Przygotowanie i porównanie architektur rozwiązujących problem predykcji odpowiedzi klienta oraz inwestygacja 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/YuriiBatkovych/MachineLearning2022_2023/tree/main/project

Podczas wykonania projektu udało się:

- przeprowadzić inwestygację danych oraz preprocessing danych , niezbędny dla ich wykorzystania w trenowaniu modeli sieci neuronowych

- przeprowadzić manualny oraz automatyczny (za pomoczą 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 Normalzacją 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 róznice 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.

W ostateczej analizie były brane pod uwagę trzy trójwarstwowe modele sieci neuronowych o architekturze 25-5-2, z funkcją aktywacji relu po pierwzszej i drugiej warstwie (simple architecture). Drugi z trzech modeli dodatkowo miał zdefioniowane BatchNormalization po pierwszej i drugiej warstwie (neural network with normalisation), ostatni model miał zdefiniowany Dropout po pierwszej i drugiej warstwie z prawdopodobieństwem wyłączenia neurona 0.3 (neural network with dropout)

Nazwa modelu	Accuracy	Precision	Recall
Simple – cały dataset	90.462%	0.814	0.385
Simple – manualny	88.045%	0.774	0.194
Simple - automatyczny	86.704%	0.857	0.048
Dropout – cały dataset	88.972%	0.840	0.231
Dropout – manualny	88.380%	0.857	0.194
Dropout - automatyczny	86.145%	0.532	0.266
BN – cały dataset	89.866%	0.689	0.462
BN – manualny	88.827%	0.740	0.298
BN – automatyczny	86.592%	0.532	0.266

Jak można było się spodziewać, najlepsze accuracy osiągały modele wyuczone na podstawowym zbiorze danych. Zaskakująco, wyniki trenowania modeli na manualnie stworzym zbiorze były znacząco lepsze od wyników, które pokazywały modele wytrenowane na automatycznie stworzonym zbiorze. Problem, który pozostał nie rozwiązany przez żadną architekturę, są bardzo niskie wartości recall.

Dla sprawdzania wpływu pruningu została wybrana trójwarstwowa fully connected architektura oraz architektura z Batch Normalizacją po każdej warstwie. Pruningowane modele były trenowane na podstawowym zbiorze danych oraz zbiorze danyh z manualnie wybranym zbiorem cech.

Były stosowane następujące metody pruningu: losowy niestrukturalny pruning parametrów wybranej warstwy modelu, iteracyjny strukturalny pruning wybranej warstwy modelu, niestrukturalny losowy pruning parametrów każdej warstwy modelu, globalny pruning paramentrów 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 danyh 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.

Architektura-Dataset	Accuracy	Precision	Recall
losowy niestrukturalny pruning parametrów wybranej warstwy modelu			
Simple-cały dataset	86.438%	0.715	0.345
Simple – manualny	90.015%	0.731	0.418
BN – cały dataset	89.866%	0.702	0.440
BN – manualny	89.568%	0.648	0.505
iteracyjny strukturalny pruning wybranej warstwy modelu			
Simple-cały dataset	90.611%	0.684	0.571
Simple – manualny	90.015%	0.740	0.407
BN – cały dataset	90.462%	0.714	0.495
BN – manualny	89.121%	0.714	0.330
globalny pruning paramentrów modelu z wyznaczonym poziomem wyłączenia parametrów			
Simple-cały dataset	91.356%	0.780	0.505
Simple – manualny	90.164%	0.755	0.407
BN – cały dataset	88.972%	0.613	0.505
BN – manualny	88.227%	0.625	0.330
niestrukturalny losowy pruning parametrów każdej warstwy modelu,			
Simple-cały dataset	90.313%	0.732	0.451
Simple – manualny	89.419%	0.727	0.352
BN – cały dataset	89.568%	0.652	0.495
BN – manualny	88.525%	0.694	0.275

Najlepszy wpływ na rozwiązanie pokazał globalny pruning, który podniósł accuracy do 91% , zaś recall był na poziomie 0.505. Z innej strony, niestrukturalny 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 szczególne negatywnym wskaźnikiem w naszym przypadku.

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

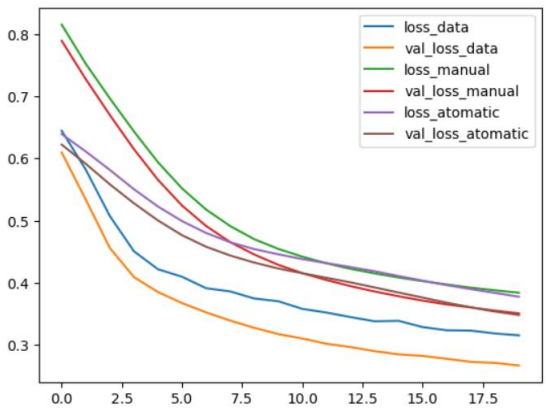
Zaskakująco, tym razem manualnie wybrany dataset znów okazał się lepszy od automatycznie stworzonego. Nie występowało problemu tak jawnie małych, niezadowalających wartości recall. Wpływ różnych metod pruningu można opisać jako wręcz identyczny, co prawda mniej widoczny na wynikach.

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

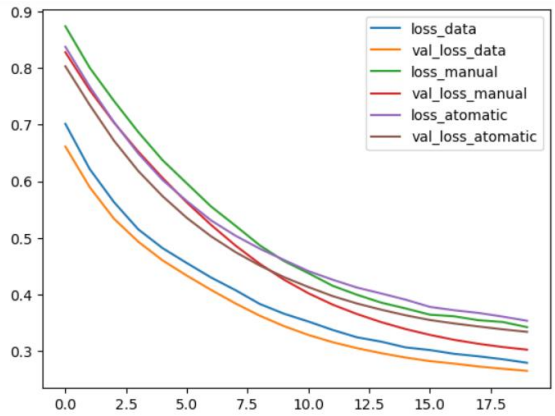
Trenowanie bez pruningu:

Trójwarstwowa fully connected architektura

zwykła



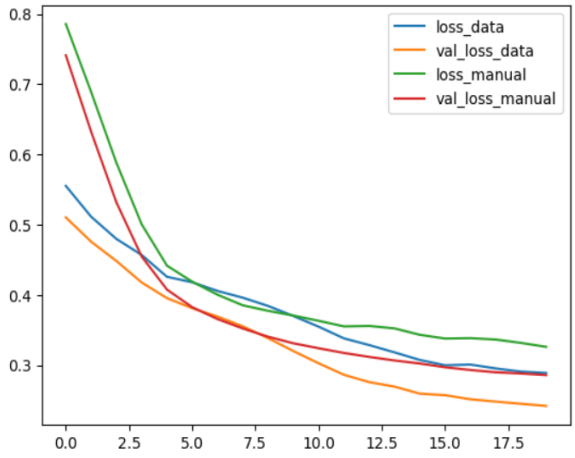
z wykorzystaniem DropOut



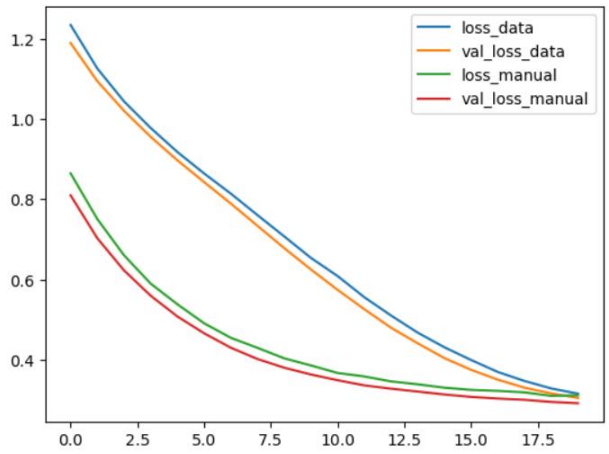
Trenowanie z pruningiem:

Global pruning

zwykła



z wykorzystaniem Batch Normalizacji



Yurii Titov