Dokumentacja wstępna

Zespół nr **16**: Jakub Polak, Hanna Smach

Zadanie: Generowanie obrazów samochodów przy użyciu sieci GAN. Do zaliczenia wystarczy wariant bezwzględny, na maksymalną ocenę parametryzowane np. poprzez typ auta.

1. Założenia projektowe

W ramach projektu zostaną zaimplementowane 2 sieci – generatora i dyskryminatora, składające się na architekturę GAN - Generative Adversarial Network. Obie zostaną oparte na konwolucyjnych sieciach neuronowych, których parametry zostaną dobrane optymalnie w celu uzyskania najlepszych wyników, co zostanie sprawdzone poprzez odpowiednie miary statystyczne (punkt 6.). Ponadto, na funkcje straty zostanie wybrana entropia krzyżowa ze względu na przypadek binarny. Następnie podjęta zostanie próba parametryzacji sieci, co wyjaśniono w punkcie 3.

2. Zbiór danych

Do realizacji projektu wybrano zbiór danych "The Car Connection Picture Dataset", dostępny pod adresem:
https://github.com/nicolas-gervais/predicting-car-price-from-scraped-data/tree/master/picture-scraper
Zbiór ten zbiór zawiera 64 467 zdjęć samochodów w formacie .jpg, a także kilka informacji, na podstawie których będzie można uczyć sieć generować zdjęcia parametryzowane (np. model, marka, typ, rok produkcji). Informacje te zakodowane zostały w nazwie plików. Gdy brakuje informacji o danej właściwości samochodu, w nazwie pliku występuje oznaczenie nan. Wtedy zdjęcia tak opisane należy odrzucić, w zależności od wybranego parametru. Jako pierwsza podjęta zostanie próba parametryzacji poprzez typ. Poniżej znajdują się typy samochodów w wybranym dataset wraz z ich liczbą. Pominięto samochody z oznaczeniem typu jako nan.

'Van': 1790,
'Pickup': 6173,
'Station Wagon': 395,
'4dr': 21144,
'3dr': 211,
'2dr': 6596,
'SUV': 20297,
'Convertible': 7547

3. Opis architektury sieci

W projekcie zaimplementujemy sieć o architekturze Conditional Generative Adversarial Network (CGAN). Sieć tego typu składa się tak naprawdę z dwóch mniejszych sieci, nazywanych generatorem i dyskryminatorem, których rywalizacja stanowi proces uczenia. Generator jest siecią złożoną z warstw splotu transponowanego. Na wejściu otrzymuje ona wektor zawierający szum losowy, z którego generuje obraz. Dyskryminator to klasyfikator binarny obrazów, złożony z warstw gęstych

poprzedzonych warstwami splotowymi. Jego zadanie stanowi rozpoznanie, czy zdjęcie wprowadzone na wejściu jest prawdziwe (pochodzi ze zbioru treningowego), czy jest fałszywe (zostało wygenerowane przez generator).

Podczas procesu uczenia, najpierw generator produkuje zestaw obrazów z szumu losowego, a następnie dyskryminator klasyfikuje obrazy z takiego kompletu, jak i te ze zbioru treningowego. Po klasyfikacji w obu sieciach aktualizowane są wagi, zgodnie z celami podsieci, gdzie zadaniem generatora jest oszukanie dyskryminatora tak, aby uznał fałszywy obraz za prawdziwy, a funkcją dyskryminatora jest prawidłowe sklasyfikowanie obrazów jako prawdziwe lub fałszywe. W celu optymalizacji i określania błędu modelu stosuje się funkcję straty, którą poddaje się minimalizacji. Po zakończeniu procesu trenowania dyskryminator jest odrzucany, a generator pozwala na tworzenie obrazów podobnych do tych zawartych w zbiorze treningowym.

Elementem odróżniającym sieć typu CGAN od zwykłej sieci GAN jest podanie na wejściu zarówno generatora, jak i dyskryminatora, dodatkowej informacji o obrazie (np. modelu samochodu). Zabieg ten pozwala użytkownikowi na zażądanie od wytrenowanej sieci uzyskania obrazu o wyszczególnionych cechach (np. w przypadku generowania samochodów, Fiata 500, lub przy generowaniu zdjęć ludzi, persony o czarnych włosach).

4. Wybrane biblioteki

W projekcie zdecydowaliśmy się wykorzystać framework PyTorch języka Python, który udostępniany jest na licencji OpenSource, oraz pozwala na szybką i prostą implementację sieci neuronowych.

5. Etapy projektu i eksperymenty do przeprowadzenia

Pierwszym etapem projektu będzie implementacja generatora oraz dyskryminatora sieci. Następnie sieć zostanie poszerzona o możliwość parametryzacji generowanego zdjęcia rodzajem samochodu, jaki chcemy uzyskać (użytkownik może zażądać wygenerowania np. SUV-a). Jeżeli parametryzację uda się zrealizować poprawnie, planujemy dodać do parametryzacji sieci rok produkcji samochodu.

Eksperymentami, które planujemy przeprowadzić poza strojeniem hiperparametrów sieci, są:

1. Zbadanie, jak rozmiar sieci wpływa na proces uczenia i końcowe wyniki.
2. Zbadanie, czy przy parametryzowaniu mniej liczne kategorie będą prawidłowo generowane.
3. Wygenerowanie zestawienia po N obrazów każdej klasy w celu wizualizacji rezultatów.

Każdy eksperyment zostanie podsumowany wnioskami i komentarzami w dokumentacji końcowej projektu.

6. Miary ewaluacji sieci GAN

Do ewaluacji sieci planujemy wykorzystać metrykę *Fréchet Inception Distance* (FID). Jest to miara odległości pomiędzy rozkładami cech wyekstrahowanych z obrazów ze zbioru uczącego oraz tych wygenerowanych przez badaną sieć. Do ekstrakcji cech wykorzystywana jest sieć Inception v3, którą wytrenowano na zbiorze ImageNet. FID pozwala na ocenę jakości obrazów wyjściowych. Im niższa jej wartość, tym wyższa jakość.

Inną przydatną miarą może być Inception Score (IS), która szacuje, jak bardzo realistyczny jest wyjściowy obraz. Im wyższa wartość IS, tym lepiej, gdyż oznacza to, że sieć potrafi wygenerować wiele różnych (np. różnych typów samochodów, a nie w większości jeden typ) i jednocześnie wyraźnych (takich, które jasno przypominają któryś z typów samochodu) obrazów. W przypadku IS również wykorzystuje się wspomnianą sieć Inception v3.