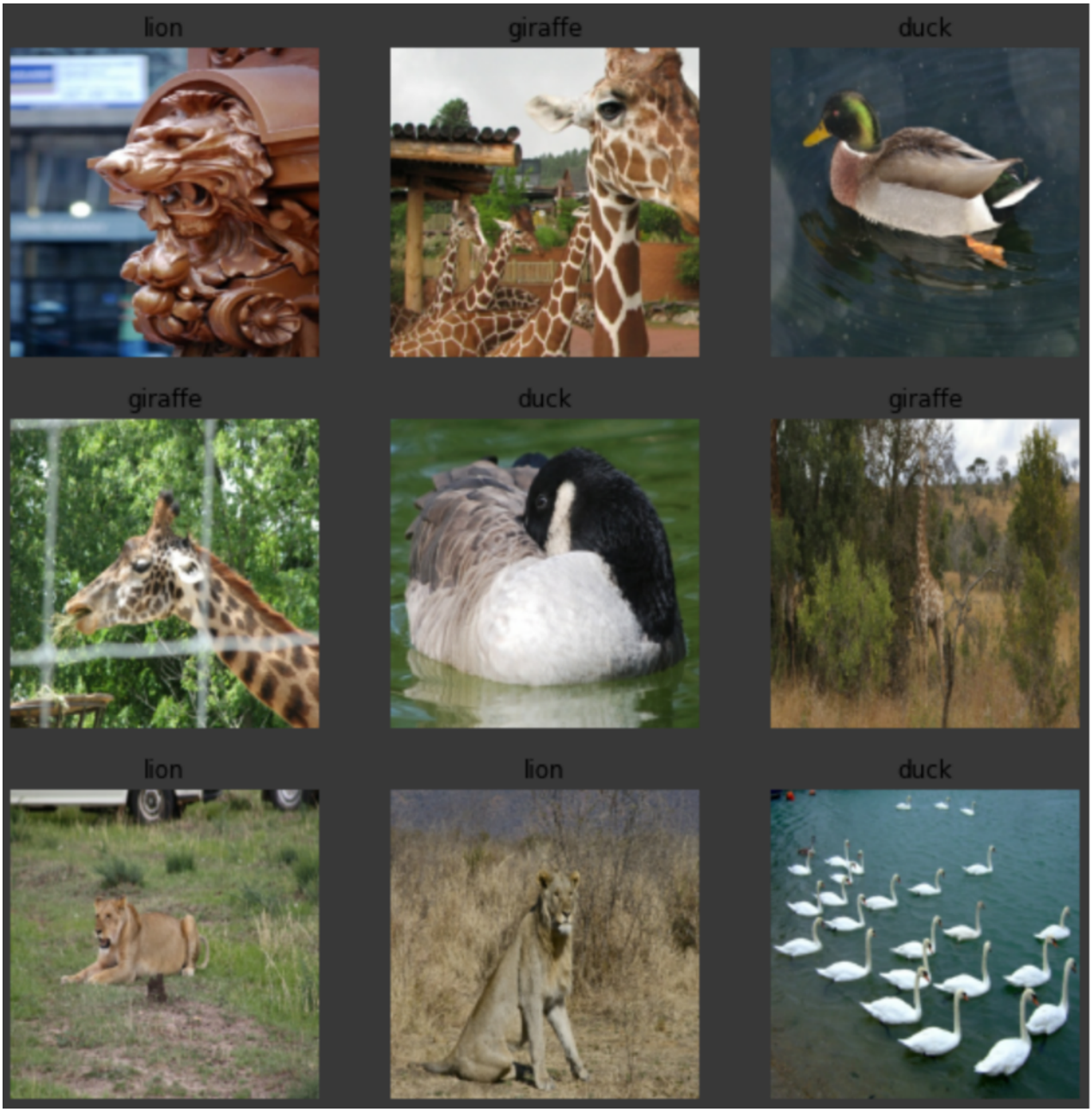
Metody Rozpoznawania Obrazu

Raport 4

Oscar Teeninga

1. Dane treningowe

   Najłatwiej było pobrać dane z Open Images Dataset udostępnianego przez google za pomocą OIDv4-ToolKit. Wybrane przeze mnie klasy to żyrafy, kaczki i lwy. Następnie przeskalowałem wszystkie obrazy do rozmiaru 400x400.



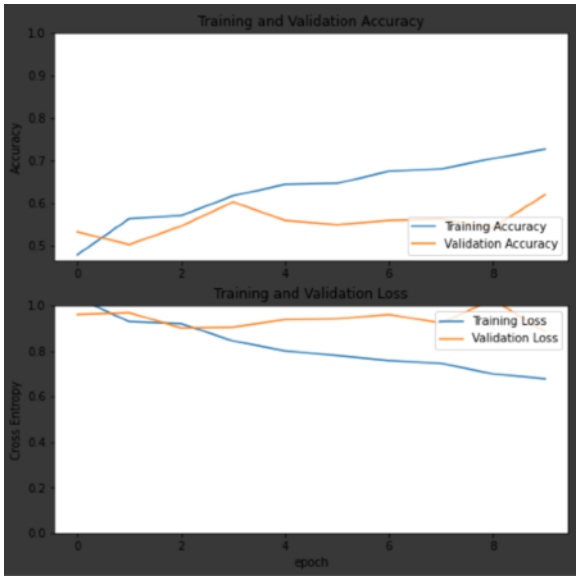
2. MobileNetV2

   Najpierw przetestowałem uczenie na sieci MobileNetV2, następnie wyłączyłem uczenie, a na końcu dodałem warstwy. Wybrana optymalizacja to było SGD, natomiast loss to categorical crossentropy dostosowane do 3 klas.

```
x=tf.keras.layers.Dense(1024,activation='relu')(x)
x=tf.keras.layers.Dense(1024,activation='relu')(x)
x=tf.keras.layers.Dense(512,activation='relu')(x)
preds=tf.keras.layers.Dense(3,activation='softmax')(x)
```

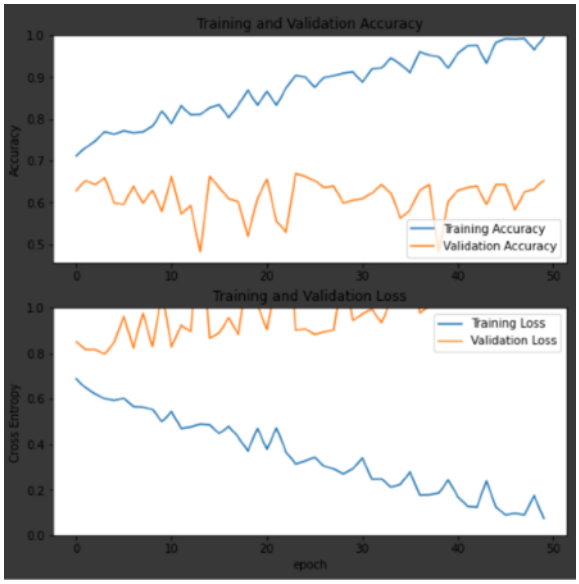
   A. Liczba iteracji - 10

      Można zauważyć, że nie dało to zadowalających efektów, nawet na zbiorze treningowym nie udało się osiągnąć wysokiego współczynnika accuracy, natomiast widać, że miary jakości klasyfikacji dla zbioru testowego nie zmieniają się w jakikolwiek sposób.



   B. Liczba iteracji - 50

      W przypadku zwiększenia liczby iteracji nie zaobserwowałem pozytywnych zmian dla zbioru testowego, loss nawet się pogarszał, natomiast zbiór treningowy z iteracji na iteracje nieznacznie lepiej reagował na przetrenowaną przez siebie sieć.

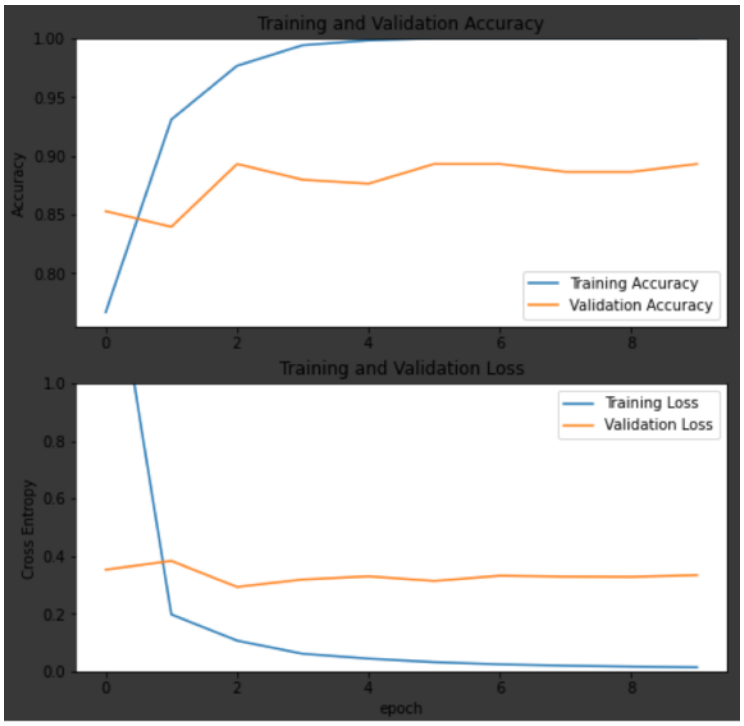


3. VGG19

   Następnie skorzystałem z większej sieci i również wyłączyłem trenowanie na niej, biorąc jej okrojoną wersję bez "topowych" warstw zastępując je tymi samymi co dla MobileNetV2. Tym razem również do optymalizacji używałem SGD oraz do loss categorical crossentropy.

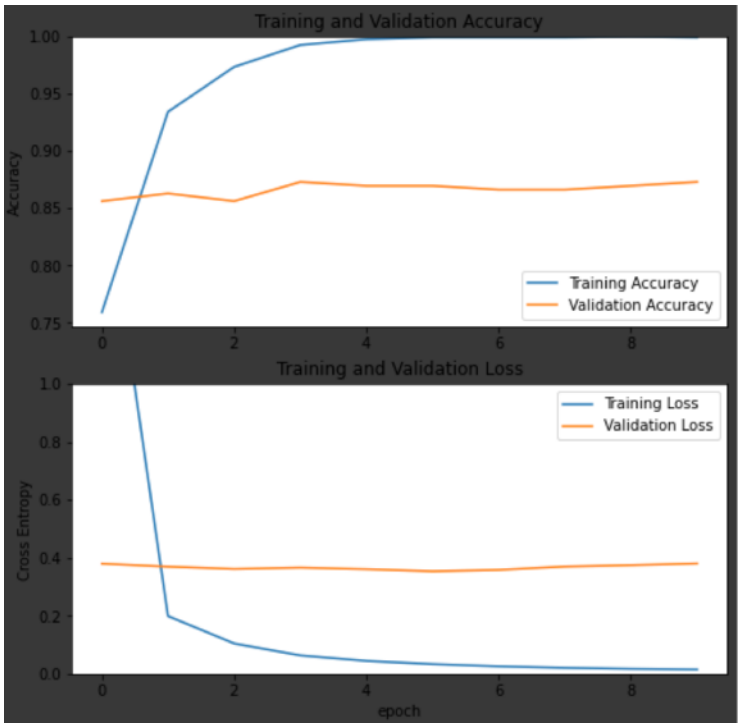
   A. Liczba iteracji - 10

      Tym razem efekty są nieco lepsze, ponieważ sama sieć bez dodatkowego trenowania sama daje zadowalającą 85% dokładność, po 10 iteracjach osiągamy mniej-więcej 90%.



   B. Augmentacja

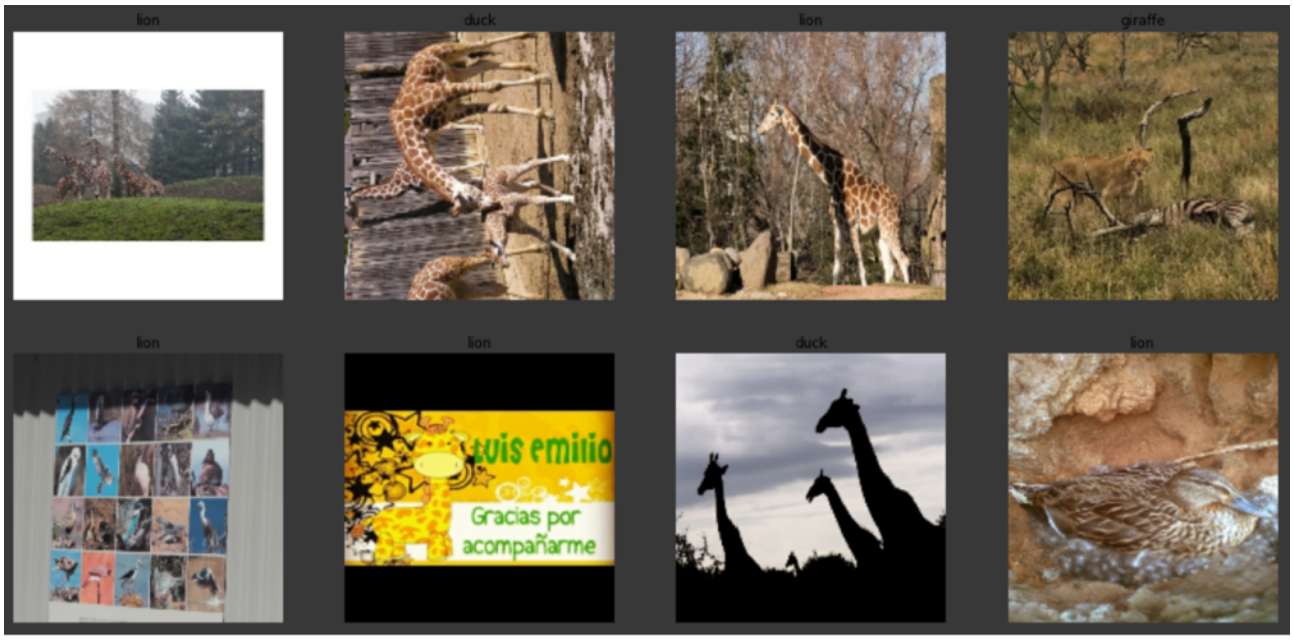
      Zastosujemy zaimplementowane przekształcenia. Tym razem osiągnąłem gorszą klasyfikację na zbiorze treningowym osiągając ~ 87% accuracy i stały loss 0,4.



4. Błędne klasyfikacje

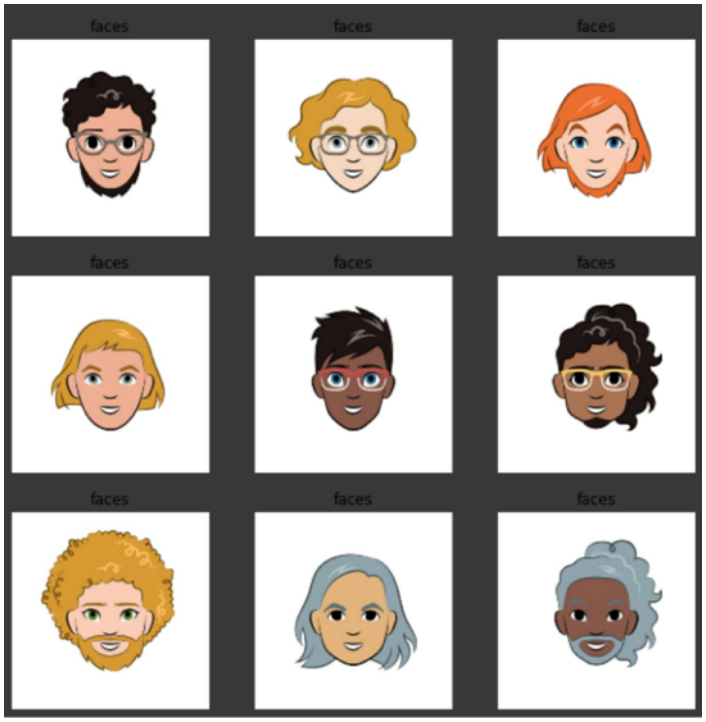
   Do dalszej analizy wybrałem model CGG19 bez augmentacji.

   Błędnie kwalifikowane obrazy okazały się tymi "złośliwymi", więc zrozumiały jest fakt, że klasyfikacja zawodzi. Najczęściej klasyfikowało coś jako lew, szczególnie żyrafy, co może oznaczać, że środowisko ma znaczenie, ponieważ większość z testowych danych dla tych dwóch klas wyróżniało się pustynno-sawannym otoczeniem dookoła obiektu.



5. Dodatkowy zbiór treningowy

Znalazłem zbiór twarzy animowanych postaci, po raz kolejny jest to 500 elementów. Dodatkowym wyróżnikiem jest białe tło, które pozwoli ułatwić sieci zadanie.



Następnie opierając się na VGG19 stworzyłem dodatkową warstwy, których zadaniem było dodanie kolejnej klasy i zrobienie na wynikach softmax.

Sprawdziłem ile spośród danych treningowych zostało źle zakwalifikowanych. Stworzyłem licznik dla każdej klasy, które był inkrementowany w momencie, kiedy na danej klasie została przeprowadzona błędna predykcja, oraz kiedy błędną predykcją okazała się ta klasa. Bez zaskoczenia wyniki prezentują się następująco.

Samo accuracy zwiększyło się względem danych bez "faces" do poziomu ok. 92%

klasa	błąd
faces	2
duck	72
giraffe	97
lion	75