WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Maciej Pracucik

Kierunek: informatyka
Specjalność: informatyka stosowana
Ścieżka dydaktyczna: systemy mobilne
Numer albumu: 410731

Wykorzystanie sieci typu GAN na potrzeby generowania gestów rąk.

Praca magisterska

wykonana pod kierunkiem
dr Krzysztof Podlaski
w Katedrze Informatyki WFiIS UŁ

Łódź 2025

Spis treści

1 Wprowadzenie

1.1 Problematyka

W dzisiejszych czasach bardzo modnym tematem jest sztuczna inteligencja, która zaczyna się wkradać w każdy aspekt naszego życia. Możemy ją spotkać w formie chatów, podpowiedzi do pisanego kodu, asystentów internetowych, systemów rozpoznawania głosów, czy nawet we własnej lodówce! Każda firma żeby zaistnieć i pozostać istotną inwestuje w tę część technologii. Jednakże to z czym AI radzi sobie najgorzej są ręce.
W tym projekcie chodzi o stworzenie sieci typu GAN, która pozwoli na generowanie obrazów rąk, w jak najlepszej jakości. Dodatkowo sieć ma za zadanie nauczyć się, żeby móc modyfikować istniejące zdjęcia i nadawać im zupełnie inny gest, przy zachowaniu jakości i realizmu. Szczególnie ten drugi aspekt pozostaje dla sztucznej inteligencji problematyczny. Myślę, że każdy z nas spotkał się ze zdjęciami, które dosłownie wyglądają, jak żywe, ale to co najczęściej zdradza, że jednak to AI maczało w nim palce są ręce. Za długie palce, dziwne ich ułożenie, ilość, czy nawet totalnie odrealniony wygląd. Przeróżne firmy, jak i naukowcy stale ulepszają sieci, i rozwiązania, żeby i to przestało być problemem. Niniejsza praca również podejmuje się tego niełatwego zadania.

1.2 Cel i zakres pracy

Celem niniejszej pracy jest stworzenie sieci neuronowej typu GAN, która pozwoli na generowanie obrazów gestów rąk, w jak najlepszej jakości.
Docelowo również, wygenerowane zdjęcia będą wykorzystywane do stworzenia animacji przechodzenia z jednego gestu w inny.

1.3 Struktura pracy

Pierwszy rozdział przybliży to czym są sieci neuronowe, a dokładniej typu GAN, jakie są analogiczne rozwiązania, oraz o samym generowaniu zdjęć. Następny opowie jakie narzędzia, biblioteki i technologie zostały wykorzystane do realizacji projektu. Trzeci zaś mówi o tym jak wyglądał proces tworzenia projektu. Co po koleji zostało zrobione, jakie po drodze wystąpiły komplikacje, oraz jak zostały rozwiązane i finalnie jak wygląda projekt. Przedostatni rozdział to przedstawienie wyników, rezultatów realizowanego projektu, analiza i omówienie ich. Ostatni rozdział zawiera wnioski końcowe i podsumowanie.

2 Sieci GAN, analiza konkurencji i techniczne aspekty realizacji

2.1 Sieci neuronowe i AI

sieci

2.2 Sieci typu GAN

GAN

2.3 Generowanie obrazów rąk

generowanie

2.4 GestureGAN

GestureGAN

2.5 PoseGAN

PoseGAN

3 Narzędzia i technologie wybrane do realizacji projektu

3.1 Python

Językiem programowania wykorzystanym do realizacji projektu jest Python. Jest to bardzo oczywisty wybór, ponieważ jest to najpopularniejszy język programowania do tworzenia sieci neuronowych w dzisiejszych czasach. Przybliżmy go jednak, według oficjalnej dokumentacji, jest "łatwym do nauczenia się i potężnym językiem programowania. Posiada wydajne struktury danych wysokiego poziomu oraz proste, ale skuteczne podejście do programowania obiektowego"[**'python-docs'**]. Co niejako wyróżnia go na tle innych języków to fakt, że jest dynamicznie typowany oraz jest językiem interpetowanym. To

oznacza, że nie posiada kompilatora a interpeter, który nie kompiluje programu do pliku wykonywalnego, a kod jest wykonywany w czasie rzeczywistym. Czyni to Python językiem łatwym w testowaniu, kompilowaniu czy wykonywaniu, przenośny, co oznacza, że ten sam kod uruchomi się niezależnie od systemu operacyjnego czy urządzenia. Sam język Python jest zbudowany na bibliotece C, co oznacza, że jest bardzo szybki i wydajny. Został stworzony przez Guido van Rossuma w roku 1991. Co ciekawe jego nazwa wywodzi się z starych serii skeczów grupy Monty Python's Flying Circus, a zyskał na popularności w momencie gdy firma Google, powiedziała, że wykorzystuje go do własnych, wewnętrzych celów.

Jednakże czemu akurat to on jest najczęściej wykorzystany do AI? Odpowiedź jest tak prosta jak sam Python jest prosty. Wynika to z tego, że Python ma bardzo prostą i czytelną składnię, co pozwala developerom skupianie się na logice i samym problemie, a nie na składni[**'python-ai'**]. Dodatkowo Python sam w sobie nie wymaga dużej ilości kodu. Najprostsza sieć neuronowa może zostać stworzona i uruchomiona w zaledwie 4 linijki! Kolejnym powodem przemawiającym dlaczego to własnie Python jest najczęściej używany, jest bardzo duża ilość bibliotek, czy to wbudowanych, czy stworzonych przez społeczność. Biblioteki takie jak NumPy, SciPy, Matplotlib, czy wykorzystane w tym projekcie PyTorch czy MediaPipe, o których będzie w kolejnych podroździałach. Także kolejnym i osatnim już aspektem, o którym chcę wspomnieć, jest wyżej wymieniona społeczność. To dzięki dużej liczbie osób i ogromnym zebranym doświadczeniu, tworzenie dowolnego projektu staje się znacznie prostsze. Praktycznie każdy projekt, aspekt projektu czy problem natrafiało wcześniej duża część ludzi, dzięki czemu możemy szybko i sprawnie rozwiązaywać problem.

3.2 PyTorch

"PyTorch to w pełni funkcjonalny framewordk do tworzenia modeli do deep learningu, kóry jest typem machine learningu, najczęściej wykorzystywanym w aplikacjach takich jak rozpoznawanie obrazów czy procesowanie języka."[**'pytorch-nvidia'**] Biblioteka ta została napisana w Pythonie, przez developerów z Facebook AI Research, oraz ma doskonałe wsparcie do wykorzystywania GPU, szczególnie dla GPU od firmy Nvidia, który posiadam. Co czyni ją idealną biblioteką dla tego projektu. Jednak projekt dotyczy generowania obrazów rąk, czyli dobre wykorzystanie GPU jest bardzo wskazane. Początkowo

jednak używana była konkurencyjna biblioteka, a dokładnie Tensorflow, jednakże była zdecydowanie wolniejsza i mniej klarowna od PyTorch co zaważyło na finalnym wyborze.

3.3 MediaPipe

"MediaPipe Solutions to zestaw bibliotek i narzędzi, które umożliwiają szybkie stosowanie w aplikacjach technik sztucznej inteligencji (AI) i uczenia maszynowego (ML). "[**'mediapipe'**] Opis ten pochodzi z oficjalnej dokumentacji Mediapipe, jednakże sam opis jest zbyt ogólny. Ta biblioteka ma masę możliiwości i zastosować. Do nich można zaliczyć np. rozpoznawanie obrazu, nie chcemy pisać sieci do rozpoznawania, czy na danym obrazku jest pies czy kot? Mediapipe daje gotowe rozwiązanie! Poza tym do wyboru jest też klasyfikacja obrazu, segmentacja obrazu, wykrywanie twarzy, rozpoznawanie gestu, oraz to co było niezbędne w tym projekcie to wykrywanie rąk. To dzieki tej bibliotece udało się wyłuskać szkielet ręki na każdym z obrazów ze zbioru danych, nałożyć go na zdjęcie wraz z maską i przekazane do modelu. Później te informacje były użyte do tworzenia kolejnych klatek animacji przechodzenia z jednego gestu w drugi.

3.4 OpenCV

OpenCV jest to biblioteka do machine learningu oraz computer vision, posiada w swoim arsenale dostęp do takich narzędzi jak wykrywanie i rozpoznawanie twarzy, identyfikowanie obiektów, śledzenie ruchów kamery, śledzenie obiektów 3D, usuwanie czerwonych oczu ze zdjęć, czy nawet łączenie zdjęć razem, by uzyskać obraz całej sceny o wysokiej rozdzielczości.[**'opencv'**] Jednak to żadna z tych funkcji nie została użyta w projekcie. Wykrywanie i tworzenie szkieletu to zadanie Mediapipe, tworzenie modelu to działka PyTorch. OpenCV miał znacznie prostsze zadanie, został wykorzystany do ładowania zdjęć, czy to dla preprocessingu, czy już bezpośrednio do modelu. Dalej zapisywał każdy kolejny obraz w zależności od epoki, dzięki czemu można było śledzić poczynania i na koniec sklejał wszystkie klatki i tworzył z nich animację.

4 Proces tworzenia projektu

4.1 Dobór zbioru danych

zbior

4.2 Wybór architektury sieci

architektura

4.3 Preprocesowanie danych

Preprocesowanie

4.4 Tworzenie modelu

model

4.5 Tesotowanie modelu

Tesotowanie

4.6 Realizacja projektu

realizacja

5 Wyniki i dyskusja

wyniki

6 Podsumowanie

6.1 Zalety i wady przyjętych rozwiązań

Zalety

6.2 Napotkane trudności

trudności

6.3 Możliwości rozwoju

rozwoj

6.4 Wnioski końcowe

wnioski

Spis rysunków

Spis tabel

Listingi