.cn, suzhong@cn.ibm.com

Abstract

A mind-map is a diagram used to represent ideas linked to and arranged around a central concept. It's easier to visually access the knowledge and ideas by converting a text to a mind-map. However, highlighting the semantic skeleton of an article remains a challenge. The key issue is to detect the relations amongst concepts beyond intra-sentence. In this paper, we propose a multi-grained framework for automatic mind-map generation. That is, a novel neural network is taken to detect the relations at first, which employs multi-hop self-attention and gated recurrence network to reveal the directed semantic relations via sentences. A recursive algorithm is then designed to select the most salient sentences to constitute the hierarchy. The human-like mind-map is automatically constructed with the key phrases in the salient sentences. Promising results have been achieved on the comparison with manual mind-maps. The case studies demonstrate that the generated mind-maps reveal the underlying semantic structures of the articles.

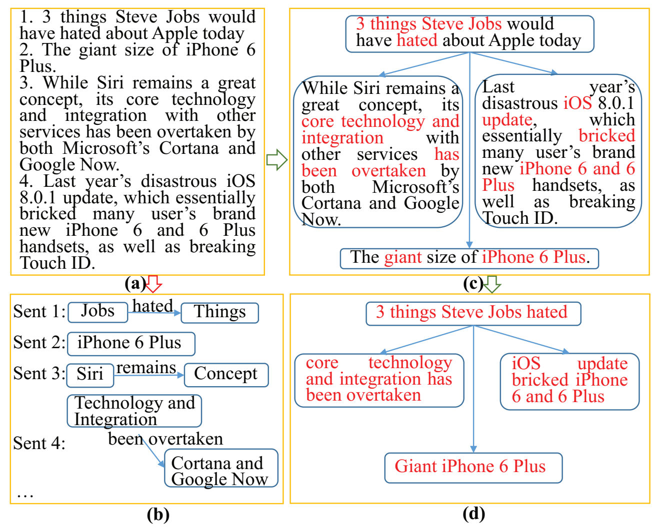

Figure 1: Examples of mind-map. Given the original text (a), generate parser-based mind-map (b), salient-sentence-based one (c) and key-snippet-based one (d).

1 Introduction

A mind-map is widely known as a hierarchical map in which the major ideas are directly connected to the central concept, and other ideas branch out from those [Kudelić *et al.*, 2011]. Research has shown that cognitive structures of knowledge are better in learning with mind-maps than traditional way of plain text [Dhindsa *et al.*, 2011]. Mind-maps highlight key concepts by putting them in upper nodes, and are therefore used as an aid to studying and organizing information, making decisions and writing [Willis and Miertschin, 2006].

Many editors, such as FreeMind, MindMapper, Visual Mind, etc., are developed to help people make mind-map manually [Kudelić *et al.*, 2011]. Semi-automatic systems find and suggest elements of a map, and a person has to do the rest by hand [Zubrinic *et al.*, 2012]. To entirely release the manual efforts on reading, comprehension, and writing, some automatic methods [Brucks and Schommer, 2008; Rothenberger *et al.*, 2008; Zubrinic *et al.*, 2012; Elhoseiny and Elgammal, 2013; Falke and Gurevych, 2017] are proposed. A typical pipeline for automatic mind-map generation includes morphological analysis, syntactic parsing, discourse analysis, and co-reference resolution. These methods usually focus on intra-sentence relations and connect the subgraphs of sentences as a whole by co-reference resolution. Such a simple connection is not sufficient in the mind-map generation. As shown in Fig. 1(b), the parsed diagrams of sentences share no concepts. It remains a challenge to detect the association cross sentences. As shown in Fig. 1(d), the node "3 things Steve Jobs hate" is supported by its three successors, where the relationship between the nodes is the part-of relation. For the sentence pair "New iPhone is released" and "Apple's share increases", the connection is the cause-effect relation. The commonality between the above various relations is that both the precursors are the original ideas the authors talk about. A precursor is generally the origin of its successor. Verifying the origin is a cognitive problem which could be learned case-by-case.

Since a whole sentence contains richer semantic information than its scattered internal snippets, we try to capture the

*Contact Author.

deep semantic association cross sentences at first. We present a multi-grained framework for mind-map generation: (1) Detect the pairwise semantic linkages among sentences; (2) Generate the hierarchical salient-sentence-based mind-map (SSM, see Fig.1(c)); (3) Compress SSM to the key-snippet-based mind-map (KSM, see Fig.1(d)).

In the salient-sentence-based mind-map, a sentence originates from some semantic perspectives of its precursor and expands with more specific and various ideas. In the training, to learn the semantic association evidence for connecting two given sentences, we employ multi-hop self-attention to encode the latent semantic perspectives mentioned in the precursor sentence. We take the gated recurrence module to dig up the evidence from the successor sentence with attention to the encoded perspectives of its precursor. The hierarchical salient-sentence-based mind-map is obtained by recursively identifying the most salient sentence as the ancestor of the other sentences. The key-snippet-based one is generated by extracting key phrases from the salient sentence nodes.

The key contributions of this paper include:

1. we construct a framework for multi-grained mind-map automatic generation. Experimental results demonstrate the efficiency of the relation-driven framework;

2. we propose a novel neural network model for relation detection of mind-maps, which employs multi-hop self-attention and gated recurrence network to detect semantic association evidence among the sentences;

3. two kinds of mind-maps are provided: the mind-map with salient sentences is easy to read; the mind-map with key phrases highlights the critical points of an article.

2 Related Work

A mind-map is easily confused with other concept maps. One prominent perspective comes from the formal reasoning. So-Called conceptual graphs are interchangeable with predicate calculus [Olney *et al.*, 2011]. The level of granularity given to nodes and relationships is very small, which turns out to be a relevant differentiator with mind-map. Another prominent perspective comes from the psychology literature [Graesser and Clark, 1985] with some emphasis on modeling question asking and answering. In this formulation of conceptual graphs, nodes themselves can be propositions, and relations are generally limited to a generic set of propositions for a given domain. While in a mind-map, nodes and the edges linking them are not restricted [Novak and Canas, 2006]. Like the tree of phrases in our KSM, the hierarchical topic models [Zhang *et al.*, 2010] organize words into a hierarchical tree. Such a hierarchy is the visualization of a corpus, while a mind-map is the visualization of an article.

Natural language sentence matching (NLSM) is the task of comparing two sentences and identifying the relationship between them. It is a fundamental technology for a variety of tasks, e.g., paraphrase identification task [Yin *et al.*, 2016], natural language inference [Lin *et al.*, 2017], question answering [Kumar *et al.*, 2016] and information retrieval [Wang *et al.*, 2016]. We employ the hot-spot structure comprising of embedding, encoding, interaction, aggregation, and prediction layers [Wang *et al.*, 2017; Gong *et al.*, 2018] as our relation detector. We further utilize the multi-hop attention [Lin *et al.*, 2017] to extract multi-perspective information, and modify the episodic memory module as our gated recurrence network [Kumar *et al.*, 2016] to explore implicitly relevant information of sequential data judging by a reference.

The design of the salient-sentence-based mind-map generator borrows the idea from some extractive summarization methods [Erkan and Radev, 2011; Tavallali *et al.*, 2015]. To find the most salient sentences in an article, these methods evaluate the salience scores according to the pairwise relationship between sentences, where the relationship is restricted to the similarities between sentences. A graph of sentences could also be derived with these methods. However, such a graph cannot be converted into a mind-map. As shown in Fig. 1(a), the similarities between the four sentences would be zero. This will be further confirmed by the experiment with LexRank [Erkan and Radev, 2011].

3 Multi-grained Framework for Mind-map Generation

3.1 Problem Definition

In this paper, we focus on mining mind-maps of news articles. The task of mind-map generation can be defined as:

$$A \to \mathbf{M}(\mathbf{C}, \mathbf{E}), \tag{1}$$

where A denotes the input news article; $\mathbf{M}$ is the mind-map graph of A; $\mathbf{C}$ represents the collection of nodes in $\mathbf{M}$; $\mathbf{E}$ denotes the collection of edges in $\mathbf{M}$.

Generally, the nodes in $\mathbf{C}$ are semantic nodes, which are text snippets, such as words, phrases, sentences, etc. Edges are directed. $\forall c_i, c_j \in \mathbf{C}$, if c_j is implied by c_i, then $\exists e(c_i, c_j) \in \mathbf{E}$. We name the precursor c_i as the semantic governor, the successor c_j as the semantic governed, and $e(c_i, c_j)$ as the *governing relationship*.

Hierarchical mind-map is easy to read [Buzan, 2006], where major ideas are connected directly to the central concept, and other ideas branch out from those. More precisely, there is at most one precursor per node; no node is isolated from the others.

Definition 1 $\mathbf{M}(\mathbf{C}, \mathbf{E})$ *is the mind-map of A if it satisfies the following restrictions:*

1. $\forall c_i, c_j, c_k \in \mathbf{C}$, *if* $e(c_j, c_i) \in \mathbf{E}$ *and* $e(c_k, c_i) \in \mathbf{E}$ *then* $c_j = c_k$;
2. $\forall \mathbf{M}_1, \mathbf{M}_2 \subseteq \mathbf{M}, \exists c_i \in \mathbf{C}_1, c_j \in \mathbf{C}_2, e(c_i, c_j) \in \mathbf{E}$.

3.2 Architecture

The proposed multi-grained mind-map generation framework is shown in Fig. 2. The framework consists of Governing Relationship Detector (GRD), Salient-Sentence-Based Mind-map (SSM) generator and Key-Snippet-based Mind-map (KSM) generator. Given a raw text as input, GRD detects inter-sentence governing relations, which computes the probability $P(e(s_i, s_j))$ that sentence s_i is the governor of s_j. The Governing Matrix $\mathbf{G}_{m\times m}$ is output, where $\mathbf{G}(i, j) = P(e(s_i, s_j))$, m is the number of sentences in the text. With $\mathbf{G}$, SSM generator arranges the sentences to a hierarchical mind-map. KSM generator prunes the salient-sentence-based mind-map to the key-snippet-based mind-map.

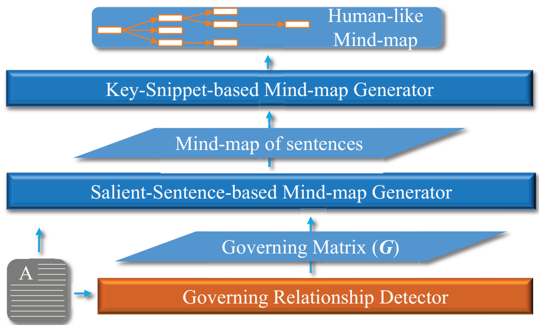

Figure 2: Architecture of the multi-grained framework.

4 Multi-perspective Recurrence Detector Learning

In this section, we present a Mulit-perspective Recurrence Detector (MRD) for the governing relationship detection. This section first gives an overview of MRD. Then the core layers of MRD are illustrated in detail. Finally, we show how MRD is learned from CNN news corpus.

4.1 Model Structure

We consider governing relationship detection as a classification task, which estimates a conditional probability $P(y|s_i, s_j)$ based on the training set, and predicts the relationship $y^* = argmax_{y \in (0,1)} P(y|s_i, s_j)$. Here $y = 1$ means that s_i is the governor of s_j, namely $P(1|s_i, s_j) = P(e(s_i, s_j))$. A high-level illustration of MRD is shown in Fig. 3, which is composed of the five layers below.

Embedding layer. This layer represents each word in the input sentences s_i and s_j with a d-dimensional vector. We construct the vector with the word embedding of word2vec [Mikolov *et al.*, 2013b]. The output of this layer is two sequences of word vectors $[\mathbf{w}^i_1, \ldots, \mathbf{w}^i_{|s_i|}]$ and $[\mathbf{w}^j_1, \ldots, \mathbf{w}^j_{|s_j|}]$, where $|s|$ is the number of words in s.

Encoding layer. It is one of the core layers in our model. It encodes the representations of sentences by incorporating contextual information of word embedding sequences. As shown in Fig. 3, the latent semantic perspectives of the governor s_i are computed as multiple weighted sums of hidden states from a BiLSTM. The contextual embeddings of the governed s_j are the hidden states from another BiLSTM, which are considered as the semantic units of s_j with no information loss.

Interaction layer. It is another core layer. It compares the representations of two given sentences and outputs the factor matrix of relation. As shown in Fig. 3, the gated recurrence module chooses which parts of the contextual embeddings to focus on through attention to each of the perspectives covered by s_i. Each time-step updates the module with the newly relevant information of the input contextual embedding. The evidence to verify the relationship is the final state of the module.

Aggregation layer. This layer combines the matching results into a fixed-length matching vector. We do this by concatenating the rows in the factor matrix.

Prediction layer. It evaluates the probability distribution $P(y|s_i, s_j)$. We employ a linear layer to classify the matching vector to two classes and apply the $softmax$ function in the layer.

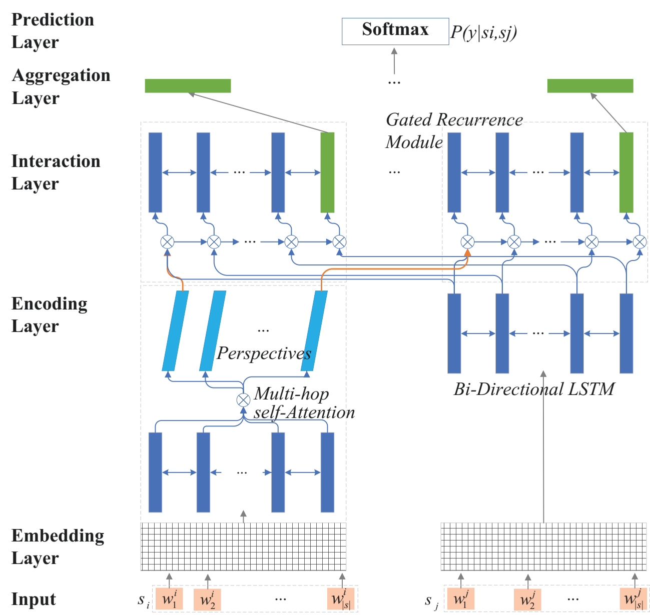

Figure 3: Network of Multi-perspective recurrence detector.

4.2 Multi-perspective Recurrence Detection

We employ the multi-hop self-attention mechanism and gated recurrence module to detect the semantic association evidence, which are instantiated in the encoding and interaction layers.

Encoding Layer

In this layer, we first utilize a BiLSTM [Wang *et al.*, 2017] to encode the contextual embeddings for each time-step of the governed sentence s_j. That is,

$$\mathbf{h}^j_k = [\overrightarrow{\mathbf{h}}_k; \overleftarrow{\mathbf{h}}_{|s|-k+1}], \tag{2}$$

where $\overrightarrow{\mathbf{h}}_k = lstm(\overrightarrow{\mathbf{h}}_{k-1}, \mathbf{w}_k)$; $\overleftarrow{\mathbf{h}}_k = lstm(\overleftarrow{\mathbf{h}}_{k+1}, \mathbf{w}_k)$.

Then we perform multiple hops of self-attention to retrieve the latent perspectives of the governor s_i [Lin *et al.*, 2017]. Say we want r different parts to be extracted from the sentence, the self-attention annotation matrix is calculated as $\mathbf{A} = softmax(\mathbf{W}_2 tanh(\mathbf{W}_1 \mathbf{H}_i^T))$. Here $\mathbf{H}_i = (\mathbf{h}^i_1, \ldots, \mathbf{h}^i_{|s|})_{|s| \times 2u}$, it is the hidden states of s_i computed with another BiLSTM. $\mathbf{W}_1$ is a weight matrix with a shape of $d_a \times 2u$, $\mathbf{W}_2$ is a $r \times d_a$ weight matrix, d_a is a hyperparameter we can set arbitrarily. Since $\mathbf{H}_i$ is sized $|s| \times 2u$, the annotation matrix will have a size $r \times |s|$. We compute the r weighted sums by multiplying the annotation matrix $\mathbf{A}$ and hidden states $\mathbf{H}_i$, the resulting matrix is the sentence embedding with a shape of $r \times 2u$:

$$\mathbf{S}_i = \mathbf{A}\mathbf{H}_i. \tag{3}$$

Interaction Layer

In this layer, we extract the evidence to verify the relationship of input sentences. The factor matrix of relation is obtained from the governed sentence s_j by referring to all the perspectives of the governor s_i.

$$\mathbf{F}_{rel} = (\mathbf{f}_1, \ldots, \mathbf{f}_r), \tag{4}$$

where $\mathbf{f}_t(t = 1, \ldots, r)$ is the evidence with respect to the t-th perspective of s_i. We take a modified LSTM over the sequence of contextual embeddings of s_j, weighted by the gates, which attends over some perspective of governor to compute $\mathbf{f}_t$. That is,

$$z(\mathbf{h}_k^j, \mathbf{f}_{k-1}, \mathbf{S}_t^i) = [\mathbf{h}_k^j; \mathbf{f}_{k-1}; \mathbf{S}_t^i; \mathbf{h}_k^j \cdot \mathbf{f}_{k-1}; \mathbf{h}_k^j \cdot \mathbf{S}_t^i; |\mathbf{h}_k^j - \mathbf{f}_{k-1}|; |\mathbf{h}_k^j - \mathbf{S}_t^i|; (\mathbf{h}_k^j)^T \mathbf{W}_5 \mathbf{f}_{k-1}; (\mathbf{h}_k^j)^T \mathbf{W}_5 \mathbf{S}_t^i], \tag{5}$$

$$g_k^t = \sigma(\mathbf{W}_4 tanh(\mathbf{W}_3 z(\mathbf{h}_k^j, \mathbf{f}_{k-1}, \mathbf{S}_t^i) + b_1) + b_2), \tag{6}$$

$$\mathbf{f}_k^t = g_k^t lstm(\mathbf{h}_k^j, \mathbf{f}_{k-1}^t) + (1 - g_k^t)\mathbf{f}_{k-1}^t, \tag{7}$$

$$\mathbf{f}_t = [\overrightarrow{\mathbf{f}_{|s_j|}^t}; \overleftarrow{\mathbf{f}_1^t}]. \tag{8}$$

z captures a variety of similarities between $\mathbf{h}_k^j$ (the input contextual embedding of the k-th word in the governed sentence s_j), $\mathbf{f}_{k-1}$ (the previous memory vector) and $\mathbf{S}_t^i$ (the t-th perspective of s_i) [Kumar *et al.*, 2016], the operation $\cdot$ is the element-wise product; g_k^t is the gate for the k-th time-step of s_j referring to the t-th perspective embedding of s_i, it also takes as input $\mathbf{h}_k^j$, $\mathbf{f}_{k-1}$, and $\mathbf{S}_t^i$; $\mathbf{f}_k^t$ is the evidence updated with the information of the k-th word in the governed sentence referring to the t-th perspective of the governor.

4.3 Training

We construct the training corpus from CNN news articles [Cheng and Lapata, 2016; Hermann *et al.*, 2015]. A CNN news consists of title, highlights, and content. Since the highlights summarize the critical aspects of the news, they could be regarded as the governors of the sentences in the corresponding paragraphs. The title could also be regarded as the governor of all the highlights. We sampled 20 docs to validate the assumption with human experts. They verified the assumption and found that a highlight and the matched sentence often share some key phrases. So we used a TFIDF matching strategy to find the matching pairs. A highlight is assigned to govern a paragraph due to its similarity to one or some sentences in the paragraph. The negative samples are generated randomly. With this method, we build a real-world, large-scale training corpus of 2,213,212 governing pairs from 90k articles, which saved much manual effort. The training learns to assign high probabilities to the sentences expressing the idea of a highlight, not just the similar ones with some matched words.

We split the corpus into the training, develop, and test set with 80%, 10%, and 10% portions, respectively. We initialize the size of word embedding as 300, the hidden size as 200 for all BiLSTM dimension layers. We set the batch size as 64 and the dropout as 0.5. We apply ADADELTA [Zeiler, 2012], an adaptive learning rate optimizer, to tune the parameters.

5 Multi-Grained Mind-map Generation

In this section, we describe the salient-sentence-based mind-map (SSM) generator and the key-snippet-based mind-map (KSM) generator in detail. SSM generator produces the hierarchical mind-map with the salient sentences. KSM generator extracts key phrases from the output of the SSM generator to form a human-like mind-map.

5.1 Salient-Sentence-based Mind-map Generator

SSM generator builds the final sentence-level mind-map from the bi-directional graph generated by MRD, in which $\mathbf{G}(i, j) \neq \mathbf{G}(j, i)$. To convert the graph to SSM, a simple strategy is to prune the extra edges. For example, remove the edge $e(s_i, s_j)$ if $\mathbf{G}(i, j) < \mathbf{G}(j, i)$, and discard the weakest edge when there is a circle. However, it is lack of confidence to infer the governing relations between sentences through the single values in $\mathbf{G}$ independently. A governor in the hierarchy is chosen depending on its governing probabilities to all of its successors. Thus we propose a recursive algorithm to determine the governing relationship by the overall properties of sentences (see Algorithm 1). The main idea is below.

At the very beginning, we make the root be null (see Step 17). The salience score is computed as the aggregation value of the probabilities to govern the other sentences (see Step 3, which accumulates $\mathbf{G}$ along the rows with selected columns corresponding the sentences in $\mathbf{B}$). In each iteration, we take the most salient one with the highest salience score ($>=$ a predefined confidence threshold T) as the governor (see Steps 4-7). If the selected governor is the leaf node, the governor needn't be verified (see condition $k = 1$ in Step 5). Then we remove the governor from the set of sentences $\mathbf{B}$, and take the governor as the root for the rest (see Steps 8 and 9). By clustering the left nodes into two groups with k-means algorithm (see Step 11), we can find the root concepts for each sub-group recursively. When the number of sentences in a sub-group is less than two, the iteration stops (see Step 10).

As Algorithm 1 divides sentences into two groups per iteration, there would be $log_2(m)$ iterations, where m is the number of sentences. The threshold T determines the depth of a mind-map. If $T = 0$, at least one sentence in each iteration is regarded as the root node. If the value of T increases, the root node for a sub-group may be regarded as fake root, the depth of the mind-map decreases.

5.2 Key-Snippet-based Mind-map Generator

KSM generator extracts key phrases from the sentences in the generated SSM and replaces the sentences with these phrases. We use the algorithm proposed by Rose [Rose *et al.*, 2010] as the extractor, which extracts the key phrases and calculates their importance scores in each sentence. The key phrase with the highest score is selected as a replacement of the corresponding sentence. If no key phrase is found, the whole sentence is kept in the mind-map. A global keyword set is maintained to avoid duplicate phrases in the mind-map. When the selected phrase already appears in the current mind-map, the phrase with the second highest score will be selected.

Algorithm 1 Salient-Sentence-based Mind-map Generator

Require: the set of sentences $\mathbf{B}_s$
Ensure: the nodes $\mathbf{C}_s$, and the edges $\mathbf{E}_s$
function RECURSIVEFUNCTION($\mathbf{B}, \mathbf{C}, \mathbf{E}, root$)
  $k \leftarrow length(\mathbf{B})$
  $\mathbf{g} \leftarrow$ salientVector($\mathbf{G}$)
  $governor \leftarrow s_i, s.t., \mathbf{g}(i) = \max_{s_j \in \mathbf{B}-\mathbf{C}}\{\mathbf{g}(j)\}$
  if $k > 0$ and ($k = 1$ or $\mathbf{g}(i)/k > T$) **then**
    $\mathbf{C} \leftarrow \mathbf{C} \cup \{governor\}$
    $\mathbf{E} \leftarrow \mathbf{E} \cup \{(root, governor)\}$
    $\mathbf{B}' \leftarrow \mathbf{B} - \{\mathbf{s}_i\}$
    $root \leftarrow governor$
  if $k <= 1$ **then return**
  $\mathbf{B}_1, \mathbf{B}_2 \leftarrow$ clustering($\mathbf{B}', 2$)
  recursiveFunction($\mathbf{B}_1, \mathbf{C}, \mathbf{E}, root$)
  recursiveFunction($\mathbf{B}_2, \mathbf{C}, \mathbf{E}, root$)
function MIND-MAPGENERATOR($\mathbf{B}_s$)
  $\mathbf{G} \leftarrow$ GRD($\mathbf{B}_s$)
  $\mathbf{C}_s \leftarrow \varnothing, \mathbf{E}_s \leftarrow \varnothing$
  recursiveFunction($\mathbf{B}_s, \mathbf{C}_s, \mathbf{E}_s, Null$)
  return $\mathbf{C}_s, \mathbf{E}_s$

Algorithm 2 Similarity Evaluation for Mind-maps

Require: the edges of the manual mind-map $\mathbf{E}_1$, the edges of the generated mind-map $\mathbf{E}_2$
Ensure: the similarity score r, the most similar pair msp
1: **function** SIM($\mathbf{E}_1, \mathbf{E}_2$)
2:   $sim \leftarrow 0$
3:   truncate($\mathbf{E}_2$), $s.t.|\mathbf{E}_2| = |\mathbf{E}_1|$
4:   **for** $e(s_i, s_j)$ **in** $\mathbf{E}_1$ **do**
5:     $msp \leftarrow (None, None)$
6:     **for** $e(s_x, s_y)$ **in** $\mathbf{E}_2$ **do**
7:       **if** $r(s_i, msp[0]) + r(s_j, msp[1])$
8:         $< r(s_i, s_x) + r(s_j, s_y)$ **then**
9:         $msp \leftarrow (s_x, s_y)$
10:     $sim \leftarrow \frac{(r(s_i, msp[0]) + r(s_j, msp[1]))}{2} + sim$
11:     $\mathbf{E}_2 \leftarrow \mathbf{E}_2 - e(msp[0], msp[1])$
  return $sim/|\mathbf{E}_1|$

6 Experiments

This section describes the experimental setting, and analyzes the experimental results and provides the case study.

6.1 Experimental Setup

Experimental Methods

In the experiments, we evaluate the proposed Multi-grained Framework equipped with the Multi-perspective Recurrence Detector (MRDMF). We compare it with the traditional parsing-based algorithm, similarity-based method, and the various neural network based methods.

ParserA [Rothenberger *et al.*, 2008] parses each sentence into the tuples of subject, verb and object. As shown in Fig. 1(b), a subject and its object are regarded as the governor and governed nodes, respectively, and the edge is the verb.

ParserB [Brucks and Schommer, 2008] is an extension of ParserA, which uses pronoun resolution technique to recover the missed relations among nodes incurred by the cause that the same subject/object in an article is referred to by different words. To enrich the information of each node, the adjacent adjective(s) of both subjects and objects are also extracted to constitute the nodes together with their subjects and objects.

LexRank calculates the governing probabilities between the sentences by the cosine similarity of their TFIDF vectors. It follows the well-known LexRank algorithm in document summarization domain [Erkan and Radev, 2011] that the PageRank algorithm is used to select the salient sentences. Then the mind-maps are generated with the SSM generator and KSM generator.

Dual LSTM Mind-map Generation (DLMG) is a simplified version of MRDMF. It simplifies the encoding and interaction layers. In the encoding layer, DLMG detects the governing relationship without considering the multi-perspectives covered by the sentences. It utilizes BiLSTM to compute the representations of both sentences. The last time-step of the contextual embeddings is taken as the representation $\mathbf{S}_i = (\mathbf{h}^i_{|s_i|}); \mathbf{S}_j = (\mathbf{h}^j_{|s_j|})$. In the interaction layer, DLMG first multiplies each row in the matrix embedding of a sentence by a different weight matrix. Repeating it over all rows corresponds to a batched dot product between a 2-D matrix and a 3-D weight tensor [Mikolov *et al.*, 2013a; Lin *et al.*, 2017]. We call the resulting matrix as a factor. That is, $\mathbf{F}_i = batcheddot(\mathbf{S}_i, \mathbf{W}_{gor})$, and $\mathbf{F}_j = batcheddot(\mathbf{S}_j, \mathbf{W}_{ged})$, where $\mathbf{W}_{gor}$ and $\mathbf{W}_{ged}$ are the weight tensors for governor embedding and governed embedding. The factor of relation is then obtained by taken the element-wise product $\mathbf{F}_{rel} = \mathbf{F}_i \odot \mathbf{F}_j$.

Multi-hop Attention Mind-map Generation (MAMG) takes into account the multiple perspectives of a sentence. Different from MRDMF, MAMG independently encodes the perspectives of both sentences to be detected in advance. In the encoding layer, it computes the latent semantic perspectives in both sentences with the multi-hop self-attention mechanism defined in Eq.3 in Section 4.2. In the interaction layer, it follows DLMG. The overall governing relationship detection employs the natural language inference algorithm proposed by [Lin *et al.*, 2017].

Evaluation Metric

Borrowing the idea from ROUGE [Lin, 2004] to evaluate subjective outputs, we design a recall based method (see Algorithm 2) to compute the similarities between the generated mind-maps and the artificial ones. ROUGE is widely used for automatic evaluation of summary tasks, which effectively evaluates how similar the generated summary is to the ground truth. By dividing mind-maps into minimum sub-graphs with two nodes, comparing sub-graphs is equivalent to compare their plain text expressed in the nodes, which is exactly the same task as summarization evaluation does. We use a greedy strategy to find the matching parts of two mind-maps. Every sub-graph in one mind-map is compared with all the sub-graphs in the other. The proposed method truncates a generated mind-map to the same size of the artificial one by removing the least salient nodes (see Step 3). It avoids the potential trick to improve the similarity score by putting more nodes in

	SSM	KSM		SSM	KSM
ParserA	0.059	0.048	DLMG	0.41	0.39
ParserB	0.059	0.049	MAMG	0.43	0.40
LexRank	0.37	0.34	MRDMF	**0.54**	**0.52**

Table 1: Experimental Results (Similarity Score)

a mind-map. For each pair of the governor and its governed node in the manual mind-map, the algorithm finds the most similar pair (msp) in the generated mind-map, accumulates the similarity, and removes the selected pair (see Step 7-10, where $r(x, y)$ denotes the ROUGE score between x and y). The final score is the average of the total similarity scores of all the matched pairs. The proposed similarity metric returns a score ranging from zero to one.

Data Set

In the experiment, we built a test corpus by randomly selecting 100 news articles from CNN news corpus. The size of the test dataset is about 60,000 words. The average length of the news article is about 956 words. For every article, one annotator writes the mind-map and the other reviews the output. If the reviewer doesn't agree with any content of the mind-map, they discussed to reach consensus. Two examples of the manual mind-maps are shown in Fig. 5 in Section 6.3. It is a very challenging task for the experts to write mind-maps for the long articles because various perspectives might be implied in the long text.

6.2 Experimental Results

The experimental results are shown in Table 1. The proposed framework is performed with the confident threshold $T = 0.4$. Experimental results show that our proposed model achieves much better performance than the other methods.

MRDMF has achieved 49.44%, 32.52%, and 27.79% improvements compared with LexRank, DLMG and MAMG, respectively. LexRank employs inter-sentence relations. However, the initial state of LexRank is built with the TFIDF method, which computes the shallow unidirectional similarity between sentences. The proposed relation detection method is more powerful on the implicit directed relation detection. Compared with DLMG, multi-perspective extraction in MAMG and MRDMF can better capture the underlying semantic perspectives in a sentence. The superiority of MRDMF to MAMG largely attributes to the interaction schema. MAMG generates the perspectives of the sentence pair of the governor and governed independently in advance. Different from MAMG, during the generation process of the perspectives of the governed sentence, our proposed MRDMF employs a gated recurrence module to encode the semantic association evidence in the governed sentence by incorporating each perspective of the governor sentence. This can effectively capture the deeper semantic association between the governor and the governed sentences.

With ParserA and ParserB, the concepts are selected with respect to their frequencies and locations in an article. The experimental results demonstrate that this strategy severely mismatched the mechanism of manual mind-maps. In addition, the parser-based algorithms fail to connect the nodes as a whole because some sentences have not shared subjects or objects with the others (e.g., see the case shown in Fig. 1(b)). The parser-based algorithms are not good at finding associate relations cross sentences.

Threshold Determination

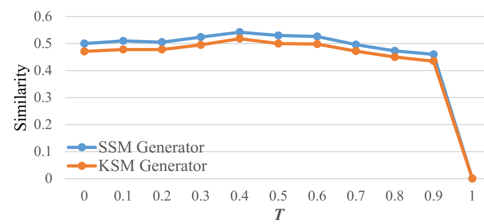

Figure 4: Performance variations with the different values of the confident threshold T.

We also investigate the variations of the performances of the proposed MRDMF with respect to the confident threshold T (see Fig. 4). The variations for both SSM and KSM generators are similar. Both of them obtain good results with the median values of T. T determines the depth of a mind-map. Small values of T make the mind-map too complicated while big values of T ignore too many governing relationships. Hence, we set the confident threshold T around 0.4 in the application.

In addition, the salience score of a sentence is normalized to [0,1] by the number of sentences it governs (see Step 5 in Algorithm 1). When the value of T is one, the averaged salience score must be less than or equal to T, no sentences could be taken into the mind-map, and the similarity score becomes zero. That is why all curves converge to point (1,0) at the end.

6.3 Case Study

Two mind-maps generated with MRDMF are shown in Fig. 5. The case shown in Fig. 5 (a) is a good match between the manual and generated mind-maps. The only difference between the artificial and generated mind-maps is the location of Sentence 1 (see Fig. 5 (a-1)), which is semantically equivalent to Sentence 3. We can quickly grasp the ideas and relations among the ideas of the news with the generated mind-map. Comparing the mind-maps shown in Fig. 5 (a-3) and (a-4), we can find that the two kinds of mind-maps have complementary properties. As sentences generally contain complete semantics and salient ones are selected, SSM is easy to understand; while KSM highlights the core semantics of an article.

The SSM shown in Fig. 5 (b) differs from the manual one. The hierarchy shown in Fig. 5 (b-2) is thinner and taller (with five levels), while the hierarchy shown in Fig. 5 (b-3) is flatter (with three levels). The manual mind-map lists the main points in the second level of the hierarchy and then shows the pieces of evidence to support these points. The SSM displays more opinions about the news, but the pieces of evidence are

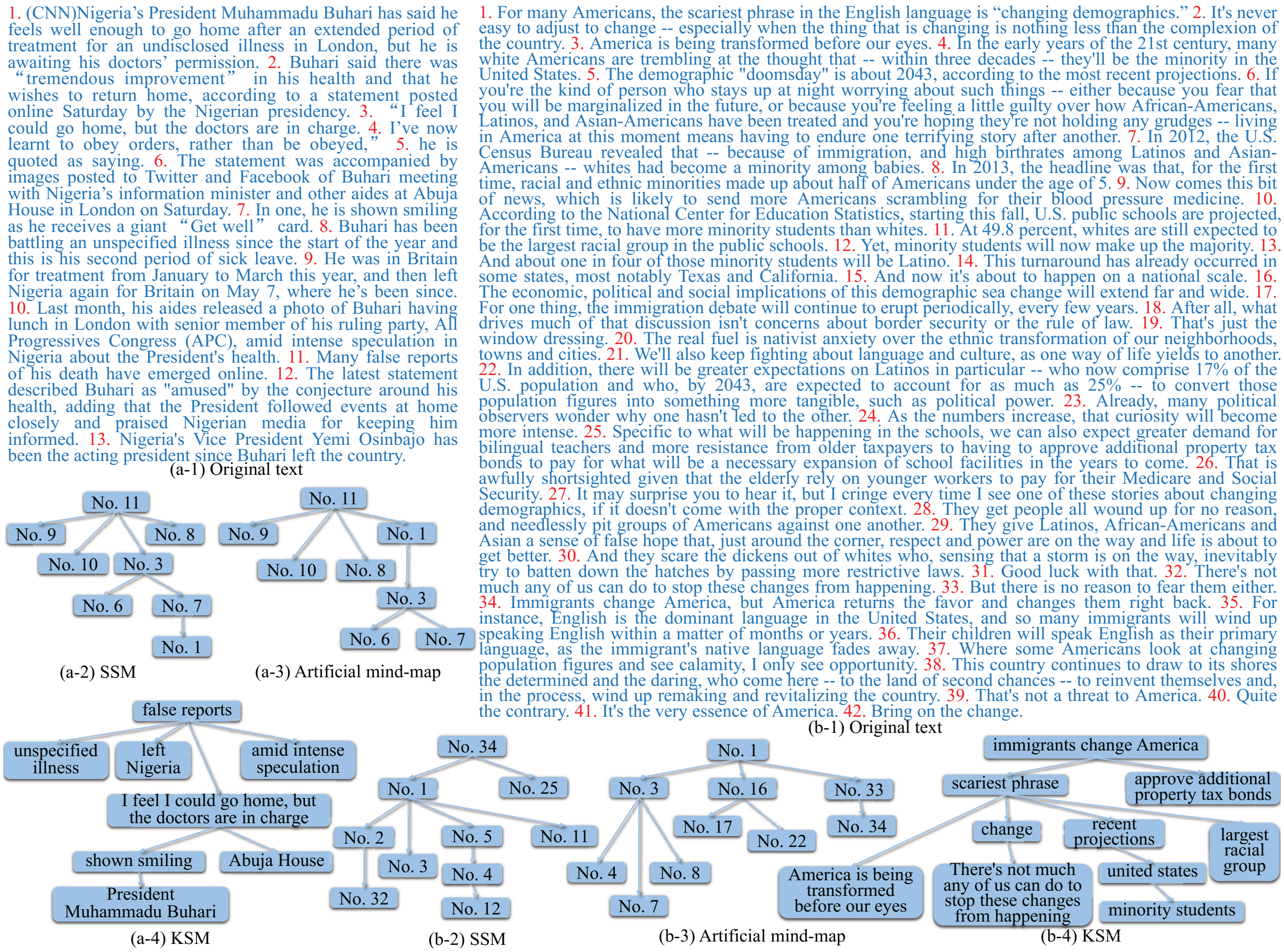

Figure 5: Mind-map cases. Article a-1 is available at *http://edition.cnn.com/2017/08/13/africa/nigeria-president-buhari-illness/index.html*; Article b-1 is available at *http://www.cnn.com/2014/08/18/opinion/navarrette-majority-minority-students-public-schools.*

not sufficient to support the opinions. As we truncate the generated mind-map by remaining the most salient concepts, the sentences of opinions are kept as they are more informative than those of facts. In this sense, the proposed method is not good at processing argumentative essays, where the proposed method is difficult to balance the importance among opinions and evidences. We will further improve the framework on the argumentative essays.

The above case studies show that the generated mind-map illustrates the concepts of an article with a hierarchical organization. We can easily grasp the key concepts and can conveniently reach more details with the directed relations. Mind-maps can help people read, organize information, and study complicated issues. Meanwhile, automatic mind-map generation methods can incorporate with other applications. For example, it can help text summarization method to focus on particular parts of an article, and aid question answering method to deal with clues scattered in long texts.

7 Conclusion

In this paper, we propose a multi-grained framework for the automatic mind-map generation. The present framework outperforms parsing-based algorithm, similarity-based method and the various neural network based methods on the real news data set. We figure out that governing relationship detection is an essential part of the mind-map generation. Our framework detects the directed relationship with multi-hop attention encoding and gated recurrence interaction. The multi-hop attention reveals the latent perspectives of the governor sentence. The gated recurrence interaction computes the semantic association evidence in the governed sentence by incorporating each perspective of the governor. The generated multi-grained mind-maps can aid people in reading, information organization, and data mining applications.

Acknowledgments

This work was done when Yang Wei was a research intern at IBM Research - China.

References

[Brucks and Schommer, 2008] Claudine Brucks and Christoph Schommer. Assembling actor-based mind-maps from text stream. *Computer Science*, 2008.

[Buzan, 2006] Tony Buzan. *The Mind Map Book*. Penguin Books, 2006.

[Cheng and Lapata, 2016] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[Dhindsa *et al.*, 2011] Harkirat S. Dhindsa, MakarimiKasim, and O. Roger Anderson. Constructivist-visual mind map teaching approach and the quality of students' cognitive structures. *Journal of Science Education & Technology*, 20(2):186–200, 2011.

[Elhoseiny and Elgammal, 2013] Mohamed Elhoseiny and Ahmed Elgammal. English2mindmap: An automated system for mindmap generation from english text. In *IEEE ISM*, pages 326–331, 2013.

[Erkan and Radev, 2011] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Qiqihar Junior Teachers College*, 22:2004, 2011.

[Falke and Gurevych, 2017] Tobias Falke and Iryna Gurevych. Utilizing automatic predicate-argument analysis for concept map mining. In *International Conference on Computational Semantics*, 2017.

[Gong *et al.*, 2018] Yichen Gong, Heng Luo, and Jian Zhang. Natural language inference over interaction space. In *ICLR*, 2018.

[Graesser and Clark, 1985] A. C. Graesser and L. F. Clark. Structures and procedures of implicit knowledge. 1985.

[Hermann *et al.*, 2015] Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *NeurIPS*, pages 1693–701, 2015.

[Kudelić *et al.*, 2011] Robert Kudelić, Mladen Konecki, and Mirko Maleković. Mind map generator software model with text mining algorithm. In *International Conference on Information Technology Interfaces*, pages 487–494, 2011.

[Kumar *et al.*, 2016] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*, pages 1378–1387, 2016.

[Lin *et al.*, 2017] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *ICLR*, 2017.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *ACL-04 workshop on text summarization*, pages 74–81, 2004.

[Mikolov *et al.*, 2013a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[Mikolov *et al.*, 2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pages 3111–19, 2013.

[Novak and Canas, 2006] Joeseph D. Novak and Alberto J. Canas. The theory underlying concept maps and how to construct them. In *Technical report, Institute for Human and Machine Cognition, January*, 2006.

[Olney *et al.*, 2011] Andrew M. Olney, Whitney L. Cade, and Claire Williams. Generating concept map exercises from textbooks. In *The Workshop on Innovative Use of NLP for Building Educational Applications*, pages 111–119, 2011.

[Rose *et al.*, 2010] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. *Text Mining*, pages 1–20, 2010.

[Rothenberger *et al.*, 2008] T. Rothenberger, S. Oez, E. Tahirovic, and C. Schommer. Figuring out actors in text streams: Using collocations to establish incremental mind-maps. *Computer Science*, abs/0803.2856:2008, 2008.

[Tavallali *et al.*, 2015] Hossein Tavallali, Saeed Yazdandoust, and Mozhdeh Yazdandoust. A survey of graphs in natural language processing. *NLE*, 21(5):665–98, 2015.

[Wang *et al.*, 2016] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. In *COLING*, pages 1340–1349, 2016.

[Wang *et al.*, 2017] Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*, 2017.

[Willis and Miertschin, 2006] Cheryl L. Willis and Susan L. Miertschin. Mind maps as active learning tools. *Journal of Computing Sciences in Colleges*, 21(4):266–72, 2006.

[Yin *et al.*, 2016] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association of Computational Linguistics*, 4(1):259–272, 2016.

[Zeiler, 2012] Matthew D. Zeiler. Adadelta: An adaptive learning rate method. *Computer Science*, 2012.

[Zhang *et al.*, 2010] Xiaoping Zhang, Xuezhong Zhou, Houkuan Huang, and Shibo Chen. A hierarchical symptom-herb topic model for analyzing traditional chinese medicine clinical diabetic data. In *BMEI*, pages 2246–49, 2010.

[Zubrinic *et al.*, 2012] Krunoslav Zubrinic, Damir Kalpic, and Mario Milicevic. The automatic creation of concept maps from documents written using morphologically rich languages. *Expert Systems with Applications*, 39(16):12709–12718, 2012.