Sieci neuronowe

19 stycznia 2024

Mikołaj Oganiaczyk 264263

Streszczenie

Badanie sieci neuronowej mającej za zadanie rozpoznawanie kotów i psów na zdjęciach dostarczonych przez użytkownika. Porównanie działania sieci dla różnych modeli oraz rozmiarów danych wejściowych.

Spis treści

1 Temat projektu

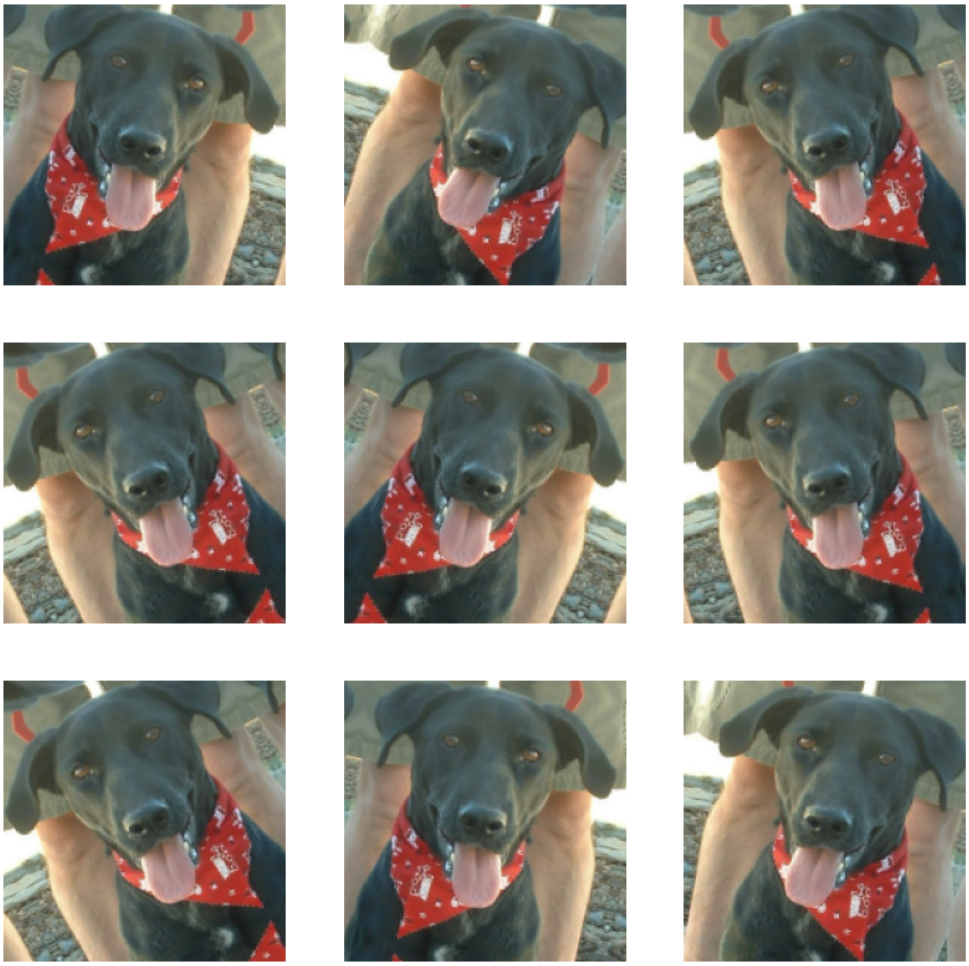
Sieć neuronowa klasyfikująca psy i koty na podstawie obrazów dostarczanych na jej wejście. Do uczenia sieci wykorzystano bibliotekę Keras oraz zbiór danych: https://www.kaggle.com/datasets/shaunthesheep/microsoft-catsvsdogs-dataset/data. Po odrzuceniu uszkodzonych plików, otrzymaliśmy 18728 plików w zestawie uczącym oraz 4682 plików w zestawie walidacyjnym.

Zbiór zawiera kolorowe obrazy psów i kotów z różnych perspektyw, co zwiększa trudność poprawej klasyfikacji dla sieci neuronowej. Obrazy zapisane są w formacie .jpg w różnych rozmiarach (zostaną one przeskalowane do rozmiaru wejścia sieci w trakcie wczytywania danych).

Przykładowe obrazy z zestawu danych:

Rysunek 1: Przykładowe zdjęcia kotów.

Rysunek 2: Przykładowe zdjęcia psów.



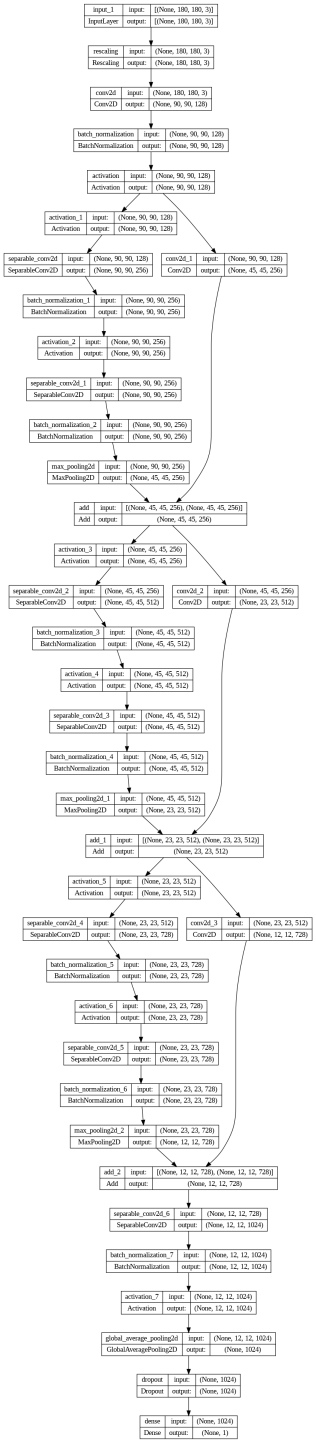
Rysunek 3: Przykładowe przekształcenia obrazu w celu zwiększenia liczby obrazów w zbiorze.

2 Przebieg uczenia sieci

1. Zaimportowanie danych do pamięci środowiska wykonawczego Google colab oraz wykluczenie ze zbioru uszkodzonych plików.
2. Przygotowanie danych dla sieci:
   - Określenie rozmiaru obrazu na wejściu sieci.
   - Podział na zbiór uczący oraz walidacyjny.
   - Ustawienie wartości seed aby uzyskać jednakowe wyniki przy przyszłych wywołaniach.
3. Dodajemy wersję obrazów obrócone o niewielki kąt lub odbite w celu zwiększenia liczby danych uczących.
4. Tworzymy model sieci o wybranym kształcie z wykorzystaniem wybranych funkcji aktywacji oraz ustawiamy jego parametry.
5. Wybieramy liczbę epok, metrykę jaką mierzymy postęp uczenia sieci oraz funkcję optymalizacji.
6. Rozpoczynamy uczenie sieci.

3 Jeden z modeli sieci

Pierwszy model sieci wykorzystany do realizacji projektu.



Rysunek 4: Graficzne przedstawienie modelu. Opisy warstw znajdują się poniżej.

4 Opis działania modelu

1. Zdefiniowanie kształtu wejścia sieci, w przypadku powyższej sieci 180;180.
2. Zdefiniowanie bloku wejściowego sieci odpowiedzialnego za:
   - Przeskalowanie wartości pikseli z zakresu 0-255 na 0-1.
   - Zastosowanie filtrów konwolucyjnych w celu wykrycia wzorców na obrazie oraz zmniejszenie przestrzennego rozmiaru obrazu z jednoczesnym zachowaniem istotnych dla sieci informacji.
   - Normalizację zwiększającą stabilność sieci oraz prędkość uczenia.
   - Wybieramy funkcję aktywacji warstwy w przypadku powyższego modelu jest to funkcja relu.
3. Dodanie do modelu wybranej ilości bloków warstw o zdefiniowanym rozmiarze w przypadku powyższej sieci są to bloki o rozmiarach 256, 512, 728. Każdy z bloków składa się z:
   - Dwóch bloków złożonych z:
     - Funkcji aktywacji w powyższym modelu - relu.
     - Warstwy SeparableConv2D pozwala ona na przeprowadzenie konwolucji podzielonej na etapy i zebranie wyników co zmniejsza ilość wymaganych obliczeń.
     - Warstwy normalizującej.
   - Warstwy MaxPooling2D, która odpowiada za zmniejszenie rozmiaru mapy cech poprzez wybranie maksymalnej wartości z obszaru o określonym rozmiarze(w przypadku powyższej sieci 3x3).
   - Warstwa konwolucyjna z dodawaniem wyniku poprzedniej funkcji aktywacji. Pozwala to na uczenie sieci cech szczególnych wybranego zestawu danych.
4. Warstwa GlobalAveragePooling2D służąca do obliczenia wartości średniej ze wszystkich cech. Pozwala na zredukowanie wymiarów przestrzennych w celu dalszego przetwarzania.
5. Warstwa dropout wyłączająca losowo wybrane fragmenty modelu na podstawie podanego prawdopodobieństwa w powyższej sieci(0.5). Ma na celu zapobieganie przeuczeniu sieci.
6. Warstwa dense odpowiedzialna za zebranie wyników z pozostałych warstw i dopasowanie wartości wejściowych do poszczególnej klasy na wyjściu.

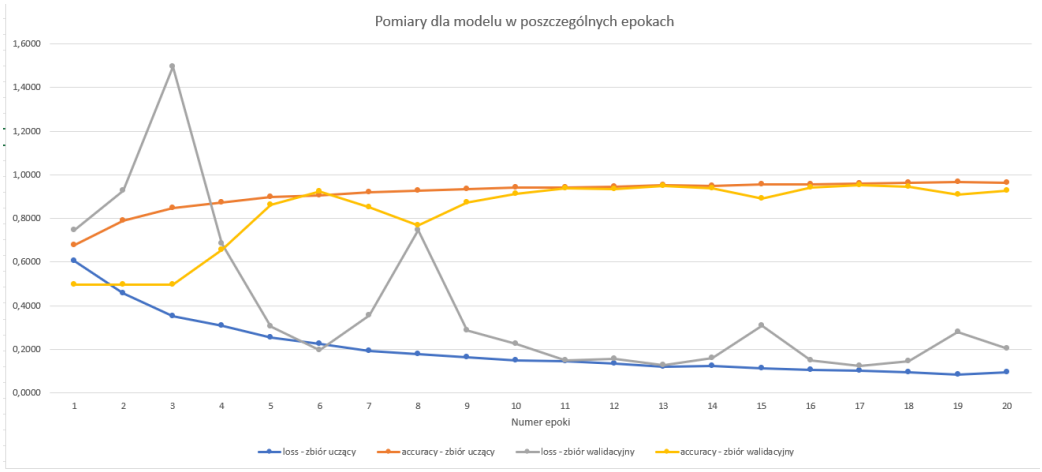
Funkcją aktywacji jest sigmoid ponieważ sieć ma za zadanie rozwiązywanie problemu z 2 klasami na wyjściu.

5 Uczenie sieci

Uczenie sieci przez 20 epok.

Nr epoki	Loss	acc	val_los	val_acc
1	0,6058	0,6761	0,7464	0,4957
2	0,4578	0,7883	0,9281	0,4957
3	0,3525	0,8469	1,4969	0,4957
4	0,3066	0,8710	0,6863	0,6572
5	0,2534	0,8976	0,3054	0,8635
6	0,2245	0,9067	0,1943	0,9227
7	0,1910	0,9211	0,3553	0,8528
8	0,1791	0,9268	0,7472	0,7668
9	0,1618	0,9344	0,2882	0,8721
10	0,1479	0,9415	0,2241	0,9114
11	0,1460	0,9398	0,1478	0,9393
12	0,1351	0,9464	0,1579	0,9346
13	0,1203	0,9533	0,1269	0,9498
14	0,1238	0,9499	0,1581	0,9376
15	0,1119	0,9545	0,3088	0,8891
16	0,1063	0,9570	0,1507	0,9413
17	0,1015	0,9594	0,1248	0,9517
18	0,0937	0,9625	0,1469	0,9460
19	0,0850	0,9651	0,2778	0,9086
20	0,0937	0,9616	0,2041	0,9263

Rysunek 5: Przebieg uczenia sieci.



Rysunek 6: Wykres przebiegu uczenia sieci.

Jako metrykę do pomiaru skuteczności działania sieci w trakcie uczenia wykorzystano klasę BinaryAccuracy z biblioteki keras. Klasa ta pozwala na wyliczenie wartości accuracy oraz loss osobno dla zbioru uczącego oraz walidacyjnego.

- Accuracy - wyliczane jest na podstawie porównania dopasowania na wyjściu sieci z rzeczywistą klasą przynależności do której należy dany obraz. Wartość ta pokazuje procentową ilość poprawnych dopasowań przez sieć.

- Loss - wyliczana jest z wykorzystaniem algorytmu wstecznej propagacji i wykorzystywana do określenia stopnia nauczenia sieci. Funkcja pozwala na modyfikację wag w procesie minimalizacji wartości loss w trakcie uczenia. Wartość loss wykorzystywana jest do określania postępów uczenia oraz wykrywania przeuczenia sieci.

Uczenie przerywane jest w momencie osiągnięcia ustawionej liczby epok lub w momęcie znacznego wzrostu wartści loss względem poprzedniej epoki.

6 Porównanie różnych modli sieci.

Model	Ilość epok	Rozpoznawanie kotów [%]	Rozpoznawanie psów [%]	Rozpoznawanie kotów (powyżej 75%)	Rozpoznawanie psów (powyżej 75%)	Dane testowe psy/koty	Średni czas uczenia [s/epokę]	Rozmiar modelu
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	1	71,22145204	28,78497091	0,00000	0,00000	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	2	84,82167247	15,27380183	84,82167	0,00000	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	3	98,07070094	2,222973536	98,07070	0,00000	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	4	83,43426091	70,87461021	71,94086	48,31111	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	5	97,09802295	50,31273509	96,09673	32,35113	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	10	78,23521256	95,71009843	67,01544	92,52997	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	15	56,2205459	94,12348236	29,71548	90,15668	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	20	91,51501512	92,70327941	86,08260	87,68141	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 180px	25	93,23673364	92,24100506	88,21572	86,86419	1000/1000	258,21	31,5MB
Rozmairy - 256,512,728 - nodropout - relu - wejście 180px	25	94,55125596	96,6245151	91,40785	95,12463	1000/1000	261,38	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - sigmoid - wejście 180px	25	96,18533148	81,67118289	94,22102	74,74626	1000/1000	254,43	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 90px	25	96,25293225	89,27062323	93,85750	82,61893	1000/1000	57,12	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 45px	25	80,18433399	93,89606519	69,61044	90,57253	1000/1000	35,76	31,5MB
Rozmairy - 256,512,728 - dropout 0.5 - relu - wejście 20px	25	63,52558904	84,47161982	38,75156	72,23589	1000/1000	23,32	31,5MB
Rozmairy - 128,256,364 - dropout 0.5 - relu - wejście 180px	25	92,66237799	94,88984541	88,37699	91,94821	1000/1000	223,32	8,1MB
Rozmairy - 64,128,182 - dropout 0.5 - relu - wejście 180px	25	83,30545197	96,87942373	73,64784	95,23220	1000/1000	215,59	2,82MB
Rozmairy - 32,64,91 - dropout 0.5 - relu - wejście 180px	25	85,7490962	86,78683655	76,79307	78,83888	1000/1000	197,53	1,19MB
Rozmairy - 16,32,64 - dropout 0.5 - relu - wejście 180px	25	93,84454439	56,15707692	90,47018	36,53950	1000/1000	179,52	786,23KB

Rysunek 7: Tabelka z pomiarami dla poszczególnych modeli.

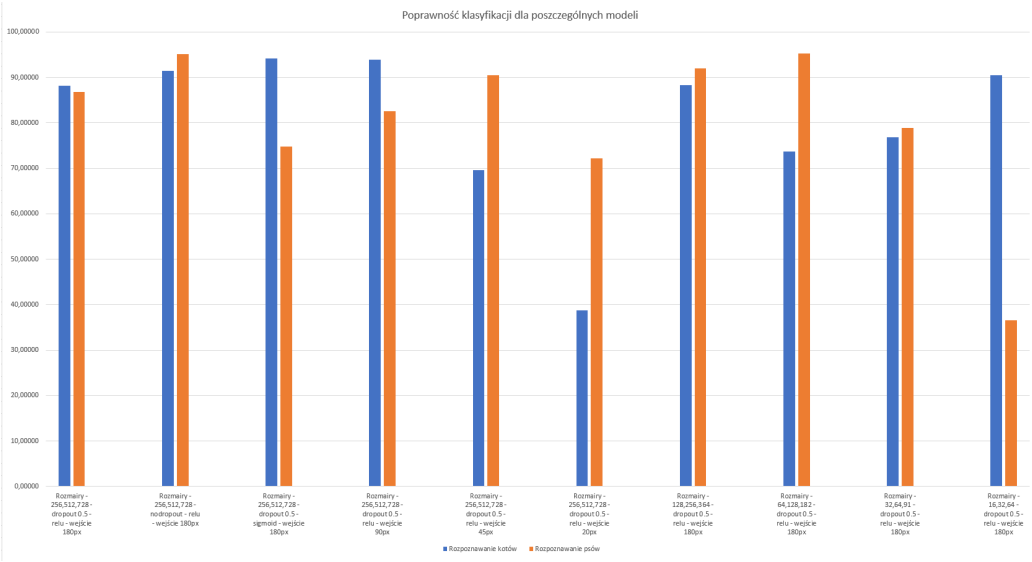
- Rozpoznawanie ... - Średnia procentowa wyjścia sieci dla zbioru testowego (Liczone są wszystkie wyniki). Liczona z sumy wyjść sieci dla kolejnych obrazów.
- Rozpoznawanie ... (powyżej 75%) - Średnia procentowa wyjścia sieci dla zbioru testowego liczona tak jak powyżej, ale uwzględniane są tylko wyniki powyżej 75%. Wyniki niższe zastępowane są 0.

6.1 Opisy modeli

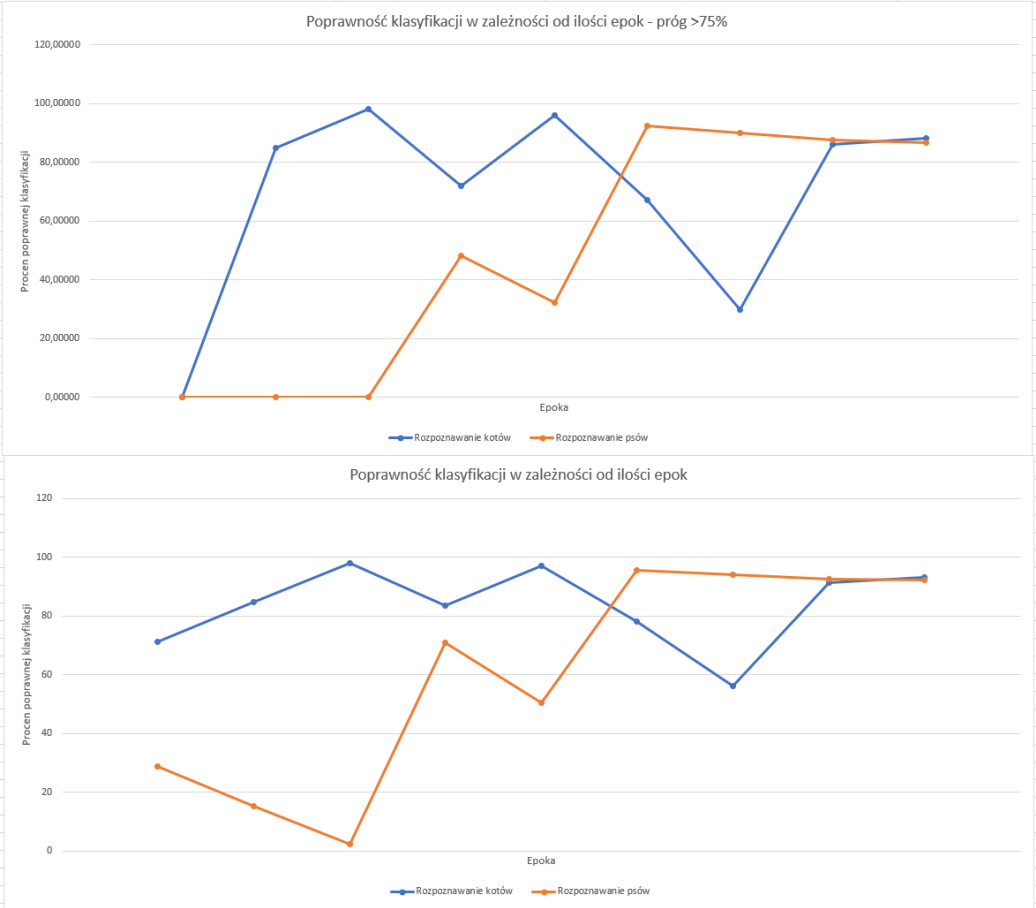
- **Model Rozmiary - 256,512,728 - dropout 0.5 - relu - wejście 180px (pomarańczowy)** - model z 3 warstwami głębokimi, dla których zastosowano filtry konwolucyjne o rozmiarach 256,512 i 728. Funkcja aktywacji to relu a warstwę dropout ustawiono na 0.5. Obrazy na wejściu mają rozmiar 180x180px.
- **Model Rozmiary - 256,512,728 - nodropout 0.5 - relu - wejście 180px (żółty)** - model z 3 warstwami głębokimi, dla których zastosowano filtry konwolucyjne o rozmiarach 256,512 i 728. Funkcja aktywacji to relu a warstwa dropout została usunięta. Obrazy na wejściu mają rozmiar 180x180px.
- **Rozmiary - 256,512,728 - dropout 0.5 - sigmoid - wejście 180px (zielony)** - model z 3 warstwami głębokimi, dla których zastosowano filtry konwolucyjne o rozmiarach 256,512 i 728. Funkcja aktywacji to sigmoid a warstwę dropout ustawiono na 0.5. Obrazy na wejściu mają rozmiar 180x180px.
- **Rozmiary - 256,512,728 - dropout 0.5 - relu - wejście 90px/45px/20px (niebieski)** - model z 3 warstwami głębokimi, dla których zastosowano filtry konwolucyjne o rozmiarach 256,512 i 728. Funkcja aktywacji

to relu a warstwę dropout ustawiono na 0.5. Obrazy na wejściu mają rozmiary kolejno 90x90px, 45x45px oraz 20x20px.

- **Rozmiary - [128,256,364], [64,128,182], [32,64,91] oraz [16,32,64] - dropout 0.5 - relu - wejście 180px (szary)** - model z 3 warstwami głębokimi, dla których zastosowano filtry konwolucyjne o rozmiarach kolejno [128,256,364], [64,128,182], [32,64,91] oraz [16,32,64]. Funkcja aktywacji to relu a warstwę dropout ustawiono na 0.5. Obrazy na wejściu mają rozmiar 180x180px.



Rysunek 8: Wykres porównujący poszczególne modele sieci. Przedstawiono osobno poprawność dopasowań kotów(kolor niebieski) i psów(kolor pomarańczowy.



Rysunek 9: Wykresy wyników dla pomarańczowych modelu z tabeli [7] w zależności od ilości epok. Przedstawiono osobno poprawność dopasowań kotów(kolor niebieski) i psów(kolor pomarańczowy.

7 Wnioski i analiza wyników

- Najlepsze wyniki osiągnął model żółty z tabeli [7]. Jest to model bez warstwy dropout. Prawdopodobnie otrzymane wyniki spowodowane są ustawieniem wartości dropout na zbyt wysoką wartość w pozostałych modelach.

- Najwyższą skuteczność rozpoznawania osiągnęła sieć żółta [7]. Sieć jest w stanie zaklasyfikować obrazy z pewnością co najmniej 75% ze skutecznością: 91.4% dla kotów oraz 95.1% dla psów.

- Niepoprawnie dobrany rozmiar sieci może doprowadzić do niskiej skuteczności modelu, lub nauczenia modelu rozpoznawanie obrazów tylko

jednej z klas np sieć potrafi rozpoznać na zdjęciach koty, ale nie jest w stanie rozpoznawać psów.

- Modele lepiej radzą sobie z problem przy wykorzystaniu funkcji aktywacji relu.
- Sieci z mniejszymi rozmiarami filtrów konwolucyjnych zajmują mniej miejsca, ale spada ich skuteczność rozpoznawania.
- Zmniejszenie rozmiaru obrazu na wejściu przyspiesza uczenie oraz rozpoznawanie po nauczeniu, ale zmniejsza ilość poprawnych dopasowań.
- Znaczne zmniejszenie rozmiarów warstw konwolucyjnych lub obrazu na wejściu sieci powoduje, że sieć jest w stanie nauczyć się dobrze rozpoznawać tylko jedną z dwóch klas.
- Układ GPU z dużą ilością VRAM znacznie przyspiesza uczenie sieci neuronowej do klasyfikowania obrazów.
- Przeanalizowanie większej ilości modeli było niemożliwe z powodu ograniczeń użytkowania usługi google colab.