Dokumentacja projektu zaliczeniowego na Studio Projektowe 1: Translacja języka naturalny na Workflowy

Łukasz Chmielewski
Maciej Adamus

Styczeń 2024

Spis treści

1 Wprowadzenie

Celem projektu było stworzenie oprogramowania bazującego na sekwencyjnej sieci neuronowej przeznaczonego do pretwarzania języka naturalnego. Program na podstawie dostarczonego tekstu opisujacego listę kroków ma za zadanie utworzyć graf przepływu pracy.

2 Przebieg Prac

Pracę rozpoczęliśmy od analizy tematu oraz przeglądu dostępnych źródeł, starając się zdobyć jak najwięcej informacji o sekwencyjnych sieciach neuronowych, takich jak RNN, LSTM i podobnych. Następnym krokiem było stworzenie szkieletu sieci i zebranie danych niezbędnych do jej wytrenowania. Jednak ten etap okazał się wyjątkowo trudny z powodu specyfiki wymaganych danych. Nie znaleźliśmy w dostępnych źródłach zbioru danych, który byłby adekwatny do naszego problemu. W związku z tym zdecydowaliśmy się samodzielnie wygenerować odpowiedni zbiór danych, wykorzystując do tego API OpenAI. Dzięki odpowiednim zapytaniom udało nam się uzyskać zbiór składający się z około 2,5 tysiąca rekordów danych wraz z pożądanymi etykietami. Niestety, ta ilość okazała się niewystarczająca do skutecznego wyszkolenia sieci sekwencyjnej, która w trakcie nauki wymagała znacznie większej ilości danych, niż byliśmy w stanie zapewnić. Wobec tego zdecydowaliśmy się wykorzystać jedną z gotowych sieci przetwarzających język naturalny. Po przeprowadzeniu testów na różnych modelach, wybraliśmy model GPT-3.5 od OpenAI, który osiągał wyniki zadowalające nasze oczekiwania.

3 Stos technologiczny

3.1 Python

Językiem którym posługiwaliśmy się podczas pisania aplikacji jest Python. Wybraliśmy go m. in. ze względu na fakt, iż jest on językiem wiodącym w tematyce sztucznej inteligencji. Ponadto posiada łatwe w obsłudze narzędzia z których skorzystaliśmy do stworzenia graficznego interfejsu użytkownika oraz dobrze udokumentowane API do korzystania z modeli udostępnionych przez OpenAI.

3.2 API OpenAI

W projekcie korzystaliśmy z API OpenAI, które umożliwia dostęp do zaawansowanych modeli przetwarzających język naturalny. W naszym przypadku, głównym modelem był GPT-3,5. Jest to zaawansowany model językowy stworzony przez OpenAI, oparty na architekturze GPT-3.

3.3 Mermaid

Mermaid to narzędzie umożliwiające tworzenie diagramów i grafów w łatwy i czytelny sposób. Wykorzystywane jest przede wszystkim do reprezentacji grafów przepływu pracy, co idealnie koresponduje z celem naszego projektu. Mermaid oferuje czytelny, deklaratywny język do definiowania struktury grafów, co sprawia, że jest łatwe w użyciu nawet dla osób niebędących ekspertami w dziedzinie tworzenia diagramów.

3.4 Post Processing

Po wygenerowaniu grafu przepływu pracy za pomocą API OpenAI, następuje etap post-processingu, którego celem jest analiza i weryfikacja poprawności uzyskanego grafu. W tym celu opracowaliśmy skrypt w języku Python, który wykonuje następujące kroki:

- **Parsowanie krawędzi grafu:** Skrypt wyodrębnia krawędzie z opisu grafu w języku Mermaid, usuwając zbędne komentarze i formatowanie.
- **Wykrywanie cykli:** Analizowany jest graf pod kątem obecności cykli, które mogą wskazywać na nieskończone pętle w procesie.
- **Sprawdzanie wierzchołków z wieloma krawędziami:** Identyfikowane są wierzchołki, które mają więcej niż jedną wychodzącą krawędź, co sugeruje obecność warunków typu "if".
- **Wykrywanie równoległych ścieżek:** Sprawdzane jest, czy w grafie istnieją równoległe ścieżki, które mogą wskazywać na współbieżne procesy.

Dzięki powyższym krokom możliwe jest wykrycie potencjalnych błędów i zapewnienie, że wygenerowany graf jest logicznie poprawny i zgodny z założeniami projektu.

3.5 Flask - Interfejs graficzny użytkownika (GUI)

Do stworzenia webowego interfejsu graficznego użytkownika (GUI) skorzystaliśmy z frameworka Flask. Flask to lekki framework do tworzenia aplikacji webowych w języku Python.

4 Architektura systemu

Głównym elementem systemu jest serwer HTTP bazujący na technologii flask. Po otrzymaniu od użytkownika formularza zawierającego zapytanie, jest ono przetwarzane a następnie przesyłane przy pomocy API do OpenAI. Po uzyskaniu i przetworzeniu odpowiedzi jest ona przesyłana do użytkownika, gdzie jest wyswietlana w formie grafu.

5 Instalacja i uruchomienie

Aby uruchomić aplikację trzeba mieć zainstalowego Pythona.

5.1 Instalacja wymaganych bibliotek

Użyj następującego polecenia do zainstalowania wymaganych bibliotek:

```
1    pip install -r nat2wf/requirements.txt
2
```

5.2 Uruchomienie Lokalne Aplikacji

Aby uruchomić aplikację lokalnie z użyciem Flask, użyj poniższego polecenia:

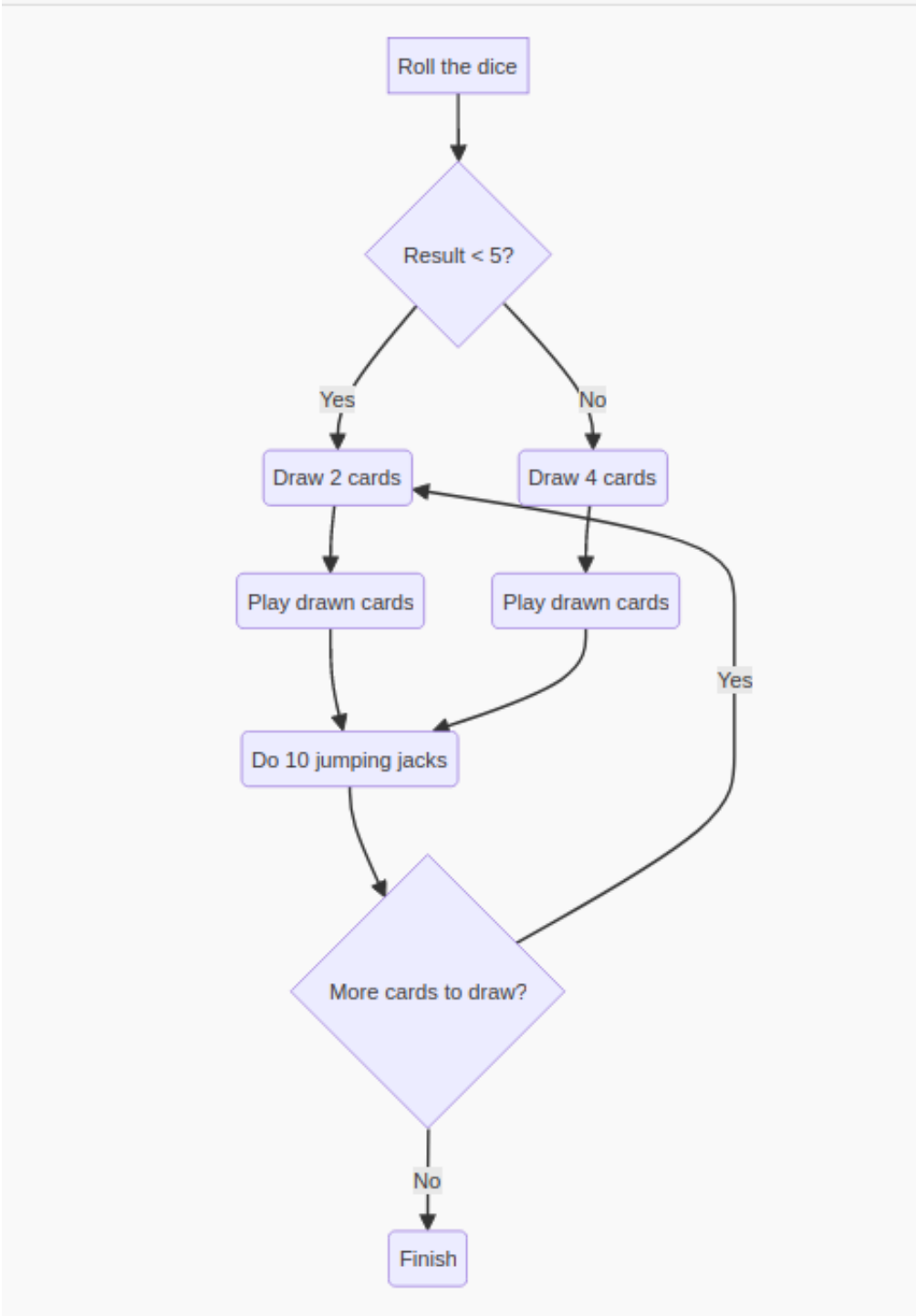
```
1    python -m flask --app flaskr run --debug
2
```

6 Wyniki

6.1 Przykłady wygenerowanych grafów

Korzystanie z aplikacji jest banalnie proste. Po uruchomieniu wystarczy wpisać w dostępne pole tekstowe scenariusz, który chcemy przetworzyć.

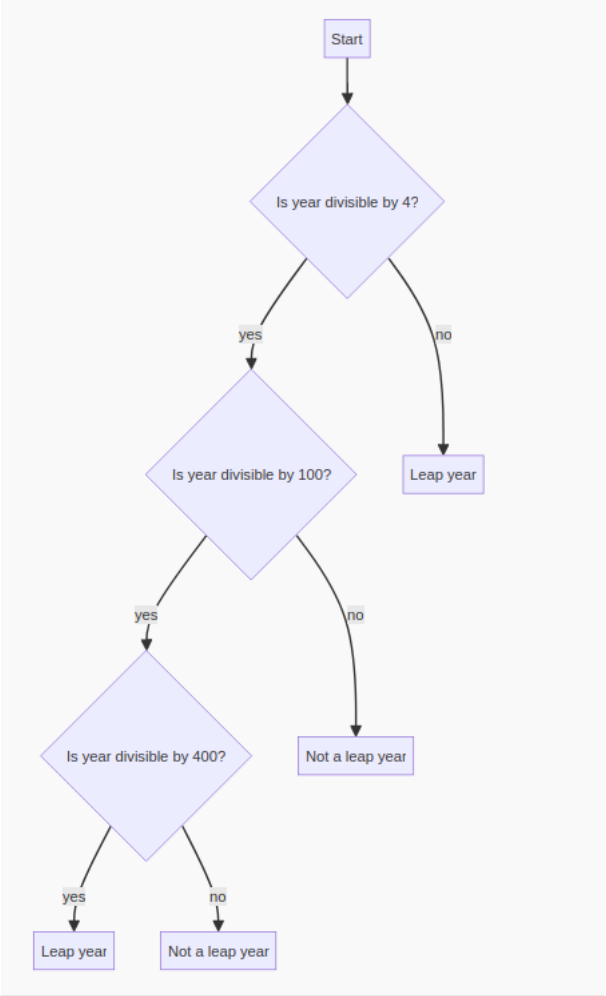
6.2 Przykłady wejścia i wyjścia



Graph Analysis Results:

- Contains cycle: Yes
- Contains 'if' conditional (vertex with multiple edges): Yes
- Contains parallel paths: Yes

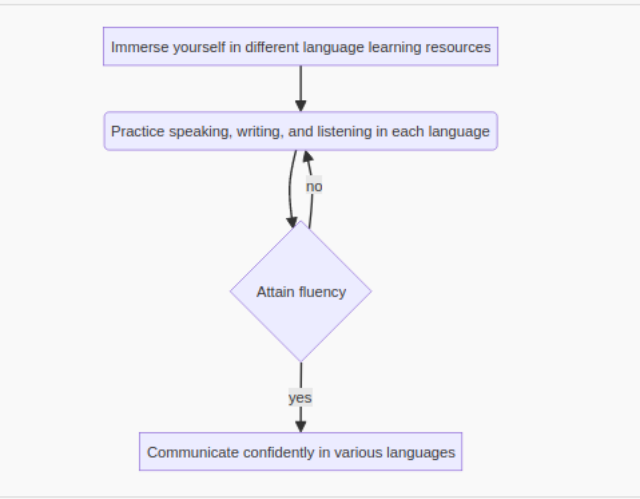
Rysunek 1: Graf uzyskany dla wejścia: *Roll the dice. If the result is less than 5, draw 2 cards, else draw 4 cards. Play drawn cards and do 10 jumping jacks.*



Graph Analysis Results:

- Contains cycle: No
- Contains 'if' conditional (vertex with multiple edges): Yes
- Contains parallel paths: No

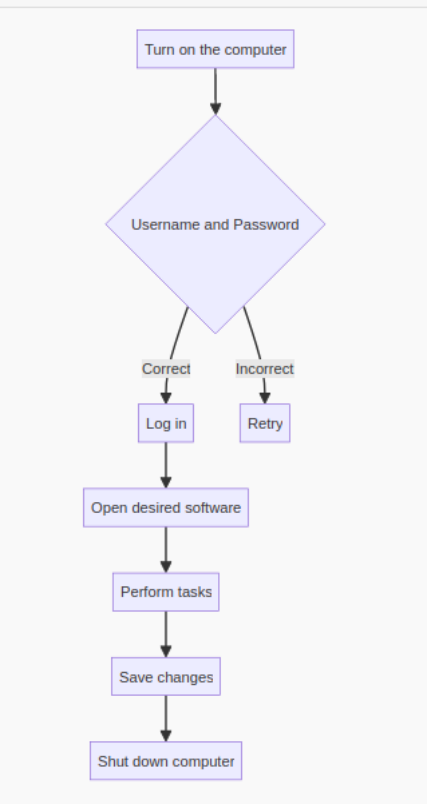
Rysunek 2: Graf uzyskany dla wejścia: *Check if a given year is a leap year or not.*



Graph Analysis Results:

- Contains cycle: Yes
- Contains 'if' conditional (vertex with multiple edges): Yes
- Contains parallel paths: No

Rysunek 3: Graf uzyskany dla wejścia: *Immerse yourself in different language learning resources. Practice speaking, writing, and listening in each language. Attain fluency and communicate confidently in various languages.*



Graph Analysis Results:

- Contains cycle: No
- Contains 'if' conditional (vertex with multiple edges): Yes
- Contains parallel paths: No

Rysunek 4: Graf uzyskany dla wejścia: *Turn on the computer. Enter the username and password to log in. Open the desired software or application. Perform the required tasks. Save any changes made. Shut down the computer.*

Wejście	Wyjście
Keep searching a list until an item with the highest value is found. Remove the item.	A((Start)) A –> B[Search List] B –> CIs item found? C –>|Yes| D[Remove item] C –>|No| B D –> E((End)
Turn on the computer. Enter the username and password to log in. Open the desired software or application. Perform the required tasks. Save any changes made. Shut down the computer.	A[Turn on the computer] B[Enter username and password to log in] C[Open desired software or application] D[Perform required tasks] E[Save any changes made] F[Shut down the computer] A –> B; B –> C; C –> D; D –> E; E –> F;
Check if a given year is a leap year or not.	A[Start] –> B[Enter year] B –> C[Is the year divisible by 4?] C – Yes –> D[Is the year divisible by 100?] C – No –> E[Is the year divisible by 400?] D – Yes –> F[Is the year divisible by 400?] D – No –> G[Is the year divisible by 400?] E – Yes –> H[Leap Year] E – No –> I[Not a Leap Year] F – Yes –> H F – No –> I G – Yes –> H G – No –> I H[Terminate] I[Terminate]

Tabela 1: Przykładowe wyniki