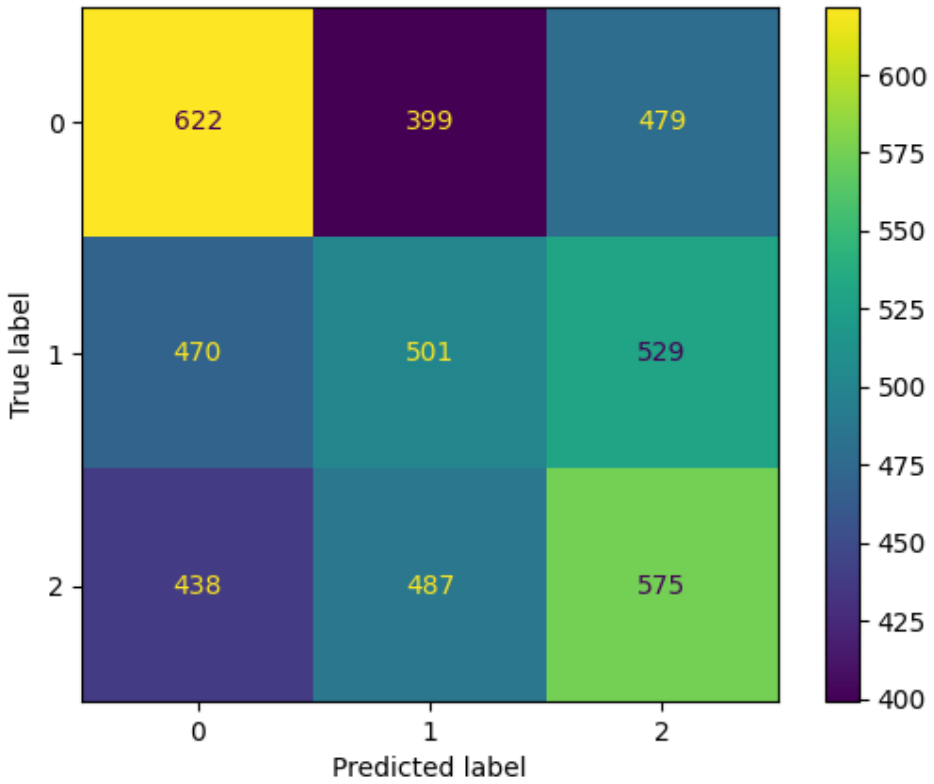
Citirea datelor(readwrite.py):

- Datele sunt citite în modulul readwrite.py cu ajutorul functiei Image.open din modulul Pillow, dupa care sunt convertite într-un array bidimensional de int8
- Opțional, acest array poate sa fie flattened prin setarea parametrului flat=True
- Pentru a putea antrena convolutional neural network-ul, datele sunt reshaped după forma (-1, 50, 50, 1), imaginile fiind grayscale si avand dimensiunea de 50x50
- Pentru a nu mai fi necesara prelucrarea imaginilor de fiecare data, acestea sunt serializate, cu ajutorul modului pickle
- Funcția write_data primește ca parametru o lista ce contine predicțiile modelului pentru setul de date ca test și le scrie într-un csv, sub forma cerută

K-nearest neighbors(knn.py):

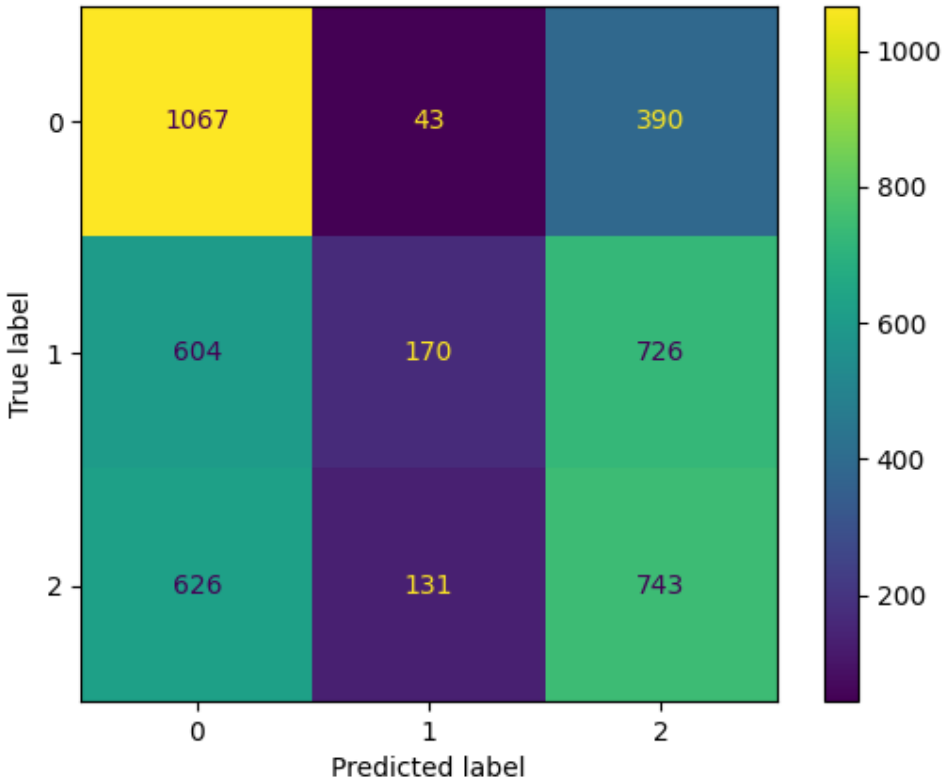
- Datele sunt citite cu ajutorul functiei read_data
- Este creat un clasificator KNeighborsClassifier cu n_neighbors=3 si este antrenat cu datele de training



- Se creaza o imagine care contine matricea de confuzie a clasificatorului pe setul de validare cu ajutorul funcției plot_confusion_matrix din modulul sklearn.metrics
- Se genereaza valori pentru setul de date de test și sunt scrise într-un csv cu ajutorul funcției write_data

K-nearest neighbors(knn.py):

- Datele sunt citite cu ajutorul funcției read_data
- Clasificatorul este identic cu cel din laboratorul 7
- Se creaza o imagine care contine matricea de confuzie a clasificatorului pe setul de validare

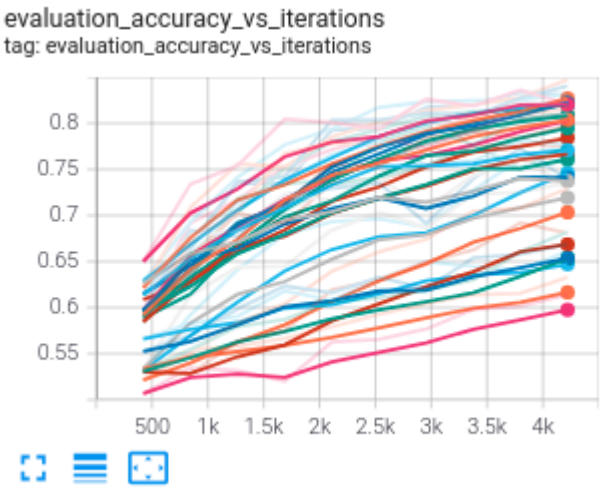


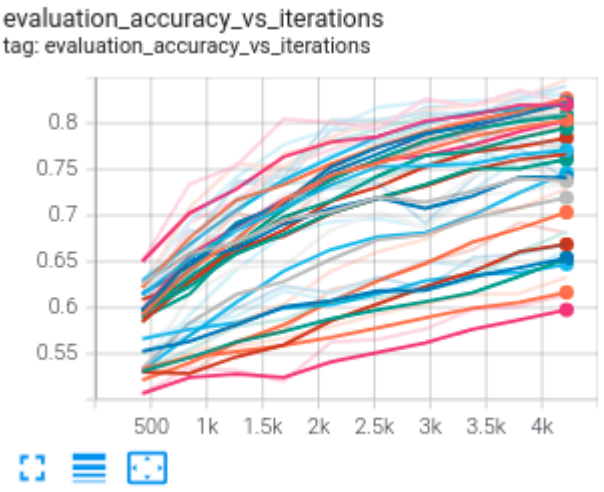
- Se genereaza valori pentru setul de date de test și sunt scrise într-un csv cu ajutorul funcției write_data

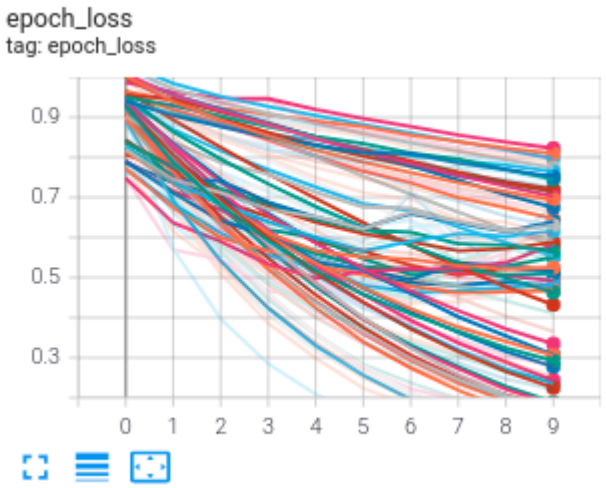
Convolutional neural network (cnn.py, morecnn.py, load_model.py):

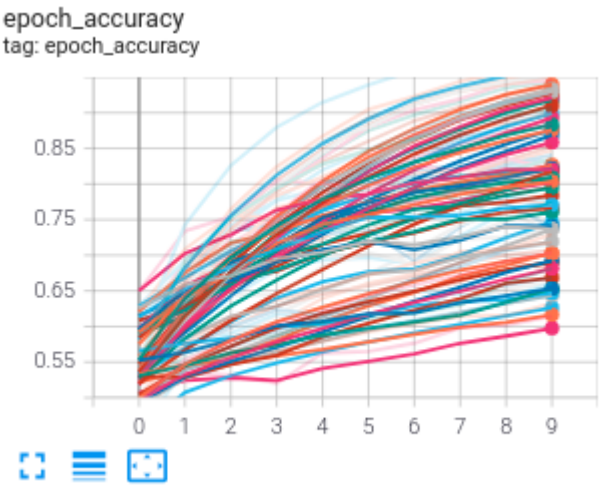
- Datele sunt deserializate din fisierele X_train.pickle, Y_train.pickle, X_validation.pickle, Y_validation.pickle, X_test.pickle

- Datele sunt normalizate prin împărțirea fiecărui valori la 255, valoare din care se scade 0.5
- Se creaza fișiere de logs pentru fiecare model, pentru a fi vizualizate cu ajutorul tensorboard









- Modelul este salvat pentru a putea fi încărcat ulterior
- Ultimul model conține 2 layere de convoluție, un layer de flatten, un layer de dense, un layer de droput și un layer final care mapeaza totul către o categorie și o rata de învățare de 0.001
- În total am testat 24 de rețele, modificand dimensiunea layerelor, numărul de layere de convolutie, numărul de layere de dropout, rata de învățare