SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Raspoznavanje znamenaka korištenjem konvolucijskih neuronskih mreža

Meko računarstvo

Laboratorijska vježba 6

Ivan Gudelj

Diplomski studij računarstva, DRB

Osijek, 2022.

1. UVOD

Cilj pete laboratorijske vježbe je projektirati i ispitati konvolucijsku neuronsku mrežu koja će naučiti raspoznavati znamenke. Za treniranje i testiranje mreže smo koristili MNIST bazu, te za implementaciju mreže je korištena Python programski jezik uz pomoć ***Keras*** biblioteke s pripadajućim ***Tensorflow*** backend-om. MNIST baza sadrži 70000 slika znamenaka od kojih se koristi 60000 za treniranje,a ostalih 10000 za testiranje. Ovoj bazi je moguće lako pristupiti preko ***Keras*** biblioteke.

2. OPIS PROBLEMA I RJEŠENJA

2.1. Konvolucijska neuronska mreža

Duboko učenje je grana strojnog učenja, dok je konvolucijska neuronska mreža konkretna implementacija dubokih neuronskih mreža koja se najčešće primjenjuje za analizu i raspoznavanje slika.

Standardne neuronske mreže nisu osobito prigodne u svrhu analize slika zbog zahtjeva da svaki neuron mora biti povezan sa svim elementima iz prethodnog sloja. Stoga ako npr. imamo sliku veličine 25 x 25 što znaci ukupno 225 piksela u konačnici će rezultirati s 225 težina koje je potrebno podesiti i optimizirati samo za jedan jedini neuron. Stoga, ako je je slika skromnih 100 x 100 piksela to je već 10000 težina po neuronu i ako imamo 100 neurona u jednom sloju ispada 1000000 težina koje se moraju podesiti. Konvolucijske neuronske mreže imaju takvu arhitekturu koja omogućava da se raspoznaju uzorci različitih dimenzija uz puno manje parametara koje je potrebno podesiti i optimizirati. Smisao konvolucijske neuronske mreže jest da se kroz slojeve neuronske mreže uče raspoznavati inkrementalno kompleksniji uzorci. Ako se uči npr. raspoznavanje ljudskih lica, na prvoj razini se raspoznaju linije, rubovi i različiti točkasti uzorci, na drugoj su to npr. dijelovi lica kao što je to nos, uho, usta, oko i sl., dok na idućoj razini su to veći dijelovi lica itd.

Drugim riječima, svaki sloj konvolucijske neuronske mreže uči različitu razinu apstrakcije ulaznih podataka.

Iako konvolucijska neuronska mreža može imati mnogo različitih dijelova, tipični dijelovi uključuje sljedeće slojeve:

- Konvolucijski sloj (engl. Convolution layer)
- Sloj sažimanja (engl. Pooling Layer),
- Potpuno povezani sloj (engl. Fully-Connected (dense) Layer)

Unutarnja struktura tipične konvolucijske neuronske mreže sastoji se od nekoliko naizmjenično poslaganih višedimenzionalnih konvolucijskih slojeva i slojeva sažimanja. Na kraju se nalazi potpuno povezan sloj, koji je jednodimenzionalan, kao i izlazni sloj.

Fundamentalna razlika između konvolucijskog sloja i potpuno povezanog sloja (istovrstan standardnoj neuronskoj mreži), jest to što potpuno povezan sloj uči uzorke globalno dok konvolucijski sloj uči lokalne uzorke unutar malih dvodimenzionalnih prozora nazvanih jezgra (engl. kernel). Tako naučne lokalne uzorke konvolucijski sloj nauči raspoznavati na bilo kojem mjestu na slici, dok bi na primjer standardne neuronske mreže naučile raspoznavati samo na točno određenom mjestu.

Stoga ako bi neka značajka promijenila svoj položaj, standardna neuronska mreža bi morala ponovo biti naučena. Još jedna bitna značajka konvolucijskih slojeva je ta da može naučiti odnosno zadržati prostorne odnose između uzoraka.

Ako se na primjer u prvom konvolucijskom sloju uče osnovni uzorci kao što su rubovi, linije, prijelazi i sl. tada se u drugom sloju mogu naučiti kompleksne kompozicije tih osnovnih uzoraka iz prethodnog sloja. Time se iz sloja u sloj povećava kompleksnost apstraktnih vizualnih koncepata.

Konvolucijski slojevi i slojevi sažimanja se baziraju na 3D tenzorima nazvanima mape značajki (engl. feature map), koje imaju širinu, visinu i dubinu. Ulazni slojevi mogu isto imati te tri dimenzije, ako se radi o crno-bijeloj slici tada pored širine i visine ima samo dubinu od 1, dok slike u boji imaju dubinu 3 (crvena, zelena i plava komponenta slike).

Konvolucijski sloj radi na principu konvolucije gdje maleni prozor „klizi“ po slici, umnaža elemente (piksele) slike i potom ih sumira, te uz dodatak bias-a „pridružuje“ neuronu na odgovarajućem mjestu konvolucijskog sloja. Vizualno, obradu krećemo s prozorom u gornjem lijevom kutu slike koji daje potrebne informacije prvom (opet, gornji lijevi kut) neuronu konvolucijskog sloja. Nakon toga

pomičemo prozor za jedno mjesto u desno i povezujemo vrijednosti s drugim neuronom u konvolucijskom sloju i tako obilazimo cijelu sliku, s lijeva na desno, od gore prema dolje.

Ako imamo ulaznu sliku 28 x 28 pikesela (npr. MNIST baza slika znamenki) i prozor od 5 x 5, tada veličina konvolucijskog sloja može biti samo 24 x 24, jer nam ta dimenzija omogućava da obiđemo sve elemente ulazne slike s našim prozorom bez da narušavamo granice slike, odnosno možemo pomaknuti prozor samo za 23 mjesta u desno (odnosno prema dolje) bez da pređemo granicu slike. Dakle prema korištenom primjeru, svaki neuron iz skrivenog sloja je povezan s odgovarajućom regijom ulazne slike preko 5 x 5 prozora. To povezivanje je definirano s 5 x 5 matricom težina koja se naziva filter i pripadajućim bias-om. Potrebno je naglasiti da se ista matrica težina i isti bias koristi za sve neurone određenog podsloja unutar nekog konvolucijskog sloja neuronske mreže. Dakle samo te težine i bias se podešavaju tijekom učenja neuronske mreže (za svaki sloj). Što npr. znači da za jedan sloj gdje se koristi 5 x 5 filter podešava 26 vrijednost (25 težina + 1 bias) za razliku od npr. standardne neuronske mreže kada imamo 28 x 28 sliku na ulazu, podešavalo bi se 784 težine za svaki neuron, a ovdje samo 26 za svaki sloj.

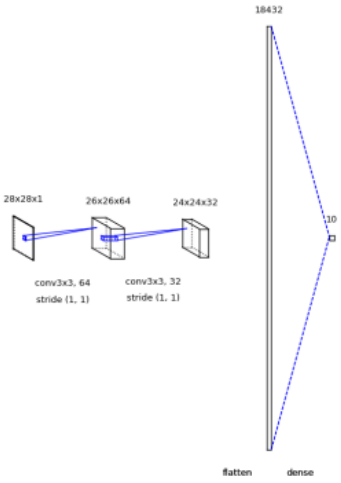
Filteri koji se ovdje koriste se mogu povezati s filterima koji se koriste za standardnu obradu slike (detekcija rubova, izoštravanje, zamućivanje i sl.). Individualni filtri mogu detektirai samo jedan tip značajki, stoga je preporučano da se koristi više filtera u isto vrijeme koji će detektirati različite značajke. U konvolucijskim slojevima se stoga nalazi više podslojeva istih dimenzija, svaki sa svojim filterom tj. težinama i bias-om.

2.2. Opis problema

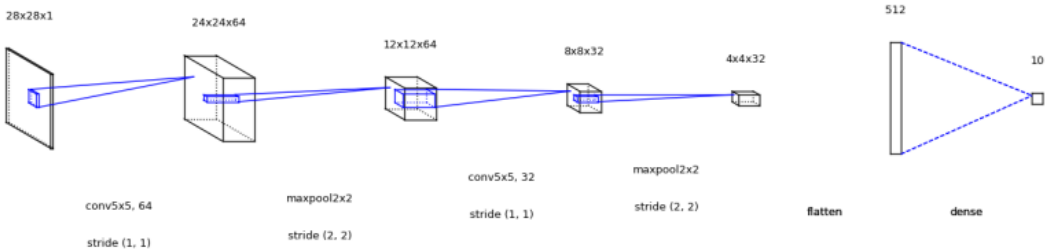
Potrebno je projektirati i ispitati konvolucijsku neuronsku mrežu koja će naučiti raspoznavati znamenke. Za treniranje i testiranje koristiti MNIST bazu, a za implementaciju koristiti Python programski jezik i Keras biblioteku s pripadajućim Tensorflow backend-om. MNIST baza sadrži 70000 slika znamenaka od kojih se 60000 koristi za treniranje, a 10000 za testiranje.

Prije korištenja podataka za treniranje i testiranje preporučljivo je skalirati ulazne podatke na interval 0→1, za razliku od originalnog intervala 0→255, a za korištenje kao ulaz konvolucijske neuronske mreže potrebno ga je pretvoriti u 3D

tenzor (28,28,1). Potrebno je implementirati 2 tipična modela konvolucijskih neuronskih mrežu koje su vidljive na sljedećim slikama.



Slika 8. 1. model konvolucijske neuronske mreže



Slika 9. 2. model konvolucijske neuronske mreže

Kao što se može vidjeti, prvi model sadrži samo dva konvolucijska sloja, prvi sa 64 filtera, a drugi s 32 filtera. Drugi model isto ima dva konvolucijska sloja s istim broj filtera kao i prethodni model, ali za razliku od prethodnog modela ima i dva sloja za sažimanje, jedan između konvolucijskih slojeva i jedan na kraju, prije potpuno povezanog modela.

Slojevi za sažimanje koriste max funkciju i veličinu prozora 2 x 2. Implementirati dva navedena modela i mijenjati im parametre na slijedeći način: Aktivacijska funkcija: 'relu', 'tanh', 'sigmoid' Veličina filtera (kernel-a) konvolucijskih slojeva: 3 x 3, 5 x 5, 7 x 7 Učiti stvorene modele kroz najviše 5 epoha. Usporediti performanse sa standardnom neuronskom mrežom za isti problem, raspoznavanje znamenaka u MNIST bazi, korištenjem scikit-learn biblioteke. Implementacija standardne neuronske mreže treba imati 100 neurona, koristiti 'logistic' aktivacijsku funkciju, 'adam' optimizacijski postupak i prolaziti kroz najviše 100 epoha za učenja.

Kod koji je korišten za pronalaženje rješenja nalazi se ispod:

#ZAJEDNIČKI KOD

activationFunctions = ["relu", "tanh", "sigmoid"]

sizeOfKernel = [3,5,7]

(X_train, y_train), (X_test, y_test) = mnist.load_data()

X_train = X_train.reshape(60000,28,28,1)

X_train = X_train.astype('float32') / 255

X_test = X_test.reshape(10000,28,28,1)

X_test = X_test.astype('float32') / 255

y_train = to_categorical(y_train)

y_test = to_categorical(y_test)

#MODEL 1

```
for singleKernel in sizeOfKernel:
    for actFunc in range(3):
        modelOne = models.Sequential()
        modelOne.add(layers.Conv2D(64, singleKernel,singleKernel), activation=activationFunctions[actFunc],
input_shape=(28,28,1)))
        modelOne.add(layers.Conv2D(32, singleKernel,singleKernel), activation=activationFunctions[actFunc])
        modelOne.add(layers.Flatten())
        modelOne.add(layers.Dense(10, activation="softmax"))
        modelOne.compile(optimizer = "sgd",loss = "categorical_crossentropy",metrics = ["accuracy"])

        modelOne.summary()
        start = time.time()
        past =  modelOne.fit(X_train,y_train,validation_data = (X_test,y_test),epochs = 5)
        end = time.time()
        difference = end - start
        modelOne.predict(X_test)
        #+++++KOD ZA ISPIS VRIJEDNOSTI++++++

        print("\nDONE WITH MODEL 2\n\n\ntime for training = ", str(difference), " seconds", "\nACCURACY =
", str(past.past['accuracy'][-1]))
        nameOfFile = "MODEL_2_KERNEL_SIZE=" + str(singleKernel)  + "x" + str(singleKernel) +
"&activationFunc=" + activationFunctions[i] + ".txt"
        dataForWriting = "RequiredTime = " + str(difference) + " seconds" + "\nAccuracy = " +
str(past.past['accuracy'][-1])
        with open(nameOfFile, 'w') as dat:
            dat.write(dataForWriting)
```

#MODEL 2

```
for singleKernel in sizeOfKernel:
    for actFunc in range(3):

        modelTwo = models.Sequential()
        modelTwo.add(layers.Conv2D(64, (singleKernel,singleKernel), activation=activationFunctions[actFunc],
input_shape=(28,28,1)))
        modelTwo.add(layers.MaxPooling2D((2, 2)))

        modelTwo.add(layers.Conv2D(32, (singleKernel,singleKernel), activation=activationFunctions[actFunc]))
        modelTwo.add(layers.MaxPooling2D((2, 2)))

        modelTwo.add(layers.Flatten())
        modelTwo.add(layers.Dense(10, activation="softmax"))

        modelTwo.compile(optimizer = "sgd",loss = "categorical_crossentropy",metrics = ["accuracy"])

        modelTwo.summary()
        start = time.time()
        past =  modelTwo.fit(X_train,y_train,validation_data = (X_test,y_test),epochs = 5)
        end = time.time()
        difference = end - start
        modelTwo.predict(X_test)
        #++++++KOD ZA ISPIS VRIJEDNOSTI++++++

        print("\nDONE WITH MODEL 2\n\n\ntime for training = ", str(difference), " seconds", "\nACCURACY =
", str(past.past['accuracy'][-1]))
        nameOfFile = "MODEL_2_KERNEL_SIZE=" + str(singleKernel)  + "x" + str(singleKernel) +
"&activationFunc=" + activationFunctions[i] + ".txt"
        dataForWriting = "RequiredTime = " + str(difference) + " seconds" + "\nAccuracy = " +
str(past.past['accuracy'][-1])
        with open(nameOfFile, 'w') as dat:
            dat.write(dataForWriting)
```

3. REZULTATI ZA TRAŽENI PROBLEM - Projektiranje i ispitivanje konvolucijske neuronske mreže

Potrebno je kreirati dva modela prema zadanim slikama iz poglavlja 2.2 te mijenjati parametre modela na sljedeći način:

- Aktivacijska funkcija: 'relu', 'tanh', 'sigmoid'
- Veličina filtera (kernel-a) konvolucijskih slojeva: 3 x 3, 5 x 5, 7 x 7

3.1. Ovisnost o aktivacijskoj funkciji - Model 1

Filter	(3,3)		
Aktivacijska funkcija	relu	tanh	sigmoid
Preciznost u zadnjoj epohi	0.978216648	0.96766668	0.918850004
Potrebno vrijeme (sec)	438.2923662	418.6585664	468.14175367

3.2. Ovisnost o aktivacijskoj funkciji - Model 2

Filter	(3,3)		
Aktivacijska funkcija	relu	tanh	sigmoid
Preciznost u zadnjoj epohi	0.972100019	0.9668499	0.880100011
Potrebno vrijeme (sec)	48.3805332	49.3661253	54.4116859

3.3. Ovisnost o veličini filtera konvolucijskih slojeva - Model 1

Aktivacijska funkcija	relu		
Filter	(3,3)	(5,5)	(7,7)
Preciznost u zadnjoj epohi	0.978216648	0.98278331	0.9844166636
Potrebno vrijeme (sec)	438.2923662	500.0768809	500.5900352

3.4. Ovisnost o veličini filtera konvolucijskih slojeva - Model 2

Aktivacijska funkcija	relu		
Filter	(3,3)	(5,5)	(7,7)
Preciznost u zadnjoj epohi	0.972100019	0.976916670	0.976199984
Potrebno vrijeme (sec)	48.38053321	54.026990890	51.31566095

3.5. Rješenje standardne neuronske mreže

Za parametre standardne neuronske mreže; Broj neurona = 100, broj skrivenih slojeva = 1 , aktivacijska = logistic , algoritam ucenja = adam su dobiveni sljedeći rezultati;

-Preciznost: 0.9622855

-Vrijeme izvođenja 175.48372745 sec

9.95918367e-01	0	1.02040816e-03	0	0	0	0	2.04081633e-03	1.02040816e-03	0
4.40528634e-03	9.88546256e-01	2.64317181e-03	0	0	0	1.76211454e-03	8.81057269e-04	1.76211454e-03	0
3.48837209e-02	9.68992248e-04	9.57364341e-01	9.68992248e-04	9.68992248e-04	0	0	3.87596899e-03	9.68992248e-04	0
2.27722772e-02	0	8.91089109e-03	9.60396040e-01	0	0	0	2.97029703e-03	1.98019802e-03	2.97029703e-03
2.54582485e-02	0	3.05498982e-03	0	9.64358452e-01	0	0	0	1.01832994e-03	1.01832994e-03
2.80269058e-02	1.12107623e-03	0	1.34529148e-02	2.24215247e-03	9.43946188e-01	2.24215247e-03	2.24215247e-03	3.36322870e-03	3.36322870e-03
1.35699374e-02	1.04384134e-03	2.08768267e-03	1.04384134e-03	3.13152401e-03	3.13152401e-03	9.73903967e-01	0	2.08768267e-03	0
3.01556420e-02	1.94552529e-03	6.80933852e-03	4.86381323e-03	2.91828794e-03	0	0	9.48443580e-01	0	4.86381323e-03
3.79876797e-02	0	1.02669405e-03	5.13347023e-03	4.10677618e-03	6.16016427e-03	0	3.08008214e-03	9.42505133e-01	0
2.77502478e-02	2.97324083e-03	0	4.95540139e-03	5.94648167e-03	2.97324083e-03	0	6.93756194e-03	9.91080278e-04	9.47472745e-01

4. ZAKLJUČAK

Rezultirajuća rješenja raspoznavanja znamenaka prikazana su u tablicama. Prikazane su ovisnosti neuronske mreže o aktivacijskim funkcijama kao i ovisnosti mreže o veličini filtera. Iz navedenih rješenja vidimo da je za model 1 najbolja kombinacija parametara; aktivacijska funkcija- **tanh,** veličina filtera - 3x3 **(**418.6585664 sec i 96.76% preciznosti**),** dok je za model 2 najbolja kombinacija parametara bila; aktivacijska funkcija - **relu** ,veličina filtera - **3x3** (48.38053 sec i 97.21% preciznosti).

Iz navedenih rezultata možemo vidjeti da je model 2 znatno brži. Što se tiče preciznosti, oba modela, u većini slučajeva, imaju preciznost preko 95%. Povećanjem veličine filtera kod oba modela dolazi do podizanja vremena potrebnog za analizu no i do povećanja preciznosti modela.

Bitno je naglasiti da je i standardna neuronska mreža dala dobre rezultate te je puno jednostavnija za implementaciju od konvolucijske neuronske mreže.