

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1115

**DUBOKE KONVOLUCIJSKE
NEURONSKE MREŽE ZA
RASPOZNAVANJE ZNAKOVA**

Matija Ilijaš

Zagreb, lipanj 2015.

SADRŽAJ

1. Uvod	1
2. Optičko prepoznavanje znakova	2
3. Umjetne neuronske mreže	3
4. Duboke konvolucijske neuronske mreže	4
5. Implementacija	5
5.1. Sustav za učenje	5
5.1.1. Torch razvojni alat	5
5.1.2. Definiranje arhitekture	6
5.1.3. Definiranje parametara učenja	7
5.2. Sustav za testiranje	7
6. Rezultati	8
7. Zaključak	9

1. Uvod

Uvod rada. Nakon uvoda dolaze poglavlja u kojima se obrađuje tema.

2. Optičko prepoznavanje znakova

3. Umjetne neuronske mreže

Umjetna neuronska mreža je algoritam strojnog učenja inspiriran strukturom i funkcionalnošću ljudskog mozga. Zasniva se na paralelnoj obradi podataka, te vrlo dobro rješava sve probleme kod kojih postoji složena nelinearna veza ulaza i izlaza. Radi s velikim brojem parametara i varijabli, te može raditi s nejasnim podacima što ju čini robusnom na pogreške. Zbog svoje robusnosti, najčešća primjena neuronske mreže je u raspoznavanju uzoraka, obradi slike i govora, nelinearnom upravljanju itd.

Mreža se sastoji od tri vrste sloja: ulaznog, izlaznog, te skrivenog koji povezuje prethodna dva. Čvorišta, tzv. neuroni, skrivenog sloja povezani su težinskim vezama s ulaznim i izlaznim neuronima. Dvije su faze rada umjetnih neuronskih mreža: učenje (treniranje) i obrada podataka (eksploatacija). Učenje je iterativan postupak predočavanja ulaznih primjera i očekivanog izlaza pri čemu dolazi do postupnog prilagođavanja težina veza neurona. Nakon što se težine skrivenog sloja prilagode postupkom treniranja, eksploatacijom neuronske mreže može se za do tad neviđeni ulazni primjer dobiti pripadajući izlaz.

4. Duboke konvolucijske neuronske mreže

5. Implementacija

U sklopu ovog rada razvijen je sustav za učenje i testiranje različitih arhitektura neuronskih mreža. Prilikom razvijanja takvog sustava bilo je važno zadovoljiti dva glavna uvjeta: brzo učenje novih arhitektura na velikoj količini podataka te detaljna analiza rada naučenih mreža. Iz tog razloga sustav je podijeljen na dva podsustava razvijena sa zasebnim alatima. Podsustav zadužen za učenje implementiran je u razvojnom alatu Torch koji omogućuje paralelno izvođenje procesa učenja na grafičkoj kartici. Podsustav koji provodi analizu rada naučenih mreža implementiran je u jeziku C++ korištenjem OpenCV biblioteke.

5.1. Sustav za učenje

Kako bi cijeli proces istraživanja različitih arhitektura mreža i podataka bio što efikasniji, potrebno je prije svega imati brzi sustav za učenje. Učenje velikih arhitektura na velikoj količini podataka može trajati tjednima, pa i mjesecima, sekvencijalnim računanjem na standardnom procesoru računala. Stoga je tek pojavom kvalitetnih biblioteka, koje omogućuju optimizirano paralelno računanje na grafičkoj kartici, postalo moguće provoditi istraživanje u području dubokog učenja. Osim brzine učenja, važna je i jednostavnost definiranja parametara mreže koja se uči, kao i parametara samog procesa učenja. Kod implementacije ovog rada odabran je razvojni alat Torch zbog svoje jednostavnosti i brzine, ali i podrške velikih kompanija kao što su Facebook i Google, koje nerijetko dijele korisne implementacije sa zajednicom.

5.1.1. Torch razvojni alat

Prilikom odabira razvojnog alata uzeti su u obzir i alati Caffe i Theano. Možda najpopularniji alat za duboko učenje Caffe ima izrazito jednostavno sučelje za odabir parametara mreže i procesa učenja, no kao posljedica toga teško je

raditi promjene u postojećoj implementaciji u svrhu istraživanja ili razvoja. Theano s druge strane pruža puno manju razinu apstrakcije te ostavlja mogućnost lake izmjene većine implementacije, no postaje nepraktičan kod česte upotrebe standardnih algoritama. Torch je po razini apstrakcije negdje između prethodno navedenih alata, dovoljno je niske apstrakcije za laku izmjenu implementacije kod razvoja i istraživanja, ali ima i sučelje za standardne algoritme koje ga čini praktičnim.

Brzina samog izvođenja procesa učenja na grafičkim karticama velikim dijelom ovisi o pozadinskoj implementaciji paralelnog računanja koja je za sve navedene alate otprilike podjednako brza, te se toliko često mijenja da nema smisla uspoređivati njihove brzine. S druge strane, važan faktor kod odabira alata je podrška akademske zajednice i industrije s obzirom da o tome ovisi koliko brzo se razvijaju implementacije novih koncepata. Torch ima implementaciju za većinu novih koncepata u dubokom učenju zbog podrške kompanija kao što su Google, Facebook i Twitter. Tako je primjerice implementirana relativno nova metoda Dropout za regularizaciju prilikom učenja mreže te ReLU funkcija prijelaza koja je pokazale bolje rezultate od standardne sigmoidalne funkcije u nedavnim istraživanjima.

Sama implementacija Torcha podijeljena je na dva dijela. Sve računske operacije na grafičkoj kartici napisane su kao C/CUDA implementacija za maksimalnu brzinu, dok je ostatak funkcionalnosti implementiran u skriptnom jeziku LuaJIT za lakše korištenje alata.

5.1.2. Definiranje arhitekture

Neuronska mreža je u Torch alatu definirana kao model koji se sastoji od sekvencijalno povezanih slojeva koji vrše različite računske operacije. Tako bi na primjer jedna standardna neuronska mreža s jednim skrivenim slojem bila definirana kao sekvencijalni niz prikazan na Slici ???.

```
-- Simple 2-layer neural network
model = nn.Sequential()
model:add(nn.Reshape(ninputs))
model:add(nn.Linear(ninputs,nhiddens))
model:add(nn.Tanh())
model:add(nn.Linear(nhiddens,noutputs))
```

Slika 5.1

Primjer jedne duboke konvolucijske mreže definirane kao Torch model prika-

zan je na Slici ???.

```
-- hidden units, filter sizes
nstates = {20,40,180}
filtsize = 3
poolsize = 2

model = nn.Sequential()

-- stage 1 : filter bank -> squashing -> max pooling
model:add(nn.SpatialConvolutionMM(1, nstates[1], filtsize, filtsize))
model:add(nn.ReLU())
model:add(nn.SpatialMaxPooling(poolsize,poolsize,poolsize,poolsize))

-- stage 2 : filter bank -> squashing -> max pooling
model:add(nn.SpatialConvolutionMM(nstates[1], nstates[2], filtsize, filtsize))
model:add(nn.ReLU())
model:add(nn.SpatialMaxPooling(poolsize,poolsize,poolsize,poolsize))

-- stage 3 : standard 2-layer neural network
model:add(nn.View(nstates[2]*filtsize*filtsize))
model:add(nn.Dropout(0.5))
model:add(nn.Linear(nstates[2]*filtsize*filtsize, nstates[3]))
model:add(nn.ReLU())
model:add(nn.Linear(nstates[3], noutputs))
```

Slika 5.2

5.1.3. Definiranje parametara učenja

5.2. Sustav za testiranje

6. Rezultati

7. Zaključak

Zaključak.

DUBOKE KONVOLUCIJSKE NEURONSKE MREŽE ZA RASPOZNAVANJE ZNAKOVA

Sažetak

Ključne riječi: Ključne riječi, odvojene zarezima.

Deep convolutional neural networks for character recognition

Abstract

Abstract.

Keywords: Keywords.