



AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Wykorzystanie sieci neuronowych do przetwarzania sygnałów biomedycznych w ocenie ryzyka patologii

Autor: Przemysław Janiszewski

Promotor: Krzysztof Regulski

Recenzent: Izabela Olejarczyk-Woźeńska

Rodzaj studiów i kierunek: stacjonarne IT, inż

Nazwa wydziału: Wydział Inżynierii Metali i Informatyki Przemysłowej

Nazwa katedry: Katedra Informatyki Stosowanej i Modelowania

Miejsce i data prezentacji: Kraków, 03.02.2025

Motywacja

➤ **Problematyka:**

Dynamicznie postępujący rozwój sztucznej inteligencji, a także narastające trudności w dostępie do opieki medycznej wynikające z problemów demograficznych, niewystarczającej ilości personelu medycznego oraz coraz większego rozwarstwiania się społeczeństw.

➤ **Cel pracy:**

Stworzenie systemu ekspertowego umożliwiającego prowadzenie badań przesiewowych dla dolegliwości cywilizacyjnych, takich jak cukrzyca, czy choroby wieńcowe, a także zapewnienie szybkich analiz i odpowiedzi na proste, rutynowe pytania w celu odciążenia personelu medycznego.

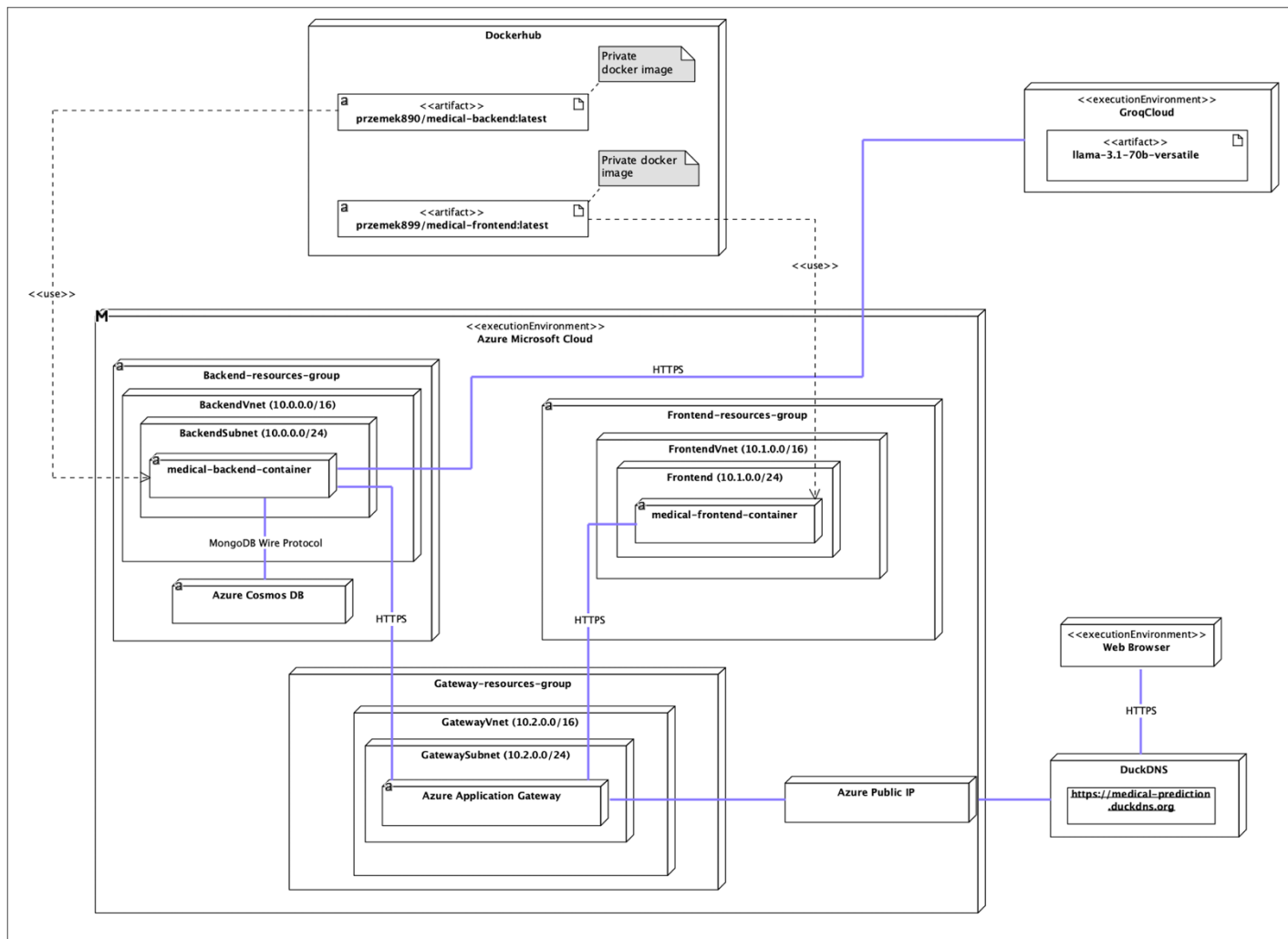
➤ **Kluczowe etapy realizacji projektu:**

- Trening modelu do oceny ryzyka wystąpienia zaburzeń sercowo-naczyniowych przy wykorzystaniu architektury MLP
- Szkolenie modelu do prognozowania ryzyka rozwoju cukrzycy przy zastosowaniu mechanizmu samouwagi architektury TabNet
- Implementacja oraz konfiguracja modelu LLaMA3.1 dla celów okołomedycznych

➤ **Założenia techniczne:**

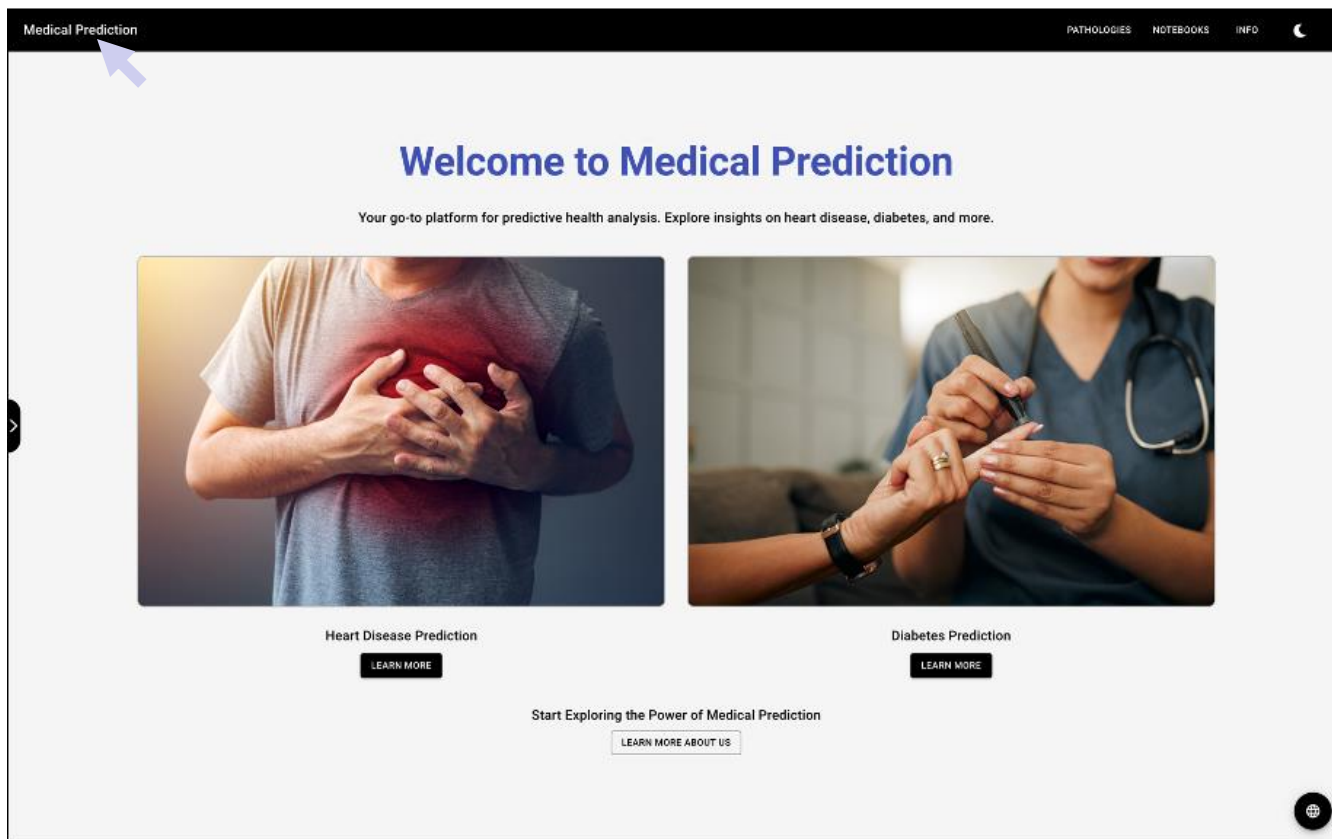
- Dostęp do systemu z dowolnego miejsca na świecie poprzez hostowanie aplikacji w chmurze Microsoft Azure
- Ochrona danych przed nieautoryzowanym dostępem poprzez szyfrowanie komunikacji przy pomocy protokołu HTTPS, wykorzystanie reverse proxy, a także grup zabezpieczeń sieciowych (NSG)

Projekt systemu

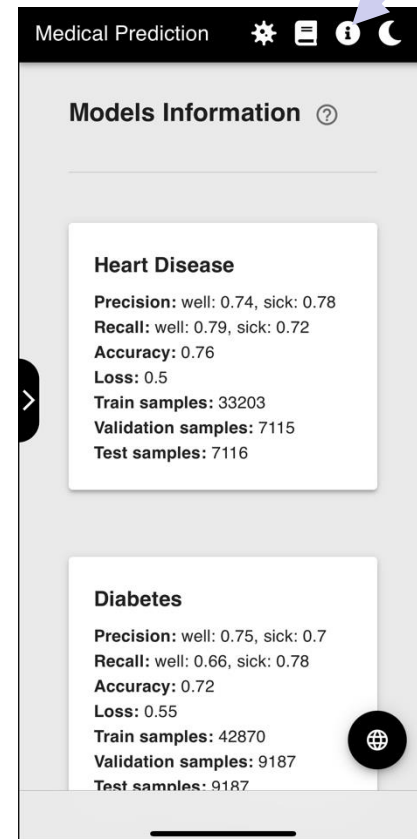




Graficzny interfejs użytkownika

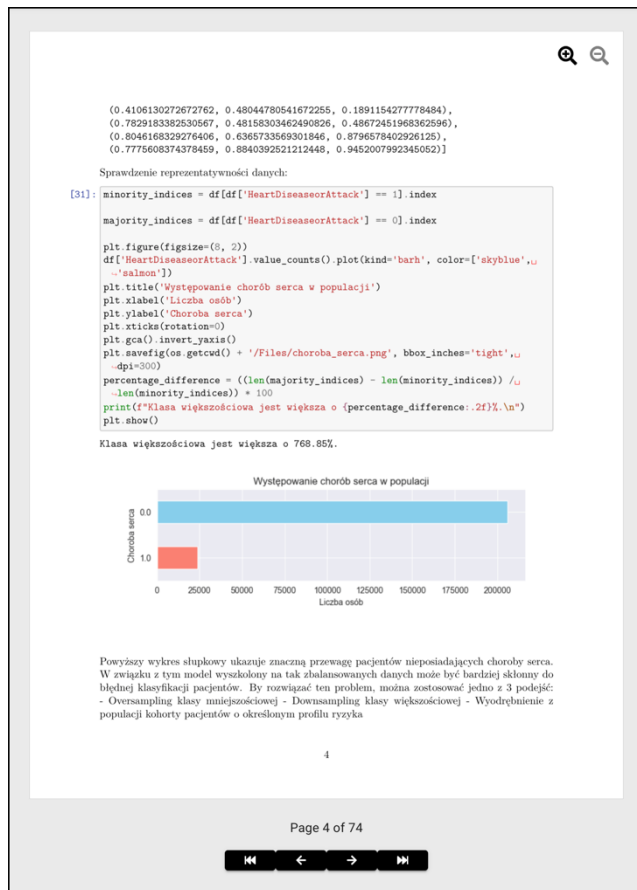


Widok dla urządzeń desktopowych

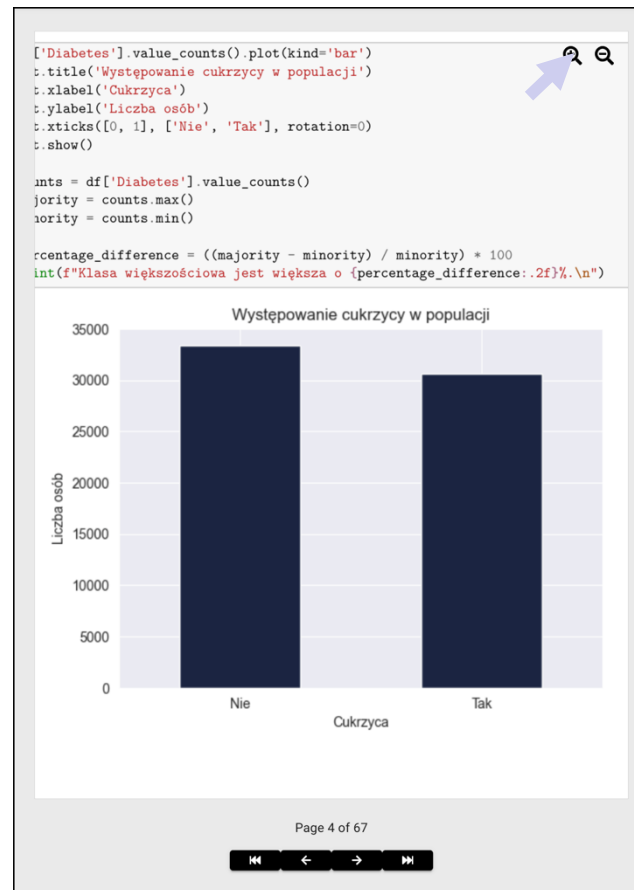


Widok dla urządzeń mobilnych

Notatniki Jupyter w postaci plików PDF, przekonwertowane z wykorzystaniem LaTeX, osadzone w oknie przeglądarki



Analiza zaburzeń sercowo-naczyniowych



Analiza ryzyka wystąpienia cukrzycy

Architektura modeli sieci neuronowych oraz obiektu żądania

```
import torch
import torch.nn as nn
#####

class Model_1(nn.Module):

    def __init__(self):
        super(Model_1, self).__init__()
        self.hidden1 = nn.Linear(21, 15)
        self.bn1 = nn.BatchNorm1d(15)
        self.dropout1 = nn.Dropout(p=0.3)
        self.act1 = nn.LeakyReLU(negative_slope=0.01)

        self.hidden2 = nn.Linear(15, 10)
        self.bn2 = nn.BatchNorm1d(10)
        self.dropout2 = nn.Dropout(p=0.3)
        self.act2 = nn.LeakyReLU(negative_slope=0.01)

        self.hidden3 = nn.Linear(10, 5)
        self.bn3 = nn.BatchNorm1d(5)
        self.dropout3 = nn.Dropout(p=0.3)
        self.act3 = nn.LeakyReLU(negative_slope=0.01)

        self.output = nn.Linear(5, 1)

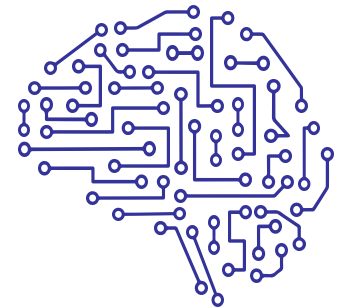
    def forward(self, x):
        x = self.act1(self.dropout1(self.bn1(self.hidden1(x))))
        x = self.act2(self.dropout2(self.bn2(self.hidden2(x))))
        x = self.act3(self.dropout3(self.bn3(self.hidden3(x))))
        x = self.output(x)
        return x
```

```
from pytorch_tabnet.tab_model import TabNetClassifier
import torch
#####

class Model_2:

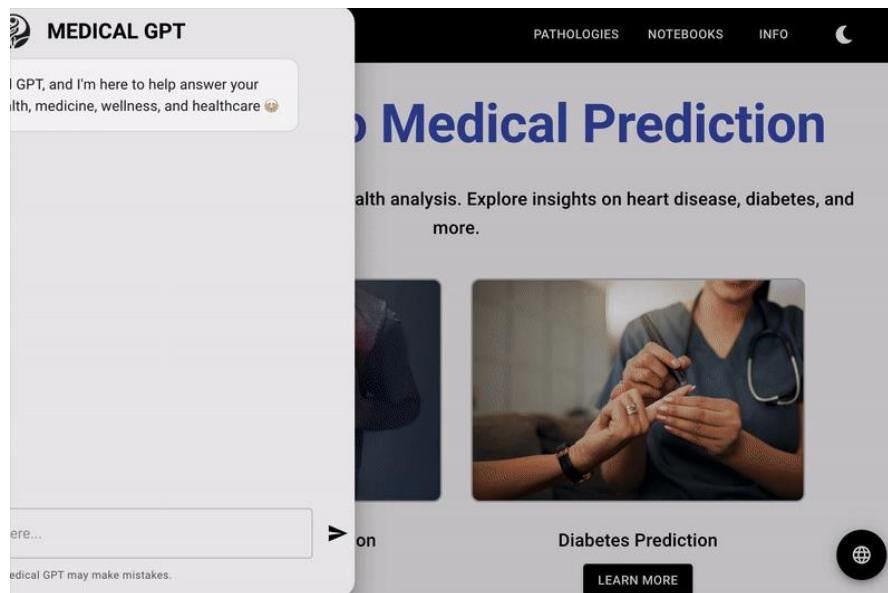
    def __init__(self, input_dim=17, output_dim=1):
        self.model = TabNetClassifier(
            input_dim=input_dim,
            output_dim=output_dim,
            n_d=8, n_a=8,
            n_steps=3,
            gamma=1.3,
            lambda_sparse=1e-3,
            optimizer_fn=torch.optim.AdamW,
            optimizer_params=dict(lr=0.0001),
            mask_type="sparsemax"
        )
```

```
model: str = os.getenv("GROQ_GPT_MODEL", "llama-3.1-70b-versatile")
completion = client.chat.completions.create(
    model=model,
    messages=[
        {"role": "system", "content": system_message},
        {"role": "user", "content": user_message}
    ],
    temperature=0.3,
    max_tokens=8000,
    top_p=0.5,
    stream=True,
    stop=None
)
for chunk in completion:
    yield chunk.choices[0].delta.content or ""
```



Testy systemu E2E

- Weryfikacja poprawności pobierania i wyświetlania parametrów szkoleniowych modeli z chmurowej bazy danych w sekcji Info
- Obsługa formularzy diagnostycznych // weryfikacja generowanych predykcji
- Obsługa i nawigacja po notatnikach zawierających informacje o procesie szkolenia modeli
- Zmiana języka // motywu aplikacji
- Testy funkcjonalności chatbota „Medical GPT”



Test utrzymywanie kontekstu rozmowy oraz wyszukiwania kluczowych informacji w bazie danych przy pomocy GPT

5

Dni złego zdrowia fizycznego w ciągu ostatnich 30 dni (1-30 dni)

Dni złego zdrowia fizycznego *

31

Trudność

Tak

Wartość nie może być większa niż 30.

Czy miałeś/aś udar? *

Nie

Wysokie ciśnienie *

Nie

ZATWIERDŹ

Male

Age *

23

Education *

Some College

Income *

\$15,000 to \$20,000

SUBMIT

Heart Disease

Probability

1

Yes

4.41%

2

No

95.59%

English

Polish

age: "23"

anyHealthcare: "0.0"

bmi: "23"

cho[Check]: "0.0"

diabetes: "0"

diffWalk: "0.0"

education: "4"

fruits: "1.0"

genHlth: "3"

highBP: "0.0"

highChol: "0.0"

hvyAlcoholConsump: "0.0"

income: "3"

mentHlth: "19"

noDocCost: "0.0"

physActivity: "0.0"

physHlth: "0"

sex: "1"

smoker: "0.0"

stroke: "0.0"

veggies: "1.0"

~P(A): 95.59

app

5m 23.7s

12

16

16

0

Theme change with GPT

28.8s

2

2

Dark Theme

15.5s

Light Theme

13.2s

Forms

26.2s

2

2

Diabetes Prediction

15.5s

cy.title().should('eq', 'MedPred');

cy.get('img[alt="Diabetes Prediction"]').click({

force: true

Generating medical code

Generates Medical Code containing Hello and World

BEFORE EACH

visit /

(fetch) POST 200 :5000/api/session

TEST BODY

get #root > div>child(4) > div

click (force: true)

get [placeholder="Type your message here..."]

type Generate medical code "Hello world". (force: true)

get body

type (enter)

wait 1000

(fetch) POST 200 :5000/api/askGPT

get div>h1[code=language-python]

get [data-testid="ContentCopyIcon"]

click

waitFor

expected and medical hello world example

print_medical_hello() prints a hello message in a medical context.

print("Hello world from Medical AI Assistant")

Call the function to print the message print_medical_hello()

When you run this code, it will print: 'Hello world from Medical AI Assistant'. This is a simple demonstration of how to generate a 'Hello world' message in a medical context using Python. If you have any specific medical questions or topics you'd like to discuss, I'm here to help.

Type your message here...

Code copied to clipboard

Diabetes Prediction

LEARN MORE

➤ Dalsze możliwości rozwoju:

- Zapewnienie autoskalowania, replikacji i równoważenie obciążenia poprzez Managed Kubernetes Service (AKS)
- Optymalizacja zapytań bazodanowych oraz zwiększenie liczby dostępnych specjalistów medycznych
- Implementację innych, nowatorskich modeli uczenia maszynowego
- Zwiększenie intuicyjności interfejsu użytkownika poprzez optymalizację UI/UX
- Udoskonalenie chatbota „Medical GPT” poprzez weryfikację dostarczanych informacji w oparciu o źródła dostępne w Internecie oraz rozwój bardziej zaawansowanych technik „prompt-to-action”
- Rozwój większej liczby testów E2E oraz ich innych rodzajów tj. jednostkowe, bezpieczeństwa, wydajnościowe, A/B ...

➤ Ocena skuteczności opracowanego rozwiązania

➤ Czy założony cel opracowanego systemu został osiągnięty?



Dziękuję za uwagę, czekam na Państwa „zapytania” 😊