Mikuš Filip

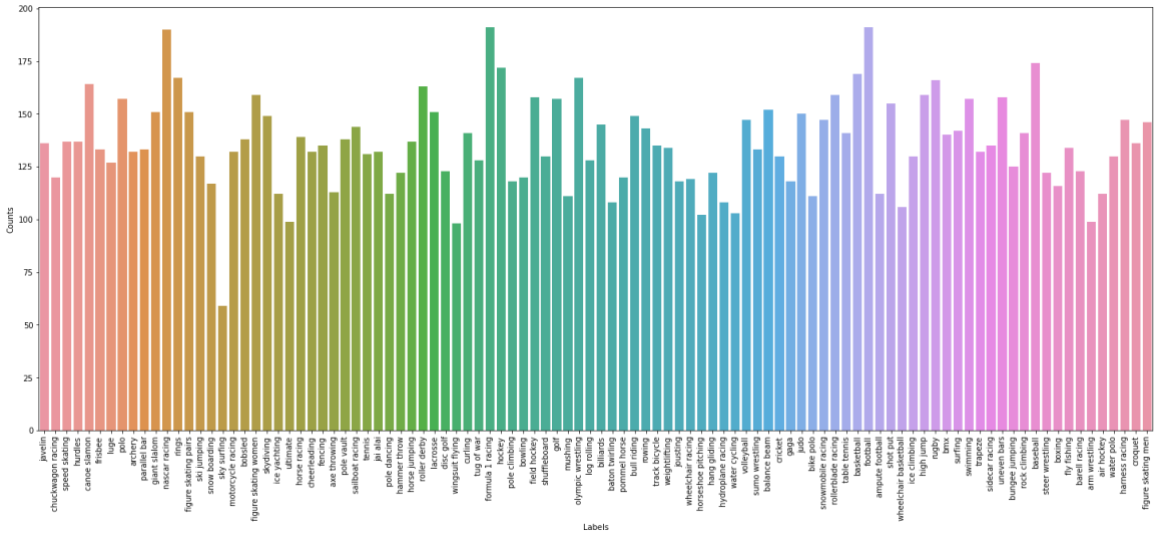
I-SUNS – Zadanie č. 3

Import datasetu

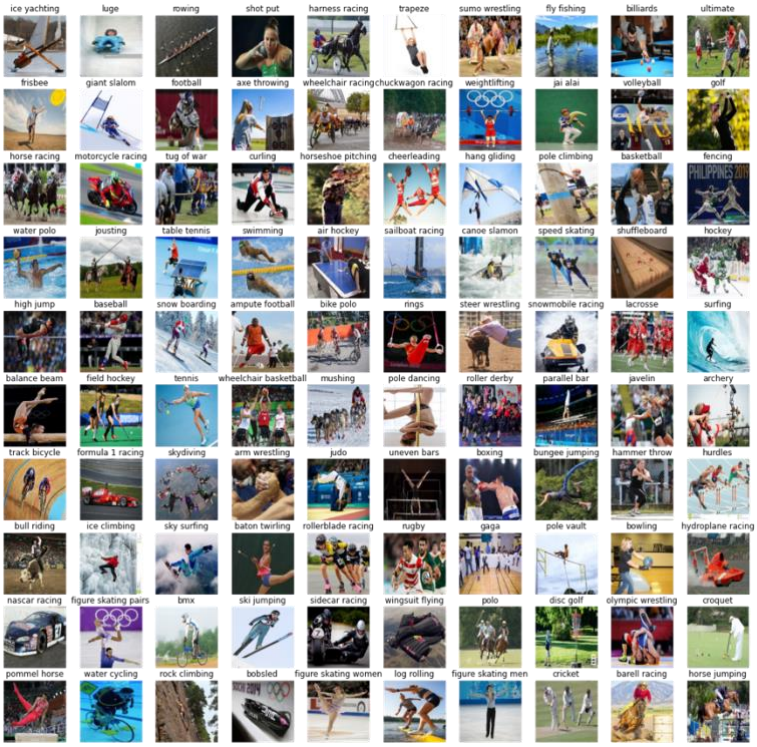
- import do ***tf.data.Dataset*** pomocou metódy ***tf.keras.utils.image_dataset_from_directory***
- dáta normalizované ***(tf.keras.layers.Rescaling(1./255))***
- aktivovaný ***shuffle***

EDA

Graf početnosti v triedach

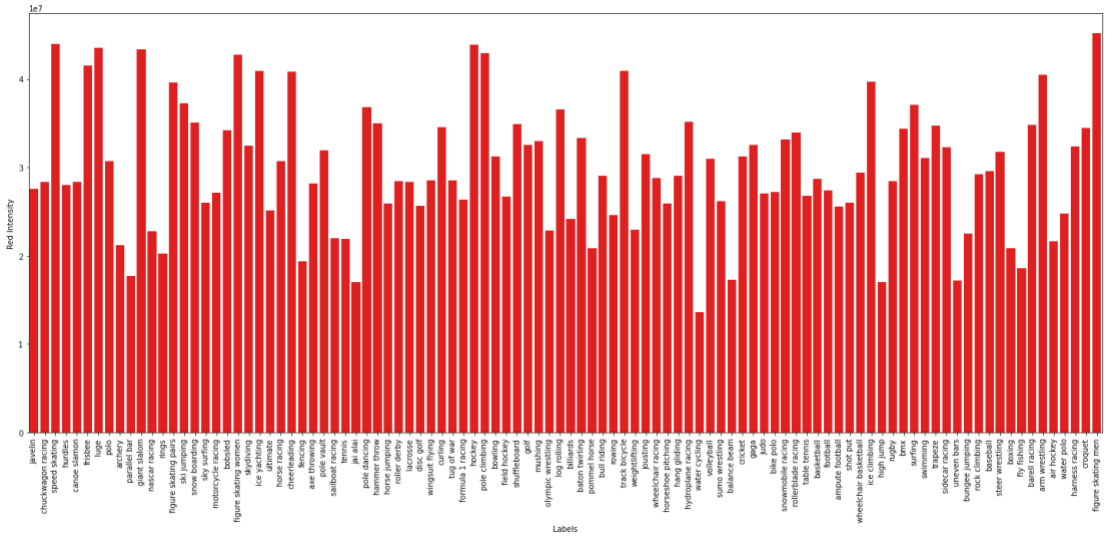


Graf zobrazenia reprezentantov tried

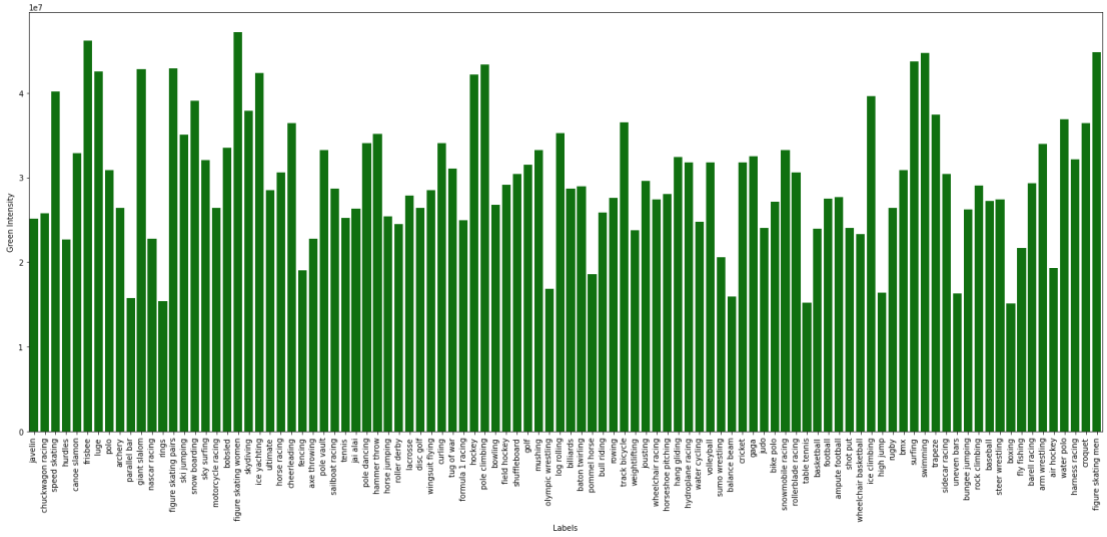


Mikuš Filip

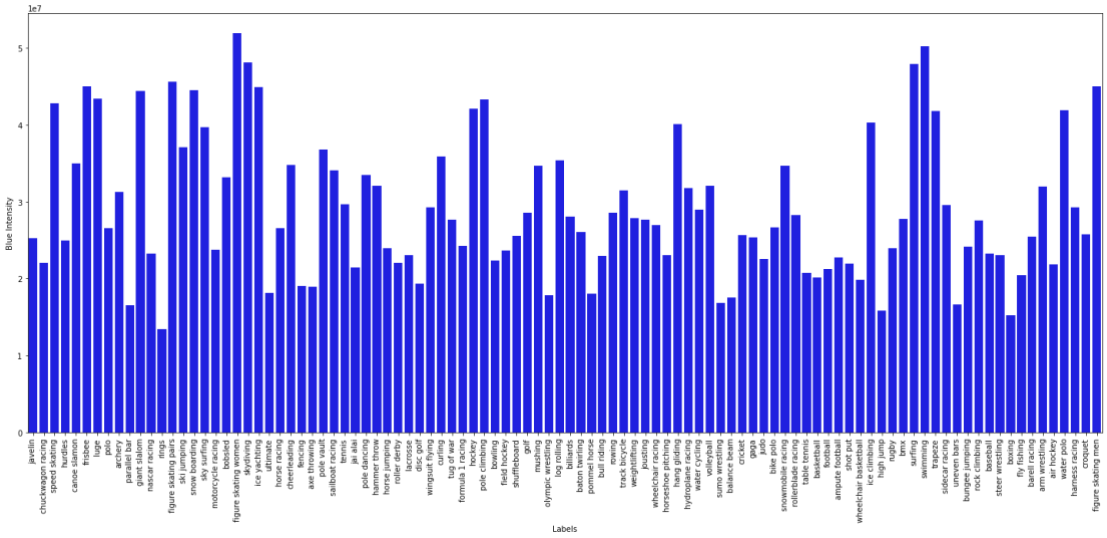
Graf hodnôt intenzity červených pixelov v triedach



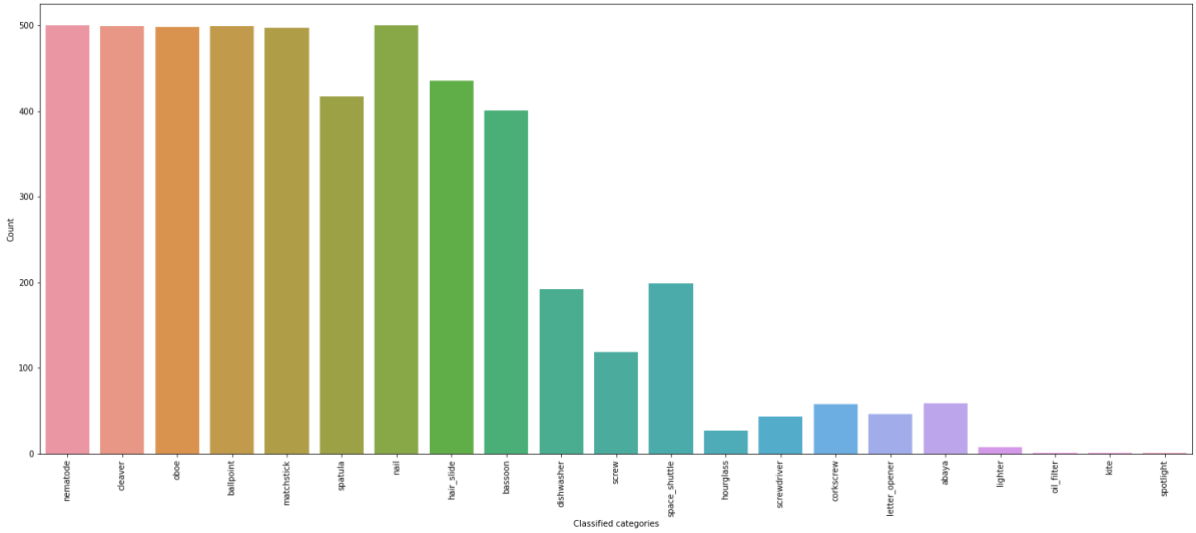
Graf hodnôt intenzity zelených pixelov v triedach



Graf hodnôt intenzity modrých pixelov v triedach



Graf predpovedaných tried pred-trénovaným modelom ResNet50



CNN

Experiment č. 1

```
[ ] cnn_model = models.Sequential()
    cnn_model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 3)))
    cnn_model.add(layers.MaxPooling2D((2, 2)))
    cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    cnn_model.add(layers.MaxPooling2D((2, 2)))
    cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    cnn_model.add(layers.Flatten())
    cnn_model.add(layers.Dense(64, activation="relu"))
    cnn_model.add(layers.Dense(100, activation="softmax"))
```

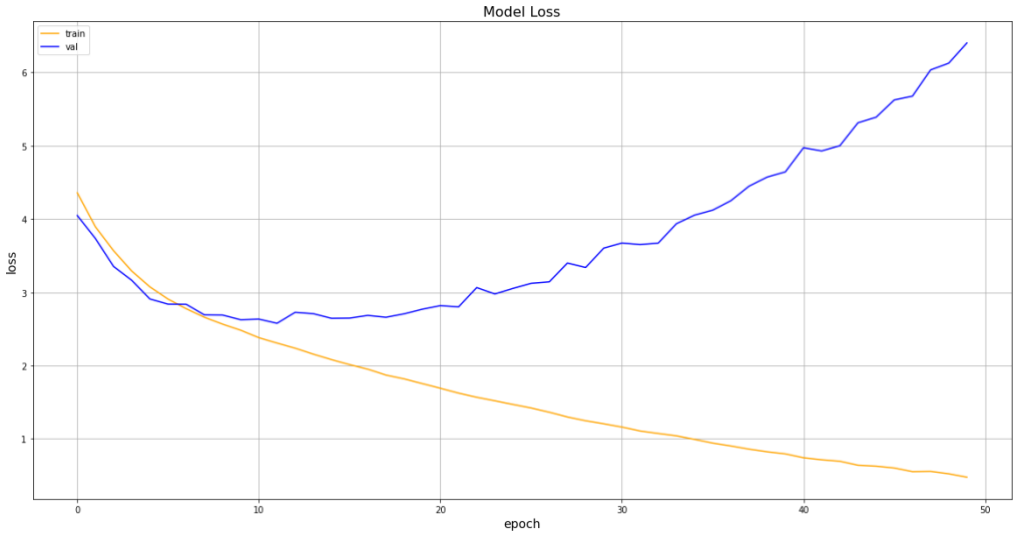
```
    cnn_model.compile(
          optimizer="adam",
          loss="sparse_categorical_crossentropy",
          metrics=["accuracy"]
    )
    cnn_model.summary()
```

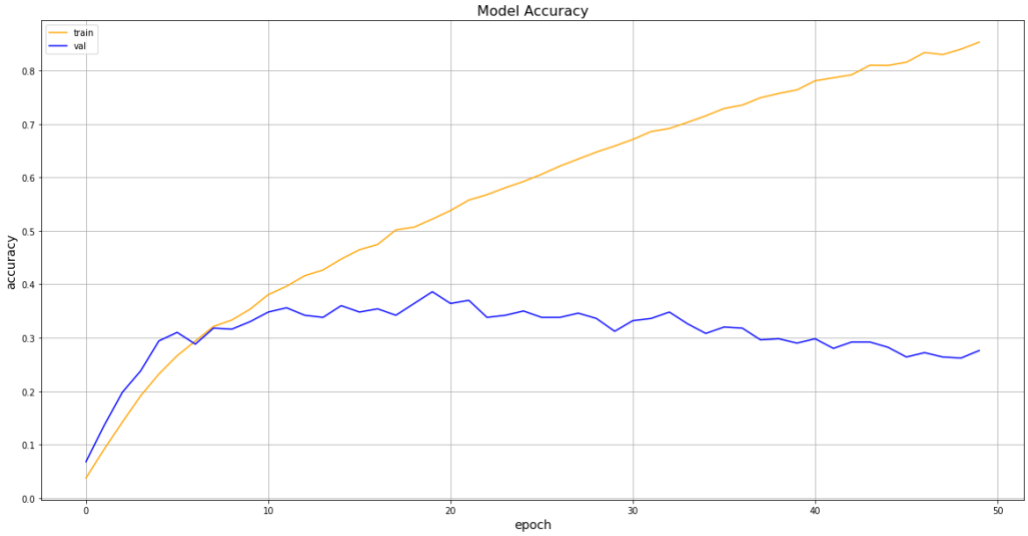
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        896

 max_pooling2d (MaxPooling2D  (None, 13, 13, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 5, 5, 64)         0
 2D)

 conv2d_2 (Conv2D)           (None, 3, 3, 64)          36928

 flatten (Flatten)           (None, 576)               0

 dense (Dense)               (None, 64)                36928

 dense_1 (Dense)             (None, 100)               6500

=================================================================
Total params: 99,748
Trainable params: 99,748
Non-trainable params: 0
_________________________________________________________________
```

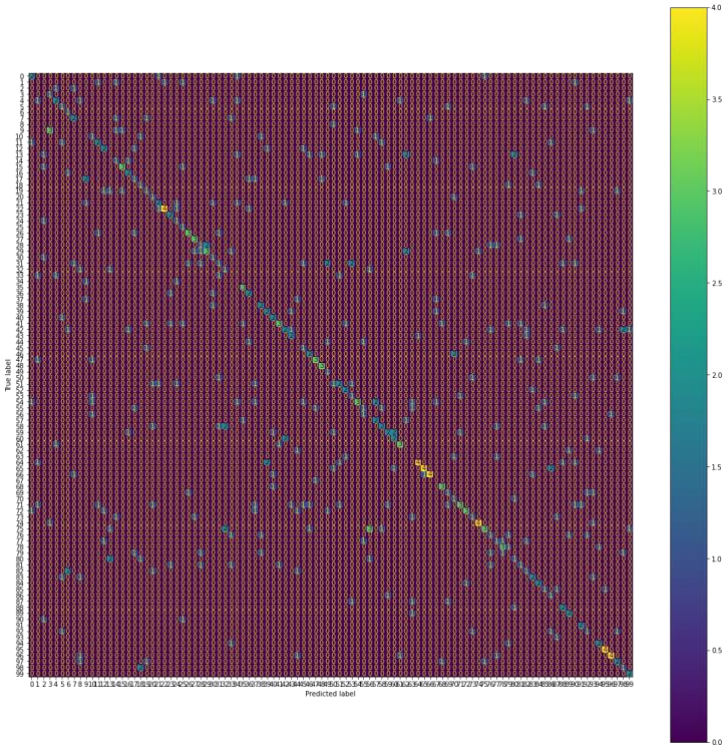
```
[ ] cnn_model_history = cnn_model.fit(
          train_ds,
          validation_data=val_ds,
          epochs=50
    )
    print(cnn_model_history.history)
```

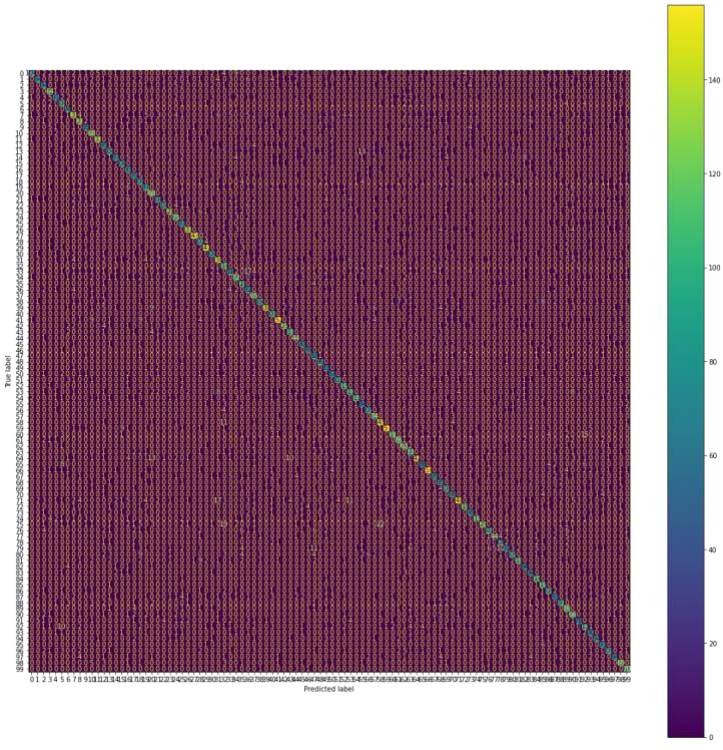
Mikuš Filip

Train Loss	Train Acc,	Test Loss	Test Acc.
0.474	0.852	5.821	0.324









Mikuš Filip

CNN
Experiment č. 2

```
[ ] cnn_model = models.Sequential()
    cnn_model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 3)))
    cnn_model.add(layers.MaxPooling2D((2, 2)))
    cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    cnn_model.add(layers.MaxPooling2D((2, 2)))
    cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    cnn_model.add(layers.Flatten())
    cnn_model.add(layers.Dense(64, activation="relu"))
    cnn_model.add(layers.Dropout(0.2))
    cnn_model.add(layers.Dense(100, activation="softmax"))
```

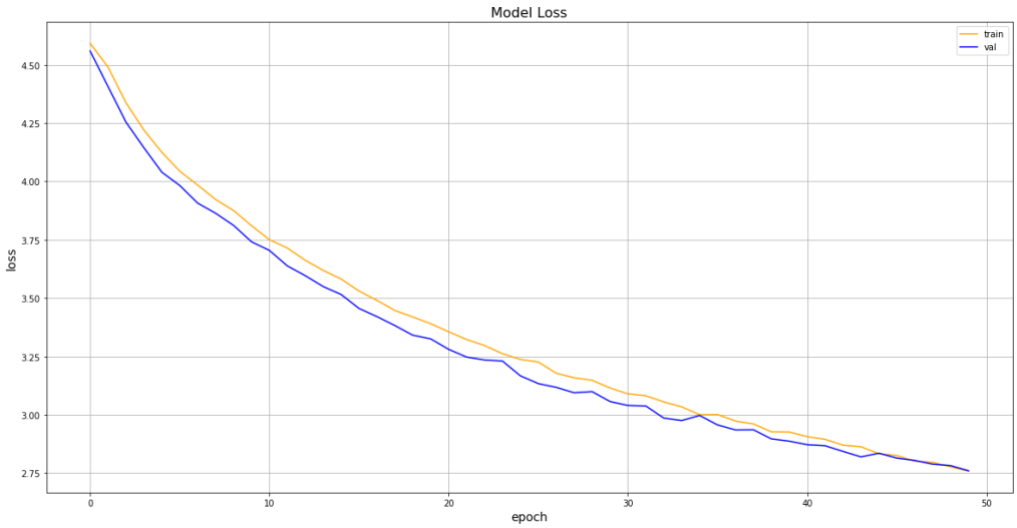
```
[ ] cnn_model.compile(
          optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
          loss="sparse_categorical_crossentropy",
          metrics=["accuracy"]
    )
    cnn_model.summary()
```

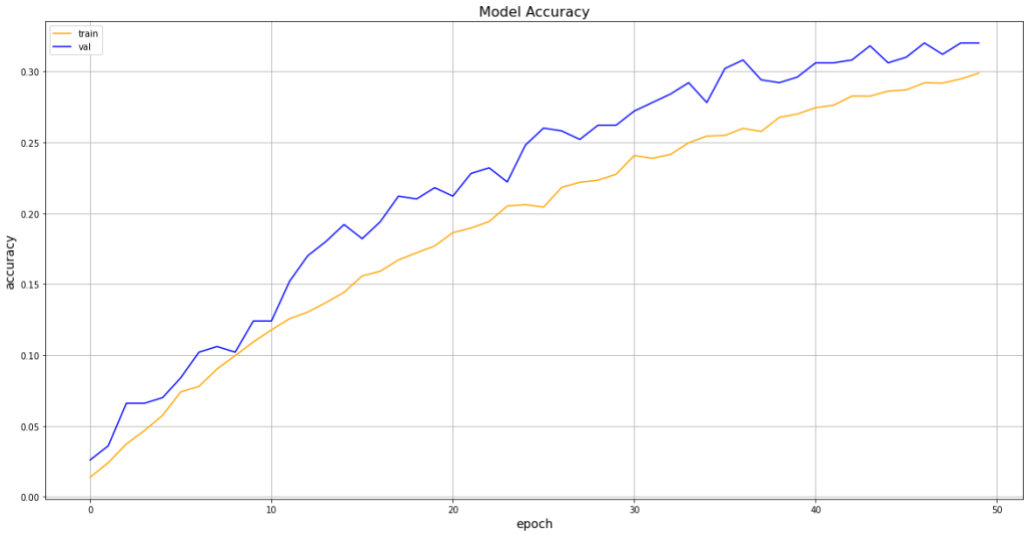
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 26, 26, 32)        896

 max_pooling2d_2 (MaxPooling  (None, 13, 13, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_3 (MaxPooling  (None, 5, 5, 64)         0
 2D)

 conv2d_5 (Conv2D)           (None, 3, 3, 64)          36928

 flatten_1 (Flatten)         (None, 576)               0

 dense_2 (Dense)             (None, 64)                36928

 dropout (Dropout)           (None, 64)                0

 dense_3 (Dense)             (None, 100)               6500

=================================================================
Total params: 99,748
Trainable params: 99,748
Non-trainable params: 0
_________________________________________________________________
```

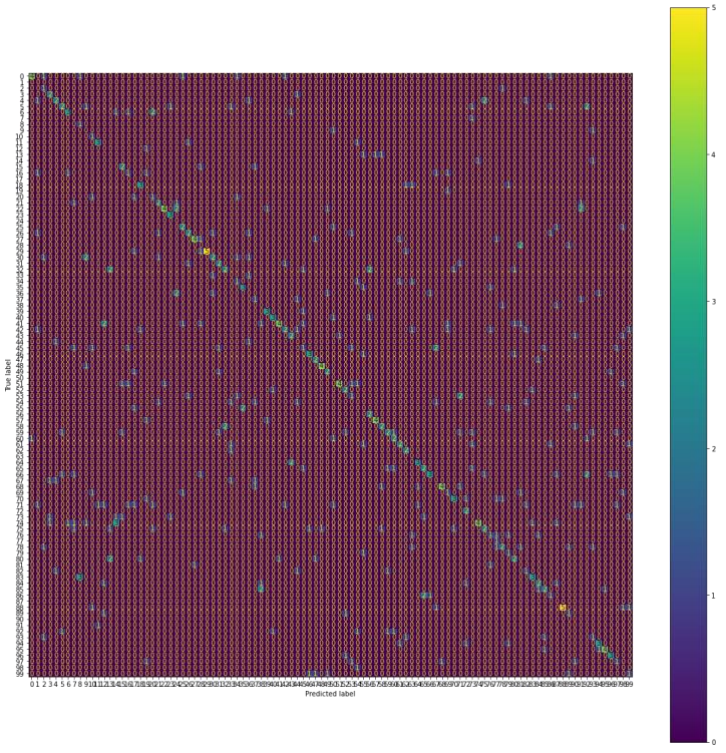
```
[ ] cnn_model_history = cnn_model.fit(
          train_ds,
          validation_data=val_ds,
          epochs=50
    )
    print(cnn_model_history.history)
```

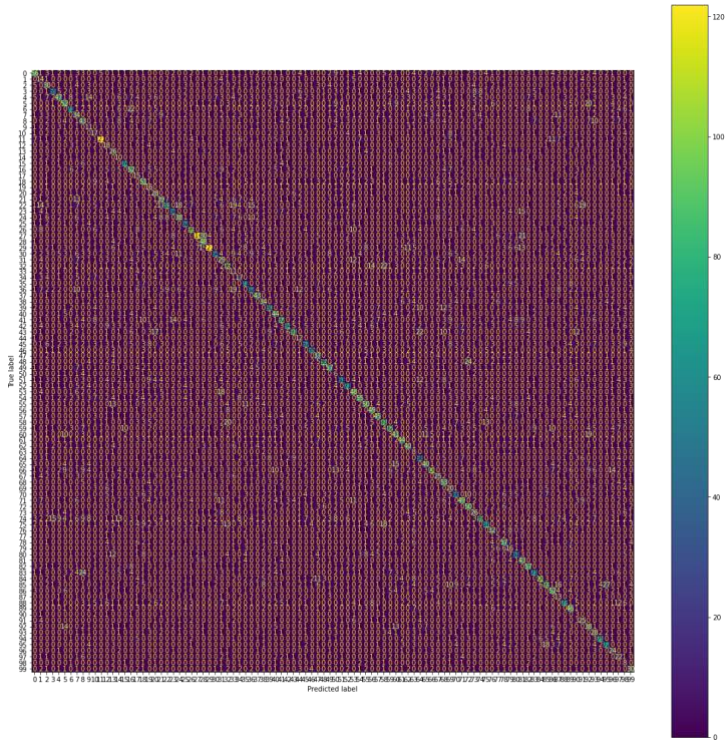
Train Loss	Train Acc,	Test Loss	Test Acc.
2.758	0.298	2.576	0.334

Mikuš Filip


Model Loss
train
val
loss
epoch
4.50
4.25
4.00
3.75
3.50
3.25
3.00
2.75
0
10
20
30
40
50


Model Accuracy
train
val
accuracy
epoch
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
10
20
30
40
50


True label
Predicted label


True label
Predicted label

Mikuš Filip

CNN
Experiment č. 3

```
[ ] cnn_model = models.Sequential()
    cnn_model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 3)))
    cnn_model.add(layers.MaxPooling2D((2, 2)))
    cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    cnn_model.add(layers.MaxPooling2D((2, 2)))
    cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    cnn_model.add(layers.Flatten())
    cnn_model.add(layers.Dense(128, activation="relu"))
    cnn_model.add(layers.Dropout(0.5))
    cnn_model.add(layers.Dense(100, activation="softmax"))
```

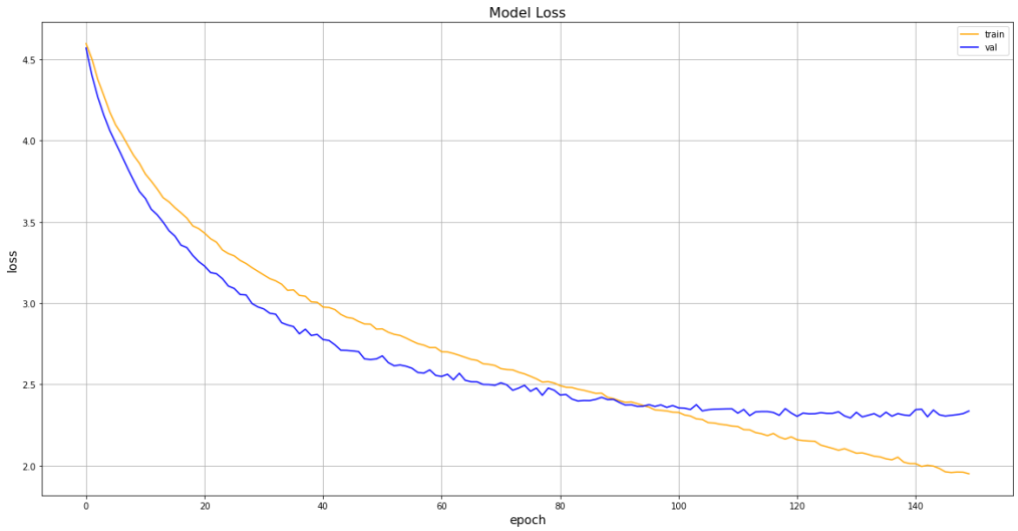
```
[ ] cnn_model.compile(
          optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
          loss="sparse_categorical_crossentropy",
          metrics=["accuracy"]
    )
    cnn_model.summary()
```

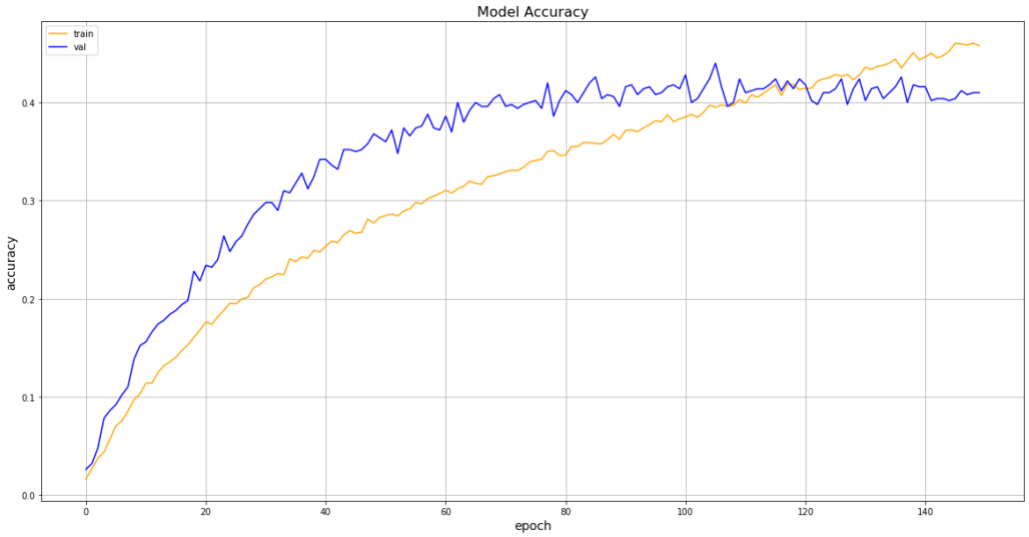
```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 26, 26, 32)        896

 max_pooling2d_4 (MaxPooling  (None, 13, 13, 32)       0
 2D)

 conv2d_7 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_5 (MaxPooling  (None, 5, 5, 64)         0
 2D)

 conv2d_8 (Conv2D)           (None, 3, 3, 64)          36928

 flatten_2 (Flatten)         (None, 576)               0

 dense_4 (Dense)             (None, 128)               73856

 dropout_1 (Dropout)         (None, 128)               0

 dense_5 (Dense)             (None, 100)               12900

=================================================================
Total params: 143,076
Trainable params: 143,076
Non-trainable params: 0
_________________________________________________________________
```

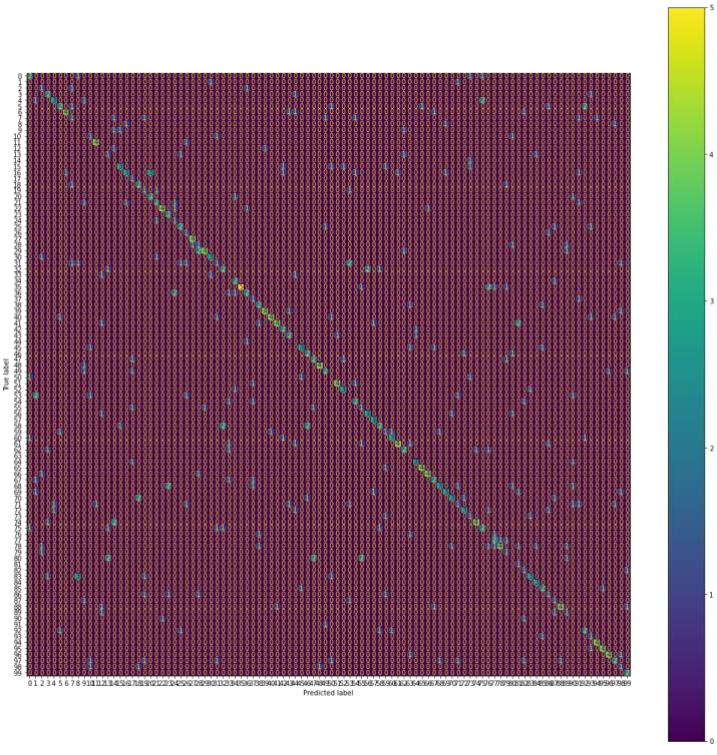
```
[ ] cnn_model_history = cnn_model.fit(
          train_ds,
          validation_data=val_ds,
          epochs=150
    )
    print(cnn_model_history.history)
```

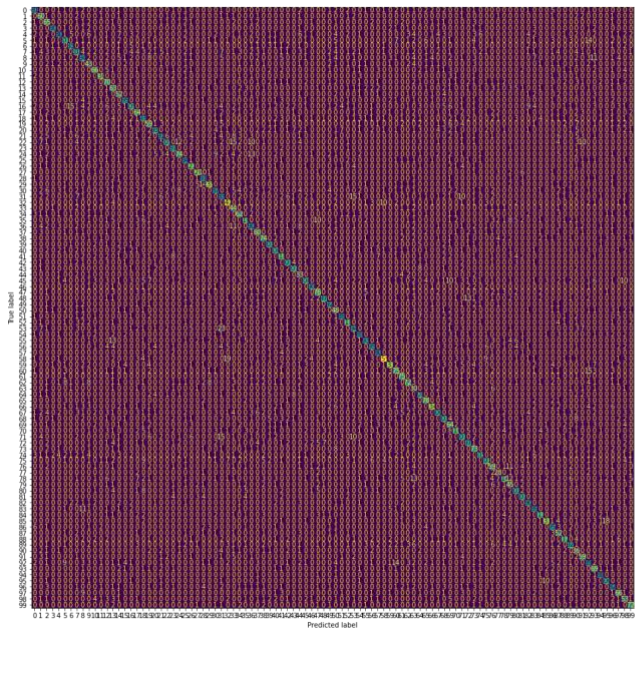
Train Loss	Train Acc,	Test Loss	Test Acc.
1.950	0.457	2.207	0.412

Mikuš Filip


Model Loss
train
val
loss
epoch


Model Accuracy
train
val
accuracy
epoch


True label
Predicted label


True label
Predicted label

Mikuš Filip

Transfer-learning CNN (VGG19)
Experiment č. 4

```
[ ] vgg19_backbone_cnn_model = tf.keras.applications.vgg19.VGG19(
        weights='imagenet',
        include_top=False,
        input_shape = (32, 32, 3)
    )

    for layer in vgg19_backbone_cnn_model.layers:
      layer.trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras
80134624/80134624 [==============================] - 1s 0us/step
```

```
transfer_cnn_model = models.Sequential()
transfer_cnn_model.add(vgg19_backbone_cnn_model)
transfer_cnn_model.add(layers.Flatten())
transfer_cnn_model.add(layers.Dense(128, activation="relu"))
transfer_cnn_model.add(layers.Dropout(0.2))
transfer_cnn_model.add(layers.Dense(100, activation="softmax"))
```

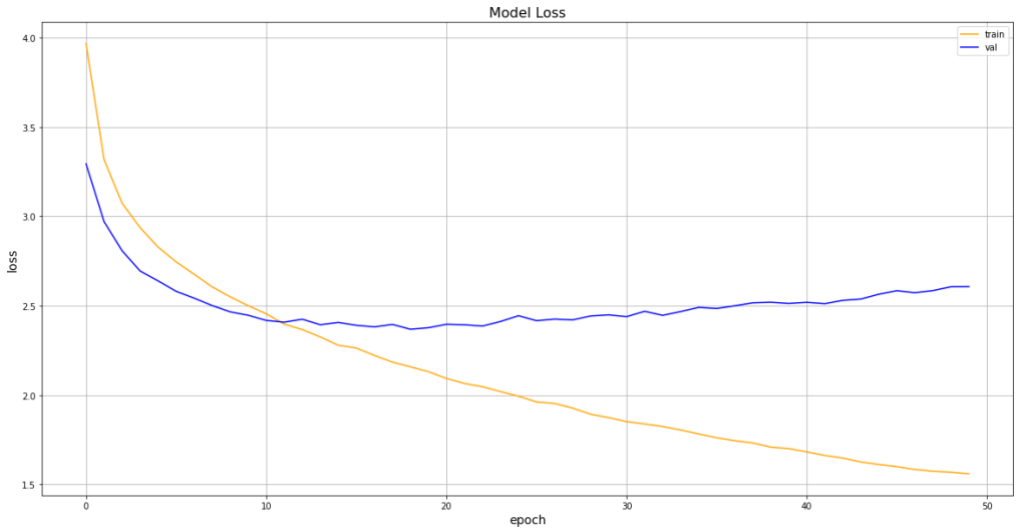
```
[ ] transfer_cnn_model.compile(
        optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
        loss="sparse_categorical_crossentropy",
        metrics=["accuracy"]
    )
    transfer_cnn_model.summary()
```

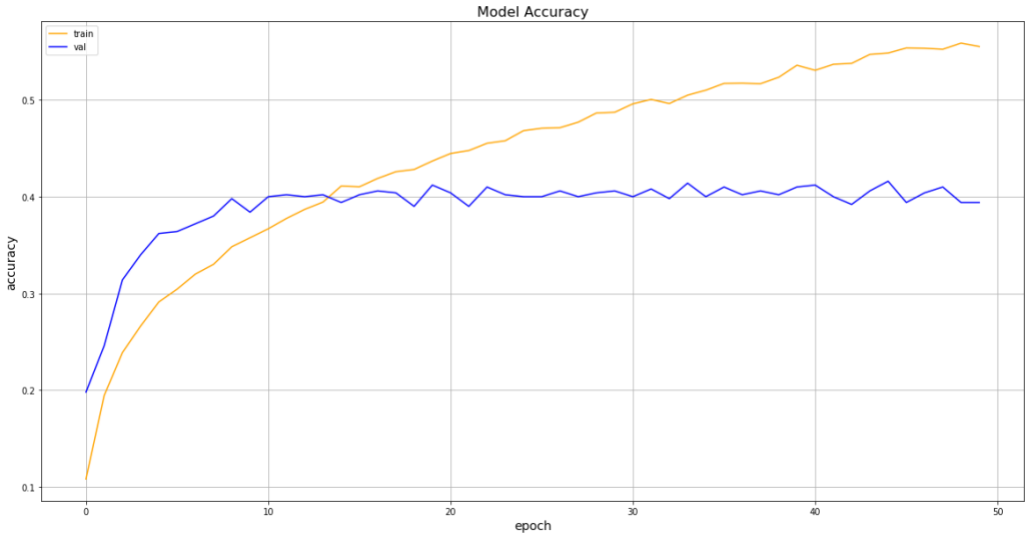
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 1, 1, 512)         20024384

 flatten_1 (Flatten)         (None, 512)               0

 dense_2 (Dense)             (None, 128)               65664

 dropout_1 (Dropout)         (None, 128)               0

 dense_3 (Dense)             (None, 100)               12900

=================================================================
Total params: 20,102,948
Trainable params: 78,564
Non-trainable params: 20,024,384
_________________________________________________________________
```

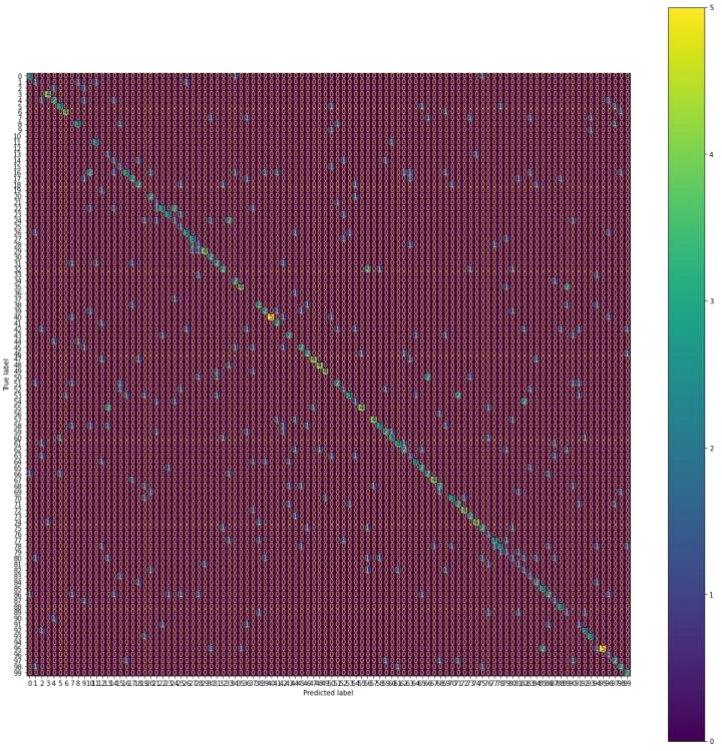
```
[ ] transfer_cnn_model_history = transfer_cnn_model.fit(
        trans_train_ds,
        validation_data=trans_val_ds,
        epochs=50
    )
    print(transfer_cnn_model_history.history)
```

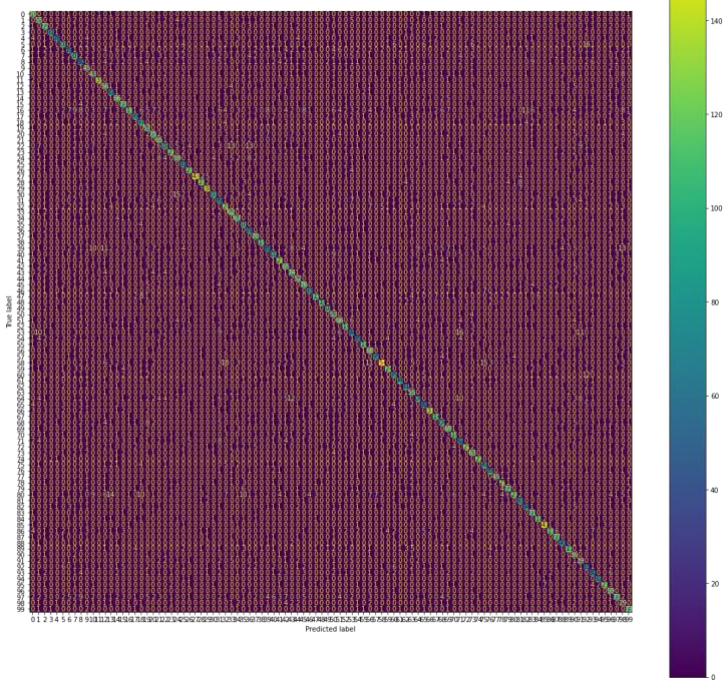
Train Loss	Train Acc,	Test Loss	Test Acc.
1.560	0.555	2.589	0.396

Mikuš Filip


Model Loss
train
val
loss
epoch
4.0
3.5
3.0
2.5
2.0
1.5
0
10
20
30
40
50


Model Accuracy
train
val
accuracy
epoch
0.5
0.4
0.3
0.2
0.1
0
10
20
30
40
50


True label
Predicted label


True label
Predicted label

Mikuš Filip

Transfer-learning CNN (VGG19)
Experiment č. 5

```
transfer_cnn_model = models.Sequential()
transfer_cnn_model.add(vgg19_backbone_cnn_model)
transfer_cnn_model.add(layers.Flatten())
transfer_cnn_model.add(layers.Dense(128, activation="relu"))
transfer_cnn_model.add(layers.Dropout(0.5))
transfer_cnn_model.add(layers.Dense(100, activation="softmax"))
```

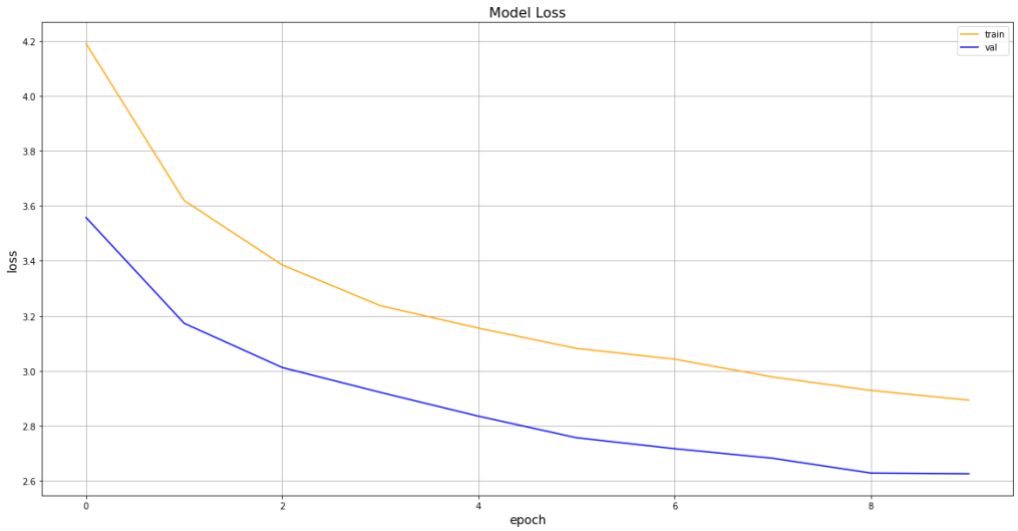
```
transfer_cnn_model.compile(
      optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
      loss="sparse_categorical_crossentropy",
      metrics=["accuracy"]
)
transfer_cnn_model.summary()
```

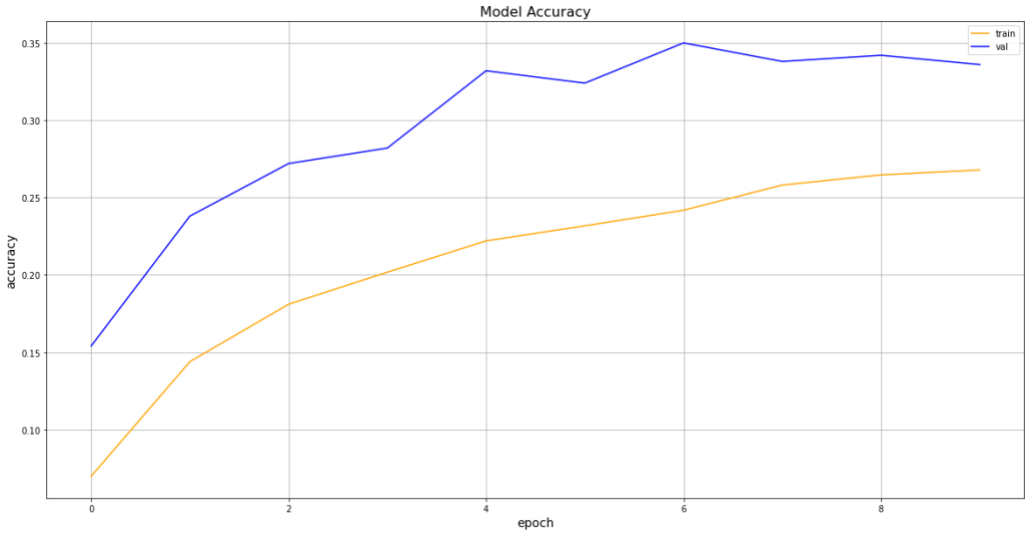
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 1, 1, 512)         20024384

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 128)               65664

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 100)               12900

=================================================================
Total params: 20,102,948
Trainable params: 78,564
Non-trainable params: 20,024,384
_________________________________________________________________
```

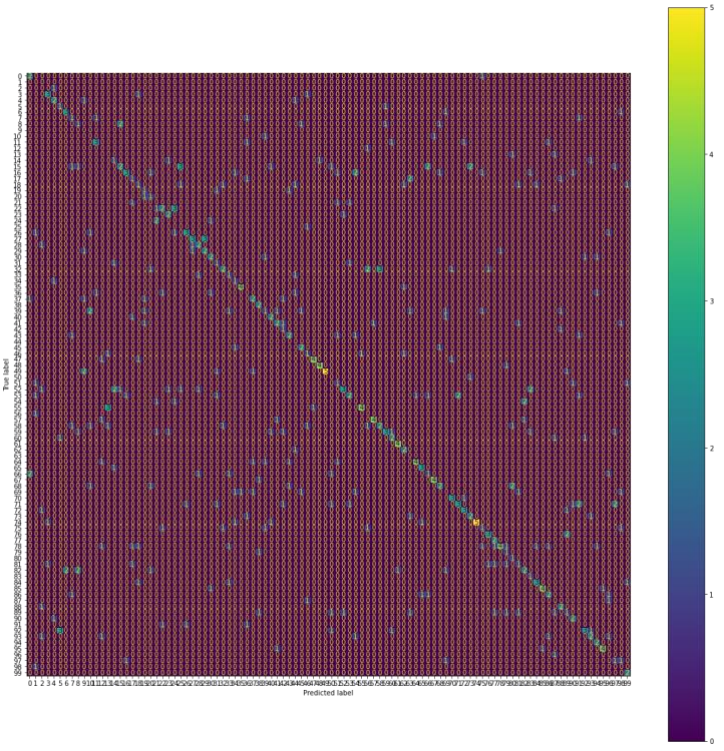
```
transfer_cnn_model_history = transfer_cnn_model.fit(
      trans_train_ds,
      validation_data=trans_val_ds,
      epochs=10
)
print(transfer_cnn_model_history.history)
```

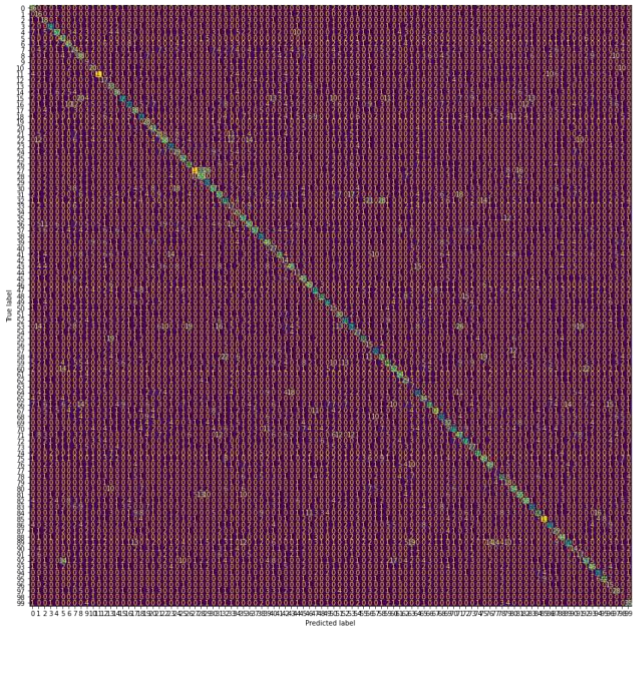
Train Loss	Train Acc,	Test Loss	Test Acc.
2.892	0.267	2.556	0.360

Mikuš Filip


Model Loss
train
val
loss
epoch


Model Accuracy
train
val
accuracy
epoch


True label
Predicted label


True label
Predicted label

Mikuš Filip

Bonus

Zobrazenie filtrov z konvolučných vrstiev

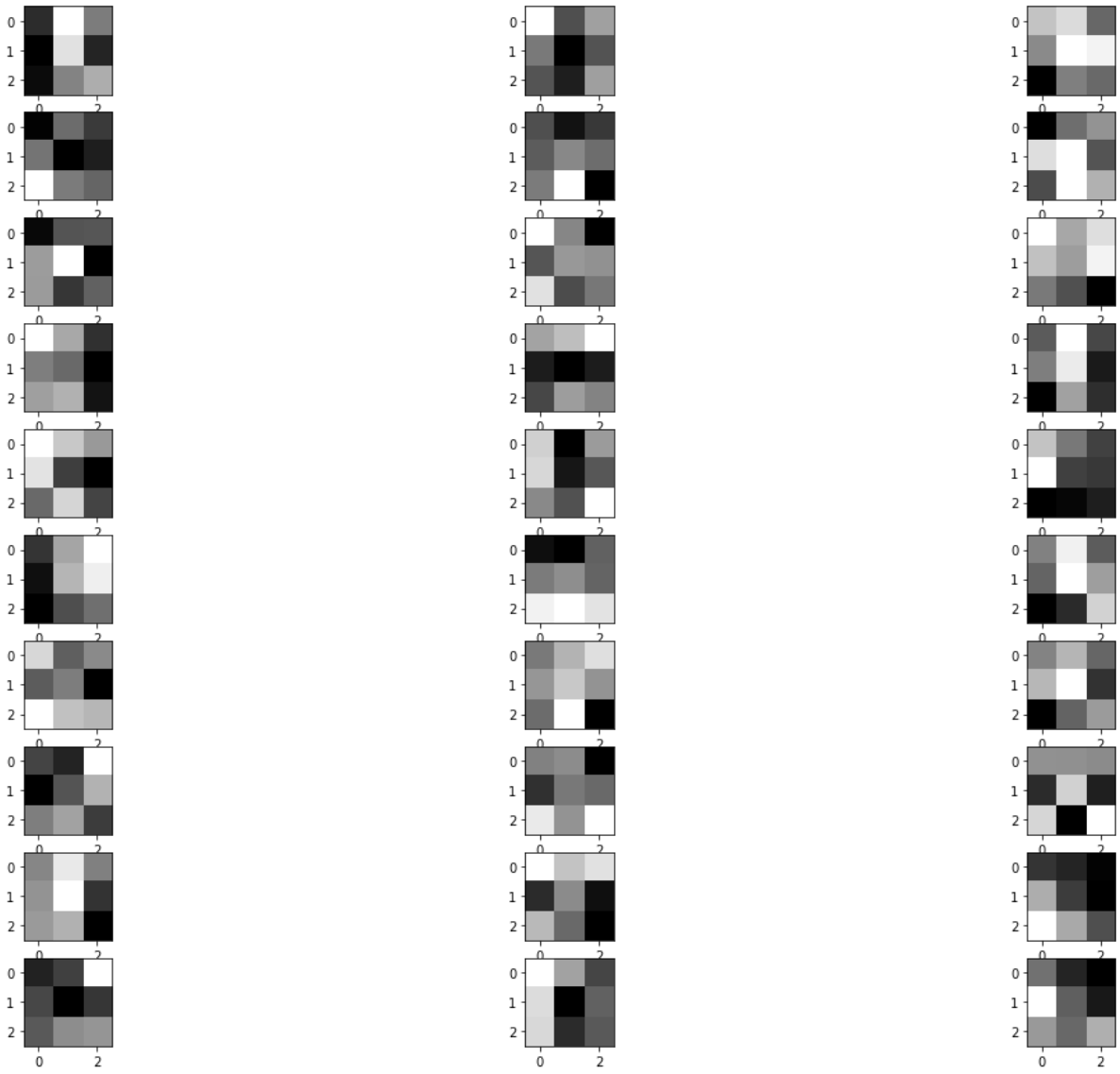
```
cnn_model = models.Sequential()
cnn_model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 3)))
cnn_model.add(layers.MaxPooling2D((2, 2)))
cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
cnn_model.add(layers.MaxPooling2D((2, 2)))
cnn_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
cnn_model.add(layers.Flatten())
cnn_model.add(layers.Dense(256, activation="relu"))
cnn_model.add(layers.Dropout(0.5))
cnn_model.add(layers.Dense(100, activation="softmax"))
```

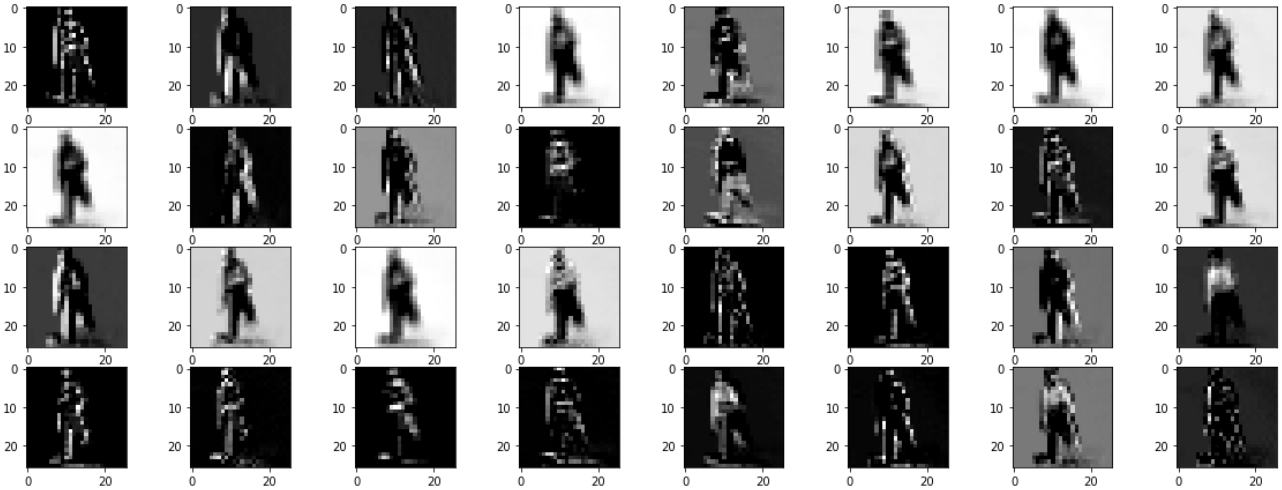
```
cnn_model.compile(
      optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
      loss="sparse_categorical_crossentropy",
      metrics=["accuracy"]
)
cnn_model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        896

 max_pooling2d (MaxPooling2D  (None, 13, 13, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 5, 5, 64)         0
 2D)

 conv2d_2 (Conv2D)           (None, 3, 3, 64)          36928

 flatten (Flatten)           (None, 576)               0

 dense (Dense)               (None, 256)               147712

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 100)               25700

=================================================================
Total params: 229,732
Trainable params: 229,732
Non-trainable params: 0
_________________________________________________________________
```

```
cnn_model_history = cnn_model.fit(
      train_ds,
      validation_data=val_ds,
      epochs=100
)
print(cnn_model_history.history)
```

Mikuš Filip





Mikuš Filip