



DATA MINING AND APPLICATIONS

Automatic Text Summarization

Lecturer: Le Ngoc Thanh
Group 09 - 19KHDL

GROUP 9 MEMBERS



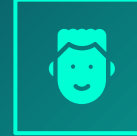
19127009

Lu The Vy



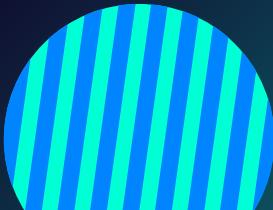
19127360

Duong Thi Xuan Dieu



19127416

Mac Van Hung



CONTENTS

Main Tasks

01.

02.

Extraction-Based
Summarization

Compression-Based Automatic
Summarization

03.

04.

Abstractive Automatic
Summarization

Query-Based
Automatic Summarization

05.

06.

Crosslingual and Multilingual
Automatic Summarization

Summary Quality
Evaluation and Evaluation
Workshops

07.



01. Main Tasks



- Automatic text summarization aims to compress the input texts and generate a condensed summary.
- The summary must satisfy the requirements of sufficient information by covering the main content of the original text and providing low redundancy and high readability.

Automatic text summarization technology can be divided into different types of summarization :

- + **Indicative:** key topics of the input document (or collections of documents)
- + **Informative:** the main information of the input document (or collections of documents)
- + **Critical summarizations:** the main information of the input document (or document sets) or key comments on the original text.

- Based on the number of input documents, automatic summarization is divided into:
 - + **Single-document:** transforms a source text into a condensed, shorter text summary
 - + **Multidocument summarization:** transforms a cluster of topic-related documents into an informative and concise summary
- Considering the relation between input and output languages, automatic summarization can be divided into:
 - + **Monolingual:** The input and output are in the same language.
 - + **Crosslingual:** The input is in one language, the output is in another.
 - + **Multilingual summarization:** The input is in multiple languages, the output is in one of the input languages.

Considering the different applications, automatic summarization technology is divided into:

- + **Generic summarization:** summarizes the main points of the original author
- + **Query-based summarization:** content closely related to the user's query.

Based on the methods generating the summary, automatic summarization technology can also be divided into types of summarization:

- + Extraction-based: produces summaries by extracting important sentences from the original text
- + Compression-based: summarizes the original text by extracting and compressing important sentences
- + Abstraction-based summarization: generates final outputs by rewriting or reorganizing the content of the original text

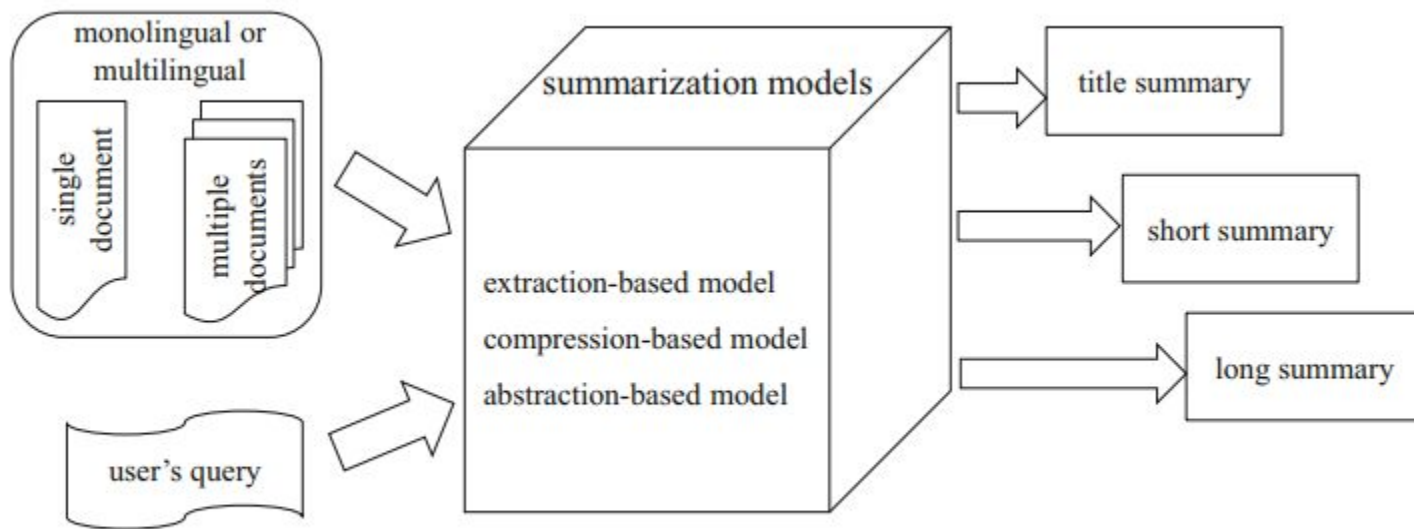


Fig. 11.1 The basic framework of automatic text summarization

- **Application:** multidocument summarization aims to compress the information expressed by multiple documents into a summary according to a compression ratio
- + Transform website content into a concise summary, allows search engines to work faster and improve efficiency
- + Reduce the cost of information storage (only store content summary, not entire document)
- + Save searching time for users



02.

Extraction-Based Summarization



Given a document set $D = \{D_i\}$, $i = 1, \dots, N$. Each document $D_i = S_{i1}, \dots, S_{ij}, \dots, S_{iM}$ consists of a sequence of iM sentences.

Extraction-based summarization chooses K sentences from the document to form a summary, where K is the number of sentences manually set or is obtained through the compression ratio. In many automatic summarization systems, the number of sentences K can be different as long as the final summary is controlled within a limited number of words (e.g., 100 words for English summaries).

- Step 1: finding the most important and informative candidate sentences
- Step 2: minimizing the redundancy of candidate sentences
- Step 3: generating a summary according to a compression ratio or summary length requirements based on sentence order constraints.

In the first step, we need to design an evaluation method to estimate the importance of sentences, and in the last two steps, we need to construct a constraint-based summary generation algorithm

Sentence Importance Estimation

Unsupervised algorithms

- word frequency-based algorithms
- document structure-based algorithms
- graph-based algorithms

A) Word frequency-based algorithm

$$\text{Score}(S_{ij}) = \frac{\sum_{w_k \in S_{ij}} \text{Score}(w_k)}{|\{w_k | w_k \in S_{ij}\}|}$$
$$\text{Score}(w_k) = \text{tf}_{w_k} = \frac{\text{count}(w_k)}{\sum_w \text{count}(w)}$$

Score **S_{ij}** denotes the importance score of the j -th sentence in the i-th document

Count(w_k) denotes the number of times it appears in the whole document D_i.

Sigma w count(w) is the occurrence of all words in the entire document.

Score (w_k) is usually called the term frequency (TF)

(next) The function has a serious problem!

some words are not important for expressing the meaning of a sentence, but they frequently appear in different documents and sentences, and their scores are high in the TF algorithm.

=> **inverse document frequency (IDF)**

$$\text{idf}_{w_k} = \log \frac{|D|}{|\{j | w_k \in D_j\}|}$$

$$\text{idf}_{w_k} = \log \frac{|D|}{|\{j | w_k \in D_j\}|}$$

idf_{wk} is a measure of word universality. If **idf_{wk}** is larger, the denominator in formula is smaller, indicating that fewer documents contain the word, then the word is more important for those specific documents.

=> But **tf_{wk} (term frequency)** and **idf_{wk} (inverse document frequency)** can only represent one aspect of the importance of the word w_k

TFIDF (term-frequency inverse document frequency)

=> more comprehensively describe the importance of the word w_k to document S_{ij}

$$\text{Score}(w_k) = \text{tf_idf}_{w_k} = \text{tf}_{w_k} \times \text{idf}_{w_k}$$

Some clue words (e.g., in a word, in short) and named entities are often used as informative features.

B) Word frequency-based algorithm

- The position of a sentence in the document and the length of the sentence are two common document structure features considered.
- For example: The first sentence in each paragraph can best reflect and express the information of the whole paragraph, especially in English critical articles. Thus, sentence position in the document is very important.
- The importance of sentence position is calculated by the following formula:

$$\text{Score}(S_{i_j}) = \frac{n - j + 1}{n},$$

j denotes the position of the sentence **S_{ij}** in the document and **n** represents the number of sentences in the document.

B) Graph-based algorithm

- **PageRank algorithm:** a ranking model based on a directed graph.
“ If a page is linked from thousands of pages or several important pages, this page is important “ (Page and Brin 1998)
- The weight of each page indicates the importance of that page. The weight can be calculated by the following formula:

$$S(V_i) = \frac{1-d}{N} + d \times \sum_{V_j \in \text{In}(V_i)} \frac{1}{|\text{Out}(V_j)|} S(V_j)$$

$$S(V_i) = \frac{1-d}{N} + d \times \sum_{V_j \in \text{In}(V_i)} \frac{1}{|\text{Out}(V_j)|} S(V_j)$$

- **For a directed graph $G(V, E)$:**
 - + V is a set of nodes in which each node represents a web page,
 - + E is a set of directed edges where each edge $e = (V_i, V_j)$ indicates that it can jump from web page V_i to web page V_j .
- **For a node V_i , $\text{In}(V_i)$** denotes the set of pages linked to V_i , and $|\text{In}(V_i)|$ is the in-degree of V_i . $\text{Out}(V_i)$ denotes the web pages that V_i can link to, and $|\text{Out}(V_i)|$ is the out-degree of V_i .
- **$d \in [0, 1]$** is the damping factor that assigns V_i a prior probability of jumping to any other node V_j from V_i . d is usually set to 0.85 in page ranking.

- **LexRank**: an extension of PageRank
- The directed graph $G(V, E)$ becomes the undirected graph, and each edge of the graph $e = (V_i, V_j)$ carries a weight W_{ij} , V denotes the set of sentences, and E denotes the set of undirected edges
- Common cosine similarity method based on TFIDF to calculate the similarity of two sentences

$$W_{ij} = \frac{\sum_{w \in V_i, V_j} (\text{tf_idf}_w)^2}{\sqrt{\sum_{x \in V_i} (\text{tf_idf}_x)^2} \times \sqrt{\sum_{y \in V_j} (\text{tf_idf}_y)^2}}$$

$$W_{ij} = \frac{\sum_{w \in V_i, V_j} (\text{tf_idf}_w)^2}{\sqrt{\sum_{x \in V_i} (\text{tf_idf}_x)^2} \times \sqrt{\sum_{y \in V_j} (\text{tf_idf}_y)^2}}$$

$\mathbf{w} \in \mathbf{V_i, V_j}$ denotes a word that simultaneously occurs in V_i and V_j . Given a weighted undirected graph $G(V, E)$, the importance score of each node (sentence) is calculated by the following formula:

$$S(V_i) = \frac{1-d}{N} + d \times \sum_{V_j \in \text{adj}(V_i)} \frac{W_{ij}}{\sum_{V_k \in \text{adj}(V_j)} W_{kj}} S(V_j)$$

$\mathbf{V_j} \in \mathbf{adj(V_i)}$ denotes the set of nodes adjacent to V_i , i.e., the set of nodes with edges linked to V_i

- **TextRank:** an extension of PageRank
- Uses the word overlap between two sentences as the similarity metric
- Common cosine similarity method based on TFIDF to calculate the similarity of two sentences

$$W_{ij} = \frac{|\{w_k | w_k \in V_i \text{ \& } w_k \in V_j\}|}{\log |V_i| + \log |V_j|}$$

$$W_{ij} = \frac{|\{w_k | w_k \in V_i \text{ \& } w_k \in V_j\}|}{\log |V_i| + \log |V_j|}$$

where $|\{w_k | w_k \in V_i \text{ \& } w_k \in V_j\}|$ denotes the number of words co-occurring in two sentences; $|V_i|$ and $|V_j|$ are the number of words in sentences V_i and V_j .

Supervised algorithms

- Human-generated summaries are usually not exactly the sentences from the documents
- For supervised algorithms, the best reference summaries should consist of sentences that appeared in the original document, and each sentence in the document is assigned a score between 0 and 1, indicating the degree to which that sentence should be included in the summary, namely, the importance score of the sentence
- Two algorithms using discrete features and distributed representation
- + **Sentence importance estimation based on the log-linear model**
- + **Sentence importance estimation based on deep neural networks**

(a) Sentence importance estimation based on the log-linear model

To estimate sentence importance using machine learning algorithms, the key issue is designing effective features. Different machine learning methods rely on different assumptions. The naive Bayesian method assumes that the features are conditionally independent when the label is given. The hidden Markov model assumes that the first-order Markov property is satisfied between sentences. In contrast, the log-linear model has no independence hypothesis for features.

The log-linear model is a discriminative machine learning method that directly synthesizes various features to model the posterior probability

$$p(sl|s) = \frac{1}{Z(s)} \exp \left\{ \sum_i \lambda_i f_i(s, sl) \right\}$$

where $Z(s)$ is the normalization factor. $f_i(s, sl)$ denotes all kinds of sentence features, and λ_i is the corresponding feature weight. sl is a Boolean-valued sentence tag, and the value `true(1)` or `false(0)` reflects whether the sentence is or is not a summary sentence.

Because the training data are extremely unbalanced (only a few positive examples in a document indicate summary sentences, and the others are all negative examples), many machine learning algorithms, including the log-linear model, tend to predict most of the test sentences will be negative (non-summary sentences). To solve this problem, we use a class priori constraint:

$$sl^* = \operatorname{argmax}_{sl} p(sl) \times p(sl|s) = \operatorname{argmax}_{sl} \left(\log p(sl) + \sum_i \lambda_i f_i(s, sl) \right) \quad (11.9)$$

(b) Sentence importance estimation based on deep neural networks

Although discrete features such as sentence length and word frequency can reflect the importance of a sentence to a certain extent, they cannot model the global semantic information of a sentence. Discrete features face a serious problem of data sparsity and cannot capture the semantic similarity between words (phrases, sentences).

Deep learning methods represent linguistic units in different granularities, such as words, phrases, sentences, and documents, to low-dimensional real-valued vectors, expecting that linguistic units with similar semantics will be close to each other in the vector space. This distributed representation paradigm avoids complicated feature engineering work.

To predict the sentence importance using deep neural networks, the central problem is deciding which kind of neural network structure should be employed to learn the semantic representation of sentences. Recurrent, recursive, and convolutional neural networks are commonly used in distributed sentence representation learning

Convolutional neural networks as an example to introduce sentence importance estimation using distributed feature representations

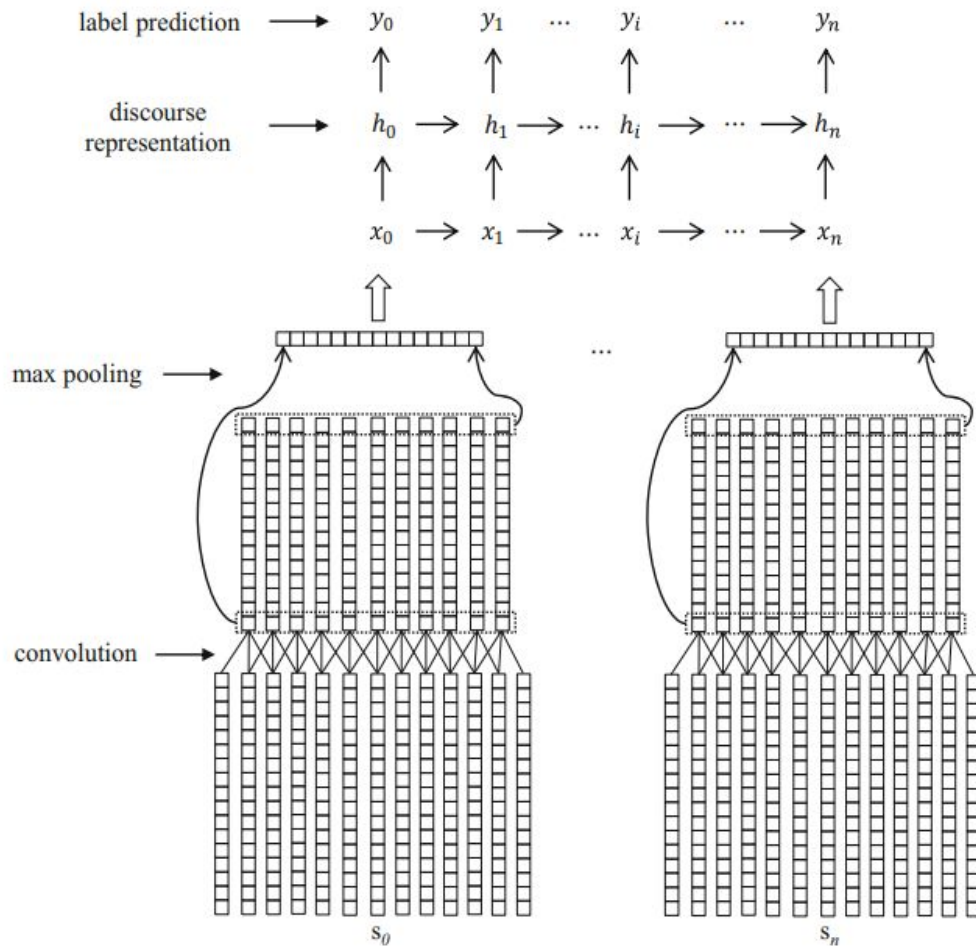
Given a sentence $s = w_0 w_1 \dots w_{n-1}$, each word is mapped into a low-dimensional vector, resulting in a sequence of vectors $X_w = [X_{w_0}, X_{w_1}, \dots, X_{w_{n-1}}]$

The convolutional neural network includes a convolution operator and a pooling operator. The convolution operator extracts local information, while the pooling operator summarizes the global information of a sentence.

The convolution operator consists of L filters $W \in \mathbb{R}^{h \times k}$, and each filter extracts local features along the window of h words $X_{w_i:i+h-1}$

$$u_i = \sigma(W \cdot Xw_{i:i+h-1} + b) \quad (11.10)$$

- σ is a nonlinear activation function (e.g., ReLU and Sigmoid)
- b denotes the bias item.
- a filter convolves from Xw_0 to Xw_{n-1} , a vector $u = [u_0, u_1, \dots, u_{n-1}]$ can be obtained.
- If L different filters are adopted, we will have L local feature vectors of n -dimension.



- **For a single-document summarization task**, choosing sentences with a high importance score can satisfy the requirement of information quantity.
Composing the extracted sentences into a summary according to the order in which they appear in the document can ensure the fluency and readability of the output.
- **For the multidocument summarization task**, if each document carries a timestamp, composing summaries first according to the sentence order in a document and then according to the chronological order of the multiple documents can also ensure outputs of high readability (Barzilay et al. 2002).

Constraint-based Summarization

Wide coverage and low redundancy are the basic requirements for automatic summarization, limited to no more than K sentences or no more than N words. Therefore, maximizing coverage is actually equivalent to minimizing redundancy under the constraint of summary length. The widely used algorithm for minimizing redundancy is derived from the idea of maximum marginal relevance (MMR) (Carbonell and Goldstein 1998).

- The MMR algorithm is mainly proposed for query-related document summarization tasks.

$$\text{MMR}(R, A) = \operatorname{argmax}_{s_i \in R \setminus A} \left\{ \lambda \text{Sim}_1(s_i, Q) - (1 - \lambda) \max_{s_j \in A} \text{Sim}_2(s_i, s_j) \right\}$$

$$\text{MMR}(R, A) = \operatorname{argmax}_{s_i \in R \setminus A} \left\{ \lambda \text{Sim}_1(s_i, Q) - (1 - \lambda) \max_{s_j \in A} \text{Sim}_2(s_i, s_j) \right\}$$

R is the set of all sentences, A denotes the set of selected summary sentences, Q stands for the user query, s_i denotes any sentence in the set of unselected sentences, and s_j means any sentence in the set of selected sentences.

$\text{Sim}_2(s_i, s_j)$ is the similarity between two sentences s_i and s_j . λ is a hyperparameter that weighs correlation and redundancy. The larger λ is, the more s_i 's relevance to the user query is emphasized; otherwise, redundancy is more emphasized.

In general automatic document summarization tasks, the redundancy calculation method is similar to the MMR algorithm. Basically, the following calculation formulas can be used:

$$\text{MMR}'(R, A) = \operatorname{argmax}_{s_i \in R \setminus A} \left\{ \lambda \text{Score}(s_i) - (1 - \lambda) \max_{s_j \in A} \text{Sim}(s_i, s_j) \cdot \text{Score}(s_j) \right\} \quad (11.14)$$

How to generate summaries given the sentence importance score in the framework of the graph-based summarization

1. We first initialize two sets, $A = \emptyset$ and $B = \{s_i | i = 1, \dots, n\}$, which denote the summary sentence set and the unselected sentence set, respectively. We then initialize the overall score of the importance and redundancy for each sentence $RS(s_i) = \text{Score}(s_i)$, $i = 1, \dots, n$ (at the beginning, the redundancy score is unknown, and the overall score only includes the importance score of the sentence).
2. Then, we rank set B according to the score from high to low in the results of $RS(s_i)$.
3. Assuming s_i is the sentence with the highest score or, say, the sentence that ranks first in B, we remove s_i from B and add it to A. Then, we update the overall score of each remaining sentence in B according to the following formula:

$$RS(s_j) = RS(s_j) - \lambda \text{Sim}(s_i, s_j) \cdot \text{Score}(s_i)$$

4. We return to step 2 and do the next iteration until set B is empty or set A meets the requirement in terms of sentence number

Conclusion

- The extraction-based automatic summarization methods take sentences as the basic unit.
- The algorithm is simple and intuitive and can maintain the fluency and readability of the summary.


Some difficult problems:

- + The irreconcilable contradiction between the coverage and the summary length.
- + The length constraints of summaries are generally expressed by the number of words or sentences contained in the final summary. Length constraints limit the number of generated sentences, while the requirement of summary coverage aims at extracting more information, that is, extracting more sentences.
- + If the sentences selected into summaries are important but contain too much useless information, which leads to longer sentences, it will directly prevent other important sentences from being included in the summary



03.

Compression-Based Automatic Summarization



The basic idea is to condense sentences by retaining important content and deleting unimportant information so that the final summary can contain more sentences under a fixed length (word count) with improved information coverage

Sentence Compression Method

- Sentence compression can be defined as a word deletion task (Knight and Marcu 2002) - deleting the unimportant words to form a compressed version of the sentence.
- There are two types of supervised sentence compression method (Knight and Marcu 2002):

(1) Sentence compression based on the noisy channel model

(2) Decision-based sentence compression method

(1) Sentence compression based on the noisy channel mode

- Assumes that the original sentence is generated by adding auxiliary information to the compressed result. Given the original long sentence s , the goal is to find the best compressed sentence t to maximize the posterior probability $p(t|s)$, which can be decomposed by the Bayesian rule:

$$p(t|s) = \frac{p(t) \cdot p(s|t)}{p(s)}$$

Searching for the best compressed result t requires solving the following optimization problem

$$t^* = \operatorname{argmax}_t p(t) \cdot p(s|t)$$

$p(s|t)$ is called the channel model, which represents the probability that the original sentence is generated from the compressed output, and $p(t)$ is called the source model, which measures the fluency of the compressed result

a) Source model

- The source model will be used to measure the degree to which the compressed parse tree conforms to the grammar. The probabilistic context-free grammars (PCFG) and bigram language model can be employed to calculate the source model

$$\begin{aligned}\hat{p}_{\text{tree}}(t_1) = & p_{\text{cfg}}(\text{TOP} \rightarrow G|\text{TOP}) \cdot p_{\text{cfg}}(G \rightarrow HA|G) \cdot p_{\text{cfg}}(A \rightarrow CD|A) \cdot \\ & p_{\text{cfg}}(H \rightarrow a|H) \cdot p_{\text{cfg}}(C \rightarrow b|C) \cdot p_{\text{cfg}}(D \rightarrow e|D) \cdot p_{\text{bigram}}(a|\text{EOS}) \cdot \\ & p_{\text{bigram}}(b|a) \cdot p_{\text{bigram}}(e|b) \cdot p_{\text{bigram}}(\text{EOS}|e)\end{aligned}$$

Pcfg (TOP→GITOP) denotes the probability of generating G from the root node TOP.

Pcfg (G→H AIG) denotes the probability of generating two nodes H A from node G with PCFG.

Pbigram (bla) represents the bigram language model probability between leaf nodes a and b.

b) Channel Model

- There are two parts in the channel model: one is the probability that the compressed tree expands to the original tree, and the other is the context-free grammar probability of the new subtree added in the original tree compared to that of the compressed tree

$$p_{\text{expand_tree}}(s|t_1) = p_{\text{exp}}(G \rightarrow HA | G \rightarrow HA) \cdot p_{\text{exp}}(A \rightarrow CBD | A \rightarrow CD) \cdot \\ p_{\text{cfg}}(B \rightarrow QR | B) \cdot p_{\text{cfg}}(Q \rightarrow Z | Q) \cdot p_{\text{cfg}}(Z \rightarrow c | Z) \cdot \\ p_{\text{cfg}}(R \rightarrow d | R)$$

=> **Pexp (G → H A | G → H A)** and **Pexp (A → CBD | A → CD)** denote the probability that binary tree $G \rightarrow H A$ remains unchanged. Binary tree $A \rightarrow CD$ expands to ternary tree $A \rightarrow CBD$, respectively

c) Search the best compressed sentence

- According to the source model and channel model, the posterior probability of the compressed tree t_1 can be calculated through the following formula

$$\hat{p}_{\text{compress_tree}}(t_1|s) = \frac{\hat{p}_{\text{tree}}(t_1) \cdot p_{\text{expand_tree}}(s|t_1)}{\hat{p}_{\text{tree}}(s)}$$

=> The compressed sentence with the highest posterior probability will be chosen. For two candidate compressed trees, t_1 and t_2 , if $\hat{p}_{\text{compress_tree}}(t_1|s) > \hat{p}_{\text{compress_tree}}(t_2|s)$, t_1 is selected and the compressed sentence can be obtained by concatenating the leaves of t_1 .

d) Model parameter training

1. The rule deduction probability of context-free grammars, e.g., $P_{\text{cfg}}(G \rightarrow H \mid G)$;
2. The tree structure extension probability, e.g., $p_{\text{exp}}(A \rightarrow CBD \mid A \rightarrow CD)$;
3. The bigram language model probability, e.g., $p_{\text{bigram}}(b \mid a)$

=> To estimate the above three kinds of probabilities, parse the training data $\{s_k, t_k\}_{k=1}^K$ to obtain the tree structures of the original sentence and the compressed sentence.

- For the probabilities of the first kind, the maximum likelihood estimation can be used to obtain the rule deduction probability of context-free grammar in the tree structures of the original and compressed sentences.
- For the probabilities of the second class, it is necessary to perform node alignment between the tree structures of s, t and then calculate the extended probability of the tree structure by using the maximum likelihood estimation method.
- For the language model probabilities, we can simply calculate the bigram probability according to the counts of words and bigrams

e) Model decoding

- For the original sentence s whose tree structure has n child nodes, there are two choices for each node in the process of sentence compression: delete or retain.
- Therefore, there are $(2^n - 1)$ choices when compressing the original sentences s . All candidate condensed sentences can be stored in a shared forest, and dynamic programming can be utilized to search for the best compressed candidate. Since all the probabilities need to be accumulated in the noisy channel model, the longer the candidate is, the smaller the posterior probability. Thus, the shorter candidate compressed sentence will be preferred by the noisy channel model.

(2) Decision-based sentence compression method

- The decision-based sentence compression method handles sentences from the perspective of tree structure rewriting. The goal of this method is to rewrite the structure tree of the original sentence s into the structure tree of the compressed sentence t_2 . The rewriting process can be achieved by a series of shift, reduce, and delete actions (similar to the shift-reduce parsing method).

- The stack (ST) and the input list (IList) are two key data structures.
- The stack is used to store the tree structure fragments corresponding to the compressed sentences so far, and the stack is empty at the beginning of the algorithm.
- The input list stores the words corresponding to the original sentence and their syntactic structure tags.

stack(ST)	Input List(IList)	Stack(ST)	Input List(IList)
	<div>$\begin{array}{ccccccc} G & & & & & & \\ H & A & & & & & \\ a & C & B & & & & \\ & b & Q & D & & & \\ & & Z & R & e & & \\ & & c & d & & & \end{array}$</div> <div>Step ①~② SHIFT; ASSIGNTYPE H</div>	<div>$\begin{array}{c} F \\ / \quad \backslash \\ H \quad K \\ \quad \\ a \quad b \end{array}$</div> <div>$\begin{array}{ccccccc} B & & & & & & \\ Q & & D & & & & \\ Z & R & e & & & & \\ c & d & & & & & \end{array}$</div> <div>Step ⑥ DROP B</div>	
<div>$\begin{array}{c} H \\ \\ a \end{array}$</div>	<div>$\begin{array}{ccccccc} A & & & & & & \\ C & B & & & & & \\ b & Q & D & & & & \\ & Z & R & e & & & \\ & c & d & & & & \end{array}$</div> <div>Step ③~④ SHIFT; ASSIGNTYPE K</div>	<div>$\begin{array}{c} F \\ / \quad \backslash \\ H \quad K \\ \quad \\ a \quad b \end{array}$</div> <div>$\begin{array}{c} D \\ e \end{array}$<div>Step ⑦~⑧ SHIFT; ASSIGNTYPE D</div></div>	
<div>$\begin{array}{cc} H & K \\ & \\ a & b \end{array}$</div>	<div>$\begin{array}{ccccccc} B & & & & & & \\ Q & & D & & & & \\ Z & R & e & & & & \\ c & d & & & & & \end{array}$</div> <div>Step ⑤ REDUCE to F</div>	<div>$\begin{array}{c} F \\ / \quad \backslash \quad \\ H \quad K \quad D \\ \quad \quad \\ a \quad b \quad e \end{array}$</div> <div>Step ⑨ REDUCE to G</div>	
		<div>$\begin{array}{c} G \\ / \quad \backslash \quad \\ F \quad D \\ / \quad \backslash \quad \\ H \quad K \quad e \\ \quad \\ a \quad b \end{array}$</div>	

Each syntactic tag is only assigned to the leftmost word it covers in the subtree. In this example, the root node in the structure tree of the original sentence s is G , and G is only attached to a , the first word of the sentence. Similarly, H is also attached to the word a . As a result, G and H are the syntactic tags related to the word a . When deleting subtrees, if a is to be removed, the whole subtree related to the word a will also be deleted.

The process of rewriting the syntactic structure tree is accomplished by the following four types of actions:

- **SHIFT:** This action moves the first word in the input list IList into the stack ST;
- **REDUCE:** Pop K tree fragments up at the top of the ST stack, merge them into a new subtree, and move the new subtree into ST;
- **DROP:** Delete the complete subtree corresponding to the syntactic tag from the input list IList;
- **ASSIGNTYPE:** Assign a new syntactic label to the top subtree of the ST stack, which is generally used to rewrite part-of-speech labels of words.

Automatic Summarization Based on Sentence Compression

- The automatic summarization method based on sentence compression includes two core algorithms: candidate sentence selection and sentence compression.
- There are three ways to combine the two algorithms:
 - 1. Select-Compress:** We first extract the candidate summary sentences according to the importance score and then simplify them with the sentence compression algorithm. In the end, we can display more information in the summary given the same length constraint.
 - 2. Compress-Select:** We first simplify all sentences in a document or documents by using a sentence compression algorithm and then select the summary sentences according to the sentence importance scores.
 - 3. Joint-Select-Compress:** A unified framework is employed to simultaneously optimize sentence selection and compression and ultimately output the simplified summary sentences.

- The final results that satisfy all the constraints can be generated by solving integral linear programming to account for both efficiency and quality.
- Compression-based automatic summarization can remove secondary information or repetitive information in sentences.
- This method is obviously more reasonable than the extraction-based summarization approach.

- **It** **still** **has** **some** **shortcomings:**

- + it cannot fuse multiple sentences expressing similar but complementary information. Assume that two sentences have high importance scores and should be included in the summary.
- + Nevertheless, the two sentences are similar, and only one can be selected due to the redundancy constraint. As a result, the complementary and important information in the other sentence will not appear in the final summary if we employ the extraction-based or compression-based summarization methods.



04. Abstractive Automatic Summarization

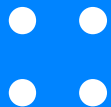




Abstractive Summarization based on Information Fusion



- Inherits and develops extraction-based and compression-based summarization methods.
- Simulates the way a human generates a summary methodology: selecting important concepts and related facts in the process of text reading, reorganizing these concepts and facts, and, finally, generating new summary sentences.






Abstractive summarization implementation



Step 1: Use the deep semantic analysis method to transform similar sentences into abstract meaning representation (AMR);

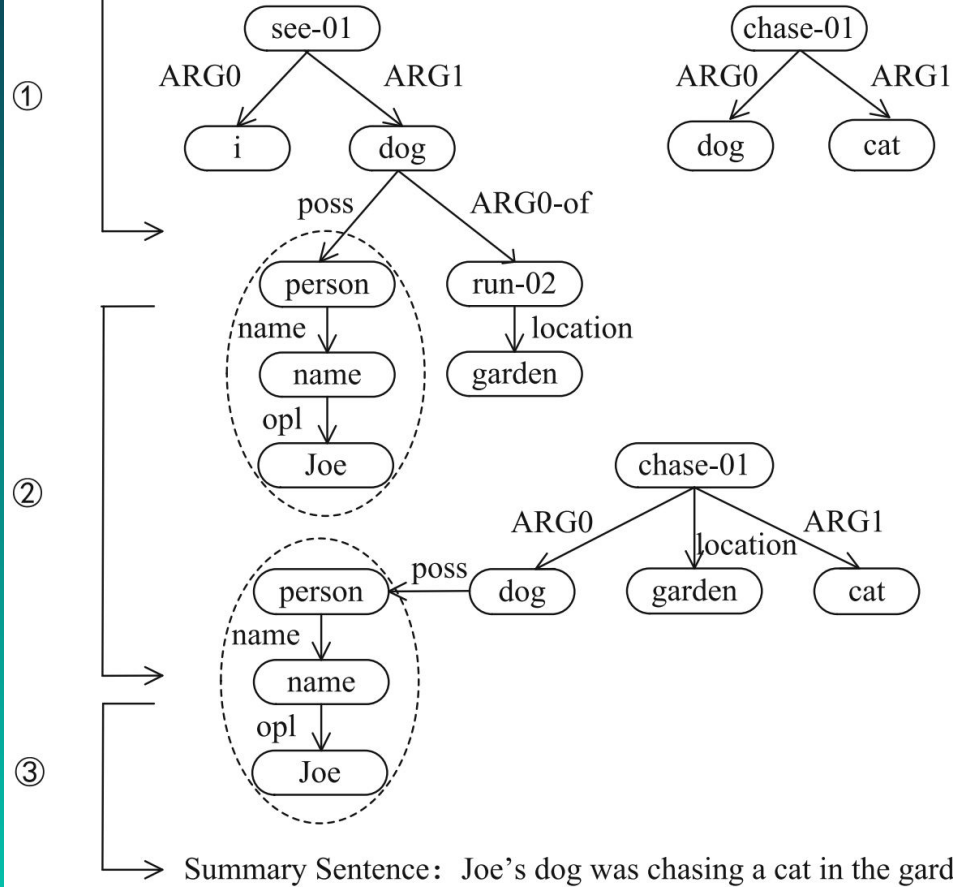
Step 2: Merge two AMR graphs into one AMR graph;

Step 3: Define concepts and facts using predicate-argument information and generate comprehensive expressions based on core arguments (e.g., dog)



Sentence 1: I saw Joe's dog, which was running in the garden

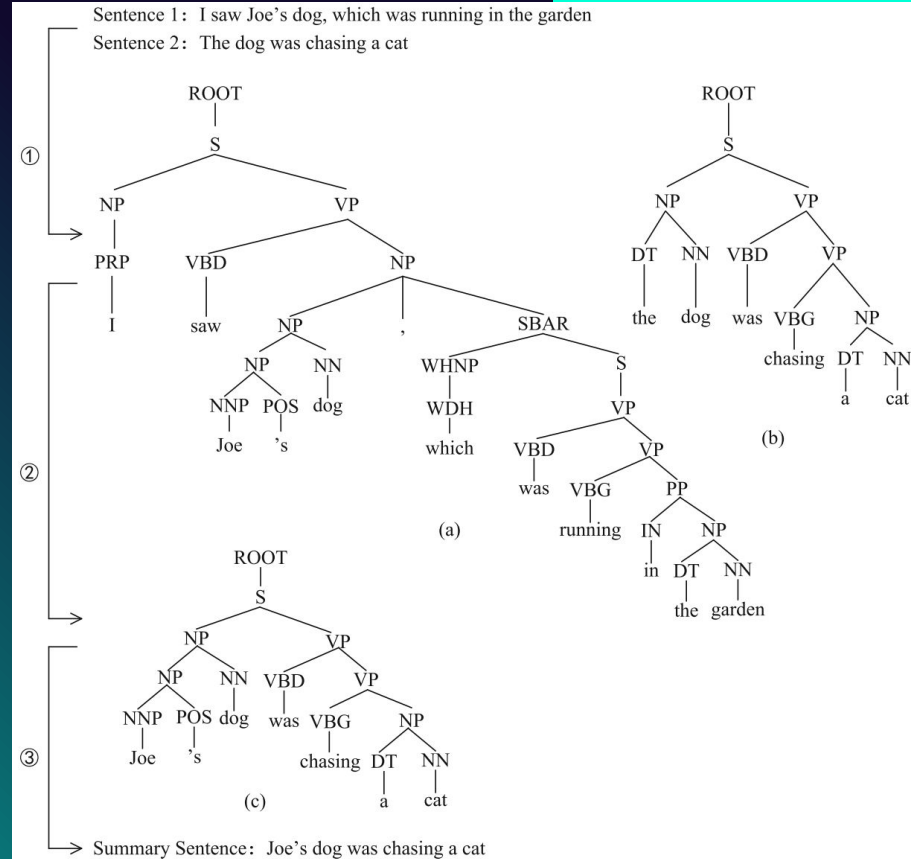
Sentence 2: The dog was chasing a cat



Definition of concepts and facts

Concepts and facts are determined using parsing trees. Generally, concepts consist of noun phrases (NP), while facts are verb phrases (VP).

Since a parsing tree is often very deep and there are many nodes rooted at the NP and VP, it is impossible to use every NP or VP phrase as a candidate for concepts or facts.





Rules to define the candidate concepts and facts using NP and VP:

(a) If NP/VP is the child node of a complete sentence or clause node (S and SBAR in Fig. 11.8), it is considered a concept/fact candidate, represented by $S(NP)$ and $S(VP)$:

(b) If the parent node of NP/VP is $S(NP)$ (or $S(VP)$), then it is regarded as a concept/fact candidate and is denoted as $S(NP(NP))$ and $S(VP(VP))$;

(c) If the parent node of NP/VP is $S(NP(NP))$ (or $S(VP(VP))$), then NP/VP is treated as a candidate concept/fact.



Importance estimation of concepts and facts

Calculate the importance scores for concepts and facts (NP and VP).

$$\text{Score (NP)} = \frac{\text{count (NE}_{\text{NP}})}{\text{count (NE}_{\text{doc}})}$$

$\text{count(NE}_{\text{NP}})$: the number of named entities contained in the NP phrases


$\text{count(NE}_{\text{doc}})$: the total number of named entities contained in the documents where NP phrases are located.


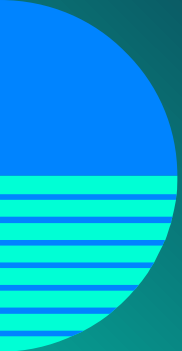


Definition of the compatibility between concepts and facts



Define more relaxed compatibility constraints:


- If NP_j is compatible with NP_i , NP_j and VP_i can be combined
 - If VP_j is compatible with VP_i , NP_i and VP_j can also be combined
- 



The compatibility of VP_i and VP_j can be determined by the co-occurrence degree of language units such as words, phrases, and named entities. Specifically, the Jaccard index can be used to calculate whether two VP VP_i and VP_j are compatible

$$J(VP_i, VP_j) = \frac{|\text{Set}_{VP_i} \cap \text{Set}_{VP_j}|}{|\text{Set}_{VP_i} \cup \text{Set}_{VP_j}|}$$

Set_{VP_i} and Set_{VP_j} denote the sets of words, bigrams, and named entities in VP_i and VP_j , respectively. If the Jaccard index is larger than a predefined threshold, VP_i and VP_j are considered compatible.



Summary generation based on concepts and facts

Abstractive summarization aims to search a set of NP and VP from all the candidate sets of NP (concepts) and VP (facts) to maximize the importance score.

$$\max \sum_i \alpha_i S_i^N - \sum_{i < j} \alpha_{ij} (S_i^N + S_j^N) R_{ij}^N + \sum_i \beta_i S_i^V - \sum_{i < j} \beta_{ij} (S_i^V + S_j^V) R_{ij}^V$$

α_i and β_i are Boolean values indicating whether NP_i and VP_i are selected.
 S_i^N and S_i^V denote the importance scores of NP_i and VP_i ,

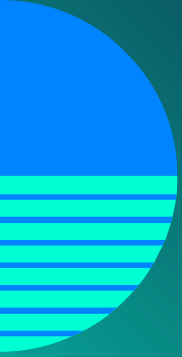
R_{ij}^N and R_{ij}^V denote the similarity score between NP_i and NP_j and VP_i and VP_j .
If NP_i and NP_j are coreferential, $R_{ij}^N = 1$. Otherwise, it can be calculated by the above Jaccard index.

Summary length constraints

$$\sum_i l(\text{NP}_i) \times \alpha_i + \sum_j l(\text{VP}_j) \times \beta_j \leq L$$

L is the predefined length limit for the output summary, such as 100 words.

$l(\text{NP}_i)$ and $l(\text{VP}_j)$ denote the lengths of NP_i and VP_j , respectively.




Includes several cascading modules, ranging from recognition of the semantic units of sentences (e.g., concepts and facts) to the importance estimation of the semantic units and, finally, to summary sentence generation by fusing the semantic units from various sources.


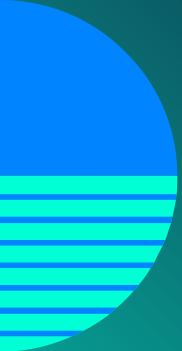
This system is complicated and strongly depends on the quality of the syntactic parsing or semantic analysis. Thus, this method is difficult to widely use.





Abstractive Summarization based on Encoder-Decoder Framework


- Inspired by the end-to-end neural machine translation framework
 - It first encodes the text in the semantic vector space to simulate the process of text understanding by humans
 - Then generates the summary word by word through the decoder network to simulate the process of natural language sentence generation.
- 


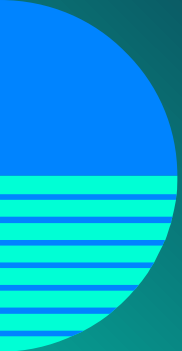


Given an original sentence $X = (x_1, x_2, \dots, x_{T_x})$, we plan to generate a simplified version $Y = (y_1, y_2, \dots, y_{T_y})$


x_j and y_i denote the low-dimensional real-valued vector of the j -th and i -th words in the sentence X and Y .

Without loss of generality, a bidirectional recurrent neural network (BiRNN) can be used to encode the original sentence X and obtain the hidden semantic representation $C = (h_1, h_2, \dots, h_{T_x})$





05. Query-Based Automatic Summarization



Query-based automatic summarization methods can be formally defined as follows:

“ Given a document or document set D and a query T expressed in a string or sentence, we attempt to generate a summary that is closely related to the query T . “

Relevance Calculation Based on the Language Model

- In the early years of research, the query-based automatic summarization method mainly focused on generating summaries of personal profiles (or biographies).
- It produces summaries according to the queries of who is X and what is X.).
- For the query who is X, this method designs a classifier that identifies whether a sentence in document D is related to the introduction of a biography:

Relevance Calculation Based on the Language Model

1. First, an unsupervised method is used to extract the biography texts from Wikipedia with information extraction technology, and the extracted biography texts are employed in a language model L_{wiki} .
2. In addition, another general language model L_{news} is trained using news documents. Then, for a sentence s in the test documents, if $L_{wiki}(s) > L_{news}(s)$, s is considered to belong to the biography information, while $L_{news}(s) > L_{wiki}(s)$ means that it does not belong.

Relevance Calculation Based on Keyword Co-occurrence

- Text summarization method for open queries, measures the importance of a sentence by calculating the number of keywords in it, and the keywords are determined by both the query and the original document set. In a query sentence, not every word deserves attention.
- The usual approach is to identify all the nouns, verbs, adjectives, and adverbs in the query sentence and regard them as query keywords; these are denoted by the set WS_{query} . The keywords in the original document set can be obtained by calculating the topic words W_{stopic} . Topic words are those words that are more likely to appear in current documents on a specific topic than in other general texts.
- Therefore, the proportion of topic words in a sentence basically reflects the importance of the sentence. The set of topic words can be determined by computing the likelihood ratio, mutual information, and TFIDF.

Graph-Based Relevance Calculation Method

- The graph-based PageRank algorithm is widely used in extraction-based generic summarization. This algorithm can be easily adapted to query-based summarization. In the graph-based PageRank algorithm, the importance score of sentences is calculated as follows:

$$S(V_i) = \frac{1-d}{N} + d \times \sum_{V_j \in \text{adj}(V_i)} \frac{W_{ij}}{\sum_{V_k \in \text{adj}(V_j)} W_{jk}} S(V_j)$$

Graph-Based Relevance Calculation Method

- To model the relevance between the query and text sentence as well, the cosine similarity between the text sentence and the query can be used as the relevance score $\text{rel}(V_i, \tau)$. It can be further employed as the initial score in the graphbased algorithm: $S(V_i|\tau) = \text{rel}(V_i, \tau)$.

$$S(V_i|\tau) = (1 - d) \times \frac{S(V_i|\tau)}{\sum_{V_k \in \text{adj}(V_i)} S(V_k|\tau)} + d \times \sum_{V_j \in \text{adj}(V_i)} \frac{W_{ij}}{\sum_{V_k \in \text{adj}(V_j)} W_{jk}} S(V_j|\tau) \quad (11.41)$$

- After calculating the score of each sentence with the above formula, the final summary sentences can be generated according to the length and redundancy constrain





06. Crosslingual and Multilingual Automatic Summarization




Crosslingual Automatic Summarization



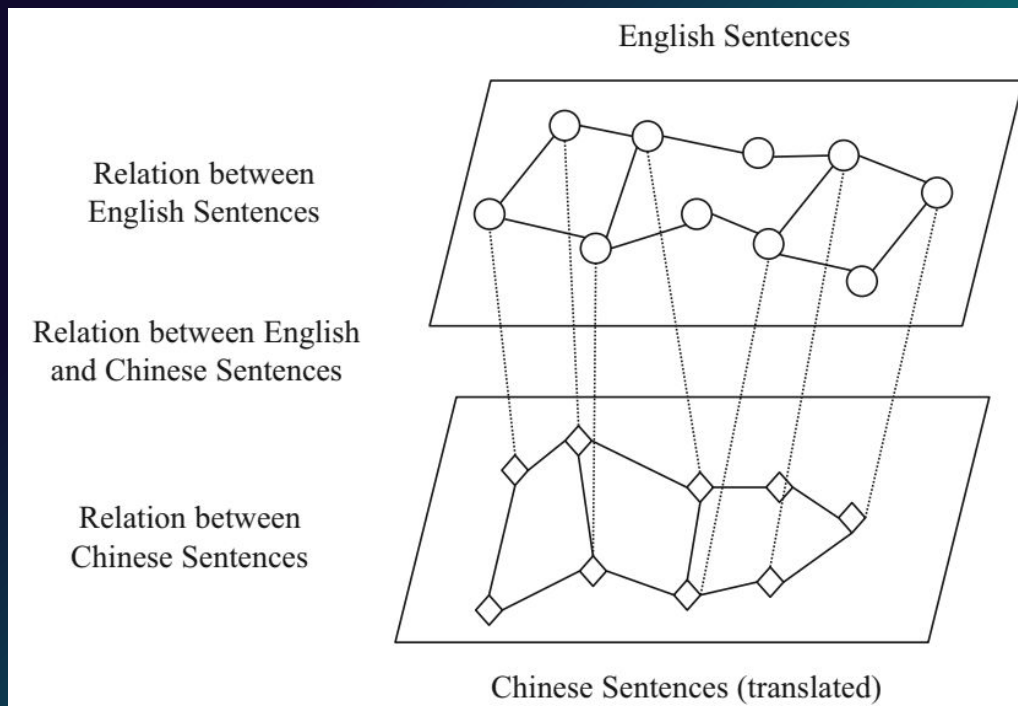
- Uses the documents in source language A as input and outputs a summary in target language B.
 - Language A summaries can be generated first and then translated into language B using machine translation.
 - Two popular methods: summarizes before translation and translates before summarization.
- 



	Summarizes before translation	Translates before summarization
Definition	<ul style="list-style-type: none"> - Extracts summary sentences from language A documents - Translates the language A summary into language B. 	<ul style="list-style-type: none"> - Translates all the sentences in the language A documents into language B - Extracts the summary sentences from the language B translations.
Pros	Make full use of language A information	Take full advantage of language B information.
Cons	<ul style="list-style-type: none"> - Contain many errors due to the unsatisfactory quality of machine translation - May be incorrectly translated. 	<ul style="list-style-type: none"> - Contain translation errors - May not correspond to the actual important information in the original English texts.



A graph-based crosslingual summarization method can simultaneously make full use of information from both languages
=> some extent avoids translation errors



Suppose $V^{en} = \{s_i^{en} | 1 \leq i \leq n\}$ and $V^{cn} = \{s_i^{cn} | 1 \leq i \leq n\}$

denotes sentences in English dataset D^{en} and D^{cn}

n : number of sentences

s_i^{cn} : Chinese translation of s_i^{en}

=> Construct an undirected graph $G = (V^{en}, V^{cn}, E^{en}, E^{cn}, E^{en cn})$

E^{en} : the relation between any two sentences in the English documents

E^{cn} : the relation between any two sentences in the Chinese documents

$E^{en cn}$: the relation between any sentence in V^{en} and any sentence in V^{cn}

Weight matrix of edges

$$W^{en} = \left(W_{ij}^{en} \right)_{n \times n}$$

$$W_{ij}^{en}$$

Similarity between the i -th sentence s_i^{en} and the j -th sentence s_j^{en} in the English documents V^{en}

$$W_{ij}^{en} = \begin{cases} \text{sim}_{\text{cosine}}(s_i^{en}, s_j^{en}), & i \neq j \\ 0, & \text{others} \end{cases}$$

$$\text{sim}_{\text{cosine}}(s_i^{en}, s_j^{en})$$

Cosine similarity between the TFIDF vectors of s_i^{en} and s_j^{en}

Weight matrix of edges

$$W^{\text{encn}} = \left(W_{ij}^{\text{encn}} \right)_{n \times n}$$

The corresponding TFIDF vectors belong to two languages and are not in the same semantic space, so their similarity cannot be calculated directly with the cosine distance.

$$W_{ij}^{\text{encn}} = \sqrt{\text{sim}_{\text{cosine}} \left(s_i^{\text{en}}, s_j^{\text{en}} \right) \times \text{sim}_{\text{cosine}} \left(s_i^{\text{cn}}, s_j^{\text{cn}} \right)}$$

Important score

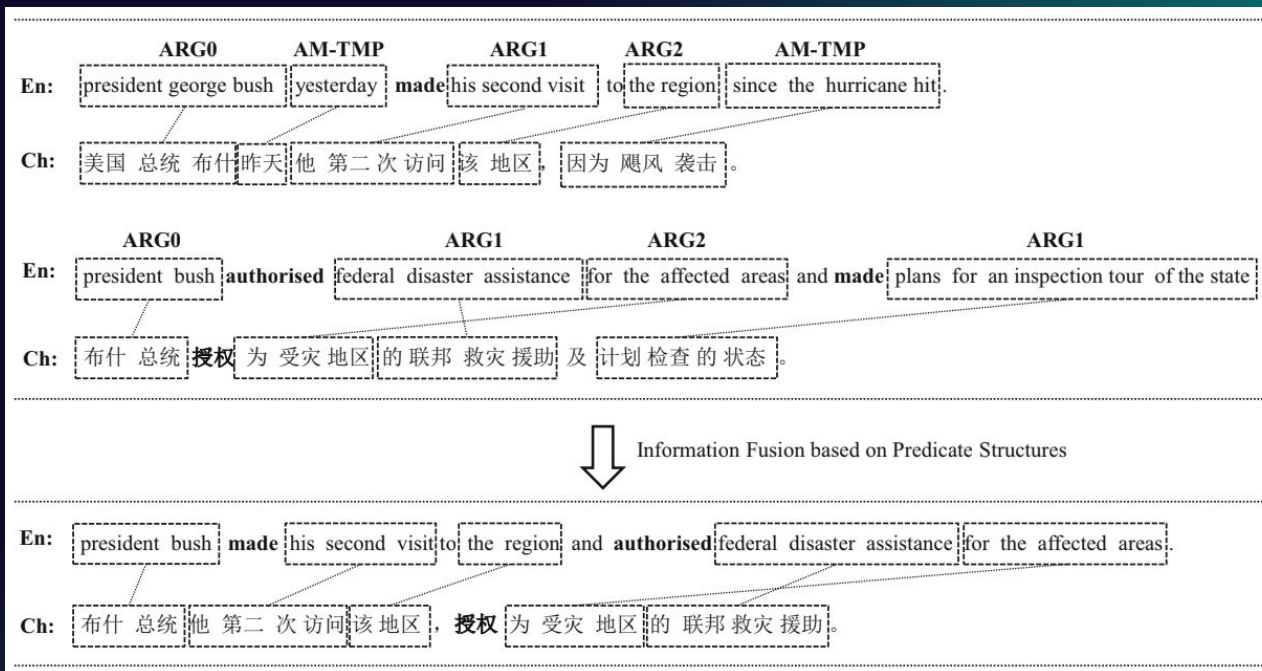
$$\begin{aligned}v(s_i^{\text{cn}}) &= \alpha \sum_j W_{ji}^{\text{cn}} v(s_j^{\text{cn}}) + \beta \sum_j W_{ji}^{\text{encn}} u(s_j^{\text{en}}) \\u(s_j^{\text{en}}) &= \alpha \sum_i W_{ij}^{\text{en}} u(s_i^{\text{en}}) + \beta \sum_i W_{ij}^{\text{encn}} v(s_i^{\text{cn}})\end{aligned}$$


$\alpha + \beta = 1$ balances the contribution of two languages

Once $v(s_i^{\text{cn}})$ is obtained

=> summary sentence selection algorithms in the generic summarization method can be used to obtain the final Chinese summary.

Crosslingual summarization method based on the information fusion model





The two key issues in the whole process are calculating comprehensive scores (importance and translation quality) for the concepts and facts and assessing the compatibility between concepts and facts.

The importance score Sim can be estimated by calculating the proportion of named entities or adopt a graph-based bilingual estimation model.

In the computation of translation quality, the word translation probability P_{lex} and the language model score P_{lm} .

Given an English concept or fact, $phen = e_0 e_1 \dots e_l$ and its corresponding translation $phcn = c_0 c_1 \dots c_m$



Word probability

$$p_{\text{lex}}(\text{ph}_{\text{cn}}|\text{ph}_{\text{en}}, a) = \left\{ \prod_{j=0}^m \frac{1}{|\{i \mid (i, j) \in a\}|} \sum_{\forall (i, j) \in a} p(c_j | e_i) \right\}^{\frac{1}{m+1}}$$

a : the translation correspondence (word alignment) between the words in ph_{cn} and ph_{en} .

$(i, j) \mid E \in a$: c_j and e_i are the word translation pair.

$p(c_j | e_i)$: lexical translation probability.

Language model probability

$$p_{\text{lm}}(\text{ph}_{\text{cn}}) = \sum_{j=0}^{m-n+1} p(c_j | c_{j-n+1} \cdots c_{j-1})$$

Translation quality

$$S_{trans} = p_{lex}(ph_{cn} | ph_{en}, a) \times p_{lm}(ph_{cn})$$

Comprehensive score of a concept or fact


$$S_{com} = \alpha S_{sim} + \beta S_{trans}$$

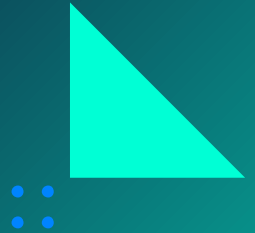
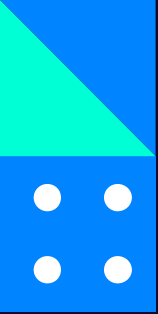
=> The ILP used in generic summarization based on the information fusing method can be employed to obtain a Chinese summary from English documents.



Multilingual Automatic Summarization



- Takes documents on the same topic in different languages as the input and outputs a summary in one of the languages.
 - For example, a Chinese summary can be generated from documents written in a mix of English, Japanese, and Chinese on the same topic.
 - Helpful for a user aiming to acquire global information.
- 



A typical method is to summarize after translation:

- First, the texts of all other languages are translated into a specific language through a machine translation system.
- Then, generic summarization methods are employed to generate the final summary.

Extracting summary sentences from machine translation results leads to poor summarization quality.

A key problem arises: how to make full use of the imperfect machine translation results?



Suppose $V^{en} = \{s_i^{en} | 1 \leq i \leq n\}$ and $V^{cn} = \{s_i^{cn} | 1 \leq i \leq n\}$

denotes sentences in English dataset D^{en} and D^{cn}

n : number of sentences

s_i^{cn} : Chinese translation of s_i^{en}

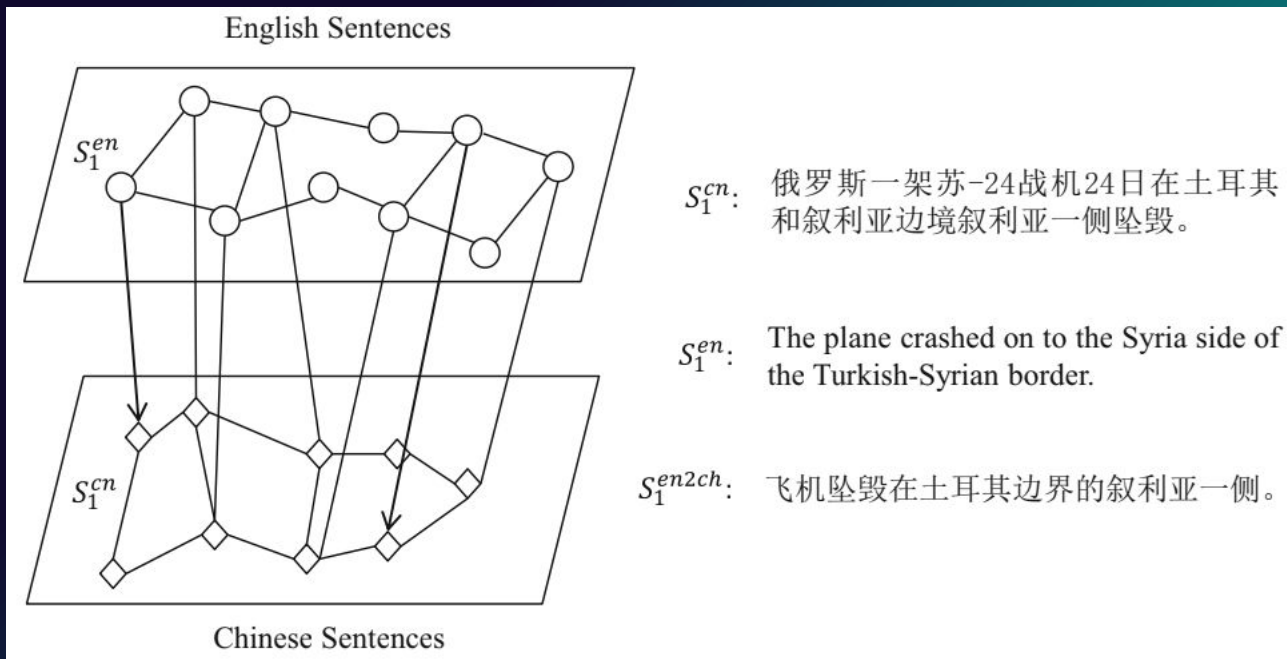
=> Construct an undirected graph $G = (V^{en}, V^{cn}, E^{en}, E^{cn}, E^{en cn})$



E^{en} : the relation between any two sentences in the English documents

E^{cn} : the relation between any two sentences in the Chinese documents

$E^{en cn}$: the relation between any sentence in V^{en} and any sentence in V^{cn}

Multilingual summarization based on the adaptive graph model





How to measure whether s_1^{cn} and s_1^{en2cn} are similar in terms of semantics:

- First, we can calculate the correlation score between s_1^{cn} and s_1^{en2cn} with the cosine similarity. If the result exceeds a prespecified threshold, these two sentences are considered to be similar to each other.
- Second, we can check if s_1^{cn} entails s_1^{en2cn} with the textual entailment method, where s_1^{cn} and s_1^{en2cn} are similar in semantics if the entailment stands.
- Third, we can also determine whether the natural Chinese sentence s_1^{cn} is a translation of the English sentence S^{en}

=> This method not only avoids choosing English sentences similar to natural Chinese sentences but also retains complementary but important English sentences by including their Chinese translations as part of the final summary.






07. Summary Quality Evaluation and Evaluation Workshops





Summary Quality Evaluation Methods



- More intractable problem because there is no perfect summary in theory.
 - For the same document or document set, different people will generate different summaries.
 - Divided into manual methods and automatic approaches.
- 



Manual evaluation methods



Focused on the following six aspects:

- grammaticality
- non-redundancy
- referential clarity
- focus
- structure
- coherence

Each score ranged from 1 to 5, in which 1 represents the worst and 5 the best.

However, the scores given by different experts in a manual evaluation are quite divergent.

=> To solve the problem, use the pyramid method






Pyramid method

The summary content unit (SCU) is the key concept, as it represents the important semantic unit at the subsentence level of the summary.

Different summaries may share the same SCUs, even though the words used maybe quite different.

Summary generated by one system:

- First step is to manually analyze the SCUs of the system summary
 - Second step is to calculate the sum of the scores of all SCUs in the reference summary
 - Third step is to calculate the ratio of the system summary score to an ideal summary score and take the ratio as the evaluation score for the system summary.
- 

A ₂ :	In the 2016 US presidential election, Trump defeated Hillary Clinton and was elected.
B ₄ :	He won the 45th US presidential election.
C ₃ :	Trump became the 45th US president.
D ₁ :	The 2016 US election was full of suspense and Trump finally won the election.

SCU ₁ :	Trump was elected the 45th US president
	A ₂ : Trump defeated Hillary Clinton and was elected
	B ₄ : He won the 45th US presidential election
	C ₃ : Trump became the 45th US president
	D ₁ : Trump finally won the election
SCU ₂ :	The US held a presidential election in 2016.
	A ₂ : In the 2016 US presidential election
	D ₁ : The 2016 US election

For an m -tier pyramid, T_i denotes the i -th layer, and all SCUs in T_i have weights of i . That is, these SCUs are mentioned in i reference summaries. $|T_i|$ denotes the number of SCUs in T_i .

$$\text{Score}_{\text{ideal}} = \sum_{i=j+1}^m i \times |T_i| + j \times \left(X - \sum_{i=j+1}^m |T_i| \right)$$

$$j = \max_k \left(\sum_{t=k}^m |T_t| \geq X \right)$$

After comparison between the automatic summary result and the reference summary, it is found that D_i SCUs at tier T_i appear in the system summary, i.e., D_i SCUs in the system summary are mentioned by i reference summaries.

$$\text{Score}_{\text{sys}} = \sum_{i=1}^m i \times D_i$$



Final evaluation score of the system summary becomes $\text{Score}_{\text{sys}}/\text{Score}_{\text{ideal}}$.

The pyramid method can minimize the influence of the differences in reference summaries.

However, manual evaluation always takes considerable time and human resources.

For example, it is reported that in a summary evaluation of the DUC, it takes approximately 3000 h to manually evaluate the quality of the summaries.



Automatic Evaluation Methods

- Two categories:
 - Intrinsic evaluation: directly evaluates the quality of the summary results.
 - Extrinsic evaluation: indirectly evaluates the summary quality by analyzing the performance of the downstream tasks that rely on the results of the summarization.
- Intrinsic evaluation methods can also be divided into two categories:
 - Form metrics focus on the grammar, coherence, and organizational structure of the summaries.
 - Content metrics focus on summary content and information, which are believed to be the most important in automatic evaluation.

ROUGE - Recall-Oriented Understudy for Gisting Evaluation

$$\text{ROUGE-}n(\text{sum}, r) = \frac{\sum_{n\text{-gram} \in r} \text{count}_{\text{match}}(n\text{-gram}, \text{sum})}{\sum_{n\text{-gram} \in r} \text{count}(n\text{-gram})}$$

n : number of words contained in the matching unit

$\text{count}_{\text{match}}(n\text{-gram}, \text{sum})$: the maximum number of times that n -gram appears in both reference summary r and system summary sum .

If there are multiple reference summaries $R = \{r_0, r_1, \dots, r_m\}$, ROUGE- n can be calculated by comparing the system summary with each reference summary, and the highest matching score is taken as the final result:

$$\text{ROUGE-}n_{\text{multi}}(\text{sum}) = \max_{r \in R} \text{ROUGE-}n(\text{sum}, r)$$

ROUGE-L calculates the matching rate of common substrings. The basic idea is that the longer the common substring is, the more similar the two sentences are.

$$\text{ROUGE-}L(\text{sum}) = \frac{(1 + \beta^2) R_{\text{LCS}} P_{\text{LCS}}}{R_{\text{LCS}} + \beta^2 P_{\text{LCS}}}$$

$$R_{\text{LCS}} = \frac{\sum_{i=1}^u \text{LCS}(s_i, \text{sum})}{\sum_{i=1}^u |s_i|}$$

$$P_{\text{LCS}} = \frac{\sum_{i=1}^u \text{LCS}(s_i, \text{sum})}{|\text{sum}|}$$

$\text{LCS}(s_i, \text{sum})$: the longest common substring of s_i and sum .
 $|s_i|$: the length of the reference summary
 $|\text{sum}|$: the length of the system summary



ROUGE-S is a special case of ROUGE-n ($n=2$), which is also called the skip bigram matching rate.



$$\text{ROUGE-S}(\text{sum}) = \frac{(1 + \beta^2) R_S P_S}{R_S + \beta^2 P_S}$$


RS and PS are the recall and precision of the skip bigram, respectively.





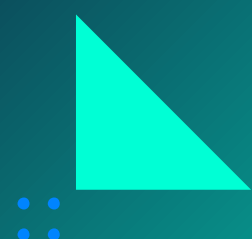
Evaluation Workshops



- The organizer releases the training data to the participants, which they can use to train their summarization systems.
 - The organizer releases the test data to the participants and asks participants to submit the outputs of their system before a prespecified deadline.
 - The organizer will evaluate the submitted summaries using both manual and automatic evaluation methods.
 - The participants are then invited to introduce their systems and communicate with each other at the evaluation workshop.
- 




The DUC evaluation workshop



NIST started the DUC in 2001, which was launched by Professor Daniel Marcu from the University of Southern California. The main task of the DUC is to evaluate the development trends in text summarization technology.

The TAC evaluation workshop



Since 2008, the TAC has organized four evaluation tasks, including text summarization, automatic question answering, text entailment, and knowledge base population.



The MSE evaluation workshop

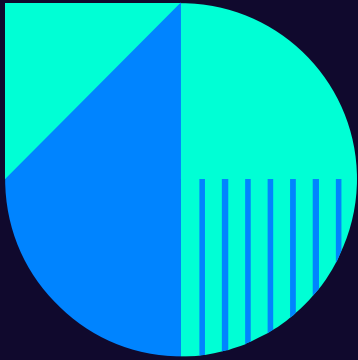


The ACL organized the MSE workshop in 2005 and 2006. The organizers provided a collection of texts on the same topic in Arabic and English, requiring the system to submit an English summary of fewer than 100 words.

The NLPCC evaluation workshop

The NLPCC conference has been organizing evaluation workshops on Chinese automatic summarization since 2015. In 2015 and 2017, the NLPCC organized an evaluation task of single document news summarization.





THANKS

Do you have any questions?

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon and infographics & images by Freepik

