Sztuczna inteligencja i inżynieria wiedzy
Lista 5

Piotr Syga

26 maja 2025

1 Cel listy

Celem ćwiczenia jest praktyczne zapoznanie się z trenowaniem i wykorzystaniem modelu perceptronu wielowarstwowego w zadaniach klasyfikacyjnych. Przykładem zadania klasyfikacyjnego będzie biometryczne porównanie tożsamości osób przedstawionych na zdjęciach.

2 Wprowadzenie teoretyczne

Sieci neuronowe stanowią jedną z najważniejszych klas systemów stosowanych w uczeniu maszynowym. Inspirowane strukturą i funkcjonowaniem biologicznych układów nerwowych i ich sposobem uczenia się. Podstawowym elementem każdej sztucznej sieci neuronowej jest neuron – jednostka obliczeniowa, która przetwarza sygnał wejściowy za pomocą funkcji aktywacji, powodując nieliniowość transformacji, i przekazuje wynik do kolejnych warstw modelu. Neurony są zorganizowane w warstwy: wejściową, jedną lub więcej warstw ukrytych oraz warstwę wyjściową. Ich wzajemne połączenia są parametryzowane przez wagi, które ulegają modyfikacji w procesie uczenia.

W kontekście niniejszej listy skupimy się na modelu perceptronu wielowarstwowego (MLP, ang. *Multilayer Perceptron*). Jest to sieć jednokierunkowa (feed-forward), w której dane przepływają od warstwy wejściowej do wyjściowej bez sprzężeń zwrotnych. Model ten znajduje zastosowanie zarówno w klasyfikacji, na której skupimy się na liście zadań, jak i w regresji – w zależności od typu zmiennej wyjściowej oraz dobranej funkcji kosztu.

Uczenie MLP odbywa się zazwyczaj metodą propagacji wstecznej (ang. *back-propagation*) w trybie nadzorowanym, a więc z wykorzystaniem oetykietowanych danych w zbiorze uczącym. W tym procesie iteracyjnie aktualizowane są wagi sieci w celu minimalizacji błędu predykcji mierzonego określoną funkcją kosztu. Do najczęściej stosowanych należą: błąd średniokwadratowy w przypadku regresji oraz funkcja krzyżowej entropii dla zadań klasyfikacyjnych. Proces ten bazuje na metodzie spadku gradientu, gdzie każdy krok optymalizacji aktualizuje wagi zgodnie ze wzorem:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta J(\theta) \ ,$$

gdzie θ oznacza wektor parametrów sieci, η to współczynnik uczenia (learning rate), a $J(\theta)$ to funkcja kosztu.

Jednym z istotnych zagadnień w projektowaniu sieci neuronowej jest odpowiednie przygotowanie danych wejściowych. Poza zapewnieniem etykiet, niezbędne jest ustandaryzowanie danych wejściowych oraz zadbanie o ich odpowiednią jakość. W przypadku przetwarzania obrazu podstawową decyzją jest ustalenie wymiarowości danych, tzn. czy zdjęcia będą monochromatyczne (np. maski binarne lub zdjęcia w szkali szarości, czy wielokanałowe (trójkanałowe w przestrzeni barw RGB, YCbCr, lub rzadziej czterokanałowe). Jakość danych będzie miała istotny wpływ na wyuczone wagi modelu, w związku z tym dla obrazów w niskiej jakości (np. rozmazanych lub zaszumionych) konieczny może być preprocessing, który zmityguje wpływ tych zakłóceń. Dla bardziej zaawansowanych zastosowań, może się zdarzyć, iż celowo, uwzględnimy w zbiorze uczącym ustaloną część danych niskiej jakości lub nawet celowo je zaburzymy (cf. augmentacja danych), tak by nauczony model był przygotowany na dane obniżonej jakości podczas testowania. Istotne jest również to, by w czasie uczenia modelu obrazy przedstawiały zróżnicowane, możliwe w docelowym środowisku, dane, jak również by wybrać tylko ten obszar zainteresowania (ROI), który jest istotny dla docelowego zadania, wycinając go z całości obrazu.

Dobór architektury modelu, liczby i rozmiarów warstw ukrytych, oraz hiperparametrów (takich jak tempo uczenia, współczynnik regularyzacji, rodzaj funkcji aktywacji) ma zasadnicze znaczenie dla efektywności procesu uczenia i zdolności generalizacji modelu. W celu oceny skuteczności modelu należy przeprowadzać eksperymenty z podziałem danych na zbiory uczący, walidacyjny i testowy, umożliwiające wykrycie problemów takich jak przeuczenie czy niedouczenie, a następnie wybrać zestaw parametrów, który okazał sie najskuteczniejszy.

Problem porównywania twarzy lub weryfikacja tożsamości na podstawie obrazu twarzy, stanowi jedno z kluczowych zagadnień biometrii i rozpoznawania wzorców. Celem zadania (w pewnym uproszczeniu) jest określenie, czy dwie dostarczone fotografie przedstawiają tę samą osobę. Formalnie jest to problem klasyfikacji binarnej, w którym dla każdej pary obrazów (I_1, I_2) należy wyznaczyć etykietę $y \in \{0, 1\}$, gdzie $y = 1$ oznacza zgodność tożsamości.

Współczesne podejścia do rozpoznawania twarzy opierają się na głębokich sieciach neuronowych uczonych metodą nadzorowaną lub samonadzorowaną. W szczególności dominującą metodologią są modele uczenia głębokiego, które mapują dane obrazowe do przestrzeni cech (embedding space), w której twarze tej samej osoby są reprezentowane wektorami bliskimi względem ustalonej metryki, np. odległości euklidesowej lub cosinusowej.

Typową architekturą stosowaną w tym kontekście jest tzw. sieć bliźniacza (ang. *Siamese Network*) lub sieć trójkowa (ang. *Triplet Network*), które uczą się

funkcji podobieństwa zamiast klasyfikatora. W przypadku sieci bliźniaczej model przyjmuje pary obrazów (I_1, I_2) i minimalizuje funkcję straty kontrastowej:

$$\mathcal{L}_{\text{contrastive}} = y \cdot \|f(I_1) - f(I_2)\|^2 + (1 - y) \cdot \max(0, m - \|f(I_1) - f(I_2)\|)^2 \ ,$$

gdzie $f(\cdot)$ to funkcja ekstrakcji cech, implementowana jako głęboka sieć konwolucyjna, a m to próg marginesowy oddzielający różne klasy. Alternatywnie stosuje się stratę trójkową, która operuje na trójkach obrazów: (baza, pozytywny, negatywny).

Po zakończeniu uczenia, porównanie dwóch obrazów sprowadza się do obliczenia odległości pomiędzy ich reprezentacjami wektorowymi. Próg decyzyjny określa granicę pomiędzy klasami „ta sama osoba" i „różne osoby" i może być ustalany empirycznie na zbiorze walidacyjnym.

Skuteczność systemów rozpoznawania twarzy zależy od wielu czynników, m.in. jakości danych wejściowych, warunków oświetleniowych, ułożenia twarzy, czy obecności masek i okularów. Wysokiej jakości modele (np. FaceNet, ArcFace, AdaFace) potrafią osiągać dokładności porównywalne, lub przekraczające skuteczność eksperta ludzkiego na standardowych zbiorach danych (np. CelebA, LFW, MegaFace). Natomiast mniejsze modele, sprawdzają się w przypadkach ograniczeń związanych z mocami obliczeniowymi, czy też dostępnością odpowiednich danych uczących. Na niniejszej liście zadań skupimy się na przetestowaniu MLP dla zadania klasyfikacyjnego, jakim jest potwierdzanie tożsamości.

3 Zadania

Wykorzystaj udostępniony plik źródłowy oraz bazę danych CelebA (lub bez pobierania) w celu wykonania poniższych zadań potwierdzenia tożsamości.

1. Naucz model MLP na losowym podzbiorze 10, 100, 500, 1000 i 5000 par zdjęć, a następnie weryfikuj skuteczność weryfikacji tożsamości dla podzbioru 200 rozłącznych par zdjęć (naturalnie, zadbaj by żadne z wykorzystanych w procesie treningu zdjęć nie było użyte do weryfikacji). Korzystając z metryk poznanych na Liście 4, zbadaj wpływ rozmiaru zbioru uczącego na skuteczność klasyfikacji (40 punktów).

2. Zbadaj wpływ wartości parametru tempa uczenia (`lr`) na skuteczność modelu, przy stałym rozmiarze zbioru uczącego (1000 par) oraz stałej liczbie epok. Przeanalizuj przynajmniej 5 różnych wartości parametru (20 punktów).

3. Zbadaj wpływ wartości parametru liczby epok na skuteczność modelu, przy stałym rozmiarze zbioru uczącego (1000 par) oraz stałym tempie uczenia. Przeanalizuj przynajmniej 5 różnych wartości parametru (20 punktów).

4. Na podstawie eksperymentów, znajdź zależność między liczebnością zbioru uczącego, tempem uczenia oraz liczbą epok. Znajdź najlepszy zestaw parametrów oraz zastanów się nad możliwością zmiany tempa uczenia w czasie, jak i warunkiem wcześniejszego zatrzymania, na podstawie zmian w wartości funkcji straty (20 punktów).

5. W czasie weryfikacji, przed wykonaniem porównania, dodaj do zdjęć testowych dodatkowe zaburzenia (np. szum Gaussa, rozmycie, zmianę jasności), a następnie porównaj skuteczność weryfikacji tożsamości na zaburzonych i niezaburzonych danych. Zmodyfikuj system weryfikacji tożsamości tak, by zmitygować wpływ zaburzeń (wypróbuj np. uwzględnienie zaburzonych przykładów w procesie uczenia, modyfikację sieci, zmianę funkcji straty, preprocessing danych) (do zdobycia 25 punktów bonusowych).

Do każdego z zadań przygotuj raport zawierający opis teoretyczny metody, wykorzystane hiperparametry, wprowadzone modyfikacje, wyniki oraz wnioski wyciągnięte na ich podstawie, materiały dodatkowe oraz krótkie opisy bibliotek wykorzystanych przy implementacji. W podsumowaniu raportu dodatkowo opisz napotkane problemy implementacyjne przy wykonywaniu zadania. Raport wyślij prowadzącemu przynajmniej na 24 godziny przed oddaniem listy.

4 Literatura

- Introduction to MLP
- Introduction to PyTorch
- FaceNet
- M. Kim et al., AdaFace: Quality Adaptive Margin for Face Recognition
- Albumentations