Dirbtinis Intelektas
Darbas 3

Rytis Petronis

Sukurta ranka rašytų skaičių atpažinimo programa

Naudotas konvoliucinis neuroninis tinklas su adam optimizacija ir kategorine kryžminės entropijos nuostolių funkcija,

Tinklo sluoksniai:
Konvoliucinis relu,
Konvoliucinis 2 relu,
Maximalaus sujungimo
Išmetimo
Išlyginimo(Flatten)
Pilnai Sujungtas relu
Išmetimo
Pilnai Sujungtas softmax

Duomenys naudojami - MNIST, apdorotų ranka rašytų skaičių nuotraukų paverstų į skaičius

Naudojama

Keras(https://keras.io/) biblioteka
su Theano(http://www.deeplearning.net/software/theano/),
per Anaconda(https://www.anaconda.com/distribution/) python distribuciją

Panaudota gidas https://elitedatascience.com/keras-tutorial-deep-learning-in-python
Ir kodas paversti nuotraukai į MNIST formatą iš https://stackoverflow.com/a/55968472

Baigus mokymąsi priima pavadinimą nuotraukos, kurioje yra skaičius, kad bandytų atpažinti, apdoroja, kad paverstų į MNIST formatą ir spausdina spėjimą pagal tuos duomenis.

Pakeitimai:

Atnaujinta kad tiktų naujesnei Keras versijai, negu gide:

Pridėta theano pasirinkimas, nes normaliai visada norėjo tensorflow

import os; os.environ['KERAS_BACKEND'] = 'theano'

Pakeista duomenų forma:

X_train = X_train.reshape(X_train.shape[0], 28, 28, 1) # buvo 1,28,28 neveikė

Atnaujinta konvoliucijos aprašymas iš Convolution2D(32, 3, 3 į Conv2D(32, (3, 3)

model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))

Pridėta sukurtų skaičių nuotraukų pavertimo į MNIST tipo ir pakeitimo į modeliui priimtinus duomenis ciklas, suformatuota nuotraukos duomenys į tinkamą modeliui formatą:

```
import imageinput
while 1:
  x = np.asarray(
     imageinput.imageprepare("C:\\source\\ocr" + '\\' + raw_input("name of image") +
".png")).reshape((-1,28,28,1))
  #print(x)
  a = model.predict_classes(x)
  print(a)
```

Duomenys:

Neuroniniam tinklui pateikiama 60000 ilgio mnist duomenų 28x28 masyvas padalintas iš 255, kad gauti reikšmes tarp 0 ir 1

Gražina 10 skaičių masyvą, kuriame yra tikimybės nuo 0 iki 1, kad tai yra tas skaičius nuo 0 iki 9 iš jo didžiausia tikimybė laikoma atsakymu

Spejimo rezultatai iš duomenų priklauso nuo užpildyto nuotraukos skaičiumi kiekio, nes MNIST duoda visus viduryje parašytus skaičius. Tinklas vis tiek galės atpažinti ne tiksliai viduryje, bet gali būti mažiau tikslus.

Pavyzdys iš mano kodo:
Didelis 1:

Mažas 1:

Abu gražina tą patį atsakymą, bet antras mažiau tikslus

```
name of image0
[1]
[[0.0006 0.9387 0.0093 0.0173 0.0014 0.0006 0.0003 0.0079 0.022  0.0019]]
name of image2
[1]
[[0.0102 0.5766 0.0239 0.0764 0.0251 0.0085 0.0153 0.1268 0.0182 0.119 ]]
```

Duomenų Padalijimas:

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

Keras turi duomenis iš anskto padalintus ir atsitiktinai išdėliotus,
60000 mokymui,
10000 testavimui

Tikslumas nustatomas išrenkant didžiausią tikimybę iš 10 kategorijų ir palyginant su teisingu atsakymu visiems duomenims, apskaičiuojamas vidurkis, aprašyta keras git kode: https://github.com/keras-team/keras/blob/c2e36f369b411ad1d0a40ac096fe35f73b9dffd3/keras/metrics.py#L13

Testavimas Neapmokius

Kad neuroninis tinklas galėtų teisingai atpažinti skaičius, paslėptuose neuronų filtruose turi būti sukalibruoti svoriai naudojant duomenis, kurie turi teisingus atsakymus. Kai yra teisingai sukalibruotas tinklas, jis galės nematytiems duomenims priskirti kategoriją, kuri turi didesnį šansą būti teisinga.

Visiškai neapmokius tinklo tikslumas - 0.08749
Apmokius nedaug(pirmoj epochoje po 9184 duomenų) tikslumas - 0.5853

Tikslumas labai mažas, nes mažai sukalibruoti paslėpti neuronai. Praėjus 1 epochą tikslumas tampa 0.9852, čia jau labai gerai sukalibruota, nes visi filtrai turėjo laiko paveikti duodamus duomenis, kad kategorizacija būtų teisinga

Apmokymas

Pabandyta sumažinti, padidinti konvoliucijos filtrų skaičių, išmetimo procentus

Tikslumas 1 - testavimo duomenys, Tikslumas 2 - mokymo duomenys

epochų	Conv2d filtrai	Išmetimas 1	išmetimas 2	Tikslumas 1	Tikslumas 2
10	32(3x3),32(3x3)	0,25	0,5	0,9911	0,9913
10	64(3x3),64(3x3)	0,4	0,75	0,9912	0,9913
10	8(3x3),8(3x3)	0,4	0,75	0,9869	0,9552
10	8(3x3),8(3x3)	0,25	0,5	0.99	0.9829
10	8(3x3),8(5x5)	0,4	0,75	0,9897	0,9658

Tikslumas visada išlieka labai panašus, programa su mažesniais filtrų skaičiais daug greičiau pereina epochas ir tik šiek tiek mažiau tiksli.

Testavimas Apmokius su savo duomenimis

Eilutė skaičius, stulpelis spėjimai 10 epochų, 8 filtrai konvoliucijai

	0	1	2	3	4	5	6	7	8	9
0.png	0,9973	0	0,0003	0	0	0	0,0002	0,0001	0,0006	0,0016
1.png	0,0039	0,4178	0,0392	0,0097	0,2359	0,0016	0,0025	0,2518	0,0109	0,0269
2.png	0	0,0001	0,9987	0,0001	0	0	0	0,0011	0	0
3.png	0	0	0	1	0	0	0	0	0	0
4.png	0	0,01	0,00	0,0002	0,7189	0,0051	0,0002	0,0021	0,1705	0,0908
5.png	0	0	0	0,0001	0	0,9997	0	0	0	0,0002
6.png	0	0	0	0,0008	0	0,9585	0,0035	0	0,0127	0,0245
7.png	0	0,0001	0,0019	0,0019	0	0	0	0,9961	0	0
8.png	0	0	0	0	0	0	0	0	1	0
9.png	0	0	0,0039	0,3548	0,0002	0,0035	0	0,0003	0,1033	0,5342

Atpažino gerai visus išskyrus 6, kuris pabandžius kelis kartus galvoja, kad yra 5, 8 ar 9
6 neatpažino geriau pakeitus konvoliucijos filtrų kiekius