

Przygotowanie i porównanie architektur rozwiązujących problem predykcji odpowiedzi klienta oraz inwestycja wpływu pruningu na jakość rozwiązania

Zadaniem projektu było:

- Stworzenie architektur sieci neuronowych, przewidujących odpowiedź klienta na kolejną ofertę (kupuję/nie kupuję) na podstawie informacji o kliencie oraz jego wcześniejszych zakupach
- Sprawdzenie wpływu pruningu na jakość rozwiązania

Repozytorium kodu projektu :

https://github.com/YuriiBatkovich/MachineLearning2022_2023/tree/main/project

Podczas wykonania projektu udało się:

- przeprowadzić inwestycję danych oraz preprocessing danych , niezbędny dla ich wykorzystania w trenowaniu modeli sieci neuronowych
- przeprowadzić manualny oraz automatyczny (za pomocą RandomForest) feature selection oraz porównać wyniki , które pokazywały modele nauczone na różnych zbiorach cech. Nasz zbiór danych posiadał 28 cech, zarówno manualnie jak i automatycznie z niego zostało wybrano 13 cech.
- stworzyć architektury sieci neuronowych fully connected z kilkoma warstwami (3-4) oraz ewentualnie Batch Normalizacją i/lub DropOut, które rozwiązywały problem na średnim poziomie accuracy – 86%
- na wybranych najlepszych architekturach zbadać wpływ pruningu na wyniki. Były stosowane 4 różne metody strukturalnego i niestrukturalnego pruningu.
- przeprowadzić powyższą procedurę na jeszcze jednym zbiorze danych , żeby zobaczyć ewentualne różnice w wynikach wynikające z charakterystyki danych

Metryki:

Podstawowymi metrykami oceny modeli oraz wpływu pruningu na wyniki były:

- Accuracy
- Precision (była to szczególnie ważna metryka, wynika to z faktu, że spodziewaliśmy się mieć raczej więcej fałszywych pozytywów (klienci, którzy nie zareagują na marketing, ale będą oznaczeni siecią pozytywnie), niż fałszywych negatywów (klienci, którzy przyjęliby ofertę, ale nie byli oznaczeni siecią pozytywnie)
- Recall

Wyniki:

Podczas manualnego oraz atomatycznego feature selection jako typu prepruningu na bazie podstawowego datasetu zostały stwożone dwa zestawy cech. „Manualny” oraz „automatyczny” zbiór cech różniły się o 6 pozycji. Wpomniane dwa zbiory danych zostały

wykorzystane do trenowania modeli wraz z podstawowym datasetem. Jak można było się spodziewać, najlepsze accuracy osiągały modele wyuczone na podstawowym zbiorze danych. Zaskakująco, wyniki trenowania modeli na manualnie stworzonym zbiorze były znacząco lepsze od wyników, które pokazywały modele wytrenowane na automatycznie stworzonym zbiorze. Tak, „manualne” modele pokazywały średnio o 2,25% większe accuracy oraz 0.25 większą wartość precision.

Problem, który pozostał nie rozwiązany przez żadną architekturę, są bardzo niskie wartości recall. One zazwyczaj znajdowały się na poziomie 0.25, jednak czasami tragicznie spadały do wartości bliskich zera (np. 0.048 dla trójwarstwowej fully connected architektury wytrenowanej na datasecie z automatycznie wybranym zbiorem cech).

Dla sprawdzania wpływu pruningu została wybrana trójwarstwowa fully connected architektura oraz podobna trójwarstwowa architektura z batch normalizacją przed każdą warstwą. Pruningowane modele były trenowane na podstawowym zbiorze danych oraz zbiorze danych z manualnie wybranym zbiorem cech.

Były stosowane następujące metody pruningu : losowy niestukturalny pruning parametrów wybranej warstwy modelu, iteracyjny strukturalny pruning wybranej warstwy modelu, niestukturalny losowy pruning parametrów każdej warstwy modelu, globalny pruning parametrów modelu z wyznaczonym poziomem wyłączenia parametrów (u nas – 0.4)

Generalnie, pruning ulepszył zarówno accuracy , jak i precision, oraz znacząco ulepszył wartości recall. Szczególnie dla modeli wytrenowanych na zbiorze danych z manualnie wybranym zbiorem cech korzystanie z pruningu zaowocowało ulepszeniem accuracy średnio na 2%. Wartości recall podwyższyły się o 0.3-0.35 dla większości przetestowanych modeli.

Najlepszy wpływ na rozwiązanie pokazał globalny pruning, który podniósł accuracy do 91% , zaś recall był na poziomie 0.505. Z innej strony, niestukturalny losowy pruning parametrów każdej warstwy modelu przyniósł najmniej korzyści, accuracy , jak i recall pozostały na poprzednich pozycjach, w niektórych przypadkach nawet spadały, też spadała wartość precision, co jest szczególnie negatywnym wskaźnikiem w naszym przypadku.

Dla zbadania ewentualnych różnic w wynikach wynikających z charakterystyki danych, postanowiłem przeprowadzić powyższą procedurę na innym zbiorze danych, zawierającym dane o klientach linii lotniczych. Ten zbiór danych był 10 razy większy do poprzedniego.

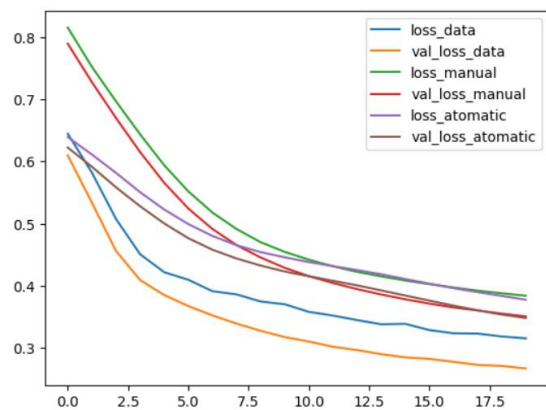
Zaskakująco, tym razem manualnie wybrany dataset znów okazał się lepszy od automatycznie stworzonego. Nie występowało problemu tak jawnie małych, niezadowalających wartości recall. Wpływ różnych metod pruningu można opisać jako wręcz identyczny, co prawda mniej widoczny na wynikach.

Na następnej stronie można zobaczyć przykładowe wykresy krzywej trenowania, pozostałe dokładne wyniki oraz wykresy proszę szukać w opisach zamieszczonych w repozytorium kodu.

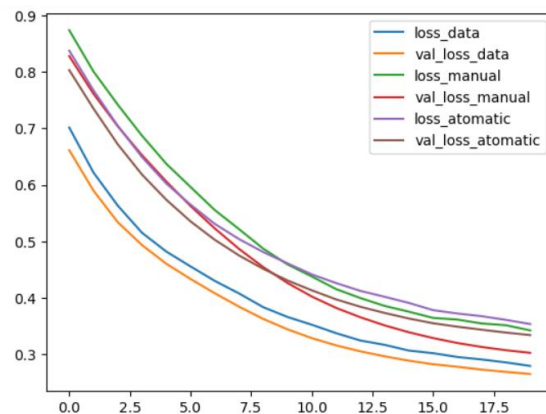
Trenowanie bez pruningu:

Trójwarstwowa fully connected architektura

zwykła



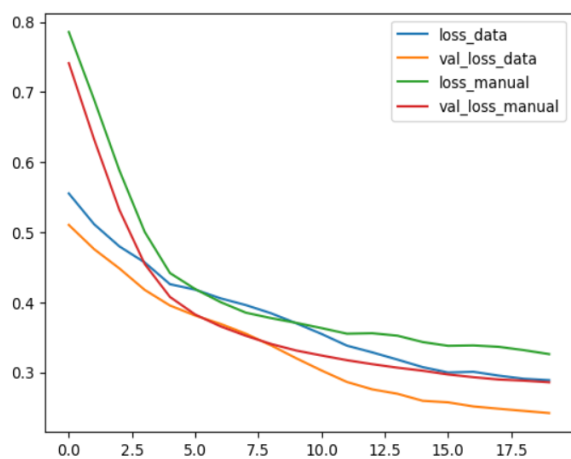
z wykorzystaniem DropOut



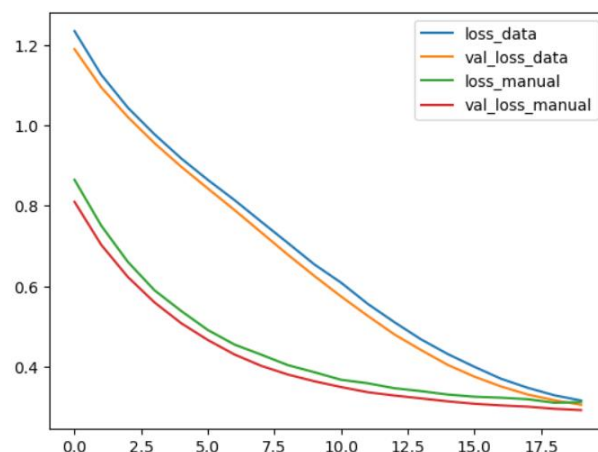
Trenowanie z pruningiem:

Global pruning

zwykła



z wykorzystaniem Batch Normalizacji



Yurii Titov