Projekt 4 - raport

Sieci Neuronowe

Urszula Żukowska
Paweł Młyniec

28 lutego 2021

Spis treści

1 Wstęp **2**

2 Przygotowanie danych **2**

3 Opis zaimplementowanych sieci neuronowych **3**

4 Opis i wyniki przeprowadzonych testów **7**
4.1 Regresja 7
4.1.1 Porównanie architektur 7
4.1.2 Wpływ danych wejściowych na wyniki 7
4.1.3 Metoda *ensembling* 8
4.1.4 Predykcja z krokiem pośrednim 8
4.2 Klasyfikacja 9
4.2.1 Porównanie architektur 9
4.2.2 Wpływ danych wejściowych na wyniki 9
4.2.3 Metoda *ensembling* 9
4.2.4 Predykcja z krokiem pośrednim 10

5 Podsumowanie **10**

1 Wstęp

Celem projektu była predykcja pogody na podstawie danych historycznych. Dane pochodzą z kolejnych dni i zawierają informacje o wilgotności, ciśnieniu, temperaturze, prędkości i kierunki wiatru oraz słowny opis pogody. Pomiary wykonywane były co godzinę w 36 miastach.

Zadanie składało się z dwóch podproblemów:

- predykcja średniej dziennej temperatury
- odpowiedź na pytanie, czy wiatr będzie silny (przyjmujemy że silny wiatr, to wystąpienie przynajmniej jednego punktu pomiarowego w danym dniu, gdzie szybkość wiatru była $\geq 8\frac{m}{s}$)

Predykcja została wykonana dla dnia następnego, czyli z pominięciem jednego dnia pomiędzy trzema dniami wykonywania pomiarów, a dniem, dla którego przeprowadzamy predykcję.

Projekt został zrealizowany przy użyciu sieci MLP z metodami uczenia bazującymi na propagacji wstecznej błędu.

W raporcie opisane została architektura zaimplementowanych sieci wykorzystanych w zadaniu predykcji, a także testy wykonane w ramach zadania.

2 Przygotowanie danych

Dane wykorzystywane do predykcji czytane są z plików *csv*. Każdy atrybut zapisany jest w oddzielnym pliku (np. wilgotność powietrza w pliku *humidity.csv*, a temperatura w pliku *temperature.csv*). Przed wprowadzeniem danych do sieci są one odpowiednio łączone, w niektórych przypadkach również przetwarzane.

Wstępna obróbka danych składa się z następujących kroków:

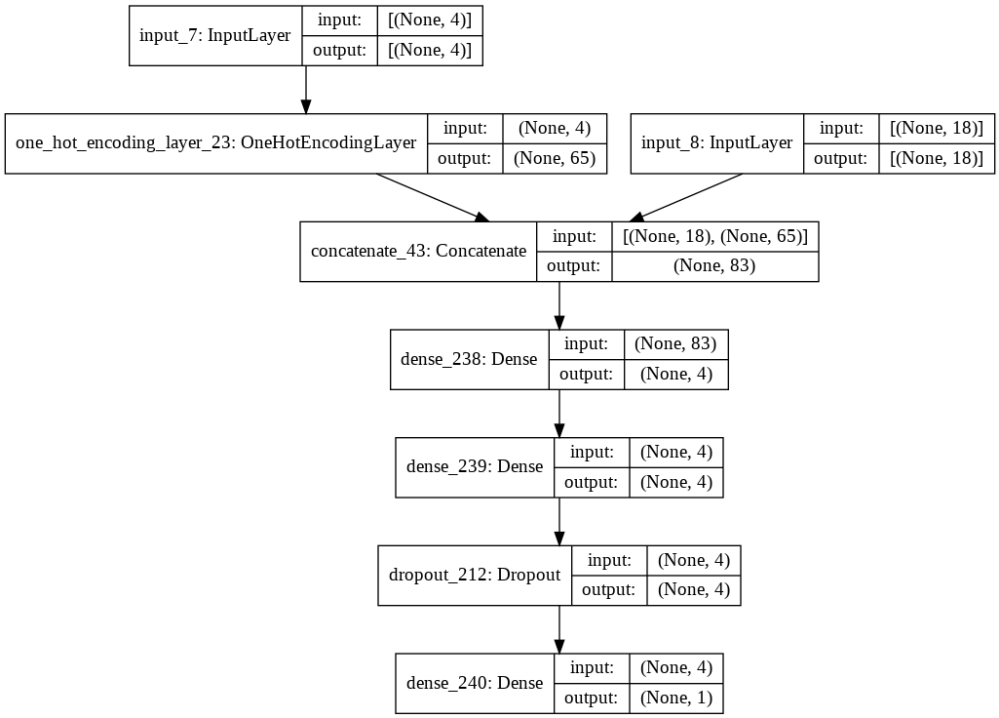
1. Dane są wczytane z plików do słownika zawierającego kolejne atrybuty.
2. Braki w danych są uzupełniane wartościami kolejnymi; jeśli nadal występują braki są one uzupełniane wartościami poprzednimi.
3. Dane z plików są obracane i łączone (tak by dla każdego miasta i daty w wierszu, były kolejne cechy pogody.
4. Dane następnie są agregowane po całym dniu. W przypadku danych numerycznych po średniej z dnia (wyjątkiem jest siła wiatru gdzie agregacji dokonuje się po średniej i maksymalnej sile). Dla danych kategorycznych jest to moda z dnia.
5. Kolejno dane są łączone w serie po 3 lub 4 dni, tak by cała seria składała się z następujących po sobie dniach oraz dotyczyła tego samego miasta.
6. Po połączeniu danych w serię wartość prędkości wiatru z piątego dnia zamieniany jest na wartość kategoryczną (w zależności od tego czy wieje silniej czy słabiej niż $8\frac{m}{s}$) i przeprowadzany jest *one hot encoding*.
7. Dla niektórych danych są też zdefiniowane nowe zmienne metodą PCA.
8. Na koniec obróbki wstępnej dane są dzielone na kategoryczne i numeryczne oraz wydzielone zostają odpowiednie etykiety.

3 Opis zaimplementowanych sieci neuronowych

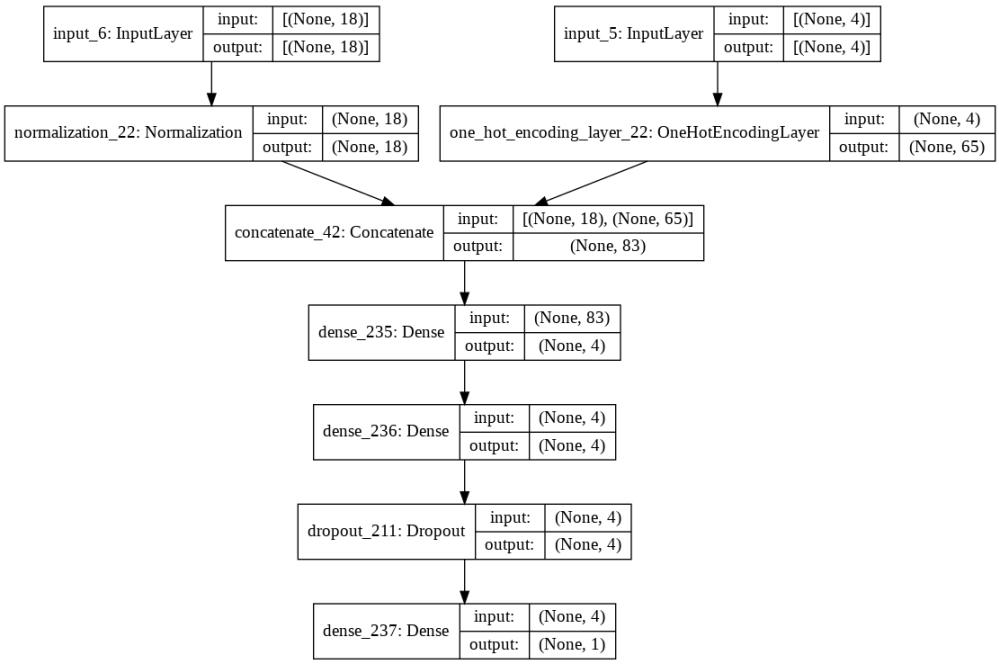
W ramach projektu problemy zostały rozwiązane na 6 różnych sposobów, po trzy na problem regresji i klasyfikacji. Do wszystkich sposobów został użyty perceptron wielowarstwowy z algorytmem uczenia *Adam* z biblioteki *Tensorflow Keras*.

Pierwszy sposób polega na predykcji siły wiatru lub temperatury piątego dnia na podstawie trzech pierwszych dni.

Schemat takiej sieci wygląda następująco.



Rysunek 1: Sieć - model podstawowy



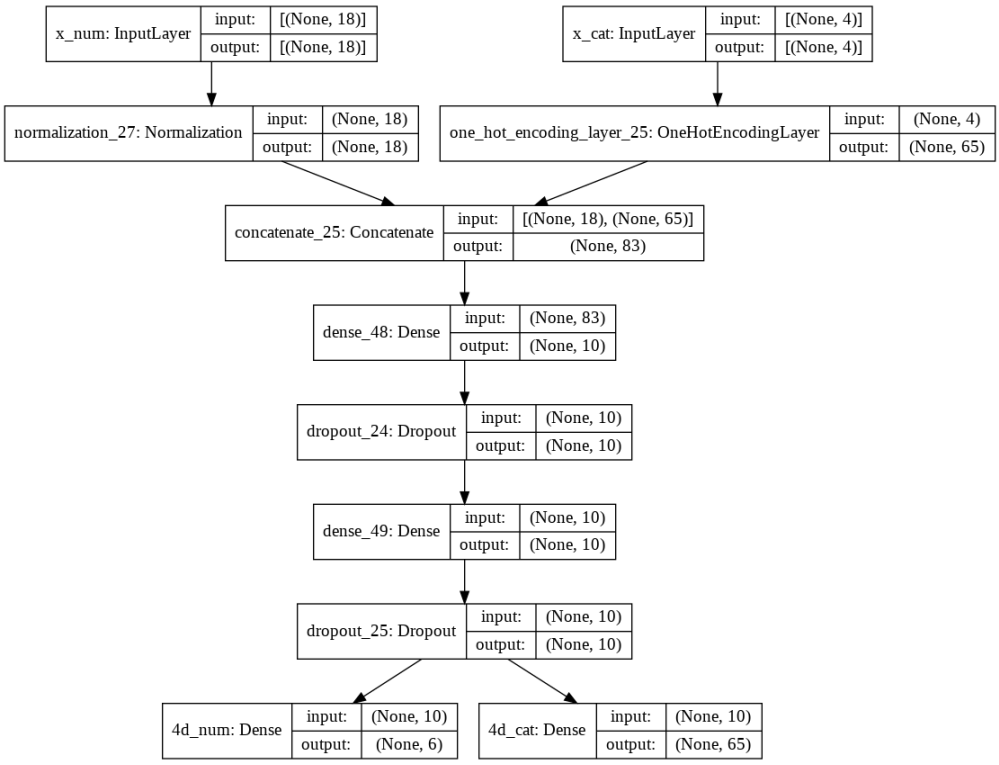
Rysunek 2: Sieć - model podstawowy z normalizacją

Model tej sieci posiada dwa wejścia. Jedno przyjmuje dane numeryczne, drugie kategoryczne. W podstawowej wersji dane numeryczne po obróbce wstępnej nie są poddawane żadnym operacjom i w niezmienionej formie są przekazywane dalej.

W kolejnej wersji dane numeryczne są normalizowane przechodząc przez warstwę *Normalization* (dodaną na schemacie 2), która to przy tworzeniu wylicza odpowiednie parametry z danych wejściowych.

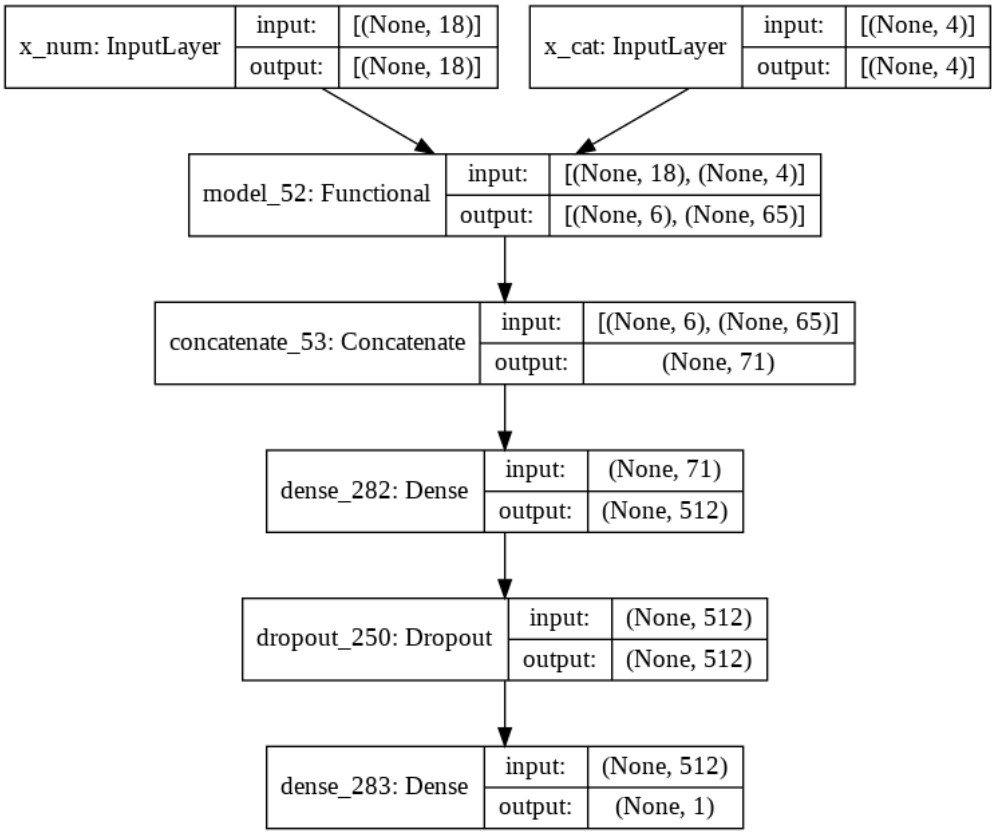
Dla danych kategorycznych została stworzona warstwa *One Hot Encoding*. Warstwa ta na początku dla wszystkich danych treningowych tworzy słownik wyrazów, który następnie jest mapowany na wektor zer i jedynek. Słownik ten posiada tyle wyrazów ile jest kategorii we wszystkich obserwacjach plus dwie na dane puste i niewystępujące w słowniku.

Po przetworzeniu w warstwach danych, wyniki z warstw są następnie łączone w jeden w warstwie *Concatinate*. Potem jest n warstw gęstych ułożonych naprzemiennie z warstwami dropoutu, by zwrócić wynik w warstwie wynikowej.



Rysunek 3: Sieć - model predykcji dnia czwartego

Drugi sposób polega na predykcji dnia pośredniego. W tym sposobie została stworzona pierwsza sieć przewidująca dane dnia czwartego (na schemacie 3). Sieć ta jest skonstruowana podobnie jak poprzednia, z wyjątkiem wyjść. Zostały skonstruowane dwa wyjścia na dane numeryczne i kategoryczne. Wyjście numeryczne przewiduje sześć cech z czwartego dnia, a wyjście kategoryczne przewiduje jedną z 65 kategorii dnia czwartego. Do obliczania funkcji straty kategorycznej wykorzystywany jest ten sam słownik który znajduje się w warstwie *One Hot Encoding*.



Rysunek 4: Sieć - model predykcji dnia czwartego

Sieć ta zostaje wstępnie wytrenowana w przewidywaniu dnia czwartego, a następnie jest umieszczana w sieci przewidującej cel tego zadania. Schemat tej sieci jest przedstawiony na rysunku 4.

Warstwa *Functional* jest warstwą składającą się z poprzedniego modelu, a dalej znajdują się naprzemienne warstwy gęste i dropoutu.

Ostatnią techniką są złożenia modeli. Do tej metody wykorzystywane są modele podstawowe z normalizacją. Wyniki uzyskane z n niezależnych sieci są następnie odpowiednio łączone i zwracany jest ostateczny wynik uzyskany metodą *ensembling*. Dokładny opis sposobu łączenia modeli i uzyskanych wyników opisany jest w rozdziale 4.

Dla wszystkich sieci jest ustalone wczesne zatrzymanie jeśli strata na zbiorze walidacyjnym nie polepsza się. Warunkiem zatrzymania jest brak poprawy o 0.001 na odcinku 10 epok.

4 Opis i wyniki przeprowadzonych testów

W ramach projektu wykonano kilka rodzajów testów. Były to porównanie różnych architektur, zbadanie wpływu danych wejściowych na uzyskane rezultaty (dane bez przekształceń, dane znormalizowane oraz poddane PCA), porównanie wyników z pojedynczych modeli do wyników uzyskanych metodą *ensembling* oraz porównanie predykcji z pominięciem dnia czwartego i predykcji z krokiem pośrednim, w którym następuje predykcja dla czwartego dnia.

4.1 Regresja

Na potrzeby testów została wprowadzona miara poprawności uzyskanych wyników. Predykcja $\hat{y}$ jest uznawana za poprawną, jeśli dla wartości prawdziwej y spełnia ona następujący warunek: $|\hat{y} - y| \leq 2$. Wartość *accuracy* to procent uzyskanych predykcji, które są poprawne zgodnie z opisanym warunkiem. Do wyznaczania tej miary została stworzona nowa metryka w Tensorflow.

4.1.1 Porównanie architektur

Dla problemu regresji (czyli predykcji średniej temperatury) porównane zostały 72 różne konfiguracje sieci. Zmieniane były następujące hiperparametry:

- liczba neuronów w warstwie = $\{128, 256, 512\}$
- liczba warstw = $\{6, 9, 12\}$
- współczynnik *droupoutu* $\in [0, 0.1]$
- współczynnik uczenia wybierany metodą *random search* $\in [10^{-5}, 10^{-2}]$

Użyta została funkcja aktywacji *ReLu* i dwie funkcje straty - *MAE* oraz *MSE*. Do aktualizacji wartości wag i uczenia sieci użyta jest metoda *Adam* (*Adaptive Moment Estimation*).

Wyniki testów różnych architektur zostały przedstawione za pomocą TensorBoarda: `https://tensorboard.dev/experiment/SaoJVRGuQbyMDm7WGsBs4Q/#scalars`. Poniżej przedstawione zostały najlepsze wyniki.

neurons	layers	lr	loss fcn	preproc	train acc	val acc	train loss	val loss	test acc	test loss
256	12	6.3e-05	mae	norm	0.23	0.34	5.96	4.69	0.35	4.58
512	9	0.0015	mae	norm	0.23	0.35	4.84	4.59	0.35	4.63
512	12	0.0034	mae	norm	0.24	0.33	5.43	4.75	0.35	4.63
512	6	6.3e-05	mae	norm	0.27	0.34	5.07	4.64	0.35	4.53
512	6	0.0034	mae	norm	0.27	0.34	5.08	4.66	0.34	4.53

Tablica 1: Wyniki testów różnych architektur dla regresji

4.1.2 Wpływ danych wejściowych na wyniki

W celu zbadania wpływu danych wejściowych na uzyskiwane rezultaty porównane zostały wyniki uzyskane dla danych niezmienionych, znormalizowanych oraz poddanych PCA. Zostały one przedstawione poniżej.

data preproc	max acc	mean acc	min acc
normalized	0.35	0.21	0.014
pca	0.22	0.16	0.08
raw	0.25	0.05	0.0007

Tablica 2: Wpływ postaci danych wejściowych na wyniki

Zdecydowanie najlepsze wyniki sieć uzyskiwała dla danych znormalizowanych. Najlepsza dokładność na zbiorze testowym wynosiła 0.35. Drugie w kolejności są wyniki otrzymane dla zmiennych uzyskanych w wyniku PCA. Maksymalna dokładność dla tego zbioru danych wynosi 0.22. Najgorsze wyniki uzyskano dla nieprzetworzonych danych. Pomimo, że najlepsza dokładność dla tych danych wynosi 0.25, czyli więcej niż dla danych z PCA, to jednak średnia dokładność jest znacznie gorsza.

4.1.3 Metoda *ensembling*

Metoda *ensembling* polega na łączeniu wyników kilku sieci i łączeniu ich w odpowiedni sposób (w zależności od rozwiązywanego problemu). W ten sposób można uzyskać lepszy wynik końcowy niż biorąc pod uwagę jedynie wynik pojedynczej sieci.

Dla problemu regresji zastosowano dwie metody łączenia wyników sieci:

- obliczanie średniej wartości wszystkich uzyskanych wyników
- obliczanie średniej wartości uzyskanych wyników odrzucając wartości maksymalne i minimalne

Testy zostały przeprowadzone dla architektur, które uzyskały najlepsze wyniki w poprzednich testach. Zostało użytych 10 modeli (10 niezależnych od siebie sieci).

neurons	layers	lr	loss fcn	preproc	all models mean acc	models w/o outliers mean acc
256	12	6.3e-05	mae	norm	0.27	0.54
512	9	0.0015	mae	norm	0.29	0.59
512	12	0.0034	mae	norm	0.33	0.66
512	6	6.3e-05	mae	norm	0.30	0.61
512	6	0.0034	mae	norm	0.33	0.65

Tablica 3: Wyniki testów metody *ensembling* dla regresji

Wyniki uzyskane pierwszym sposobem (poprzez obliczenie średniej z wyników wszystkich modeli) daje nieco gorsze wyniki niż pojedyncze modele. Natomiast wyniki uzyskane drugim sposobem są już zdecydowanie lepsze, otrzymana dokładność jest prawie 2 razy większa niż dla pojedynczych sieci.

4.1.4 Predykcja z krokiem pośrednim

Dla problemu regresji porównano 32 konfiguracje sieci. Zmieniane były następujące hiperparametry:

- liczba neuronów w warstwie $= \{128, 512\}$
- liczba warstw $= \{4, 8\}$
- współczynnik *droupoutu* $\in [0.1, 0.1]$
- współczynnik uczenia wybierany metodą *random search* $\in [10^{-5}, 10^{-2}]$
- funkcja straty $\in [mse, mae]$

Wyniki testów różnych architektur zostały przedstawione za pomocą TensorBoarda: `https://tensorboard.dev/experiment/sD7eB3eVT8OtPGC52gxUew/`. Poniżej przedstawione zostały wybrane wyniki.

neurons	layers	lr	activation	train acc	validate acc	test acc	test loss
512	4	0.00025	mse	0.21	0.32	0.349	32
128	4	0.0014	mse	0.196	0.176	0.176	57
512	4	0.00002	mae	0.199	0.32	0.35	4.18

Tablica 4: Wyniki testów różnych architektur dla regresji z predykcją dnia pośredniego

Dla najlepszych architektur wyniki są porównywalne z predykcją bez dnia pośredniego. Trafność jest na tym samym poziomie, natomiast strata na zbiorze testowym jest mniejsza (4.18 z predykcją dnia pośredniego, 4.53 bez)

4.2 Klasyfikacja

4.2.1 Porównanie architektur

Dla problemu klasyfikacji (czyli predykcji czy dany dzień był wietrzny) porównano 48 konfiguracji sieci. Zmieniane były następujące hiperparametry:

- liczba neuronów w warstwie = $\{128, 256, 512\}$
- liczba warstw = $\{4, 8, 12\}$
- współczynnik *droupoutu* $\in [0.1, 0.1]$
- współczynnik uczenia wybierany metodą *random search* $\in [10^{-5}, 10^{-2}]$

Użyta została funkcja aktywacji *ReLu* a w warstwie wyjścowej *sigmoid*i funkcja straty - *Binary Cross Entropy*. Do aktualizacji wartości wag i uczenia sieci użyto metodę *Adam* (*Adaptive Moment Estimation*).

Wyniki testów różnych architektur zostały przedstawione za pomocą TensorBoarda: `https://tensorboard.dev/experiment/xNkz8VaGQqCEQPZjKhXZrw/#hparams`. Poniżej przedstawione zostały wybrane wyniki.

neurons	layers	lr	preproc	train acc	validate acc	train loss	validate loss	test acc	test loss
512	4	0.0014	raw	0.83	0.831	0.422	0.421	0.74	0.548
128	4	0.000443	raw	0.83	0.831	0.422	0.423	0.74	0.546
512	4	0.0014	pca	0.83	0.83	0.444	0.448	0.739	0.589
512	4	0.0014	norm	0.99	0.798	0.028	1.4	0.7	2.4

Tablica 5: Wyniki testów różnych architektur dla klasyfikacji

4.2.2 Wpływ danych wejściowych na wyniki

W teście architektur najlepiej sobie poradziły dane nieprzetworzone. Dokładność na zbiorze testowym wyniosła maksymalnie 0.74. Dane znormalizowane zostały przeuczone co doprowadziło do kiepskich wyników na zbiorze testowym. Najlepsza architektura składa się z 512 neuronów w 4 warstwach, z współczynnikiem uczenia 0.0014.

data preproc	max acc	mean acc	min acc
normalized	0.703	0.69	0.67
pca	0.739	0.72	0.68
raw	0.74	0.739	0.738

Tablica 6: Wpływ postaci danych wejściowych na wyniki

4.2.3 Metoda *ensembling*

W przypadku klasyfikacji zastosowano następujące metody:

- *hard voting* - wybranie klasy przy pomocy głosowania większościowego; jako wynik ostateczny przyjmujemy tę wartość, która pojawia się najczęściej wśród wyników rozpatrywanych modeli
- *soft voting* - pod uwagę bierzemy prawdopodobieństwo z jakim otrzymany został dany wynik; wynikiem końcowym jest klasa, dla której suma prawdopodobieństw ze wszystkich modeli jest największa

Testy zostały przeprowadzone dla architektur, które uzyskały najlepsze wyniki w poprzednich testach. Zostało użytych 10 modeli (10 niezależnych od siebie sieci).

neurons	layers	lr	preproc	hard voting acc	soft voting acc
512	4	0.0014	raw	0.74	0.74
128	4	0.000443	raw	0.74	0.74
512	4	0.0014	pca	0.74	0.74
512	4	0.0014	norm	0.74	0.72

Tablica 7: Wyniki testów metody *ensembling* dla klasyfikacji

Rezultaty uzyskane za pomocą metody *ensembling* prawie w ogóle nie różnią się od wyników pojedynczej sieci. Nie można jednoznacznie stwierdzić czy zastosowanie tej metody może poprawić wynik predykcji w rozpatrywanym problemie.

4.2.4 Predykcja z krokiem pośrednim

Dla predykcji z krokiem pośrednim hiperparametry zostały sprawdzana tak jak w sekcji 4.2.1.

Wyniki są przedstawione za pomocą TensorBoarda:
`https://tensorboard.dev/experiment/BaIEUVj6QJOfN1hOQGVRUQ/`.

Niestety z powodu błędu w zapisie wyników nie zapisały się dokładnie straty na zbiorach walidacyjnych i testowych. Zapisały się natomiast wyniki na zbiorze testowym i są przedstawione poniżej:

neurons	layers	lr	train acc	validate acc	test acc	test loss
512	4	0.009	0.855	0.8	0.848	0.369
128	4	0.0059	0.843	0.82	0.840	0.393
512	8	0.009	0.83	0.83	0.831	0.374

Tablica 8: Wyniki testów różnych architektur dla klasyfikacji z predykcją dnia pośredniego

W tym wypadku wynik okazał się dużo lepszy niż w wypadku podstawowego modelu. Najlepszy wynik uzyskano dla 512 neuronów w 4 warstwach ze współczynnikiem uczenia 0.009. Dokładność na zbiorze testowym wyniosła 0.848 a strata 0.0369.

5 Podsumowanie

Na podstawie przeprowadzonych testów w zadaniu predykcji najlepiej sobie poradziła metoda *ensemblingu* z odrzucaniem wartości maksymalnych i minimalnych, zyskując trafność na poziomie 0.66)

Widać też, że modele z większą liczba neuronów radzą sobie lepiej, a wynik nie zależy tak bardzo od liczby warstw, dając czasem lepsze wyniki przy ich mniejszej liczbie. Z funkcji straty najlepszy okazał się średni błąd. Nie jest to zaskoczeniem, ponieważ chcieliśmy uzyskać wyniki jak najbardziej średnio zbliżone do docelowych wartości. Normalizacja dała najlepsze wyniki, a PCA wypadło gorzej nie przynosząc zysku ze zdefiniowania nowych wartości. Predykcja z dniem pośrednim wypadła porównywalnie do predykcji bez dnia pośredniego, uzyskując jedynie trochę lepszą wartość straty

Dla klasyfikacji najlepszy okazał się model z predykcją dnia pośredniego, uzyskując trafność na zbiorze testowym 0.848.

W tym problemie modele z mniejszymi liczbami neuronów radziły sobie porównywalnie dobrze jak te z większymi. Porównując wyniki uzyskane dla różnych typów danych wejściowym zaskoczeniem jest kiepski wynik dla danych znormalizowanych. Modele wyraźnie się przeuczały dając wprawdzie lepsze wyniki na zbiorach treningowych ale znacząco gorsze na testowych. Zdefiniowanie nowych zmiennych metodą PCA dało dobre wyniki, lecz gorsze od użycia danych nieprzetworzonych. Dość duży potencjał widać w metodzie z predykcją z dniem pośrednim i rozwijając testy w tym kierunku jest szansa na uzyskanie lepszych wyników.