Fakultet organizacionih nauka

Univerziteta u Beogradu

KORIŠĆENJE NEURONSKIH MREŽA ZA PREPOZNAVANJE TEKSTA

Prof. dr Zoran Ševarac
predmetni nastavnik

Ivan Jordanović (3735/2022)
ime i prezime (broj indeksa)

Beograd

Sadržaj

1. Uvod

Prepoznavanje rukom napisanog teksta je jedna od problema današnjice kojim se mnoge kompanije bave. Poslednjih godina napravljen je značajan napredak u razvoju modela mašinskog učenja za zadatke prepoznavanja teksta, koristeći velike skupove podataka i neuronske mreže. U ovom radu predstavljen je studijski projekat o performansi i tačnosti Tensorflow neuronske mreže za prepoznavanje rukom pisanog teksta, koristeći EMNIST Dataset.

Cilj projekta su da opiše arhitekturu i implementaciju neuronske mreže, proceni njene performanse na zadatom skupu podataka i pruži uvid u prednosti i ograničenja modela.

2. EMNIST dataset

EMNIST dataset (Extended MNIST) je proširenje klasičnog MNIST dataseta (Modified National Institute of Standards and Technology), koji je široko korišćen u oblasti prepoznavanja rukom pisanih cifara. EMNIST dataset je dizajniran da podrži zadatke prepoznavanja rukom pisanih karaktera koji pored cifara, uključuje i Abecedna slova, kako velika tako i mala.

Balanced EMNIST dataset je podskup EMNIST dataseta koji je posebno izbalansiran po broju instanci za svaku klasu. Ovaj podskup sadrži ukupno 131.600 slika i sastoji se od 47 klasa koje predstavljaju različite slova, brojeve i specijalne simbole. Svaka klasa ima približno jednaki broj instanci, što je važno za postizanje ravnoteže tokom obuke modela. Balanced EMNIST dataset je pogodan za istraživanja koja zahtevaju jednak broj uzoraka iz svake klase.

Kako bi se obezbedila konzistentnost i preciznost, EMNIST dataset je pažljivo prikupljen i anotiran. Sve slike su generisane pomoću standardnih fontova i uz pažljivo kontrolisane uslove osvetljenja. Osim toga, EMNIST dataset je dostupan u formatima koji su kompatibilni sa popularnim alatima i bibliotekama za mašinsko učenje, što ga čini pristupačnim i lako upotrebljivim za istraživanje i razvoj prepoznavanja rukom pisanih karaktera.

EMNIST dataset, posebno balanced EMNIST dataset, pruža obiman skup podataka za obuku i testiranje modela prepoznavanja rukom pisanih karaktera.

3. Tok projekta

Tok samog projekta učenja neurosnke mreže, se može predstaviti sledećim koracima:

1. Priprema dataseta
    a. Sam dataset moramo prvo dobaviti sa interneta. Iz skinute datoteke uzećemo samo **balansiran** skup podataka, odnosno sledeće datoteke
        - emnist-balanced-test.csv
        - emnist-balanced-train.csv
        - emnist-balanced-mapping.txt
2. Učitavanje i preprocesuiranje podataka
    a. U našu aplikaciju treba da učitamo CSV file, i odvojimo etikete od samih slika
    b. Potom, treba da „normalizujemo“ same podatke, što ćemo učiniti tako što ćemo svaki pixel podeliti sa 255
3. Definisanje modela i načina učenja istog
4. Samo učenje
5. Provera naučenog modela na našim test podacima

4. Problemi u toku pravljenja skripte

U poređenju sa učenjem neuronske mreže za razumevanje glasovnih komandi, učenje iste za razumevanje napisanih slova i karaktera je bila drastično lakša. Sama činjenica da su slike u ovom datasetu već bile u formatu 28x28, gde smo nakon normalizacije mogli odmah da krenemo da učimo sam model, je dosta pomogla u tome. A i sam model je imao dosta malo nivoa, gde smo morali samo da obratimo pažnju da izlazni nivo ima 47 izlaza.

Sama tačnost modela je takođe impresivna, na svega 10 epoha, tačnost se postiže sa skoro 85%, što nije bio slučaj sa glasovnim učenjem. Najveći problem su davali simboli koji su slični po prirodi kao što su malo „q“ i „9“, ili „0“ i „O“.

```
Epoch 5/10
2820/2820 [==============================] - 22s 8ms/step - loss: 0.3905 - accuracy: 0.8649 - val_loss: 0.4708 - val_accuracy: 0.8400
Epoch 6/10
2820/2820 [==============================] - 22s 8ms/step - loss: 0.3608 - accuracy: 0.8735 - val_loss: 0.4891 - val_accuracy: 0.8402
Epoch 7/10
2820/2820 [==============================] - 22s 8ms/step - loss: 0.3323 - accuracy: 0.8821 - val_loss: 0.4687 - val_accuracy: 0.8435
Epoch 8/10
2820/2820 [==============================] - 22s 8ms/step - loss: 0.3096 - accuracy: 0.8887 - val_loss: 0.4655 - val_accuracy: 0.8442
Epoch 9/10
2820/2820 [==============================] - 22s 8ms/step - loss: 0.2891 - accuracy: 0.8941 - val_loss: 0.4807 - val_accuracy: 0.8421
Epoch 10/10
2820/2820 [==============================] - 23s 8ms/step - loss: 0.2701 - accuracy: 0.8993 - val_loss: 0.4874 - val_accuracy: 0.8429
```

```
Test Accuracy: 0.8418000957497739
True Labels: ['e' '9' 'Q' 'q' 'X' 'E' 'B' '3' 'C' 'G']
Predicted Labels: ['e' '9' 'Q' '9' 'X' 'E' 'B' '3' 'C' 'G']
```

5. Učitavanje i preprocesovanje podataka

Nakon učitavanja, podaci se dalje obrađuju kako bi bili spremni za obuku modela neuronske mreže. Prvo, koristi se biblioteka pandas da bi se pročitali podaci iz CSV datoteka "emnist-balanced-train.csv" i "emnist-balanced-test.csv" i smestili u odgovarajuće pandas DataFrame-ove train_df i test_df.

Zatim, vrši se ekstrakcija podataka o slikama (vrednosti piksela) i oznakama (labelama) za skup podataka za obuku. Vrednosti piksela se normalizuju na opseg između 0 i 1 deljenjem sa 255.0. Slično se postupa i sa podacima za testiranje. Ekstraktuju se podaci o slikama i oznakama, normalizuju se vrednosti piksela i menjaju se dimenzije slika.

Na kraju, preprocesirani podaci se čuvaju u NPZ formatu (NumPy zipped format) u datoteci "preprocessed_data.npz" za kasniju upotrebu.

```
train_csv_file_path = "emnist-balanced-train.csv"
test_csv_file_path = "emnist-balanced-test.csv"

train_df = pd.read_csv(train_csv_file_path)
test_df = pd.read_csv(test_csv_file_path)

X_train = train_df.iloc[:, 1:].values.astype('float32')
y_train = train_df.iloc[:, 0].values
X_train /= 255.0
X_train = X_train.reshape(-1, 28, 28, 1)

X_test = test_df.iloc[:, 1:].values.astype('float32')
y_test = test_df.iloc[:, 0].values
X_test /= 255.0
X_test = X_test.reshape(-1, 28, 28, 1)

np.savez("preprocessed_data.npz", X_train=X_train, X_test=X_test,
y_train=y_train, y_test=y_test)
```

6. Treniranje Neuronske mreže

Nakon učitavanja preprocesiranih podataka iz datoteke "preprocessed_data.npz", podaci se dele na trening i test skupove, tj. X_train, X_test, y_train i y_test.

Zatim, definiše se model neuronske mreže koristeći Sequential Model. Model ima nekoliko slojeva, uključujući konvolucijski sloj (Conv2D) sa 32 filtera veličine 3x3 i funkcijom aktivacije ReLU, sloj za sažimanje (MaxPooling2D) veličine 2x2, sloj za ravnanje podataka (Flatten), potpuno povezani sloj (Dense) sa 64 neurona i ReLU aktivacijom, i na kraju, izlazni sloj (Dense) sa 47 neurona i softmax aktivacijom.

Nakon definisanja arhitekture, model se pokreće sa odgovarajućim parametrima. U ovom slučaju, koristi se Adam optimizator, sparse categorical crossentropy funkcija gubitka i meri se tačnost kao metrika.

Zatim, model se obučava na trening podacima sa 10 epoha, veličinom paketa od 32 uzorka i validacionim delom od 20% koji se koristi za praćenje performansi modela tokom obuke.

Na kraju, obučeni model se čuva u datoteci "handwriting_model.h5". Ovaj model može biti kasnije učitan za predikciju novih rukom pisanih karaktera.

```
data = np.load("preprocessed_data.npz")
X_train = data["X_train"]
X_test = data["X_test"]
y_train = data["y_train"]
y_test = data["y_test"]

model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28,
1)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(94, activation='relu'),
    tf.keras.layers.Dense(47, activation='softmax')
])
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2)
model.save("handwriting_model.h5")
```

7. Testiranje Neuronske mreže

Vrši se predikcija nad test podacima pomoću učitanog modela, čime se dobija sama predikcija. Kako je izlazni sloj modela definisan kao softmax sloj, primenjuje se np.argmax funkcija na izlazni vektor kako bi se dobile klasifikovane oznake (indeksi najverovatnijih klasa).

Nakon toga, računa se tačnost pogađanja poređenjem izlaznih informacijama sa stvarnim oznakama koristeći funkciju accuracy_score.

Zatim, pristupa se dekodiranju oznaka kako bi se dobile ljudski čitljive vrednosti. Prvo, čitaju se mapiranja oznaka iz datoteke "emnist-balanced-mapping.txt" i smeštaju u rečnik. Zatim se vrši dekodiranje oznaka kako bi se dobile odgovarajuće karakterne vrednosti za stvarne oznake i predviđanja.

```
data = np.load("preprocessed_data.npz")
X_test = data["X_test"]
y_test = data["y_test"]

model = tf.keras.models.load_model("handwriting_model.h5")


y_pred = model.predict(X_test)
y_pred_labels = np.argmax(y_pred, axis=1)
accuracy = accuracy_score(y_test, y_pred_labels)
print("Test Accuracy:", accuracy)

label_mapping = {}
with open("emnist-balanced-mapping.txt", "r") as f:
    lines = f.readlines()
    for line in lines:
        label, char = line.strip().split()
        label_mapping[int(label)] = chr(int(char))

y_test_decoded = np.array([label_mapping[label] for label in y_test])
y_pred_decoded = np.array([label_mapping[label] for label in y_pred_labels])

print("True Labels:", y_test_decoded[:10])
print("Predicted Labels:", y_pred_decoded[:10])
```