

Narzędzie do tworzenia podsumowań tekstu

Magdalena Gazda, Maciej Król, Wiktor Kuś

Abstrakt

W naszym raporcie opisujemy skrypt, który generuje podsumowanie zadanego przez użytkownika tekstu. Pozwala on zaoszczędzić sporo czasu wybierając tylko najważniejsze informacje z podanych artykułów. Przedstawimy schemat działania oraz efekt implementacji naszego programu.

Wstęp

Celem naszego projektu jest stworzenie narzędzia do tworzenia podsumowań tekstów. Zadaniem narzędzia jest zredukowanie tekstu tak, aby zawierał wszystkie najistotniejsze aspekty, bez utracenia znaczenia tekstu.

Część Teoretyczna

Zajmujemy się zagadnieniem sumaryzacji, czyli stworzeniu tekstu, który zawiera tylko kluczowe informacje. Takim zabiegiem redukujemy znacząco ilość początkowego tekstu, a znaczenie całości pozostaje bez zmian.

Wyróżniamy metody ekstrakcyjną i abstrakcyjną, które pozwalają nam generować streszczenia tekstów.

Metoda abstrakcyjna - wybiera słowa oparte na rozumieniu semantycznym. Generuje więc nowe zadania, które mogą nie być częścią początkowego tekstu.

Abstractive Summarization



Metoda ekstrakcyjna - wyodrębnia podstawowe słowa z oryginalnego tekstu, a następnie łączy je tworząc podsumowanie.



Część Praktyczna

W naszym programie zdecydowaliśmy się użyć metody ekstrakcyjnej. Przedstawiamy kod źródłowy i opis poszczególnych funkcji.

Kod importuje niezbędne biblioteki i moduły, takie jak os, nltk, numpy i networkx.

```
1  import os
2  import nltk
3  from nltk.corpus import stopwords
4  from nltk.cluster.util import cosine_distance
5  import numpy as np
6  import networkx as nx
7
```

Funkcja - `read_article(file_name)` odczytuje plik tekstowy, a następnie dzieli jego zawartość na zdania i przetwarza każde zdanie na listę słów.

```
7
8  ✓ def read_article(file_name):
9      with open(file_name, "r") as file:
10         filedata = file.read()
11
12         sentences = filedata.replace("[^a-zA-Z]", " ").split(". ")
13         if sentences[-1] == '':
14             sentences.pop()
15
16         processed_sentences = [sentence.split() for sentence in sentences]
17         return processed_sentences
```

Funkcja - `sentence_similarity(sent1, sent2, stop_words)`, oblicza podobieństwo kosinusowe między dwoma zdaniami oraz pomija powszechne słowa stopu, aby porównanie było bardziej efektywne.

```
18
19  ✓ def sentence_similarity(sent1, sent2, stop_words=[]):
20      sent1 = [w.lower() for w in sent1 if w.lower() not in stop_words]
21      sent2 = [w.lower() for w in sent2 if w.lower() not in stop_words]
22
23      all_words = list(set(sent1 + sent2))
24      vector1 = [sent1.count(word) for word in all_words]
25      vector2 = [sent2.count(word) for word in all_words]
26
27      return 1 - cosine_distance(vector1, vector2)
```

Funkcja - `build_similarity_matrix(sentences, stop_words)` tworzy macierz podobieństwa, porównując każde zdanie z tekstu z każdym innym zdaniem.

```
28
29  ✓ def build_similarity_matrix(sentences, stop_words):
30      similarity_matrix = np.zeros((len(sentences), len(sentences)))
31      for idx1, sentence1 in enumerate(sentences):
32          for idx2, sentence2 in enumerate(sentences):
33              if idx1 != idx2:
34                  similarity_matrix[idx1][idx2] = sentence_similarity(sentence1, sentence2, stop_words)
35      return similarity_matrix
36
```

Funkcja - generate_summary(file_name, top_n=5) wykorzystuje Narzędzie do Przetwarzania Języka Naturalnego (nltk), aby filtrować słowa stopu, generuje streszczenie najlepszych n zdań na podstawie ich ważności, ustalonej za pomocą algorytmu PageRank zastosowanego do macierzy podobieństwa.

```
36
37  ✓ def generate_summary(file_name, top_n=5):
38      nltk.download("stopwords")
39      stop_words = set(stopwords.words('english'))
40      sentences = read_article(file_name)
41      similarity_matrix = build_similarity_matrix(sentences, stop_words)
42      similarity_graph = nx.from_numpy_array(similarity_matrix)
43      scores = nx.pagerank(similarity_graph)
44      ranked_sentences = sorted(((scores[i], s) for i, s in enumerate(sentences)), reverse=True)
45      summary = [' '.join(sent) for _, sent in ranked_sentences[:top_n]]
46      return ' '.join(summary)
47
```

Funkcja - list_txt_files_in_folder() wyświetla wszystkie pliki .txt w bieżącym katalogu.

```
48  ✓ def list_txt_files_in_folder():
49      txt_files = [f for f in os.listdir() if f.endswith(".txt")]
50      if not txt_files:
51          print("No TXT files found in the current folder.")
52      else:
53          print("Available TXT files:")
54          for idx, file in enumerate(txt_files):
55              print(f"{idx + 1}. {file}")
56      return txt_files
57
```

Funkcja - select_file_to_generate_summary(txt_files) prosi użytkownika o wybranie pliku z listy plików tekstowych do streszczenia.

```
58  ✓ def select_file_to_generate_summary(txt_files):
59      while True:
60          try:
61              choice = int(input("Enter the number corresponding to the TXT file you want to summarize (0 to exit): "))
62              if choice == 0:
63                  return None
64              elif choice < 1 or choice > len(txt_files):
65                  print("Please enter a valid number.")
66              else:
67                  return txt_files[choice - 1]
68          except ValueError:
69              print("Invalid input. Please enter a number.")
```

Główna Pętla wywołuje `list_txt_files_in_folder()` aby wyświetlić dostępne pliki tekstowe. Prosi użytkownika o wybranie pliku do streszczenia, a następnie generuje i wyświetla streszczenie wybranego pliku za pomocą funkcji `generate_summary`.

```
72     while True:
73         txt_files = list_txt_files_in_folder()
74         if not txt_files:
75             break
76
77         selected_file = select_file_to_generate_summary(txt_files)
78         if selected_file is None:
79             break
80
81         summary = generate_summary(selected_file, 2)
82         print("\nSummary:", summary)
```

Przykładowe działanie programu:

Na początku program wyświetla wszystkie pliki txt znajdujące się w folderze z programem.

```
Available TXT files:
1. fb.txt
2. msft.txt
3. trump.txt
Enter the number corresponding to the TXT file you want to summarize (0 to exit):
```

Po wpisaniu konkretnego numeru generowane jest podsumowanie:

```
Summary: But Facebook also assumed extraordinary power over the personal information of its 2 billion users - control it has wielded with little transparency or outside oversight. Facebook allowed Microsoft's Bing search engine to see the names of virtually all Facebook user's friends without consent, the records show, and gave Netflix and Spotify the ability to read Facebook users' private messages.. For years, Facebook gave some of the world's largest technology companies more intrusive access to users' personal data than it has disclosed, effectively exempting those business partners from its usual privacy rules, according to internal records and interviews
Available TXT files:
1. fb.txt
2. msft.txt
3. trump.txt
Enter the number corresponding to the TXT file you want to summarize (0 to exit):
```

Dla porównania jeszcze jeden zrzut ekranu, na górze znajduje się pełny tekst, a poniżej jego podsumowanie:

```
For years, Facebook gave some of the world's largest technology companies more intrusive access to users' personal data than it has disclosed, effectively exempting those business partners from its usual privacy rules, according to internal records and interviews. The specifics detailed in hundreds of pages of Facebook documents obtained by The New York Times. The records, generated in 2017 by the company's internal audits, provide the most complete picture yet of the social network's data-sharing practices. They also underscore how personal data has become a commodity of the digital age, traded on a vast scale by some of the most powerful companies in Silicon Valley and beyond. The exchange of data for explosive growth, Facebook got more users, lifting its advertising revenue. Partner companies acquired features to make their products more appealing to Facebook users connected with friends across different devices and websites. But Facebook also assumed extraordinary power over the personal information of its 2 billion users - control it has wielded with little transparency or outside oversight. Facebook allowed Microsoft's Bing search engine to see the names of virtually all Facebook user's friends without consent, the records show, and gave Netflix and Spotify the ability to read Facebook users' private messages.

But Facebook also assumed extraordinary power over the personal information of its 2 billion users - control it has wielded with little transparency or outside oversight. Facebook allowed Microsoft's Bing search engine to see the names of virtually all Facebook user's friends without consent, the records show, and gave Netflix and Spotify the ability to read Facebook users' private messages. For years, Facebook gave some of the world's largest technology companies more intrusive access to users' personal data than it has disclosed, effectively exempting those business partners from its usual privacy rules, according to internal records and interviews
```

Jak widać podsumowanie jest ponad 2 razy krótsze od pełnego tekstu.

Podsumowanie

Pomyślnie udało się stworzyć narzędzie do generowania podsumowań tekstu. Program działa poprawnie, zwraca skrócony tekst, który posiada najważniejsze informacje, które zawierają się w tekście. Skrypt działa w prawidłowy sposób dla różnych przesłanych tekstów. W przyszłości program można byłoby rozszerzyć o nowe funkcjonalności, między innymi: dostępne inne języki, użycie sztucznej inteligencji.

Bibliografia

<https://github.com/edubey/text-summarizer>

<https://www.abstractivehealth.com/extractive-vs-abstractive-summarization-in-healthcare>