Narzędzie do generacji tytułów naukowych

Czaja Emilia

Godfryd Jakub

Krasowska Anna

Cel projektu

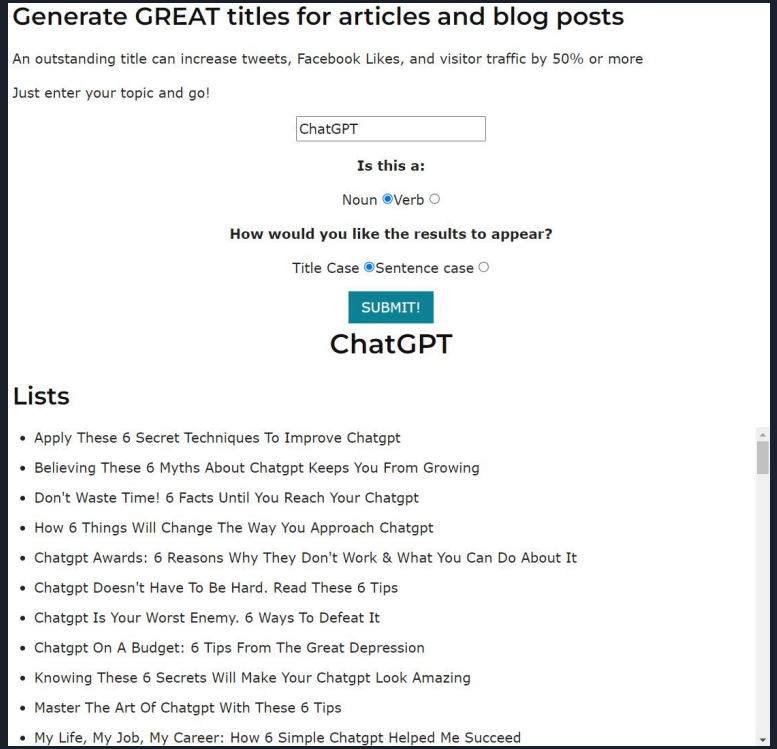
Celem narzędzia jest pomoc naukowcom w odkrywaniu istotnych elementów prac naukowych oraz inspirowania się do tworzenia własnych publikacji, poprzez generowanie tytułów artykułów naukowych opartych na słowach kluczowych za pomocą sztucznej inteligencji i architektury GPT.

Czym jest GPT-3?

GPT-3 (Generative Pretrained Transformer 3) to najnowocześniejszy model sztucznej inteligencji do przetwarzania języka opracowany przez OpenAI. Jest w stanie generować tekst podobny do ludzkiego i ma szeroki zakres zastosowań, w tym tłumaczenie języków, modelowanie języka i generowanie tekstu dla aplikacji takich jak chatboty. Jest to jeden z największych i najpotężniejszych jak dotąd modeli AI do przetwarzania języka.

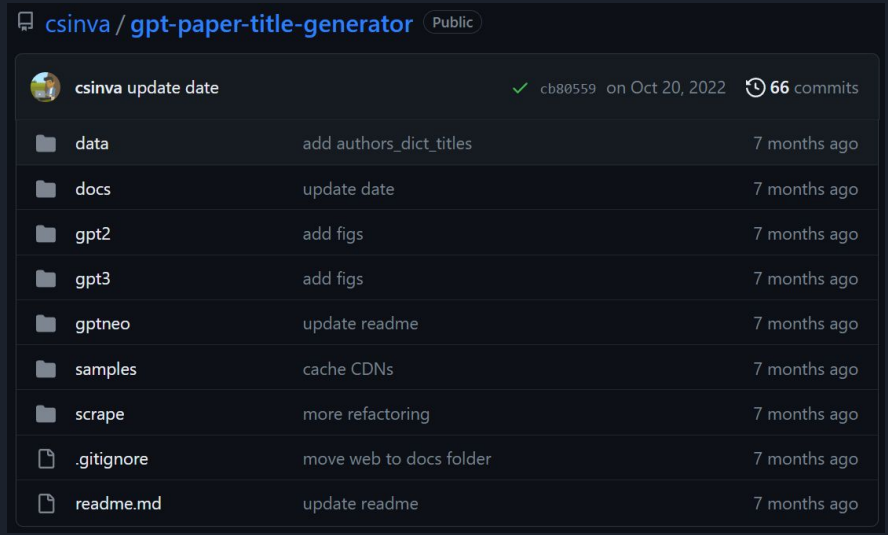
Generowanie tytułów online

Są strony internetowe, które generują tytuły do różnych tekstów, takich jak artykuły naukowe, blogi lub opowiadania. Przykłady to Portent's Content Idea Generator, Title Generator by Tweak Your Biz, Content Row Link Bait Title Generator i SEOPressor Blog Title Generator. Strony te korzystają z algorytmów sztucznej inteligencji, a tytuły są generowane na podstawie słów kluczowych lub tematów tekstu. Strony te analizują tekst i generują tytuł podobny do istniejących tytułów lub tworzą nowy tytuł, który opisuje wskazany temat.



Generowanie tytułów wykorzystując GPT-3

Projekt csinva/gpt-paper-title-generator wykorzystuje GPT-3 do generowania tytułów naukowych z różnych dziedzin. Repozytorium zawiera dane, modele i kod źródłowy, który pozwala na wygenerowanie tytułów na podstawie zadanych tematów i autorów. Model korzysta z pięciu ostatnich tytułów autorów, którzy napisali co najmniej trzy artykuły związane z uczeniem maszynowym w serwisie arXiv.



Funkcja **compute_author_stats**

Ma na celu obliczenie statystyk dotyczących autorów na podstawie metadanych artykułów naukowych pobranych z bazy danych arXiv. Funkcja przyjmuje jako argument ramkę danych (**dataframe_from_arxiv_metadata**), która zawiera metadane artykułów z arXiv. Funkcja zwraca krotkę, która zawiera słownik z informacją o autorach i indeksach ich artykułów (**authors_dict**) oraz posortowaną listę autorów (**pruned_authors_list**).

```python
def compute_author_stats(dataframe_from_arxiv_metadata: DataFrame) -> tuple[dict, list]:
    """
    Oblicza statystyki dotyczące autorów na podstawie podanych metadanych z arXiv.
    :param dataframe_from_arxiv_metadata: DataFrame zawierający metadane artykułów z arXiv.
    :return: Krotka zawierająca słownik z informacją o autorach i indeksach ich artykułów oraz posortowaną listę autorów.
    """
    def get_authors_list(s: str) -> List:
        return [
            a.lower().strip()
            for a in re.split('and |, |\n', s)
        ]

    authors_list_arr = dataframe_from_arxiv_metadata.authors.apply(get_authors_list)
    authors_dict = defaultdict(list)
    for i, l in enumerate(authors_list_arr):
        for author in l:
            authors_dict[author].append(i)

    pruned_authors_list = sorted(set(list(itertools.chain.from_iterable(authors_list_arr))))

    return authors_dict, pruned_authors_list
```

Funkcja **sanitize_author_collections**

Ma na celu oczyszczenie danych autorów, usuwając niepoprawne dane oraz informacje, które nie są związane z autorami. Funkcja ta przyjmuje dwa argumenty: **authors_dict**, który jest słownikiem autorów oraz ich ostatnich pięciu tytułów, i **authors_list**, który jest listą autorów.

```python
def sanitize_author_collections(authors_dict: dict, authors_list: list):
    """
    Usuwa niepoprawne dane (dane, które nie są autorami) z kolekcji autorów.
    :param authors_dict: Słownik autorów.
    :param authors_list: Lista autorów.
    :return: Krotka zawierająca oczyszczone dane słownika i listy autorów.
    """

    keywords = re.compile(r'universi|departm|faculty|institute|global|center|laboratory')

    def author_has_more_articles_than(author, number_of_articles):
        return len(author) > number_of_articles

    def does_not_contain_punctuaction(author):
        return not any(punctuaction in author for punctuaction in string.punctuation)

    def author_has_name_and_surname(author):
        return len(author.split()) > 1 and len(author.split()[0]) > 1

    def author_does_not_contain_keywords(author, keywords):
        return keywords.search(author) is None

    authors_list_clean = [
        author for author in authors_list
        if author_does_not_contain_keywords(author, keywords)
           and does_not_contain_punctuaction(author)
           and not contains_num(author)
           and author_has_more_articles_than(author, 3)
           and author_has_name_and_surname(author)
    ]
    authors_dict_clean = {
        author: authors_dict[author]
        for author in authors_list_clean
    }
    assert len(authors_list_clean) == len(authors_dict_clean)

    return authors_dict_clean, authors_list_clean
```

Funkcja **generate_authors_titles**

Funkcja **generate_authors_titles** jest odpowiedzialna za generowanie tytułów artykułów naukowych dla autorów na podstawie wcześniejszych tytułów autorów, wykorzystując OpenAI API. Jako parametry przyjmuje słownik z informacjami o autorach i ich wcześniejszych tytułach artykułów, funkcję filtrującą autorów na podstawie określonego kryterium, opcjonalnie maksymalną liczbę autorów, dla których zostaną wygenerowane tytuły, oraz opcjonalnie ścieżkę, do której zostaną zapisane wyniki.

```python
def generate_authors_titles(authors_titles_dict, author_filter, iteration_limit=-1, progress_bar=False, path=None):
    assert author_filter
    authors_save = {}
    authors = list(authors_titles_dict.keys())
    filtered_authors = filter_authors(authors, author_filter)
    authors_for_iteration = enumerate(tqdm(filtered_authors)) if progress_bar else enumerate(filtered_authors)
    configure_openai_api(openai_api_settings)
    for i, author in authors_for_iteration:
        if iteration_limit != -1 and i + 1 > iteration_limit:
            break
        if author not in authors_save:
            prompt = create_prompt(author, authors_titles_dict, query_settings, openai_completion_settings["stop"])
            completion = create_openai_completion(prompt, openai_completion_settings)
            authors_save[author] = [completion.choices[i].text for i in range(len(completion.choices))]
    if path is not None:
        save_results(authors_save, path)
    return authors_save
```

Wygenerowane tytuły dla danych autorów

Przykładowy output programu, gdzie znajdują się imię i nazwisko autora oraz pięć wygenerowanych tytułów dla niego:


```
{"yoshua bengio": ["Semi-Supervised Reinforcement Learning with Deep Representation Learning", "Centralized training of deep neural networks with low communication overhead", "Context-aware politeness: A learning-based approach", "Neural Variational Inference and Learning in Boltzmann Machines", "Watch Your Step: Guarding Model-Based Reinforcement Learning from Step-wise"],
"yang liu": ["Models of Explanation for Deep Neural Networks", "Neural Networks for Surrogate Modeling: A Comparative Review", "A Study on Generative Adversarial Nets with Wasserstein Distance", "On the Universality of Deep Learning", "Representation Learning for Time-Series Classification"],
"sergey levine": ["Combining Model-Based and Model-Free Updates in Deep Reinforcement Learning", "A Neural Algorithm of Artistic Style", "Quadrupedal Locomotion by Reinforcement Learning", "RLPy: A Multi-platform Toolkit for Reinforcement Learning", "Efficient Exploration in Deep Reinforcement Learning via Bootstrapped DQN"],
"bo li": ["Segmenting out Task-Independent Representations in Multi-Task Learning", "Measuring Representational Similarity in Deep Neural Networks", "...", "On the Convergence of Gradient Descent for Deep Linear Neural Networks", "On Certified Adversarial Defenses for Deep Neural Networks"],
"pieter abbeel": ["hierarchicalrl: A Framework for Multi-Level Reinforcement Learning", "Investigate Neural Representation for Sequential Decision Making Tasks", "Context-Aware Neural Machine Translation", "A Simple Approach for Time-Varying Graphical Models", "Tracking the World State with Neural Maps"],
```

Ograniczenia projektu

1. Jakość generowanych tytułów - niektóre z tytułów mogą być kreatywne i intrygujące, ale inne mogą być niejasne
2. Brak oryginalności - gdy wiele osób korzysta z tego samego narzędzia to istnieje ryzyko, że tytuły mogą być podobne lub wręcz identyczne
3. Brak merytorycznego wpływu - generator pomaga jedynie w stworzeniu ciekawego i skutecznego tytułu, co nie gwarantuje, że artykuł będzie dobrze napisany lub przyciągnie uwagę czytelników
4. Ograniczenia GPT-3 - model może mieć problem z generowaniem tytułów dla bardzo specjalistycznych dziedzin nauki, lub tematów, które są słabo reprezentowane w danych uczących

Sposoby na dalszy rozwój projektu

1. Wykorzystanie GPT-4
2. Wprowadzenie dodatkowych kryteriów filtrowania autorów
3. Dodanie bardziej zaawansowanych metod oczyszczania danych
4. Można skorzystać z dodatkowych danych o autorze, np. abstrakty poprzednich artykułów naukowych

Podsumowanie

Projekt csinva/gpt-paper-title-generator wykorzystuje architekturę GPT-3 w celu generowania tytułów artykułów naukowych na podstawie wcześniejszych tytułów autorów. Projekt ma na celu pomóc naukowcom w szybszym śledzeniu i zrozumieniu najnowszych publikacji naukowych oraz inspiracji do tworzenia własnych publikacji. Projekt został zrealizowany i dokonano refaktoryzacji kodu.

Dziękujemy za uwagę