HO CHI MINH CITY UNIVERSITY OF SCIENCE

FACULTY: INFORMATION TECHNOLOGY

PROJECT REPORT

COURSE: DATA MINING AND APPLICATIONS

HOMEWORK: PROJECT #1

CLASS: 19KHDL

INSTRUCTOR: Mr. NGUYEN THAI VU

STUDENT: LU THE VY **ID: 19127009**

STUDENT: MAC VAN HUNG **ID: 19127416**

HO CHI MINH CITY - APRIL 2022

MEMBER INFORMATION

Name	ID	Tasks	Finished (%)
Lu The Vy	19127009	- Demo - Results obtained, evaluated results	**100**
Mac Van Hung	19127416	- Summarize background knowledge learned from the topic.	**100**

TABLE OF CONTENTS

MEMBER INFORMATION ***2***

TABLE OF CONTENTS ***3***

Section 1: Knowledge summary **4**

Section 2: Demo **4**

Flow-chart 4

Prepare Python libraries 5

Implement utility functions 5

Create a dictionary for the word frequency 5

Create a dictionary contains the frequency of words in a document 6

Create a dictionary for the TF score value 6

Create a dictionary contains the IDF score value 6

Create a dictionary contains the TFIDF score value 6

Create a dictionary contains the sentence score value 7

Calculate the average sentence score from the sentence score value dictionary 7

Generate final result: summary from source document, based on sentence score and thresold 7

Section 3: Result **9**

Section 4: References **13**

Extraction-based text summarization using TFIDF method: 13

Section 1: Background knowledge learned

1. Introduction

- Automatic text summarization aims to compress the input texts and generate a condensed summary.
- 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.
  - **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.
  - **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.
  - **Generic summarization:** summarizes the main points of the original author
  - **Query-based summarization:** content closely related to the user's query.
  - **Extraction-based summarization**: produces summaries by extracting important sentences from the original text
  - **Compression-based summarization**: 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

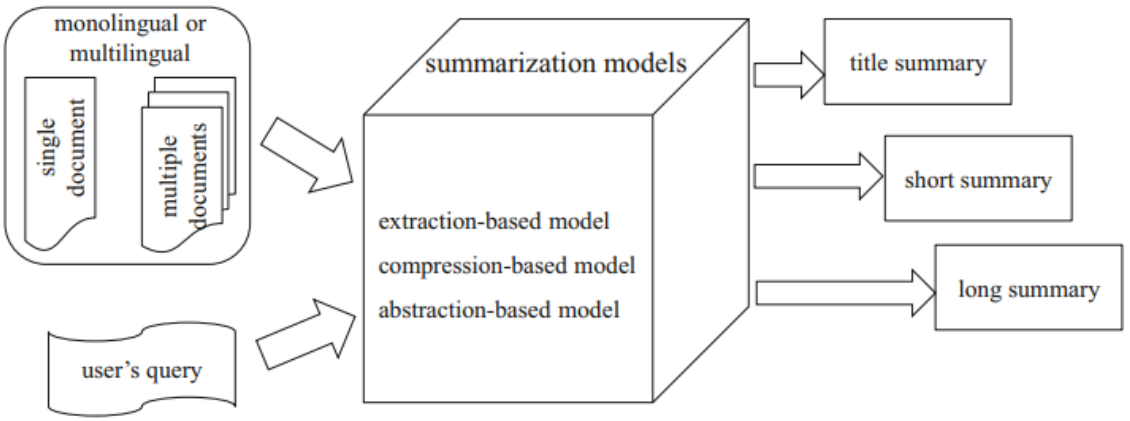


Fig. 11.1 The basic framework of automatic text summarization

- **Application:**
  + 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

2. **Extraction-Based Summarization**
   - 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.

A) Word frequency-based algorithm

$$\text{Score}\left(S_{i_j}\right) = \frac{\sum_{w_k \in S_{i_j}} \text{Score}\left(w_k\right)}{\left|\left\{w_k | w_k \in S_{i_j}\right\}\right|}$$

$$\text{Score}\left(w_k\right) = \text{tf}_{w_k} = \frac{\text{count}\left(w_k\right)}{\sum_w \text{count}\left(w\right)}$$

Score **Sij** denotes the importance score of the j -th sentence in the i-th document
Count(wk) denotes the number of times it appears in the whole document Di.
Sigma w count(w) is the occurrence of all words in the entire document.
Score (wk) is usually called the term frequency (TF)
Inverse document frequency (IDF)

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

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

TFIDF (term-frequency inverse document frequency)

$$\text{Score}\left(w_k\right) = \text{tf\_idf}_{w_k} = \text{tf}_{w_k} \times \text{idf}_{w_k}$$

B) Graph-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.

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

j denotes the position of the sentence **Sij** in the document and n represents the number of sentences in the document.

C) Graph-based algorithm

- **PageRank:**

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

- **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 = (Vi, Vj) indicates that it can jump from web page Vi to web page Vj .

- **For a node Vi, In (Vi)** denotes the set of pages linked to Vi, and |In (Vi)| is the in-degree of Vi. Out(Vi) denotes the web pages that Vi can link to, and |Out(Vi)| is the out-degree of Vi.
- **d** $\in$ **[0, 1]** is the damping factor that assigns Vi a prior probability of jumping to any other node Vj from Vi. d is usually set to 0.85 in page ranking.

- **LexRank:**

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

- The directed graph G (V,E) becomes the undirected graph, and each edge of the graph e = (Vi, Vj) carries a weight Wij, 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** ∈ **Vi, Vj** denotes a word that simultaneously occurs in Vi and Vj . Given a weighted undirected graph G (V,E), the importance score of each node (sentence) is calculated by the following formula:

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

- **Vj** ∈ **adj(Vi)** denotes the set of nodes adjacent to Vi, i.e., the set of nodes with edges linked to Vi

- **TextRank:**
- 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

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

- Where **|{wk|wk ∈ Vi & wk ∈ Vj}|** denotes the number of words co-occurring in two sentences; |Vi| and |Vj| are the number of words in sentences Vi and Vj .

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

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

R is the set of all sentences, A denotes the set of selected summary sentences, Q stands for the user query, si denotes any sentence in the set of unselected sentences, and sj means any sentence in the set of selected sentences.
Sim2(si, sj) is the similarity between two sentences si and sj . λ is a hyperparameter that weighs correlation and redundancy. The larger lambda is, the more si's relevance

to the user query is emphasized; otherwise, redundancy is more emphasized.

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

3. 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.
- **Source model**

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

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

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

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

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

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

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

- **Channel Model**

$$\begin{aligned}p_{\text{expand\_tree}}(s|t_1) = {} & p_{\text{exp}}(G \to HA|G \to HA) \cdot p_{\text{exp}}(A \to CBD|A \to CD) \cdot \\ & p_{\text{cfg}}(B \to QR|B) \cdot p_{\text{cfg}}(Q \to Z|Q) \cdot p_{\text{cfg}}(Z \to c|Z) \cdot \\ & p_{\text{cfg}}(R \to d|R)\end{aligned}$$

- **Search the best compressed sentence**

$$\hat{p}_{\text{compress\_tree}}(t_1|s) = \frac{\hat{p}_{\text{tree}}(t_1) \cdot p_{\text{expand\_tree}}(s|t_1)}{\hat{p}_{\text{tree}}(s)}$$

- **Model parameter training**
  1. **The rule deduction probability of context-free grammars, e.g., Pcfg (G → H A|G);**
  2. **The tree structure extension probability, e.g., pexp (A → CBD|A → CD));**
  3. **The bigram language model probability, e.g., pbigram (b|a)**

  => To estimate the above three kinds of probabilities, parse the training data {sk , tk} K k=1 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

- **Model decoding**

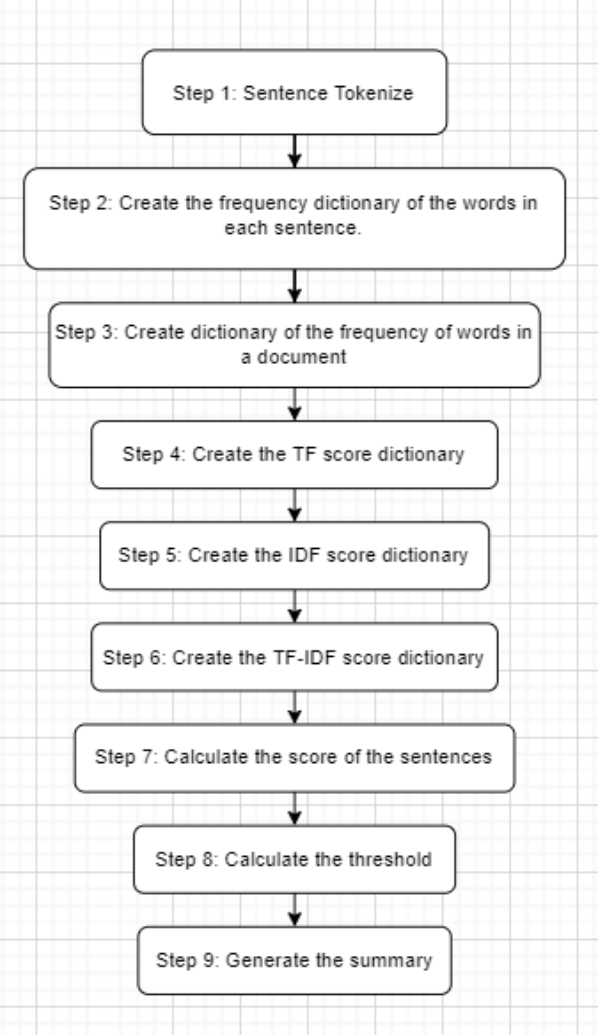
  - For the original sentence s whose tree structure has n child nodes, there are two choices for each node in the process of sentence compression: delete or retain.

  - Therefore, there are (2n − 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

Section 2: Demo

- In this section, we will implement extraction-based text summarization using TF-IDF score (sentence importance estimation) calculation in Python, from scratch.
- Python has many function libraries to in the process of text summarization. We used libraries such as: nltk (for tokenizing sentence or word, transforming words), nltk.corpus (for identifying stop-words), math (for calculating TF-IDF-TFIDF value)

1. Flow-chart



2. Prepare Python libraries

```
import math
import nltk
# nltk.download('stopwords')
from nltk import sent_tokenize, word_tokenize, PorterStemmer
from nltk.corpus import stopwords
```

- PorterStemmer: transforms a word to its original form, i.e: “cars” -> “car”

3. Implement utility functions

a. Create a dictionary for the word frequency

- Pseudo-code:

```
DEFINE FUNCTION create_frequency_dict(sentences):
    SET freq_dict TO {}
    SET stop_words TO set(stopwords.words("english"))
    SET ps TO PorterStemmer()

    FOR sentence IN sentences:
        SET freq_table TO {}
        SET words TO word_tokenize(sentence)

        FOR word IN words:
            SET word TO word.lower()
            SET word TO ps.stem(word)
            IF word IN stop_words:
                continue
            IF word IN freq_table:
                freq_table[word] += 1
            ELSE:
                SET freq_table[word] TO 1
        SET freq_dict[sentence[:15]] TO freq_table

    RETURN freq_dict
```

b. Create a dictionary contains the frequency of words in a document

- Pseudo-code:

```
DEFINE FUNCTION count_words_per_doc(freq_dict):
    SET words_per_doc_dict TO {}

    FOR sentence, f_dict IN freq_dict.items():
        FOR word, count IN f_dict.items():

            IF word IN words_per_doc_dict:
                words_per_doc_dict[word] += 1
            ELSE:
                SET words_per_doc_dict[word] TO 1

    RETURN words_per_doc_dict
```

c. Create a dictionary for the TF score value

- Pseudo-code:

```
DEFINE FUNCTION create_tf_dict(freq_dict):
    tf_dict= {}

    FOR sentence, f_dict IN freq_dict.items():
        SET value_dict TO {}
        SET count_words_in_sentence TO len(f_dict)

        FOR word, count IN f_dict.items():
            SET value_dict[word] TO count / count_words_in_sentence
        SET tf_dict[sentence] TO value_dict

    RETURN tf_dict
```

d. Create a dictionary contains the IDF score value

- Pseudo-code:

```
DEFINE FUNCTION create_idf_dict(freq_dict, count_words_per_doc, document_length):
    SET idf_dict TO {}
    FOR sentence, f_dict IN freq_dict.items():
        SET value_dict TO {}

        FOR word IN f_dict.keys():
            SET value_dict[word] TO math.log10(document_length /
float(count_words_per_doc[word]))
        SET idf_dict[sentence] TO value_dict

    RETURN idf_dict

```

e. Create a dictionary contains the TFIDF score value

- Pseudo-code:

```
DEFINE FUNCTION create_tfidf_dict(tf_dict, idf_dict):
    SET tfidf_dict TO {}

    FOR (sentence1, f_dict1), (sentence2, f_dict2) IN zip(tf_dict.items(),
idf_dict.items()):
        SET value_dict TO {}

        FOR (word1, value1), (word2, value2) IN zip(f_dict1.items(), f_dict2.items()):
            SET value_dict[word1] TO float(value1 * value2)

        SET tfidf_dict[sentence1] TO value_dict

    RETURN tfidf_dict

```

f. Create a dictionary contains the sentence score value

- Pseudo-code:

```
DEFINE FUNCTION calc_sentence_score(tfidf_dict):
    SET sentence_score_dict TO {}

    FOR sentence, f_dict IN tfidf_dict.items():
        SET sentence_score TO 0
        SET count_words_in_sentence TO len(f_dict)

        FOR word, score IN f_dict.items():
            sentence_score += score

        SET sentence_score_dict[sentence] TO sentence_score / count_words_in_sentence
    RETURN sentence_score_dict

```

g. Calculate the average sentence score from the sentence score value dictionary

- Pseudo-code:

```
DEFINE FUNCTION calc_average_score(sentence_score_dict):
    SET total TO 0

    FOR entry IN sentence_score_dict:
        total += sentence_score_dict[entry]

    SET average TO (total / len(sentence_score_dict))

    RETURN average

```

h. Generate final result: summary from source document, based on sentence score and thresold

- Pseudo-code:

```
DEFINE FUNCTION generate_summary(sentences, sentence_score_dict, threshold):
    SET sentence_count TO 0
    SET summary TO ""

    FOR sentence IN sentences:
        IF sentence[:15] IN sentence_score_dict and sentence_score_dict[sentence[:15]] >=
(threshold):
            summary += " " + sentence
            sentence_count += 1

    RETURN summary
```

Section 3: Result

- We took a document as an example for the text summarization:

  " Those Who Are Resilient Stay In The Game Longer

  "On the mountains of truth you can never climb in vain: either you will reach a point higher up today, or you will be training your powers so that you will be able to climb higher tomorrow."—Friedrich Nietzsche

  Challenges and setbacks are not meant to defeat you, but promote you. However, I realise after many years of defeats, it can crush your spirit and it is easier to give up than risk further setbacks and disappointments. Have you experienced this before? To be honest, I don't have the answers. I can't tell you what the right course of action is; only you will know. However, it's important not to be discouraged by failure when pursuing a goal or a dream, since failure itself means different things to different people. To a person with a Fixed Mindset failure is a blow to their self-esteem, yet to a person with a Growth Mindset, it's an opportunity to improve and find new ways to overcome their obstacles. Same failure, yet different responses. Who is right and who is wrong? Neither. Each person has a different mindset that decides their outcome. Those who are resilient stay in the game longer and draw on their inner means to succeed.

I've coached many clients who gave up after many years toiling away at their respective goal or dream. It was at that point their biggest breakthrough came. Perhaps all those years of perseverance finally paid off. It was the 19th Century's minister Henry Ward Beecher who once said: "One's best success comes after their greatest disappointments." No one knows what the future holds, so your only guide is whether you can endure repeated defeats and disappointments and still pursue your dream. Consider the advice from the American academic and psychologist Angela Duckworth who writes in Grit: The Power of Passion and Perseverance: "Many of us, it seems, quit what we start far too early and far too often. Even more than the effort a gritty person puts in on a single day, what matters is that they wake up the next day, and the next, ready to get on that treadmill and keep going."

I know one thing for certain: don't settle for less than what you're capable of, but strive for something bigger. Some of you reading this might identify with this message because it resonates with you on a deeper level. For others, at the end of their tether the message might be nothing more than a trivial pep talk. What I wish to convey irrespective of where you are in your journey is: NEVER settle for less. If you settle for less, you will receive less than you deserve and convince yourself you are justified to receive it.

"Two people on a precipice over Yosemite Valley" by Nathan Shipps on Unsplash

Develop A Powerful Vision Of What You Want

"Your problem is to bridge the gap which exists between where you are now and the goal you intend to reach."—Earl Nightingale

I recall a passage my father often used growing up in 1990s: "Don't tell me your problems unless you've spent weeks trying to solve them yourself." That advice has echoed in my mind for decades and became my motivator. Don't leave it to other people or outside circumstances to motivate you because you will be let down every time. It must come from within you. Gnaw away at your problems until you solve them or find a solution. Problems are not stop signs, they are advising you that more work is required to overcome them. Most times, problems help you gain a skill or develop the resources to succeed later. So embrace your challenges and develop the grit to push past them instead of retreat in resignation. Where are you settling in your life right now? Could you be you playing for bigger stakes than you are? Are you willing to play bigger even if it means repeated failures and setbacks? You should ask yourself these questions to decide whether you're willing to put yourself on the line or settle for less. And that's fine if you're content to receive less, as long as you're not regretful later.

If you have not achieved the success you deserve and are considering giving up, will you regret it in a few years or decades from now? Only you can answer that, but you should carve out time to discover your motivation for pursuing your goals. It's a fact, if you don't know what you want you'll get what life hands you and it may not be in your best interest, affirms author Larry Weidel: "Winners know that if you don't figure out what you want, you'll get whatever life hands you." The key is to develop a powerful vision of what you want and hold

that image in your mind. Nurture it daily and give it life by taking purposeful action towards it.

Vision + desire + dedication + patience + daily action leads to astonishing success. Are you willing to commit to this way of life or jump ship at the first sign of failure? I'm amused when I read questions written by millennials on Quora who ask how they can become rich and famous or the next Elon Musk. Success is a fickle and long game with highs and lows. Similarly, there are no assurances even if you're an overnight sensation, to sustain it for long, particularly if you don't have the mental and emotional means to endure it. This means you must rely on the one true constant in your favour: your personal development. The more you grow, the more you gain in terms of financial resources, status, success—simple. If you leave it to outside conditions to dictate your circumstances, you are rolling the dice on your future.

So become intentional on what you want out of life. Commit to it. Nurture your dreams. Focus on your development and if you want to give up, know what's involved before you take the plunge. Because I assure you, someone out there right now is working harder than you, reading more books, sleeping less and sacrificing all they have to realise their dreams and it may contest with yours. Don't leave your dreams to chance. "

- Depend on the value (called 'x') multiply with threshold (image below), the length of summary will be changed

```
summary = generate_summary(sentences, sentence_scores, 1.3 * threshold)
```

- For example, with **1.3** * threshold, the result is:

```
 Have you experienced this before? Who is right and who is wrong? Neither. It was at that point their biggest breakt
hrough came. Perhaps all those years of perseverance finally paid off. It must come from within you. Where are you s
ettling in your life right now? Could you be you playing for bigger stakes than you are? So become intentional on wh
at you want out of life. Commit to it. Nurture your dreams.
```

- For example, with **1.5** * threshold, the result is:

```
 Have you experienced this before? Who is right and who is wrong? Neither. It must come from within you. Where are y
ou settling in your life right now? Could you be you playing for bigger stakes than you are? Commit to it. Nurture y
our dreams.
```

- For example, with **0.5** * threshold, the result is:

```
 However, I realise after many years of defeats, it can crush your spirit and it is easier to give up than risk furt
her setbacks and disappointments. Have you experienced this before? To be honest, I don't have the answers. I can't
tell you what the right course of action is; only you will know. However, it's important not to be discouraged by fa
ilure when pursuing a goal or a dream, since failure itself means different things to different people. To a person
with a Fixed Mindset failure is a blow to their self-esteem, yet to a person with a Growth Mindset, it's an opportun
ity to improve and find new ways to overcome their obstacles. Same failure, yet different responses. Who is right an
d who is wrong? Neither. Each person has a different mindset that decides their outcome. Those who are resilient sta
y in the game longer and draw on their inner means to succeed. I've coached many clients who gave up after many year
s toiling away at their respective goal or dream. It was at that point their biggest breakthrough came. Perhaps all
those years of perseverance finally paid off. Some of you reading this might identify with this message because it r
esonates with you on a deeper level. For others, at the end of their tether the message might be nothing more than a
trivial pep talk. What I wish to convey irrespective of where you are in your journey is: NEVER settle for less. If
you settle for less, you will receive less than you deserve and convince yourself you are justified to receive it. D
on't leave it to other people or outside circumstances to motivate you because you will be let down every time. It m
ust come from within you. Gnaw away at your problems until you solve them or find a solution. Problems are not stop
signs, they are advising you that more work is required to overcome them. Most times, problems help you gain a skill
or develop the resources to succeed later. So embrace your challenges and develop the grit to push past them instead
of retreat in resignation. Where are you settling in your life right now? Could you be you playing for bigger stakes
than you are? Are you willing to play bigger even if it means repeated failures and setbacks? You should ask yoursel
f these questions to decide whether you're willing to put yourself on the line or settle for less. And that's fine i
f you're content to receive less, as long as you're not regretful later. If you have not achieved the success you de
serve and are considering giving up, will you regret it in a few years or decades from now? Only you can answer tha
t, but you should carve out time to discover your motivation for pursuing your goals. Nurture it daily and give it l
ife by taking purposeful action towards it. Vision + desire + dedication + patience + daily action leads to astonish
ing success. Are you willing to commit to this way of life or jump ship at the first sign of failure? I'm amused whe
n I read questions written by millennials on Quora who ask how they can become rich and famous or the next Elon Mus
k. Success is a fickle and long game with highs and lows. Similarly, there are no assurances even if you're an overn
```

- **Conclusion:** user can change the final result (summary) length by changing the value of '*x multiply threshold*'. The bigger '*x multiply threshold*' is, the shorter the summary is.

Section 4: References

1. Extraction-based text summarization using TFIDF method:

- Credit to:
  - NLP — Text Summarization using NLTK: TF-IDF Algorithm | by Akash Panchal | Towards Data Science
  - Process Text using TFIDF in Python | by Shivangi Sareen | Towards Data Science
  - How to Build a Text Summarizer from Scratch? | by Anjaneya Tripathi | Spider | Medium