Raport projektu: Chatbot do rozmowy o Politechnice Krakowskiej

Konrad Kowalczyk

Abstrakt

Celem projektu jest stworzenie chatbota dostępnego przez Discorda, który umożliwiałby uzyskiwanie informacji o Politechnice Krakowskiej poprzez zastosowanie odpowiednich komend. Moim celem jest umożliwienie zadawania botowi pytań w języku naturalnym, bez konieczności używania konkretnych słów kluczowych. Raport przedstawia podejście do problemu, wyniki, pomysły na dalsze udoskonalenia.

Wstęp

Pobudką do skonstruowania takiego chatbota było duże rozproszenie informacji na stronie Politechniki Krakowskiej. W mojej opinii głównymi problemami portalu uczelni są: podział między strony wydziałów, mnóstwo podstron i mało intuicyjna nawigacja między nimi. Wychodząc naprzeciw potrzebom studentów, którzy zainteresowani są konkretnymi informacjami (kierunki na uczelni, plan zajęć, kontakt itp.) postanowiłem napisać szkielet Chatbota, który będzie umożliwiał konkretne zapytania w języku naturalnym, następnie odpowiadał będzie odpowiednio zescrapowanymi danymi ze strony Politechniki Krakowskiej.

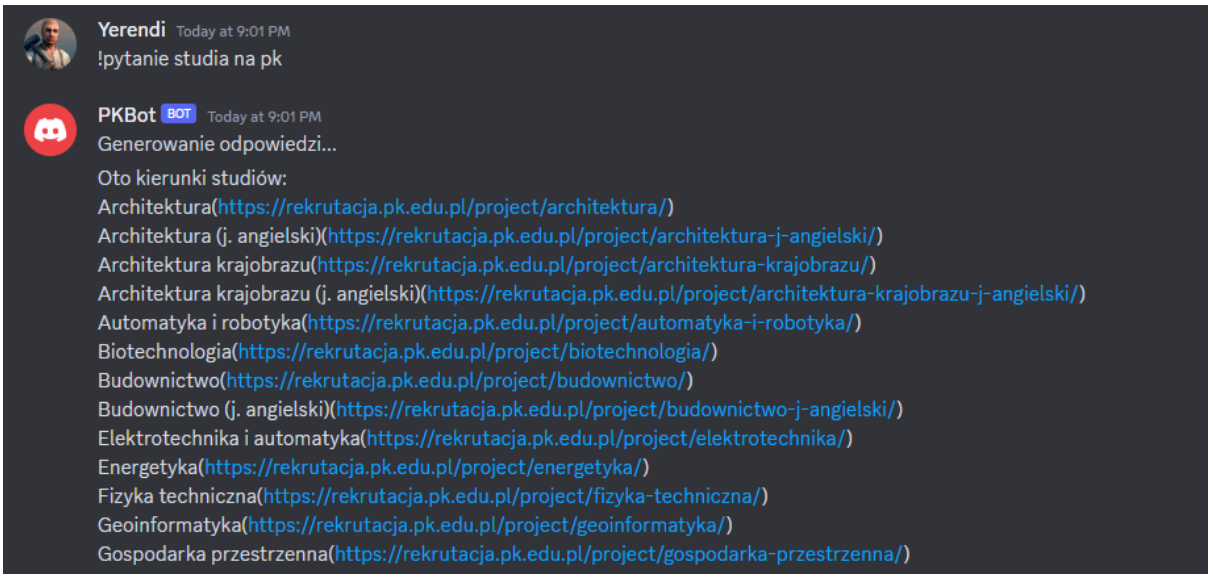
Rozwinięcie

Pisanie Chatbota zacząłem od zdecydowania jakie rzeczy zostaną zrealizowane w ramach projektu na zajęcia, a jakie będę kontynuował po zakończeniu projektu. Ostatecznie zdecydowałem się na dwa: jeden praktyczny, tzn. scrapper kierunków oraz jeden ciekawy - narzędzie podsumowywujące dowolny artykuł z głównej strony Politechniki Krakowskiej przy użyciu API OpenAI.

Scrapping kierunków był prostym zadaniem. Strona rekrutacji PK [4] ma prostą strukturę - wykonuje się żądanie GET do tego adresu przy użyciu biblioteki requests. Następnie, przy użyciu biblioteki BeautifulSoup [3] w pętli iteruje po elementach oznaczonych tagiem "h2" i klasą "et_pb_module_header". Następnie pobieram tekst znajdujący się w tym nagłówku, formatuje otrzymany tekst prostym regexem i doklejam link do strony kierunku. Poniżej kod i odpowiedź bota.

```python
def pobierz_kierunki():
    url = "https://rekrutacja.pk.edu.pl/oferta-edukacyjna/"
    response = requests.get(url)
    if response.status_code != 200:
        return "Wystąpił problem z pobieraniem informacji z witryny Politechniki Krakowskiej."

    soup = BeautifulSoup(response.content, "html.parser")

    wynik = ""
    for a_tag in soup.find_all('h2', {'class': 'et_pb_module_header'}):
        text = a_tag.get_text()
        text = re.sub(r'\s+', ' ', text)
        link = a_tag.find("a")["href"]
        text = f"{text}(<{link}>)\n"
        wynik += text

    return wynik
```

Kod funkcji pobierającej kierunki studiów



Zapytanie bota o kierunki studiów wraz z fragmentem odpowiedzi

Pobieranie artykułów wraz z ich podsumowaniem było trudniejszym zadaniem. Sam scrapping strony głównej PK [5] był nieco trudniejszy. Bez wchodzenia w szczegóły (sam proces został opisany dla kierunków) wklejam kod scrappera:

```python
def pobierz_newsy(liczba_newsy=5):
    response = requests.get(URL)
    if response.status_code != 200:
        return "Wystąpił problem z pobieraniem informacji z witryny Politechniki Krakowskiej."

    soup = BeautifulSoup(response.content, "html.parser")
    newsy = soup.find_all("div", class_="category-title")

    wynik = []
    for idx, news in enumerate(newsy[:liczba_newsy]):
        tytul = news.find("a", class_="mod-articles-category-title").get_text().strip()
        link = "https://www.pk.edu.pl" + news.find("a", class_="mod-articles-category-title")["href"]
        wynik.append(f"{tytul}\nLink: {link}")

    return wynik
```

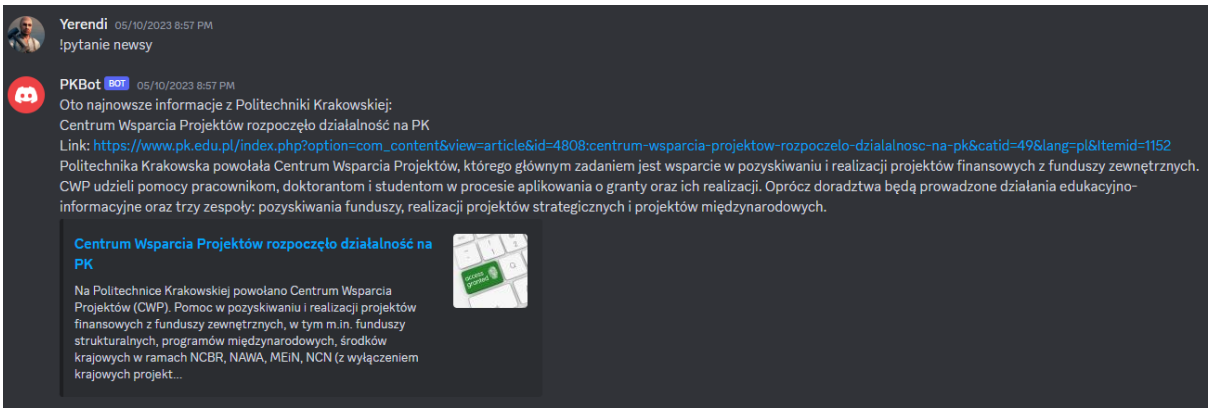
Scrapper newsów

Jak widać na załączonym kodzie, scrapper zwraca listę newsów, nie pojedynczego newsa. Jest to związane z tym, że do odpowiedzi (i wygenerowania podsumowania newsa) wybierany jest jeden losowy, żeby nie każda odpowiedź bota była taka sama.

Pobieranie opisu wymaga odpowiedniego obrobienia tekstu oraz użycia biblioteki OpenAI. Formatowanie tekstu przed wysłaniem jest robione tylko jeżeli jego długość jest większa niż 3000 znaków (limit ten wynika z tego, że odpowiedź od OpenAI przy użyciu modelu text-davinci-003 może mieć tylko 4096 tokenów, a do ilości tych tokenów liczą się również tokeny z prompta), jednak na szczęście w testach systemu wyszło, że artykuły na stronie PK najczęściej nie przekraczają 2000 znaków. Obróbka tekstu jest dość toporna, tzn. obcinane są linki, niepotrzebne znaki i stopwordsy [1] z języka polskiego, w ostateczności tekst jest po prostu ucinany. Następnie przy użyciu odpowiedniego prompta wysyłamy zapytanie do OpenAI i dostajemy odpowiedź [2]. Odpowiedź ta jest następnie przesyłana do bota discordowego [7]. Poniżej prezentuje kod odpowiedzialny za pobieranie opisu, jak również odpowiedź bota na zapytanie o aktualności.

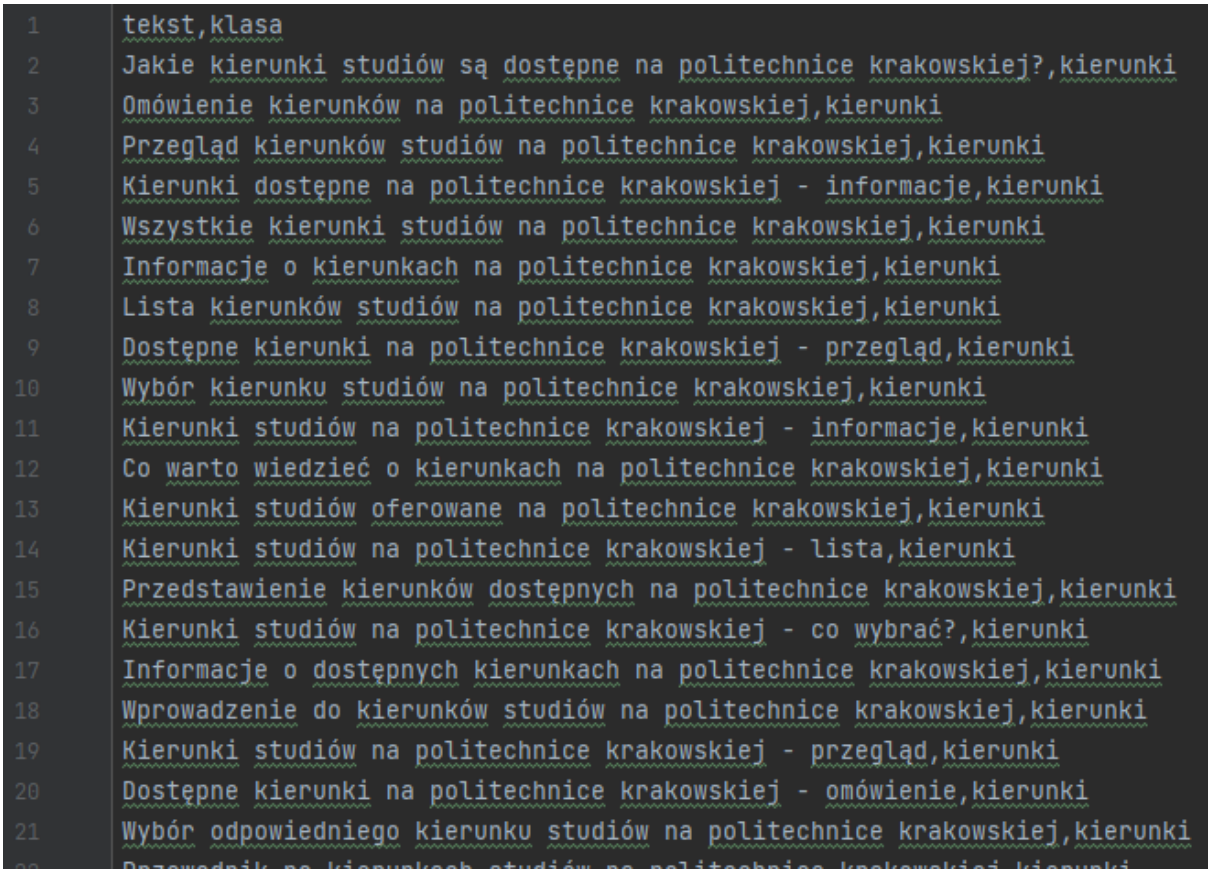
```python
def pobierz_opis(link):
    response = requests.get(link)
    if response.status_code != 200:
        return ""

    soup = BeautifulSoup(response.content, "html.parser")
    opis = soup.find("div", itemprop="articleBody")

    if opis:
        tresc = opis.get_text().strip()

        if len(tresc) > 3000:
            tresc = tresc.lower()
            tresc = re.sub(r'[^\w\s]', '', tresc)
            tresc = re.sub(r"http\S+", "", tresc)
            tokens = tresc.split(' ')
            filtered_text = [word for word in tokens if not word in stop_words]
            tresc = " ".join(filtered_text)

        if len(tresc) > 3000:
            tresc = tresc[:3000]

        prompt = f"Podsumuj artykuł w jak najbardziej zwięzły sposób, ale nie przekraczający 100 słów (lub 300 tokenów)." \
                 f" Odpowiedź wygeneruj w języku polskim. Treść artykułu: {tresc}"
        response = openai.Completion.create(engine="text-davinci-003", prompt=prompt, max_tokens=500, n=1, stop=None,
                                            temperature=0.7)
        podsumowanie = response.choices[0].text.strip()

        return podsumowanie
    else:
        return ""
```

Scrapper artykułu i generowanie podsumowania



Wynik zapytania bota o newsy

Istotną częścią projektu jest również analizowanie wiadomości użytkownika. Bazuje ono na sieci neuronowej uczonej zestawem danych wygenerowanym przez ChatGPT [6]. Fragment zestawu danych oraz kod odpowiedzialny za naukę sieci oraz pobieranie predykcji:



```
tekst,klasa
Jakie kierunki studiów są dostępne na politechnice krakowskiej?,kierunki
Omówienie kierunków na politechnice krakowskiej,kierunki
Przegląd kierunków studiów na politechnice krakowskiej,kierunki
Kierunki dostępne na politechnice krakowskiej - informacje,kierunki
Wszystkie kierunki studiów na politechnice krakowskiej,kierunki
Informacje o kierunkach na politechnice krakowskiej,kierunki
Lista kierunków studiów na politechnice krakowskiej,kierunki
Dostępne kierunki na politechnice krakowskiej - przegląd,kierunki
Wybór kierunku studiów na politechnice krakowskiej,kierunki
Kierunki studiów na politechnice krakowskiej - informacje,kierunki
Co warto wiedzieć o kierunkach na politechnice krakowskiej,kierunki
Kierunki studiów oferowane na politechnice krakowskiej,kierunki
Kierunki studiów na politechnice krakowskiej - lista,kierunki
Przedstawienie kierunków dostępnych na politechnice krakowskiej,kierunki
Kierunki studiów na politechnice krakowskiej - co wybrać?,kierunki
Informacje o dostępnych kierunkach na politechnice krakowskiej,kierunki
Wprowadzenie do kierunków studiów na politechnice krakowskiej,kierunki
Kierunki studiów na politechnice krakowskiej - przegląd,kierunki
Dostępne kierunki na politechnice krakowskiej - omówienie,kierunki
Wybór odpowiedniego kierunku studiów na politechnice krakowskiej,kierunki
```

Zestaw uczący

```python
data = pd.read_csv('dane.csv', sep=',')
X = data['tekst'].values
y = data['klasa'].values

vectorizer = CountVectorizer()
X = vectorizer.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = MultinomialNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)


def znajdz_klase(tekst):
    nowe_dane = vectorizer.transform([tekst])
    predykcja = model.predict(nowe_dane)
    return predykcja[0]
```

Uczenie sieci

Podsumowanie

Projekt udało się zrealizować w planowanym zakresie. Chatbot jest w stanie odpowiadać na pytania w zakresie kierunków i aktualności. Problemem jest mała baza danych do nauki sieci neuronowej, co powoduje, że nie wszystkie zapytania użytkowników spotykają się z prawidłową odpowiedzią.
Kolejne iteracje projektu będą miały na celu wygenerowanie lepszej bazy uczącej dla sieci neuronowej oraz rozwinięcie bota o nowe funkcjonalności (plan zajęć poszczególnych kierunków, kontakt do prowadzących, strony prowadzących itp.)

Bibliografia

[1] Stopwords dla języka polskiego
https://www.kaggle.com/datasets/heeraldedhia/stop-words-in-28-languages (dostęp 13.05.23)
[2] Dokumentacja OpenAI
https://platform.openai.com/docs/introduction (dostęp 13.05.23)
[3] Dokumentacja BeautifulSoup
https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (dostęp 13.05.23)
[4] Kierunki na PK
https://rekrutacja.pk.edu.pl/oferta-edukacyjna/ (dostęp 13.05.23)
[5] Strona główna PK
https://www.pk.edu.pl/ (dostęp 13.05.23)
[6] ChatGPT
https://chat.openai.com/ (dostęp 13.05.23)
[7] Dokumentacja Discorda
https://discord.com/developers/docs (dostęp 13.05.23)