Detekcija i izračunavanje matematičkih izraza

Vladimir Inđić

Motivacija

Računanje matematičkih izraza koji su prikazani na slici ili napisani na papiru nije jednostavan proces. Zahteva ili manuelno računanje ili unošenje izraza u digitron. I jedan i drugi proces zahtevaju određeno vreme i podložni su greškama. Zar ne bi bilo lepo kada bismo samo mogli da slikamo izraz i dobijemo njegovo rešenje? Upravo to je motivacija za razvoj programa koji prepoznaje matematički izraz prikazan na slici i određuje njegov rezultat.

Konceptualno rešenje

Detekcija izraza prikazanog na slici zasniva se na dva osnovna principa:

- Obrada slike - podrazumeva da se slika u boji prebaci u crno-belo (engl. *grayscale*), uklone nebitni detalji sa nje (engl. *threshold*), izdvoje konture koje sadrže karaktere, odradi *resize* izdvojene konture u sliku odgovarajuće veličine.
- Prepoznavanje karaktera - tipičan problem koji može biti rešen upotrebom klasifikatora. U ovom programskom rešenju biće korišćene neuralna mreža (skraćeno *NN*) i konvolucijska neuralna mreža (skraćeno *CNN*).

Implementacija klasifkatora za detekciju matematičkih izraza

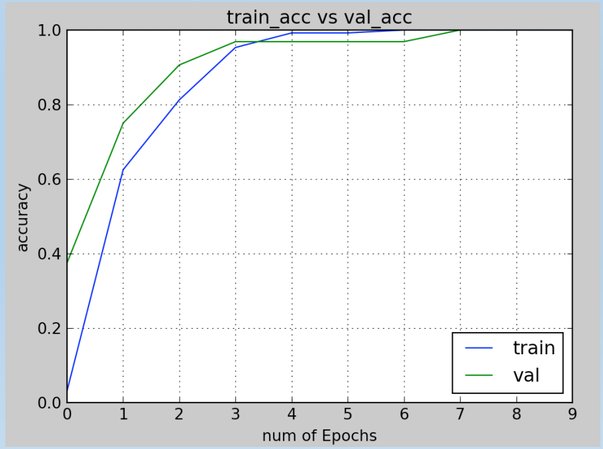
Dataset

- 160 slika dimenzija 32x32 napravljenih uz pomoć programa ***Pages*** i ***OpenCV*** biblioteke za kompjutersku viziju programskog jezika ***Python***
- 16 klasa – po jedna klasa za svaki karakter koji se može naći u izrazu:
  - cifre **0-9**,
  - operatori **+**, **-**, ***** i /,
  - male zagrade **(,)**
- Svaku klasu čini 10 slika istog karaktera prikazanog različitim fontom (*Helvetica, Times, Arial, Times New Roman, PT Mono, America Typerwriter, Apple Chancery, Comic Sans MS, Verdana, Andale Mono*)
- 80% slika za treniranje
- 20% slika za testiranje procesa treniranja

Slika 1. Nekoliko slika koje čine *dataset*

Neuralna mreža*

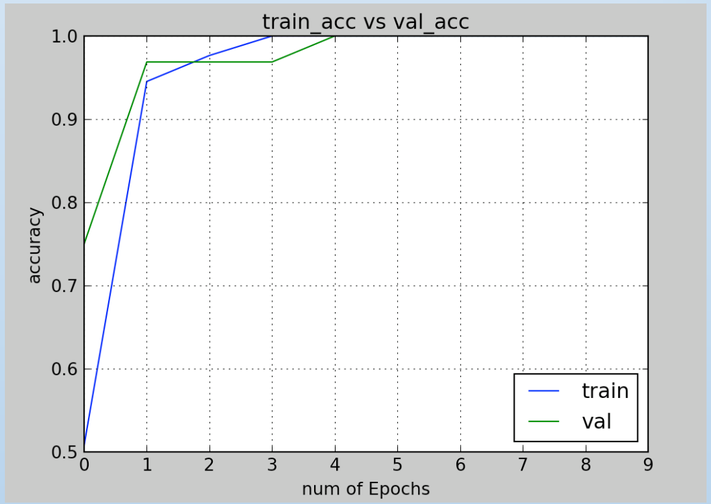
- Ulazni sloj od 1024 neurona
- Skriveni sloj od 1024 neurona
- Izlazni sloj od 16 neurona
- 10 epoha
- Uspesnošt treniranja 100%, postignuta u 8. epohi
- Vreme treniranja: **1,36s**



Slika 2. Uspešnost *NN* tokom procesa treniranja

Konvolucijska neuralna mreža*

- Konvolucijski sloj (*Conv2D*): 32 filtera, veličina prozora 5x5
- *Pooling* sloj (*MaxPooling2D*): veličina prozora 2x2
- *Dropout* sloj - zadužen za smanjenje overfittovanja (parametar *rate* postavljen na 0,2)
- *Flatten* sloj – matricu prevodi u niz
- Potpuno povezani sloj od 256 neurona
- Izlazni sloj od 16 neurona
- 10 epoha
- Uspešnost treniranja 100%, postignuta u 5. iteraciji
- Vreme treniranja: **12,87s**



Slika 3. Uspešnost *CNN* tokom procesa treniranja

Evaluacija klasifikatora

- 50 slika sa slučajno generisanim izrazima kojima je manuelno određena vrednost.
- Nekoliko novih fontova: *Georgia, Arial Hebrew, Apple Braile, Helti SC, Apple SD Gothic Neo, Courier, Apple Gothic, Comic Sans MS*
- Računanje vrednosti prepoznatog izraza pomoću ***Pythonove*** *eval* funkcije
- Uspešnost oba klasifikatora (*NN* i *CNN*) je 94-96%
- Greške ulgavnom na novim fontovima na kojima nije vršen proces treniranja (*Apple Gothic*, *Courier*)

Zaključak i dalja unapređenja programskog rešenja

Razvijeno programsko rešenje za detekciju izraza koristi dva klasifikatora. Oba su trenirana na relativno malom *datasetu*, ali i pored toga postižu zapažene rezultate. Očigledno da je za ovakav problem bolje koristiti *NN* umesto *CNN*. Prvenstveno zato što je vremenski period treniranja otprilike **10** puta manji. Treba uzeti u obzir i to da su slike dimenzija 32x32 jednostavne i da će se na malom *datasetu CNN overfittovati*, te ne može postići bolje rezultate od obične neuralne mreže za nove fontove.

Implementacija koncepta izloženog u ovom tekstu pokazala se kao veoma uspešna. Programsko rešenja bi se moglo unaprediti i za detekciju daleko većeg skupa izraza. Neophodno bi bilo uvesti veći broj fontova za treniranje klasifikatora, usložiti njihovu arhitekturu i uvesti pojedine algoritme za prepoznavanje redova u izrazu (recimo *KNN*), kako bi se mogli prepoznati i višelinijski izrazi.

*Implementirana korišćenjem ***Pythonove*** biblioteke ***Keras,*** koja je namenjena za *Deep Learning*.