Text Summarization for Bond Statements

Alastair Blake Doggett

Department of Mathematics,
Courant Institute,
NYU

07/19/2016

Introduction

- Automatic summarization is a key task in natural language processing;
- The problem of automatic text summarization can be formulated as follows (taken from Wikipedia): *Automatic summarization is the process of reducing a text document with a computer program in order to create a summary that retains the most important part of the original document.*

Introduction

- Automatic summarization is a key task in natural language processing;
- The problem of automatic text summarization can be formulated as follows (taken from Wikipedia): *Automatic summarization is the process of reducing a text document with a computer program in order to create a summary that retains the most important part of the original document.*

Introduction

Automatic text summarization tasks can be divided into categories based on several criteria,

- **Extractive Summarization:** Selecting a subset of the most important sentences;
- **Abstractive Summarization:** Paraphrasing the most important content;
- **Single Document Summarization:** Working over a single document;
- **Multi-document Summarization:** Working over multiple documents.

Introduction

Automatic text summarization tasks can be divided into categories based on several criteria,

- **Extractive Summarization:** Selecting a subset of the most important sentences;
- **Abstractive Summarization:** Paraphrasing the most important content;
- **Single Document Summarization:** Working over a single document;
- **Multi-document Summarization:** Working over multiple documents.

Introduction

Automatic text summarization tasks can be divided into categories based on several criteria,

- **Extractive Summarization:** Selecting a subset of the most important sentences;
- **Abstractive Summarization:** Paraphrasing the most important content;
- **Single Document Summarization:** Working over a single document;
- **Multi-document Summarization:** Working over multiple documents.

Introduction

Automatic text summarization tasks can be divided into categories based on several criteria,

- **Extractive Summarization:** Selecting a subset of the most important sentences;
- **Abstractive Summarization:** Paraphrasing the most important content;
- **Single Document Summarization:** Working over a single document;
- **Multi-document Summarization:** Working over multiple documents.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Extraction vs Abstraction

Extraction

- Majority of summarization systems are extractive;
- Many resources available;
- Rich in open source software;
- Lends itself to modular architecture;

Abstraction

- Hard to develop;
- Very few works exist;
- Ultimately, this is the aim of automatic text summarization.

Problem

Problem

For any given bond statement, generate a condensed and relevant summary.

Problem

Conjecture

***Single document extractive summarization* can be used to produce condensed and relevant summaries of bond statements.**

Overview of the Pipeline

An automatic summarization process can be divided into three steps,

1. **Pre-processing:** A structured representation of the original text is obtained;
2. **Selection:** An algorithm transforms the text structure into a summary structure;
3. **Generation:** The final summary is obtained from the summary structure.

Selection

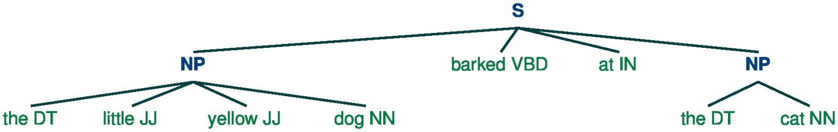
Once pre-processing is complete, a *summarization method* must be chosen. These can be split into two types:

1. **Shallow approaches:** Restrict to a snytactic level of representation and try to extract salient parts of the text in an efficient way;
2. **Deeper approaches:** Assume a semantic level of representation and involve linguistic processing at some level.

Methods involved in deeper approaches include: part-of-speech (POS) tagging, noun phrase (NP) chunking.

Deeper Approach Example

1. **Consider the sentence:** *sent = 'The little yellow dog barked at the cat'*
2. **POS tagging:** *tagged_sent = nltk.pos_tag(nltk.word_tokenize(sent)) = [('The', 'DT'), ('little', 'JJ'), ('yellow', 'JJ'), ('dog', 'NN'), ('barked', 'VBD'), ('at', 'IN'), ('the', 'DT'), ('cat', 'NN')]*
3. **Define a tag pattern:** *pattern = "NP: {<DT>?<JJ>*<NN>}"* (translated: optional determiner (DT) followed by any number of adjectives (JJ) and then a noun (NN)).
4. **Create a chunk parser:** *NPChunker = nltk.RegexpParser(pattern)*
5. **Parse the sentence:** *result = NPChunker.parse(sent)*



Selection

Shallow approaches are simpler to implement and lend themselves well to extractive summarization and to trainable machine learning algorithms.

Pre-processing

Aim: Reduce dimensionality of the representation space to make the document easier to manage.

- Data cleaning,
  - Assumed plain text format;
  - Remove info-boxes, tables, lists etc.
- Lexicon reduction,

$$\text{text} \longrightarrow \text{segments, sentences, paragraphs} \longrightarrow \text{tokens, stopwords.}$$

- Vector space model (VSM) to represent each sentence as an N-dimensional vector (Google's word2vec). This will allow us to endow a *metric* upon sentences;
- (Optional) Filter summarizable statements with respect to a precomposed compression rate,

$$\tau = \frac{|\text{Summary}|}{|\text{Source}|} \in [\text{tol}_{\text{lb}}, \text{tol}_{\text{ub}}] .$$

Pre-processing

Aim: Reduce dimensionality of the representation space to make the document easier to manage.

- Data cleaning,
  - Assumed plain text format;
  - Remove info-boxes, tables, lists etc.
- Lexicon reduction,

$$\text{text} \longrightarrow \text{segments, sentences, paragraphs} \longrightarrow \text{tokens, stopwords.}$$

- Vector space model (VSM) to represent each sentence as an N-dimensional vector (Google's word2vec). This will allow us to endow a *metric* upon sentences;
- (Optional) Filter summarizable statements with respect to a precomposed compression rate,

$$\tau = \frac{|\text{Summary}|}{|\text{Source}|} \in [\text{tol}_{\text{lb}}, \text{tol}_{\text{ub}}] .$$

Pre-processing

Aim: Reduce dimensionality of the representation space to make the document easier to manage.

- Data cleaning,
  - Assumed plain text format;
  - Remove info-boxes, tables, lists etc.
- Lexicon reduction,

$$\text{text} \longrightarrow \text{segments, sentences, paragraphs} \longrightarrow \text{tokens, stopwords.}$$

- Vector space model (VSM) to represent each sentence as an N-dimensional vector (Google's word2vec). This will allow us to endow a *metric* upon sentences;
- (Optional) Filter summarizable statements with respect to a precomposed compression rate,

$$\tau = \frac{|\text{Summary}|}{|\text{Source}|} \in [\text{tol}_{\text{lb}}, \text{tol}_{\text{ub}}] .$$

Pre-processing

Aim: Reduce dimensionality of the representation space to make the document easier to manage.

- Data cleaning,
  - Assumed plain text format;
  - Remove info-boxes, tables, lists etc.
- Lexicon reduction,

$$\text{text} \longrightarrow \text{segments, sentences, paragraphs} \longrightarrow \text{tokens, stopwords.}$$

- Vector space model (VSM) to represent each sentence as an N-dimensional vector (Google's word2vec). This will allow us to endow a *metric* upon sentences;
- (Optional) Filter summarizable statements with respect to a precomposed compression rate,

$$\tau = \frac{|\text{Summary}|}{|\text{Source}|} \in [\text{tol}_{\text{lb}}, \text{tol}_{\text{ub}}] .$$

Machine Learning Approach

ML approach can be envisaged if we have a collection $\mathcal{D}$ consisting of statements and their corresponding reference extractive summaries;

Problem: What if the reference summaries are not extractive? What if they are author-provided? How do we get our hands on the desired collection $\mathcal{D}$?

Solution: We can produce a size-K reference extractive summary consisting of the K most similar sentences to the author-provided summary.

Machine Learning Approach

- The summarization task can be seen as a two-classification problem;
- If $s \in \mathcal{S}$ is a sentence, then

$$\text{label}(s) = \begin{cases} 1 & \text{if } s \text{ belongs to the reference extractive summary} \\ 0 & \text{else;} \end{cases}$$

- The trainable summarizer is expected to *learn* the patterns which lead to the summaries by identifying *feature values* of sentences that are highly correlated to either 1 or 0;
- When a new statement is given to the system, the learned patterns are used to classify each sentence of that statement into either class 1 or class 0;
- The sentences labeled 1 are collated and an extractive summary is formed.

Machine Learning Approach

- The summarization task can be seen as a two-classification problem;
- If $s \in \mathcal{S}$ is a sentence, then

$$\text{label}\left(s\right) = \begin{cases} 1 & \text{if } s \text{ belongs to the reference extractive summary} \\ 0 & \text{else;} \end{cases}$$

- The trainable summarizer is expected to *learn* the patterns which lead to the summaries by identifying *feature values* of sentences that are highly correlated to either 1 or 0;
- When a new statement is given to the system, the learned patterns are used to classify each sentence of that statement into either class 1 or class 0;
- The sentences labeled 1 are collated and an extractive summary is formed.

Machine Learning Approach

- The summarization task can be seen as a two-classification problem;
- If $s \in \mathcal{S}$ is a sentence, then

$$\text{label}\,(s) = \begin{cases} 1 & \text{if } s \text{ belongs to the reference extractive summary} \\ 0 & \text{else;} \end{cases}$$

- The trainable summarizer is expected to *learn* the patterns which lead to the summaries by identifying *feature values* of sentences that are highly correlated to either 1 or 0;
- When a new statement is given to the system, the learned patterns are used to classify each sentence of that statement into either class 1 or class 0;
- The sentences labeled 1 are collated and an extractive summary is formed.

Machine Learning Approach

- The summarization task can be seen as a two-classification problem;
- If $s \in \mathcal{S}$ is a sentence, then

$$\text{label}(s) = \begin{cases} 1 & \text{if } s \text{ belongs to the reference extractive summary} \\ 0 & \text{else;} \end{cases}$$

- The trainable summarizer is expected to *learn* the patterns which lead to the summaries by identifying *feature values* of sentences that are highly correlated to either 1 or 0;
- When a new statement is given to the system, the learned patterns are used to classify each sentence of that statement into either class 1 or class 0;
- The sentences labeled 1 are collated and an extractive summary is formed.

Machine Learning Approach

- The summarization task can be seen as a two-classification problem;
- If $s \in \mathcal{S}$ is a sentence, then

$$\text{label}\,(s) = \begin{cases} 1 & \text{if } s \text{ belongs to the reference extractive summary} \\ 0 & \text{else;} \end{cases}$$

- The trainable summarizer is expected to *learn* the patterns which lead to the summaries by identifying *feature values* of sentences that are highly correlated to either 1 or 0;
- When a new statement is given to the system, the learned patterns are used to classify each sentence of that statement into either class 1 or class 0;
- The sentences labeled 1 are collated and an extractive summary is formed.

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Feature Values

A crucial issue in this framework is how to obtain the relevant set of features. A large variety can be found in text-summarization literature,

- **Sentence length:** Penalize sentences that are too short, since these are no expected to belong to the summary;
- **Sentence position:** Start / end of a document generally contain what the document is / was about;
- **Term frequency:** Multiple appearances of a term have a high probability of being important content;
- **Cue-phrases:** Phrases like 'conclusion' or 'in particular' are often followed by important information;
- **Key word extraction:** The statistic tf-idf (term frequency - inverse document frequency) reflects how important a word is to a document;
- **Similarity to title:** Employing VSM we can use the title of the document as a 'query' against all the sentences in the document, the similarity of which, can be computed using an appropriate metric;
- **Sentence centrality:** The centrality of a sentence implies its similarity to other sentences in the document. If a sentence has higher centrality, it is likely to be about an important topic in the document;

Machine Learning Approach