

PROPOZCYJE PROJEKTÓW 2024/25

INFORMATYKA – STUDIA II STOPNIA – NIESTACJ.

CEL PROJEKTU

Celem projektu jest wykorzystanie technik poznanych na zajęciach w większym, samodzielny zadaniu. W szczególności chodzi o następujące techniki/algorytmy/modele:

- Preprocessing, normalizacja, augmenacją danych
- Klasifikacja prostymi klasyfikatorami (NB, kNN, DecTree)
- Klasifikacja sieciami neuronowymi
- Klasifikacja obrazów konwolucyjnymi sieciami neuronowymi (i ewentualnie innymi modelami)
- Transfer learning
- Object Detection
- Generatywne AI – grafika (GAN i inne)
- Video / Audio / Tekst
- Algorytmy analizy tekstu (bag of words, wyciąganie wydźwięku / opinii, tematu z tekstu)
- Rekurencyjne sieci neuronowe (RNN, LSTM) oraz transformery lub inne modele deep learning
- Algorytmy genetyczne
- Inteligencja roju (PSO, ACO)

Projekt można zrealizować na dwa sposoby:

RAPORT BADAWCZY

Dla wybranego problemu/bazy danych – sprawdzam jakie modele/techniki działają próbując osiągnąć jak najlepsze wyniki klasifikacji. Robię porównanie wydajności w formie sprawozdania.

APLIKACJA/SERWIS

Stosuję wybrany, najlepszy model klasifikujący w aplikacji/serwisie do praktycznych zastosowań. Prezentuję aplikację z jakąś prostą dokumentacją.

REALIZACJA PROJEKTU

Kilka zasad realizacji projektu:

- Projekt powinien dotyczyć tematyki wskazanej wyżej.
- Projekt można realizować w dowolnym języku, Python nie jest wymagany.
- Projekt realizujemy samodzielnie. W szczególnych przypadkach (np. spory stopień skomplikowania) można za zgodą prowadzącego realizować w parach.
- Projekt powinien być unikalny dla każdej osoby (zespołu) w grupie. Prowadzący założy konwersację dla każdej grupy, gdzie należy rezerwować swój temat (któ pierwszy ten lepszy).
- Należy projekty przechowywać na repozytorium Gitlab i dodać prowadzącego do repozytorium, login [gmadejski-ug](#)
- Czas na wykonanie projektu: do ostatnich zajęć.
- Projekty będą rozliczane podczas ostatnich zajęć. Możliwe jest indywidualne prezentowanie rozwiązania prowadzącemu zajęcia, lub prezentacja na rzutniku przed grupą. Zaleca się, by przynajmniej parę osób zaprezentowało swój projekt publicznie.

OCENA PROJEKTU

Niestety w przypadku projektów o różnej tematyce ciężko wyznaczyć jednoznaczna miarę oceniania. Prowadzący zajęcia postarają się wystawić sprawiedliwe punkty, biorąc pod uwagę następujące kryteria:

- Czy wkład studenta (czas, energia, własny kod programistyczny) w projekt był duży czy mały?
 - Tutaj warto, żeby student wskazał co jest zrealizowane samodzielnie, co skopiowane z samouczka, aco wygenerowane przez AI. Należy też uwzględnić wszystkie źródła, z których korzystaliśmy.
- Czy projekt jest oryginalny/nowatorski, czy projekt jest raczej dobrze zbadany/odtwórczy?
- Czy projekt jest dobrze zrealizowany (zawiera wszystkie istotne komponenty: preprocessing, algorytmy, modele, funkcjonalności)?
- Czy projekt sięga po stare oklepane schematy, czy raczej student starał się korzystać z najnowocześniejszych technik, algorytmów udostępnianych w artykułach naukowych, blogach naukowych itp.?
- Czy student był w stanie dobrze zaprezentować projekt?

WYMAGANIA SZCZEGÓŁOWE

Poniżej podane są wymagania szczegółowe. Student nie musi realizować wszystkich wymagań szczegółowych, ale są one dobrym wyznacznikiem czy projekt jest dobrze realizowany.

RAPORT BADAWCZY	APLIKACJA/SERWIS
Baza danych Czy baza danych jest ciekawa/oryginalna/słabo zbadana? Czy samodzielnie ją stworzę, czy jest ściągnięta? Większość z Państwa pewnie ściągnie gotowca. Warto wybrać bazy danych duże ($>10\ 000$ próbek), nietrywialne (dużo kolumn, niełatwie obrazki), może też z błędami wymagającymi naprawy.	Baza danych Trocę jak po lewej, ale wymagania są nieco mniejsze. Jeśli aplikacja jest prototypem, to właściwie baza danych może być ręcznie stworzona i nie musi być duża. Warto jednak skolekcjonować przynajmniej 100 próbek.
Preprocessing Bazę danych należy naprawić (usuwanie brakujących danych, błędów, wartości odstających). Należy sprawdzić balans klas i ewentualnie zbalansować dane (imputacja, downsampling, upsampling, augmentacja obrazów).	Preprocessing Warto rozpatrzyć kroki wymienione po lewej stronie.
Klasyfikacja Należy porównać kilka algorytmów klasyfikujących, w tym sieci neuronowe. Warto je przebadać różnymi miarami (accuracy, confusion matrix, learning curve, itp.). Eksperymenty należy powtórzyć wielokrotnie i uśrednić wynik. Można stosować cross-validation.	Klasyfikacja & Optymalizacja Właściwie wypada skupić się na klasyfikatorach, które najlepiej pasują do naszego zadania. Taki klasyfikator trzeba dobrze skonfigurować: powinien być nie tylko precyzyjny, ale również lekki. Czas obliczeń jest bardzo ważny, zwłaszcza na słabych maszynach/aplikacjach mobilnych.
Sprawozdanie Eksperymenty opisz w przejrzystym sprawozdaniu podzielonym na rozdziały. Można je zrobić w formie docx/pdf lub notatnika jupyterowego, albo nawet prezentacji powerpoint czy pliku latexowego. Sprawozdaniu powinny znajdować się Twoje objaśnienia, komentarze, wstawki kodu (najbardziej istotne), wykresy i grafiki, tabelki z wynikami. Dobrze będzie jeśli sprawozdanie rozpoczęcie się krótkim wstępem i zakończy konkluzjami podsumowującymi eksperymenty. Dołącz bibliografię z źródłami.	Dokumentacja/Sprawozdanie Opisz wszystkie funkcjonalności aplikacji. Opisz eksperymenty tak jak powiedziano po lewej, ale może nie tak dokładnie jak w przypadku raportu. Możesz napisać instrukcję dla użytkownika aplikacji (readme).

1) KLASYFIKACJA NA DATASECIE NUMERYCZNO-KATEGORIALNYM

#uczenie-nadzorowane

#klasyfikacja

#numeryczno-kategorialne

Jest to projekt typu standard, gdzie pobieramy lub tworzymy dataset w postaci tabelki, a następnie klasyfikujemy rekordy, tak jak robiliśmy to dla iris.csv lub diabetes.csv.

Projekt należy zacząć od wybrania bazy danych. Jest tutaj kilka możliwości. Możemy wybrać dataset ze strony np.

- <https://www.kaggle.com/datasets> wpisując odpowiednie słowa np. „classification” i „numeric”
<https://www.kaggle.com/datasets?search=classification+numeric>
- <https://archive.ics.uci.edu/datasets>

Kilka dość oklepanych przykładów z Kaggle (były jako projekty w zeszłych latach):

- <https://www.kaggle.com/datasets/uciml/adult-census-income>

Rozpoznawanie ile zarabia osoba.

- <https://www.kaggle.com/datasets/blastchar/telco-customer-churn> Klasyfikacja klientów telefonii komórkowej: czy przedłużą umowę, czy zrezygnują?
- <https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009> Jaka jakość ma czerwone wino? (można zmienić liczbę klas z 1-10 na good/bad).
- <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset> Diagnoza choroby serca.
- <https://www.kaggle.com/datasets/emmanuelfwerr/thyroid-disease-data> Diagnoza choroby tarczycy.
- <https://www.kaggle.com/datasets/uciml/mushroom-classification> Klasyfikacja grzybów jadalnych i niejadalnych.

Należy taką bazę danych przebadać pod katem klasyfikacji i wyniki przedstawić w czytelnej formie w formie raportu lub prezentacji. Można podzielić raport na rozdziały:

- **Baza danych:** struktura, znaczenie kolumn, co jest klasyfikowane, jakie są wartości, wyświetlenie statystyk dla kolumn lub nawet wykresów. Objasnienie.
- **Preprocessing:** czy baza danych wymaga naprawy? Jakie są błędy? Czy są wartości odstające, brakujące? Jaki rozdział normalizacji lub modyfikacji danych działać będzie najlepiej? Czy zbiór trzeba balansować dla klas? Jeśli tak, to jaką metodą?
- **Klasyfikacja:** testowanie różnych klasyfikatorów poznanych na zajęciach. Dla każdego klasyfikatora można rozpatrzyć różnorakie konfiguracje parametrów. Ewaluacja klasyfikatorów powinna być przeprowadzona za pomocą różnych sensownych miar. Można skorzystać z modeli pre-trained.
- **Reguły asocjacyjne:** można poszukać w bazie danych jakichś ciekawych zależności.
- **Podsumowanie i interpretacja wyników:** co wyszło, co nie wyszło? Co działa, a co nie? Czy są jakieś interesujące wnioski z badań?

Im ciekawszy, bardziej szczegółowy i dociekliwy raport, tym lepsza ocena za niego. Będą brane takie aspekty jak:

- Czy baza danych jest interesująca, oryginalna, stworzona własnoręcznie?
- Czy na bazie danych dokonano jakiegoś istotnego preprocessingu?
- Czy klasyfikatory zostały dostatecznie szczegółowo przebadane (pod katem parametrów, miar, wersji bazy danych, krzywych uczenia, itp.)?
- Czy dodano reguły asocjacyjne?

2) KLASYFIKACJA OBRAZÓW

#uczenie-nadzorowane

#klasyfikacja

#obrazy

#deep-learning



W zasadzie opis będzie podobny do tego powyżej. Bazę danych można próbować tworzyć ręcznie lub pobrać z Kaggle. Kilka przykładów:

- <https://www.kaggle.com/datasets/gpiosenka/100-bird-species> Klasyfikacja gatunków ptaków
- <https://www.kaggle.com/datasets/phucthaiv02/butterfly-image-classification> Klasyfikacja gatunków motyli
- <https://www.kaggle.com/datasets/alexattia/the-simpsons-characters-dataset> <https://www.kaggle.com/datasets/kostastokis/simpsons-faces> Klasyfikacja postaci z Simpsonów
- <https://www.kaggle.com/datasets/andyczhao/covidx-cxr2> Diagnoza COVID na podstawie zdjęcia RTG płuc
- <https://www.kaggle.com/datasets/csafrift2/plant-leaves-for-image-classification> Klasyfikacja roślin po obrazie liścia

Istnieje wiele innych: <https://www.kaggle.com/search?q=image+classification+in%3Adatasets> (polecam przejrzeć)

Wymagania jak w propozycji A, choć preprocessing może wyglądać trochę inaczej (augmentacja zdjęć, kompresja, konwersja do szarości lub nie, filtry, itp.).

Najlepszym klasyfikatorem będzie tutaj pewnie CNN z różnymi konfiguracjami. Ale warto też potestować inne klasyfikatory (zwykłe NN, kNN) i może różne modele transfer learning.

Krzywe uczenia są tutaj mile widziane.

W ramach dodatkowego bonusa do tego projektu, można spróbować wytrenować sieć GAN do generowania obrazów z badanej bazy danych.

3) TWORZENIE OBRAZÓW GENERATYWNYM AI

#generatywne-ai

#obrazy

#deep-learning



Celem projektu jest przetestowanie różnych technik tworzenia grafiki za pomocą generatywnej AI. Na zajęciach poznaliśmy GAN. Chcemy rozszerzyć te badania o modele wykraczające poza materiał zajęć.

Szczegółowy opis (wygenerowany przez ChatGPT – może mieć błędy)

Celem projektu jest zapoznanie się z różnymi metodami generowania grafiki za pomocą modeli deep-learning. Studenci mają za zadanie wytrenować model generujący grafiki na podstawie wybranej bazy danych obrazów (np. z Kaggle) i porównać różne podejścia pod kątem jakości generowanych wyników, stabilności treningu i efektywności.

Zakres projektu:

1. Pobranie i przygotowanie danych:

- Studenci wybierają bazę danych grafik dostępnych na Kaggle (np. zdjęcia, obrazy rysunkowe, ikonki itp.).
- Dokonują wstępnej analizy danych oraz ich przetwarzania (normalizacja, skalowanie, augmentacja danych).

2. Implementacja modelu bazowego:

- Jako podstawowy model sugeruje się implementację **Generative Adversarial Network (GAN)**.
- Studenci muszą zaimplementować dwa komponenty GAN: generator i dyskryminator, a następnie przetestować jego działanie.

3. **Testowanie alternatywnych metod:** Studenci mają możliwość zaimplementowania innych modeli generatywnych, takich jak:
- **Variational Autoencoders (VAE):** Model probabilistyczny uczący się rozkładu danych. Jest łatwiejszy w implementacji niż GAN i stabilniejszy w treningu.
 - **Conditional GAN (cGAN):** Rozszerzenie GAN, które pozwala na generowanie obrazów warunkowanych na dodatkowych etykietach (np. generowanie obrazów kotów, psów lub innych obiektów w zależności od klasy).
 - **Deep Convolutional GAN (DCGAN):** Usprawniona wersja GAN wykorzystująca splotowe warstwy neuronowe (Convolutional Neural Networks, CNN), co poprawia jakość generowanych obrazów.
 - **StyleGAN:** Zaawansowany model do generowania realistycznych obrazów z możliwością manipulacji stylami.
 - **Denoising Diffusion Probabilistic Models (DDPM):** Nowoczesny model generatywny oparty na procesie dyfuzji, pozwalający uzyskać wysoką jakość wyników.
 - **Neural Radiance Fields (NeRF):** Model do generowania grafiki 3D, który może być opcjonalnie zaimplementowany w bardziej zaawansowanej wersji projektu.
4. **Ewaluacja wyników:**
- Jakość generowanych obrazów może być oceniana za pomocą metryk takich jak:
 - **Frechet Inception Distance (FID):** mierzący podobieństwo dystrybucji wygenerowanych i prawdziwych obrazów.
 - **Inception Score (IS):** oceniający różnorodność i jakość wygenerowanych obrazów.
 - Można również przeprowadzić subiektywną ocenę jakości generowanych grafik przez zespół.
5. **Raport końcowy:**
- Opis wykorzystanych modeli.
 - Porównanie wyników różnych metod (jeśli studenci testowali więcej niż jedną).
 - Wnioski dotyczące jakości, stabilności i trudności implementacji różnych podejść.

Przydatne linki

1. **Generative Adversarial Networks (GAN):**
 - Artykuł przeglądowy: <https://arxiv.org/abs/2111.13282>
 - Kurs na Hugging Face: <https://huggingface.co/learn/computer-vision-course/en/unit5/generative-models/gans>
2. **Variational Autoencoders (VAE):**
 - Wykład "Variational Autoencoders": <https://www.cs.columbia.edu/~zemel/Class/Nndl-2021/files/lec13.pdf>
3. **Porównanie VAE i GAN:**
 - Artykuł przeglądowy: <https://arxiv.org/abs/2103.04922>
4. **StyleGAN:**
 - Artykuł na temat StyleGAN: <https://en.wikipedia.org/wiki/StyleGAN>
5. **Denoising Diffusion Probabilistic Models (DDPM):**
 - Artykuł przeglądowy: https://en.wikipedia.org/wiki/Diffusion_model
6. **Neural Radiance Fields (NeRF):**
 - Artykuł przeglądowy: https://link.springer.com/chapter/10.1007/978-981-97-2550-2_23

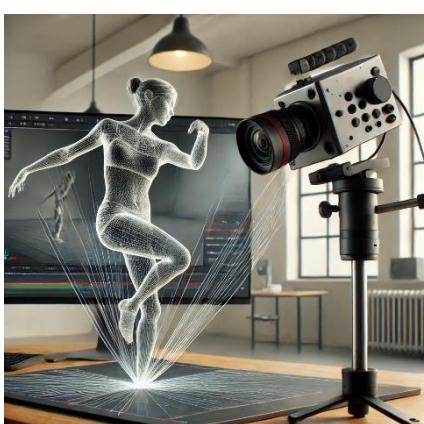
4) WYKRYWANIE OBIEKTÓW NA VIDEO I STEROWANIE

#deep-learning

#obrazy

#video

#sterowanie



Fajnie by było grać w grę nie klikając w myszkę, komórkę, klawiaturę czy konsolę, a wyginając ciało śmiało 😊

Super film na zajawkę o co chodzi:

https://www.youtube.com/watch?v=Vi3Li3TkUVY&ab_channel=EverythingIsHacked

Celem projektu byłoby stworzenie appki, a właściwie jej prototypu – wystarczy zgrubnie napisany program, która będzie jakąś grą (ściągniętą lub napisaną własnoręcznie) z możliwością sterowania obiektem w grze za pomocą ruchu.

O jakich ruchach mowa? Mogą to być gest dlonią (trocę łatwiejsze), machanie jakimś przedmiotem ze znacznikami, poruszanie całym ciałem. Można dla ułatwienia

zatożyć, że osoba jest w odpowiedniej odległości lub odpowiednio ubrana. Projekt jest ciekawy, ale trzeba nałożyć sobie parę zdraworozsądkowych ułatwień.

Można przetestować kilka opcji (tak jak ten gość z filmiku) – jeśli nie uda się z machaniem rękami, to może skupmy się na dloniach? Jeśli nasz własny model się nie wytrenował to może zastosujmy transfer learning? Kilka linków do przejrzenia:

- <https://huggingface.co/datasets/sayakpaul/poses-controlnet-dataset>
- <https://huggingface.co/spaces/hysts/mediapipe-pose-estimation>
- <https://developers.google.com/mediapipe> + <https://github.com/google/mediapipe>

5) INŻYNIERIA PROMPTÓW I PRZEGŁĄD NARZĘDZI DO TWORZENIA WIDEO

#generatywne-ai

#obrazy

#wideo

#audio

#prompty

#deep-learning



Tematem projektu jest przejrzenie narzędzi generatywnego AI do tworzenia filmów, porównanie ich i wybranie najlepszych narzędzi do stworzenia krótkiego filmu. W ramach tego projektu należy stworzyć dwie rzeczy:

• **Krótki raport** (które narzędzie testowaliśmy? Jakie były wygenerowane rezultaty? Czy spełniły nasze oczekiwania? Dlaczego tak/nie? Jakie prompty były stosowane do naszego filmu? Opisz krok po kroku jak był tworzony film.)

• **Film krótkometrażowy** na wybrany temat. Może to być krótki film fabularny, albo promocyjny (reklama), animowana bajka, itp.

Inspiracje:

- a. <https://www.facebook.com/groups/846203050725189>
- b. <https://www.facebook.com/groups/929232341613889>

Tworzenie filmu to proces dość skomplikowany. Film trzeba złożyć z kilku komponentów: wideo + audio (muzyka) + audio (dialog). Będzie dobrze jeśli wszystkie trzy komponenty pojawią się w naszym filmie.

1) Wideo.

Wykorzystaj narzędzie do tworzenia wideo. Przykładowe narzędzia (nie jest to wyczerpująca lista):

- Runway ML : <https://runwayml.com/>
- Minimax / Hailuo AI: <https://hailuoai.video/>
- Vidu: <https://vidu.studio/>
- Kling AI: <https://klingai.com/>
- Luma Labs: <https://lumalabs.ai/dream-machine>
- Haiper AI: <https://haiper.ai/>
- Na Huggingface są też modele do generowania filmów, np. CogVideoX wyszedł ostatnio i jest ponoć dobry. Wymaga to jednak ściagnięcia dużego (kilka GB) modelu na komputer, odpalenia i czekania aż film się wygeneruje. Trzeba mieć dobrą maszynę. Istnieje wiele narzędzi do automatyzowania procesu generowania filmu, np. ComfyUI czy DreamBooth. Osoby bardziej ambitne mogą spróbować się z nimi zmierzyć.
- Inne, wyżej nie wspomniane? Poszukaj w nectie.

Kilka podpowiedzi:

- Filmy generować można za pomocą prompta tekstowego (Text-to-Video), albo za pomocą prompta tekstu z obrazkiem, który staje się pierwszą klatką filmu: (Image-to-Video). Należy się zastanowić, która metoda nas bardziej interesuje. Są fajne narzędzie do generowania obrazów, które mogą nam taką pierwszą klatkę filmu wygenerować i często widać ludzi którzy generują obraz w Midjourney, a potem używają innego narzędzia do generowania wideo np. Minimax.
- Narzędzia te nie wygenerują pewnie całego filmu. Trzeba generować scenę po scenie. Z reguły narzędzi oferują tylko kilkusekundowe filmy. Jeśli chcemy wygenerować dłuższą scenę, musimy wydłużyć początkowy film. Jeśli narzędzie ma taką opcję, można skopiować ostatnią klatkę pierwszego wygenerowanego filmu i na jej bazie odpalić Image-To-Video i dogenerować resztę sceny.
- Problemem może być niezapamiętywanie obiektu między scenami. W jednej scenie nasz bohater może mieć czapkę, a w drugiej ona mu znika (albo zmieni kolor). Jeszcze gorzej jest jak twarz się zmienia ☺ Tutaj trzeba skorzystać z narzędzi typu „daj referencję twarzy / wyglądu” i generuj film pamiętając tę twarz. Niektóre narzędzie takie coś oferują. Inny sposób: generować tak długo (powtarzać próby z doprecyzowaniem szczegółów wyglądu) aż bohater w kolejnej scenie upodobni się do tego z pierwszej sceny.

- 2) **Audio: mowa.** Wykorzystaj narzędzie do generowania głosu (voice generator) np. <https://elevenlabs.io/> (są też inne które można przetestować: <https://zapier.com/blog/best-ai-voice-generator/>, <https://dorik.com/blog/best-ai-voice-generators>). Narzędzie to, powinno wygenerować track audio do naszego filmu, z rozmową pomiędzy bohaterami, lub (troczy łatwiejsza opcja) z głosem narratorki opowiadającym historię. W przypadku pierwszej i trudniejszej opcji (tzn. dialogu między bohaterami), można zadbać o to, żeby twarze/usta poruszały się zgodnie z wygłaszanym tekstem (sprawdź czy jest możliwość synchronizacji głosu z wideo).
- 3) **Audio: muzyka.** Dodaj pasujący do filmu podkład muzyczny wygenerowany za pomocą AI. Wykorzystaj narzędzie do tworzenia muzyki np.
 - <https://www.udio.com/>
 - <https://suno.com/>

Wszystkie trzy komponenty należy złożyć jakimś programem do edycji wideo. Tutaj nie podpowiadam, proszę znaleźć dobre narzędzie. Idealnie by było gdyby filmik miał przynajmniej parę scen i minimum 30 sekund. Może komuś uda się złożyć nawet minutę lub więcej? 😊

6) INŻYNIERIA PROMPTÓW I CHATBOT

#generatywne-ai

#obrazy

#wideo

#audio

#prompty



Celem projektu jest wykorzystanie LLM (large language model), do generowania odpowiedzi w specyficznym kontekście/wirtualnym środowisku. Przykłady:

- Chcemy chatbota na stronę Inf.ug.edu.pl i chatbot powinien znać wszystkie sylabusy i odpowiadać studentom lub kandydatom na studia na pytania na temat studiów.

Użytkownik: „Co przerabiane jest na przedmiocie Inteligencja obliczeniowa?”
Chatbot (LLM): „Zgodnie z sylabusem dla kierunku Informatyka, na Inteligencji obliczeniowej przerabiane jest uczenie maszynowe, algorytmy metaheurystyczne, deep learning”

Użytkownik: „Możesz wyjaśnić co to za pojęcia”

Chatbot (LLM): „Tak, oto wyjaśnienie: ...”

- Chcemy postać w grze (NPC, non-playable character) typu RPG, która będzie z nami miała interakcję lepszą niż tylko prosty dialog typu „wybierz odpowiedź a lub b”. Chcemy, żeby postać z nami rozmawiała swobodnie, ale żeby dialog pasował do świata przykład:

Gracz: „Kowalu, czy gdzie w tym mieście jest karczma?”

NPC (LLM): „Musisz iść na północ, mości panie. Karczma jest za młynem.”

Gracz: „Dzięki, ziomal. Lubisz grać na kompie?”

NPC (LLM): „Cóż mówisz, wędrowcze. Nie rozumiem tych przedziwnych słów!”

Kolejna rozmowa można nawiązywać do poprzedniej, o ile jest zapamiętana:

NPC (LLM): „O znowu przybywasz. Mów tym razem bardziej zrozumiale, mości panie”

Projekt można zrobić naprawdę ambitnie, testując różne metody i robiąc wszystko „od podszewki”. Można też skorzystać z gotowych skryptów i rozwiązań, co będzie sporym ułatwieniem. Jeśli ktoś podejdzie do tego ambitniej, możemy się umówić, że ten projekt zaliczy oba projekty. W przypadku sporego zaangażowania, można pewnie to rozwinąć w kierunku pracy mgr lub napisać raport naukowy (w czym mogę pomóc i się trochę zaangażować).

Przykłady Projektów i Technik z linkami (podpowiedzi od ChatGPT 😊, warto dokładniej przeszukać internet).

1. Dynamiczna generacja dialogów

Modele LLM, takie jak GPT-3 i GPT-4, mogą tworzyć dialogi kontekstowe zamiast korzystać z sztywnych linii dialogowych. Dzięki temu NPC są bardziej naturalni i immersyjni.

https://lutpub.lut.fi/bitstream/handle/10024/167809/bachelorthesis_Huang_Junyang.pdf?sequence=1

2. Dialogi świadome kontekstu

NPC generują odpowiedzi uwzględniające stan gry lub wcześniejsze interakcje z graczem, co zwiększa zaangażowanie.

https://projekter.aau.dk/projekter/files/536738243/The_Effect_of_Context_aware_LLM_based_NPC_Dialogues_on_Player_Engagement_in_Role_playing_Video_Games.pdf

3. Role-playing w modelach językowych

NPC są projektowani z określonymi „rolami”, co pozwala na realistyczne zachowania i spójność w interakcjach.
<https://arxiv.org/abs/2305.16367>

4. Interaktywne rozmowy z NPC w grach

Wytyczne dla projektantów pokazują, jak wykorzystać LLM, by NPC generowali odpowiedzi adekwatne do sytuacji w grze.

https://link.springer.com/chapter/10.1007/978-3-031-54975-5_10

5. Generowanie narracji proceduralnych

Framework PANGeA używa LLM do tworzenia narracji i postaci NPC z cechami osobowości, np. zgodnie z modelem Wielkiej Piątki.

<https://arxiv.org/abs/2404.19721>

6. AI Dungeon i mody AI w grach

Gry jak „AI Dungeon” oraz mody do „Skyrim” i „Stardew Valley” pokazują, jak LLM mogą tworzyć nieograniczone rozmowy z graczami.

- AI Dungeon: https://en.wikipedia.org/wiki/AI_Dungeon

- Mody w grach: <https://www.theverge.com/2024/10/17/24268007/modders-ai-companions-stardew-valley-skyrim>

7. Hugging Face – Fine-Tuning LLM dla Twojego Chatbota

Hugging Face oferuje szczegółowy przewodnik, jak dostroić istniejący model językowy (np. GPT-2, GPT-3) na własnym zestawie danych, co pozwala stworzyć spersonalizowanego czatbota.

Link: <https://huggingface.co/blog/how-to-train>

8. OpenAI – Tworzenie zaawansowanych promptów z API GPT

OpenAI dokumentuje, jak korzystać z prompt-engineering, by sterować zachowaniem modeli GPT bez konieczności fine-tuning. Zawiera przykłady z użyciem API.

Link: <https://platform.openai.com/docs/guides/completion>

9. GitHub – Awesome Chatbot Projects

Lista otwartoźródłowych projektów chatbotów z przykładami fine-tuning i prompt-engineering. Doskonały punkt startowy do nauki i eksperymentów.

Link: <https://github.com/futuga/awesome-chatbots>

10. Artykuł o Fine-Tuning GPT-2: „How to Fine-Tune GPT-2 for Specific Applications”

Ten artykuł wyjaśnia krok po kroku, jak dostroić GPT-2 do własnych danych, aby stworzyć niestandardowego czatbota.

Link: <https://towardsdatascience.com/how-to-fine-tune-gpt-2-12420adcf63f>

11. LangChain – Framework dla zaawansowanego prompt-engineering i pamięci czatbota

LangChain umożliwia budowanie chatbotów z „pamięcią” konwersacji i dynamicznymi promptami. Nadaje się do eksperymentów z narracyjnymi chatbotami.

Link: <https://www.langchain.com/>

12. FastAPI z OpenAI GPT – Budowanie własnego API do czatbota

Przewodnik, jak stworzyć backend do czatbota wykorzystującego OpenAI GPT z pomocą frameworka FastAPI.

Link: <https://towardsdatascience.com/building-a-chatbot-with-openai-gpt-and-fastapi-d8db5b82a0f8>

13. Fine-Tuning GPT-3 przy użyciu OpenAI CLI

OpenAI umożliwia fine-tuning GPT-3 z własnym zestawem danych przez interfejs wiersza poleceń. Przewodnik pokazuje, jak przygotować dane i przeprowadzić trening.

Link: <https://platform.openai.com/docs/guides/fine-tuning>

14. Real-Time Chatbot z Flask i GPT-3

Tutorial pokazuje, jak stworzyć prosty interaktywny chatbot za pomocą GPT-3, z integracją w czasie rzeczywistym na stronie internetowej.

Link: <https://realpython.com/flask-by-example-part-1-project-setup/>

Techniki Kontroli Odpowiedzi LLM

1. Fine-tuning (dostrajanie modelu)

Polega na trenowaniu modelu na specyficznych danych, aby lepiej dopasować jego zachowanie do potrzeb projektu.

- Przykłady:
 - Dostrajanie do specyficznego uniwersum gry (np. styl mowy w świecie fantasy).
 - Dodawanie osobowości NPC (np. zgodnie z modelem Wielkiej Piątki).
- Zalety: Bardzo precyzyjne dostosowanie.
- Wady: Drogie i czasochłonny proces.

2. Prompt Engineering (inżynieria promptów)

Tworzenie odpowiednich podpowiedzi bez modyfikacji modelu.

- Przykłady:

- Definiowanie postaci NPC w podpowiedzi, np.:

„Jesteś kowalem Galenem, który jest sceptyczny wobec obcych. Reaguj zgodnie ze swoją osobowością.”
- Dynamiczne podpowiedzi oparte na stanie gry.
 - Zalety: Łatwość i niskie koszty.
 - Wady: Może być trudniejsze w bardziej skomplikowanych projektach.

3. Podejścia hybrydowe

Połączenie obu technik, np. fine-tuning dla ogólnych zachowań NPC i prompt engineering do interakcji kontekstowych.

Rekomendacje: Jeśli projekt jest mały a komputery słabe, zacznij od **prompt engineering** – jest elastyczny i szybki.

- Przy większych projektach z zasobami warto rozważyć **fine-tuning**, szczególnie gdy potrzebne są specyficzne zachowania NPC.
- Korzystajcie z otwartych narzędzi i frameworków jak **Hugging Face**, aby łatwo eksperymentować.

7) ROZWIĄZYWANIE PROBLEMU OPTYMALIZACYJNEGO

Bierzemy problem do rozwiązania np. wspomniany na wykładzie nonogram/obrazek logiczny, albo jakiś graf, a następnie szukamy rozwiązania tego problemu. Projekt polegałby na tym, że testujemy skuteczność paru algorytmów i porównujemy ich skuteczność (czy zawsze rozwiązują problem, niezależnie od jego wielkości?) oraz czas (czy wykonują się szybko?). W zależności od problemu warto porównać różne wersje algorytmu genetycznego (np. różne typy kodowania chromosomów, różne funkcje fitness), algorytm PSO lub ACO, a nawet (jeśli to możliwe) algorytmy RL. Można do tego dać jakiś bazowy algorytm typu brute force. Dobreby było porównać 3 do 5 algorytmów dla kilku wersji problemu o różnym stopniu skomplikowania. Podsumowaniem takiego projektu mogłyby być taka tabela z wynikami, np.:

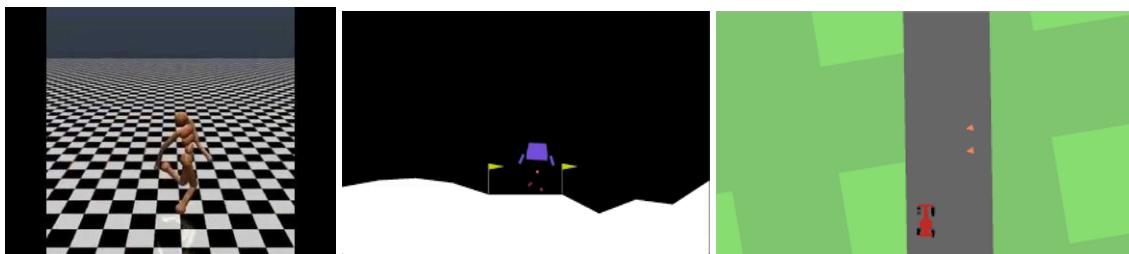
	Przeszukiwanie grafu wszerz	Algorytm genetyczny (Fitness 1)	Algorytm genetyczny (Fitness 2)	ACO	Q-Learning
Problem 5x5	100% (1s)	100% (10s)	100% (8s)	100% (15s)	100% (20s)
Problem 10x10	100% (5s)	90% (15s)	100% (10s)	90% (20s)	100% (30s)
Problem 15x15	100% (70s)	50% (30s)	90% (15s)	60% (30s)	100% (60s)
Problem 20x20	0% (--) timeout	0% (--)	20% (60s)	10% (50s)	90% (200s)

Takie liczby powinny być średnią wielu odpaleń algorytmu (minimum 10, lepiej więcej). W przypadku gdy algorytm znalazł rozwiązanie, można zapisać jego czas działania. Można do tego zrobić też fajne wykresy z wielkością problemu na osi X (5, 10, 15, 20) i skutecznością algorytmów na osi y (linie w różnych kolorach dla różnych algorytmów).

Pomysły na problemy do rozwiązania są niżej w sekcji Inspiracje tematów.

8) STEROWANIE AGENTEM W WIRTUALNYM (LUB PRAWDZIWYM) ŚRODOWISKU

Celem projektu jest opracowanie algorytmu, który będzie dobrze sterował jakimś obiektem w danym środowisku (wirtualnym lub prawdziwym). Ponieważ nie mamy do dyspozycji robotów, dronów i innych hardwarowych kreatur, to musimy ograniczyć się do środowiska wirtualnego. Przykładem jest tu pokazana na wykładzie gra w czołgi, jak i inne gry. Można zresztą grę napisać samodzielnie i wytrenować do niej różne algorytmy. Są jednak środowiska oferujące gotowe gry lub wirtualne roboty. Przykładem jest Gym / Gymnasium (<https://gymnasium.farama.org/index.html>) oferujące przeróżne problemy do rozwiązania: sterowanie samochodem, poruszanie robotem, różne gry arcade itp.



Gymnasium ma tę fajną cechę, że może odpalać grę wielokrotnie, trenując przy tym algorytmy. Można też wyłączyć interfejs graficzny, co przyspieszy proces uczenia.

W projekcie należałoby przetestować kilka algorytmów i porównać ich skuteczność, tj. sprawdzić jak dobrze sterują obiektem. Jeśli jest możliwość wystawienia oceny takim algorytmem (np. w ilu przypadkach doszły do celu gry, albo jak wysoko podniósł się wstający robot, albo jak daleko zaszliśmy w space shooterze) to warto zestawić te oceny w tabelce tak jak powyżej w pomyśle (1).

Jakie algorytmy warto przetestować:

- **Swój własny skrypt.** Można zapomnieć o AI i napisać prosty program do sterowania obiektem. Czasami proste rozwiązania są najlepsze. Oczywiście pytanie jest takie: czy nasz skrypt będzie na tyle elastyczny, że zadziała w każdej sytuacji i odnajdzie się w zmieniającym się środowisku?
- **Algorytmy RL.** Uczenie przez wzmacnianie, i nagradzanie agenta za wykonywanie dobrych akcji jest jedną z najpotężniejszych technik do stosowania w tym typie problemu. W zależności od typu gry i danych warto rozważyć Q-Learning lub Deep-Q-Network.
- **Kontroler rozmyty.** To wyzwanie innego gatunku: zamiast uczyć obiekt, opracowujemy rozmyty model, który ma szanse dobrze sterować agentem. Jeśli fuzzy kontroler umie sterować klimatyzacją czy parkować samochód, to może też będzie umiał wylądować sondą na księżycu (Lunar Lander). Taki rozmyty kontroler trzeba dobrze przemyśleć i pewnie wielokrotnie poprawiać. Warto dodać go jako technikę, zwłaszcza do problemów w których liczba zmiennych nie jest zbyt duża. Do sterowania kilkunastoma przegubami robota pewnie już będzie ta metoda za trudna do implementacji.
- **Algorytm genetyczny lub inteligencja roju.** Sterowanie obiektem można potraktować jako problem optymalizacyjny: szukamy jak najlepszego zestawu akcji, który gwarantuje dobre wykonanie akcji (np. wstanie przewróconego robota, wylądowanie sondy). Warto rozważyć, któryś z tych dwóch bio-inspirowanych metod metaheurystycznych.
- **Neuroewolucja (NEAT).** Jest technika, która wykracza poza materiał przedmiotu ale też może być bardzo ciekawa. Łączymy algorytmy genetyczne z sieciami neuronowymi w tzw. algorytmie NEAT. Algorytm genetyczny tworzy populację sieci neuronowych o różnych topologiiach, a następnie ocenia ich fitness poprzez używanie ich w rozwiązywaniu problemu (np. graniu w grę).

Wybierając grę/wirutalne środowisko z Gymnasium, zwróć uwagę na to jakimi danymi opisane jest środowisko, oraz jakie akcje może wykonywać obiekt. Dane dyskretnie/skończone z reguły są łatwiejsze niż zmiennoprzecinkowe.

9) ANALIZA TEKSTU Z MEDIÓW SPOŁECZNOŚCIOWYCH

Ciekawym projektem jest analiza wypowiedzi użytkowników mediów społecznościowych (Twitter/X, Facebook, Reddit, Telegram). W projekcie trzeba zebrać dużą bazę danych postów (najlepiej powyżej 10 tysięcy). Jest to trudne, bo API wielu serwisów się zmienia lub blokuje narzędzia scrapujące. Rok temu paczka snsccrape świetnie pobierała tweety, ale od paru miesięcy nie jest to możliwe. Głównym wyzwaniem tego projektu jest więcej zgromadzenie dobrej i dużej bazy danych. Narzędzia do web-scrapingu (np. Selenium, Beautiful Soup) mogą się okazać dobre. Istnieją też gotowe bazy danych tweedów (np. na Kaggle) ale są one już dość mocno wykiszploatowane.

Tematyka bazy danych może być naprawdę różnorodna i obejmować różne dyscypliny życia. Przykłady pytań.

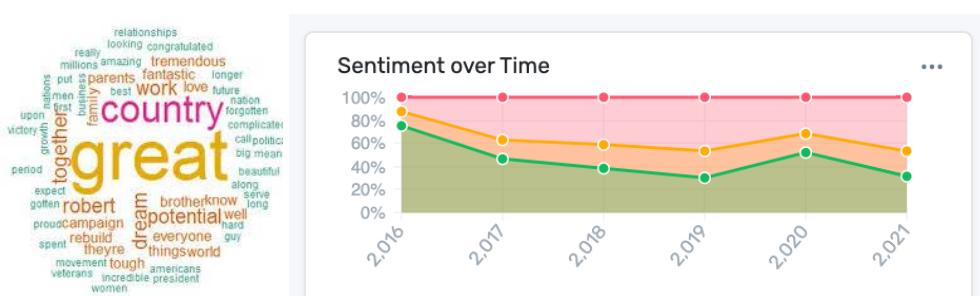
- **Rewolucja AI i emocje.** Jak zmienia się nastawienie ludzi do sztucznej inteligencji na przestrzeni ostatnich miesięcy? #chatgpt #ai #genai

- **Sondaże wyborcze.** Która partia jest bardziej lubiana przed nadchodzącymi wyborami do Europarlamentu? Czy natężenie opinii pokrywa się z sondażami wyborczymi? #KO #PIS #KoalicjaObywatelska #PrawoiSprawiedlowsc #Tusk #Kaczynski
- **Musk vs Bitcoin, Musk vs Tesla.** Czy opinie nt Elona Muska wpływają na akcje Tesli lub cenę Bitcoina? Sprawdź czy istnieje korelacja w czasie. #Musk #ElonMusk #Tesla #Bitcoin
- **Witcher? Rings of Power? Game of Thrones?** Wybierz popularny serial, najlepiej budzący różne emocje i sprawdź jakie emocje wywołuje wśród użytkowników mediów społecznościowych. Można pomierzyć emocje przed i po konkretnym sezonie, lub konkretnym odcinku i sprawdzić jak zmienia się nastawienie.

Jak widać tematy mogą być polityczne, z showbiznesu, rozrywki czy finansów. Wybierz taki, który Cię szczególnie interesuje. Opinie najlepiej zebrać z różnych okresów w czasie. Musisz przemyśleć jak je rozbić? Z różnych dni/tygodni/miesięcy? A może wystarczy rozbić na dwa okresy: przed i po jakimś wydarzeniu.

Zebrane posty można analizować pod różnymi kątami:

- Jakie słowa w nich występują? Bag of words, Word cloud, Word frequency.
- Jaki wydźwięk mają? Vader/Text2Emotion itp. Czy wydźwięk się zmienia w czasie? (Wykres liniowy).
- Analiza tematyki opinii i klasteryzacja.



10) PREDYKCJA LUB GENERACJA DLA SZEREGÓW CZASOWYCH (TEKST, MUZYKA, MOWA, GIEŁDA, ITP.)

Jak tytuł wskazuje, chcemy wybrać jakiś typ danych:

- Tekst (z jakiego źródła? Książki? Źródła internetowe?)
- Muzyka (MIDI? MP3? Jaki format? Jakie źródło?)
- Dźwięk (Mowa? Odgłosy zwierząt?)
- Dane finansowe (Ceny akcji?)

Następnie na podstawie tych danych możemy wyuczyć jakiś model przewidywać lub generować kolejne słowa/dźwięki/nuty/itp., albo klasyfikować dany obiekt (pisarz dla książki, genre dla kawałka muzycznego, gatunek ptaka dla odgłosu, ryzyko inwestycyjne dla akcji giełdy, itp.).

W tym projekcie warto przetestować poznane modele deep learning, czyli:

- Sieci rekurencyjne (zwykłe).
- Sieci LSTM.
- Transformery.

Szczególnie interesujące (zwłaszcza w kontekście tekstu) może być wykorzystanie gotowych modeli (np. pobranych z huggingface) i dotrenowanie ich na naszej bazie danych. Taki fine-tuning może być jednak obciążający czasowo, ale są różne techniki, które przyspieszają trenowanie, np. LoRA (Low Rank Adaptation). W projekcie warto uwzględnić jeden taki eksperyment z fine-tuningiem.

W tym projekcie kuszące są różne zastosowania aplikacyjne. Przykład: trenuję chatbota, którego umieszczam w grze RPG jako postać NPC. Gracz może rozmawiać z taką postacią na różne tematy, ale ważne żeby chatbot zachowywał się jakby był faktycznie postacią w RPGu. Chatbot na stronę UG, który pomagał użytkownikom strony znaleźć przydatne informacje też byłby ciekawym pomysłem.

Jeśli chodzi o modele tekstowe to dość popularne są:

- Llama <https://huggingface.co/meta-llama/Meta-Llama-3-8B>
- Mistral https://huggingface.co/docs/transformers/model_doc/mistral
- Polski model: Bielik <https://huggingface.co/speakleash/Bielik-7B-Instruct-v0.1>