Rozpoznawanie znaków drogowych

Na podstawie sieci neuronowych

Merski Szymon

Czerwiec 2022

Spis treści

1 Wstęp

Stworzenie, wyuczenie oraz porównanie różnych konwolucyjnych sieci neuronowych, których zadaniem było rozpoznanie znaków drogowych

1.1 Przykładowe znaki

Pełna baza zdjęć

2 Przedstawienie różnych modeli

Każdy model został przetrenowany na 20 epokach, przy wielkości obrazu $100x100$.

2.1 Model 1

Model składa się z następujących warstw:

```
model = models.Sequential()
model.add(layers.Rescaling(1./255))
model.add(layers.Conv2D(128, (3, 3), activation='relu', input_shape=(image_size, image_size, 3)))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(128 ,(3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(256, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(64, activation = 'relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(128, activation = 'relu'))
model.add(layers.Dense(len(test_labels), activation = 'softmax'))
```

2.2 Model 2

Model składa się z następujących warstw:

```
model = models.Sequential()
model.add(layers.Rescaling(1./255))
model.add(layers.Conv2D(128, (3, 3), activation='relu', input_shape=(image_size, image_size, 3)))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(128 ,(3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(256, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(64, activation = 'relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(128, activation = 'relu'))
model.add(layers.Dense(len(test_labels), activation = 'softmax'))
```

2.3 Model 3

Model składa się z następujących warstw:

```
model = models.Sequential()
model.add(layers.Conv2D(128, (3, 3), activation='relu', input_shape=(image_size, image_size, 3)))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(256, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(64, activation = 'relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(128, activation = 'relu'))
model.add(layers.Dense(len(test_labels), activation = 'softmax'))
```

2.4 Model 4

Model składa się z następujących warstw:

```
model = models.Sequential()
model.add(layers.Conv2D(128, (3, 3), activation='relu', input_shape=(image_size, image_size, 3)))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(256, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(64, activation = 'relu'))
model.add(layers.Dropout(0.2))

model.add(layers.Dense(128, activation = 'relu'))
model.add(layers.Dense(len(test_labels), activation = 'softmax'))
```

3 Porównanie modeli

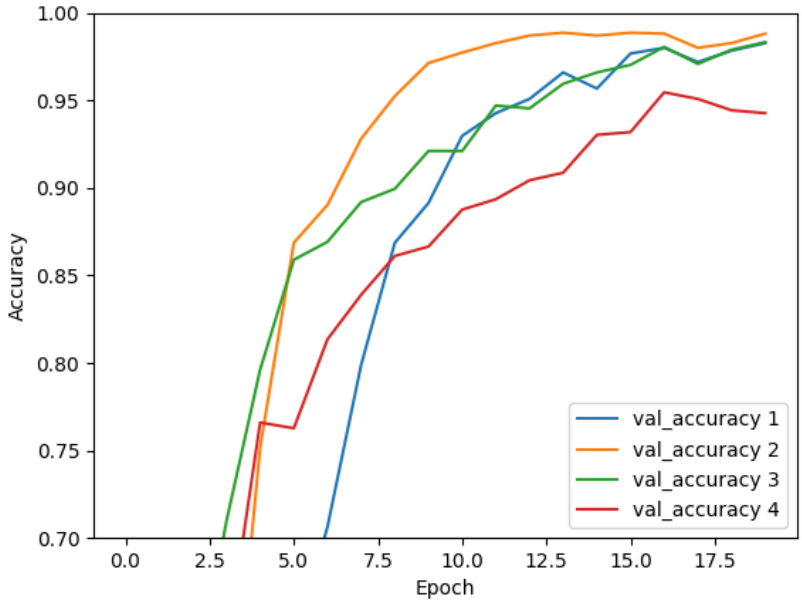
Przy porównaniu modeli zostało wzięte pod uwagę jak długo dany model się uczył oraz dokładność przy rozpoznaniu zdjęć.

3.1 Czasy

Ilość epok: 20

model 1	123.7733s
model 2	124.8371s
model 3	120.1058s
model 4	127.1615s

3.2 Dokładność



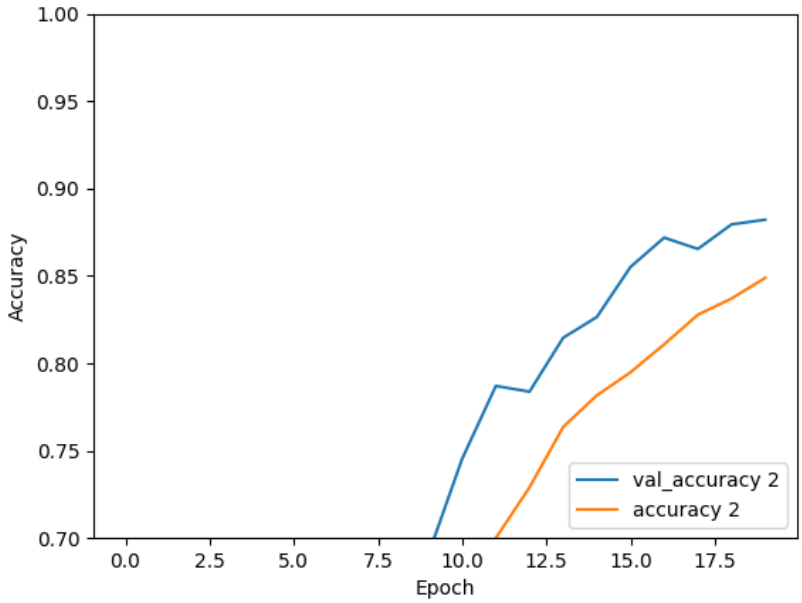
4 Dodatkowe podejście

Ten model znacznie różni się od pozostałych. Przed trenowaniem modelu obrazy zostały zmienione w binarne (czarno-białe). Czas oraz dokładność został mierzony przy takiej samej liczbie epok (20), co przyniosło następujące skutki:

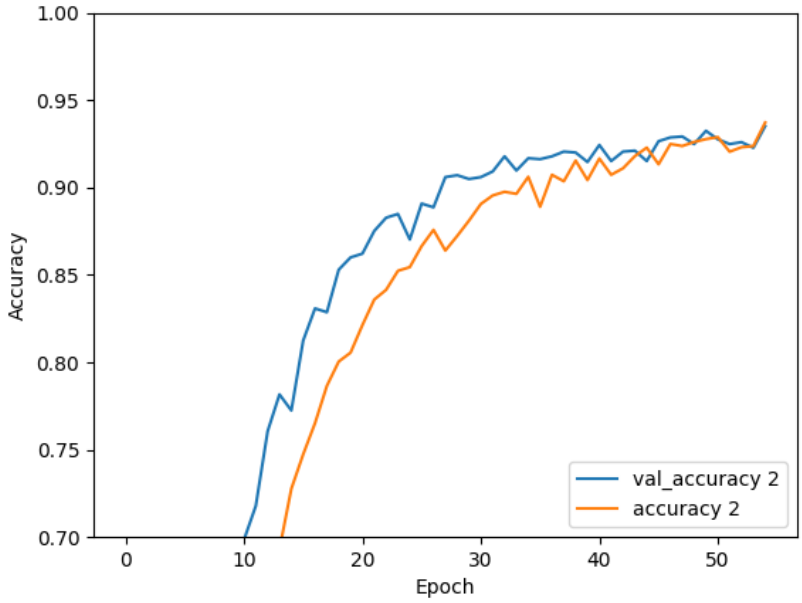
4.1 Czas

extra model	129.9953s

4.2 Dokładność



Żeby osiągnąć przyzwoitą dokładność, trzeba było sieć uczyć przez ponad 50 epok.



5 Podsumowanie

Modele 1,2,3 osiągnęły bardzo podobne wyniki dokładności, po 20 epokach. Model 4 znacząco odstaje w zestawieniu dokładności, jak również w przypadku czasu potrzebnego do przetrenowania sieci.
Model dodatkowy został zdeklasyfikowany ze względu na jego niewydajność oraz skuteczność.

6 Źródła

- Dane
- TensorFlow CNN