AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ

Interfejsy człowiek-komputer	
Interfejs sterowania głosem	
Imię i nazwisko:	**Zespół**: VOICE
Zuzanna Zielińska	**Data wykonania:** semestr letni 2023
Konrad Stalmach	**Data zaliczenia:**
Dawid Mazur	**Ocena:**

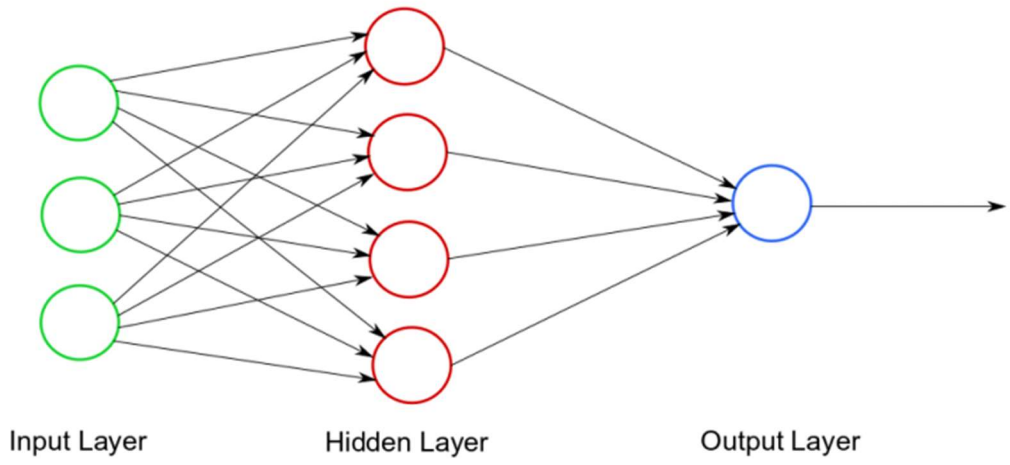
1. Wstęp

W raporcie został przedstawiony proces tworzenia systemu serowania za pomocą komend głosowych do gry ping pong. System został napisany w języku programowania *Python* 3.9.

2. Badania literaturowe

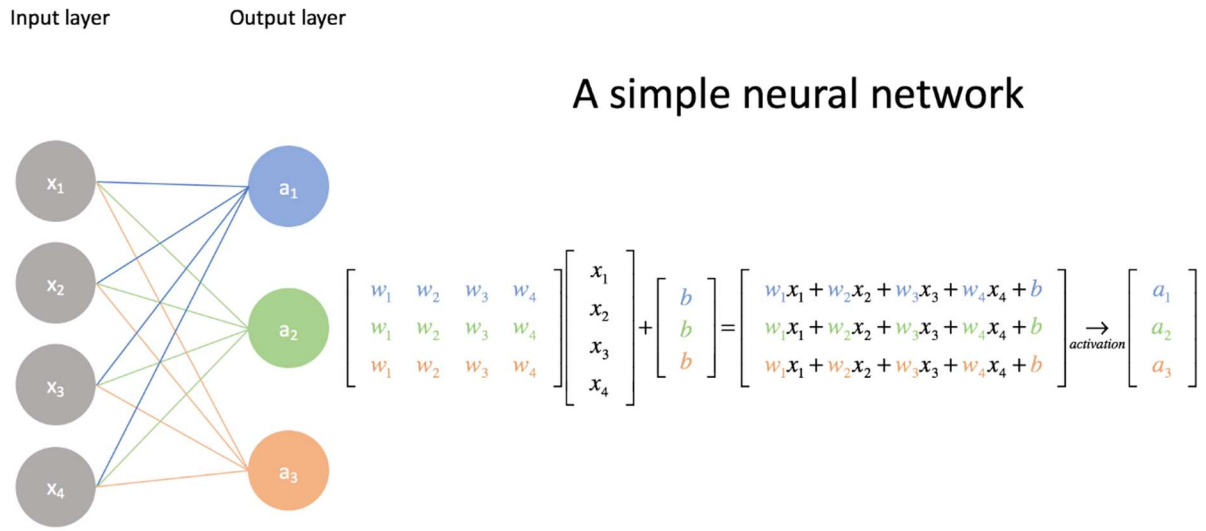
2.1 Sieci neuronowe

Sieć neuronowa jest to struktura matematyczna naśladująca działanie ludzkiego mózgu. Składa się z elementów nazywanych sztucznymi neuronami, które są grupowane w warstwy. Każdy neuron jednej warstwy jest połączony z każdym neuronem kolejnej warstwy oraz z żadnym neuronem warstwy, w której się znajduje. Są trzy rodzaje warstw wejściowa, ukryta (opcjonalna) i wyjściowa.



Rys 1.1 Schemat sieci neuronowej [1]

Wartości neuronu są mnożone przez wagi połączeń i dodawane w neuronach kolejnej warstwy. Dodatkowo do każdego neuronu dodawana jest wartość zwana obciążeniem (ang. „bias”). To czy wartość neuronu zostanie uwzględniona w późniejszych obliczeniach decyduje funkcja aktywacji (ang. „activation function”). Proces zmieniania wag połączeń oraz obciążeń aż osiągnie się najlepsze wyniki w warstwie wyjściowej nazywamy uczeniem się sieci.



Rys. 1.2. Opis połączeń neuronów [2]

2.2 Samodzielne uczenie sieci

Istnieje wiele gotowych bibliotek do uczenia maszynowego oferujące sieci neuronowe do wytrenowania. Największą trudnością w wytworzeniu modelu w ten sposób, jest duża ilość informacji, potrzebna do głębokiego uczenia.

Biblioteki do głębokiego uczenia:

- *Matlab - Deep Learning Toolbox*
- *Python – TensorFlow*
- *Python – Keras*
- *Python – PyTorch*

2.3 Gotowe modele do rozpoznawania mowy

Ze względu na dużą ilość danych potrzebną do wyuczenia własnej sieci neuronowej, została podjęta decyzja o wykorzystaniu biblioteki z już wytrenowanym modelem rozpoznawania mowy.

Istnieje wiele bibliotek, które zawierają modele do rozpoznawania mowy, takie jak: *Whisper*, *SpeechBrain* czy *SpeechRecognition*. Początkowo została wybrana biblioteka *SpeechBrain* ze względu na bardzo obszerną ilość funkcji i dostępnych, pomocnych materiałów. *SpeechBrain* to biblioteka *Pythona* typu open-source do budowania systemów przetwarzania mowy. Dostarcza ona jednolite ramy do budowania całościowych potoków przetwarzania

mowy, od akwizycji danych do szkolenia modeli. Pozwala użytkownikom na łatwe dostosowanie i rozszerzenie istniejących komponentów lub dodanie nowych. Dostarcza również zestaw wstępnie wytrenowanych modeli i zbiorów danych do rozpoznawania mowy, syntezy mowy i innych zadań. Bazuje na bibliotece *PyTorch*.

Pomimo wielu zalet biblioteki *SpeechRecognition*, nie została wykorzystana w ostatecznym projekcie ponieważ, ważnym wymaganiem było działanie biblioteki offline. Jedyny model biblioteki, który je spełniał to *Sphinx* i niestety nie dawał dobrych rezultatów. Częstą sytuacją było, że wypowiedziane wyrazy były przez program przekształcane. Model też działał wyłącznie w języku angielskim.

Po dłuższym czasie szukania znaleziono bibliotekę *VOSK*. To zestaw do rozpoznawania mowy typu open-source w trybie offline. Umożliwia rozpoznawanie mowy w ponad 20 językach. Z ich strony Internetowej [3] pobrano model rozpoznawania mowy i załączono do skryptu.

3. Wykonanie projektu

3.1 Sposoby sterowania

Są różne sposoby sterowania aplikacją. Do nich należą sterowanie:

- **Ciągłe** (względne) – podczas mówienia zdania lub wyrazu lub głoski (np. aaaaaaaa) wysyłany jest sygnał ciągły. Kiedy skończy się mówić, sygnał się urywa. Liczy się długość dźwięku.
- **Krokowe** (względne) – wydaje się komendę i wysyłane jest polecenie przejścia na kolejny krok polecenia (np. przesunięte położenie paletki o konkretną odległość, zwiększenie prędkości o konkretną wartość).
- **Bezpośrednie** (bezwzględne) – podaje się instrukcje, żeby ustawić sterowanie na konkretną wartość (np. paletka teleportuje się na konkretną pozycję, prędkość jest ustawiana na konkretną wartość).

Interfejs głosowy w aplikacji służy do sterowania prędkością piłki oraz resetowaniem tablicy wyników. Wykorzystuje więc połączenie sterowania krokowego oraz bezwzględnego.

3.2 Opis sterowania

Projekt będzie się składał z modelu do zamiany mowy na tekst i jeśli zostanie wykryte użycie komendy, wysłanie odpowiedniego sygnału sterującego grą.

Rozpoznawane komendy to „reset”, „wolniej”, „szybciej” oraz liczby od zera do pięćdziesięciu, co pięć. Komenda „reset” przywraca wynik gry do jego wartości początkowej. Pozostałe hasła sterują prędkością piłki. Za pomocą wymówienia liczb ustawia się prędkość na wymówioną wartość. Komendy „wolniej” i „szybciej” służą do krokowego ustawiania prędkości – jedno wymówienie komendy odpowiednio zmniejsza lub zwiększa o jeden wartość prędkości.

Komendy:

- „**reset**” – reset tablicy wyników,
- „**wolniej**” – zmniejszenie prędkości o jeden,
- „**szybciej** – zwiększenie prędkości o jeden,
- „**zero**” – ustawienie wartości prędkości na 0,
- „**pięć** - ustawienie wartości prędkości na 5,
- „**dziesięć** - ustawienie wartości prędkości na 10,
- „**piętnaście**” - ustawienie wartości prędkości na 15,
- „**dwadzieścia**” - ustawienie wartości prędkości na 20,
- „**dwadzieścia** pięć” - ustawienie wartości prędkości na 25,
- „**trzydzieści**” - ustawienie wartości prędkości na 30,
- „**trzydzieści pięć**” - ustawienie wartości prędkości na 35,
- „**czterdzieści**” - ustawienie wartości prędkości na 40,
- „**czterdzieści pięć**” - ustawienie wartości prędkości na 45,
- „**pięćdziesiąt**” - ustawienie wartości prędkości na 50.

Tabela 3.1. Komendy głosowe sterujące aplikacją

3.3 Dokładność

Wybrany przez nas model wymaga wyraźnego wypowiadania słów do mikrofonu. Jeśli te warunki nie są spełnione, to interfejs będzie pomijał komendy.

3.4 Komunkacja z serwerem

Aplikacja komunikuje się z grą ping pong za pomocą protokołu REST API, korzystając z biblioteki *requests*. Zmiany wartości sterowania wysyłane są za pomocą komendy POST.

4. Wnioski

W dzisiejszych czasach łatwo jest zrobić interfejs głosowy składający się z prostych komend, dzięki szerokiemu wyborowi wydrenowanych modeli rozpoznawania mowy. Wystarczy jedynie pobrać model, wpisać odpowiednie funkcje uruchamiające go oraz przy rozpoznaniu wybranych komend, wykonać pożądaną czynność. Problemem jest znalezienie odpowiednio działającego modelu. Niektóre modele nie działają w czasie rzeczywistym, niektóre działają tylko online, niektóre tylko czasami rozpoznają podane przez nas komendy. Szczególnie ważne jest więc znalezienie modelu spełniającego potrzeby użytkownika.

Do wykonania całej aplikacji, wymagana jest regularna komunikacja wszystkich grup. Od grup wykonujących interfejsy potrzeba informacji o specyfice swojego rodzaju komunikacji i jak poczynione postępy wpływają na ostateczne sterowanie. Od grupy wykonującej aplikację, jakie dokładnie dane do sterowania są wymagane.

5. Bibliografia

[1] Maciej Mazurek, *Simple neural network* [online], aktualizacja 10.01.2020 [dostęp 28.04.2023], Dostępny w Internecie: https://impicode.com/blog/simple-neural-network/

[2] Jeremy Jordan, *Neural networks: representation* [online], aktualizacja 28.06.2017 [dostęp 28.04.2023], Dostępny w Internecie: https://www.jeremyjordan.me/intro-to-neural-networks/

[3] *Models* [online], *Vosk*, [dostęp 28.04.2023], Dostępny w Internecie: https://alphacephei.com/vosk/models