**FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK**

Obrada slike i računalni vid

**SEMINARSKI RAD**

Prepoznavanje objekata pomoću dubokih neuronskih mreža

Dino Brkić

**Osijek, 2020.**

1. **Uvod**

Duboko učenje dio je obitelji metoda strojnog učenja temeljenog na umjetnim neuronskim mrežama s reprezentativnim učenjem. Učenje može biti nadzirano, nenadzirano ili podržano. Arhitekture dubokog učenja poput dubokih neuronskih mreža, povratnih neuronskih mreža i konvolucijskih neuronskih mreža primijenjene su na polja koja uključuju računalni vid, prepoznavanje govora, obradu prirodnog jezika, filtriranje društvenih mreža itd.

Duboka neuronska mreža je umjetna neuronska mreža s više slojeva između ulaznog i izlaznog sloja. Sastoje se od više razina nelinearnih operacija, poput neuronskih mreža s mnogim skrivenim slojevima.

Prepoznavanje objekata je tehnologija u području računalnog vida i obrade slika koja se bavi otkrivanjem primjera semantičkih objekata određene klase u digitalnim slikama i video zapisima. Uvelike se koristi u zadacima računalnog vida kao što su prepoznavanje aktivnosti, obilježavanje slike, detekcija lica, prepoznavanje lica, kosegmentacija video objekata. Također se koristi u praćenju objekata kao npr. osoba i automobila u prometu i kolnicima.

U ovom radu će se za prepoznavanje objekata koristiti tri unaprijed istrenirane duboke mreže te testirati na datasetu od nekoliko desetaka slika.

1. **MODELI DUBOKIH MREŽA**
   1. **VGG16**

VGG neuronske mreže su neuronske mreže bazirane na AlexNet neuronskoj mreži. Dok su se ostale derivacije AlexNeta usredotočivale na manje veličine prozora i korake u prvom konvolucijskom sloju, VGG se usredotočivao drugim važnijim aspektima konvolucijskih neuronskim mreža tj. dubinom. Dataset na kojem je VGG treniran je djelić ImageNet dataseta. Arhitektura VGG se sastoji od ulaznog sloja, konvolucijskog, potpuno povezanog i skrivenog sloja.

Ulazni sloj prima RGB sliku veličine 224×224 piksela. Konvolucijski slojevi koriste vrlo male konvolucijske filtre (3×3 piksela najmanja moguća veličina bez redundancija). Postoje također i 1×1 piksela konvolucijski filtri koji djeluju kao linearna transformacija ulaza, nakon čega slijedi ReLU aktivacijska funkcija. Konvolucijski korak je fiksiran na 1 piksel, tako da se nakon konvolucije zadrži prostorna rezolucija. VGG ima tri potpuno povezana sloja; prva dva imaju 4096 kanala, a treći ima 1000 kanala, jedan za svaku klasu. Kod svih skrivenih slojeva VGG-a se koristi ReLU aktivacijska funkcija, koja znatno smanjuje vrijeme učenja. VGG uglavnom ne koristi lokalnu normalizaciju jer lokalna normalizacija povećava potrošnju memorije i vrijeme učenja bez posebnog povećanja točnosti.

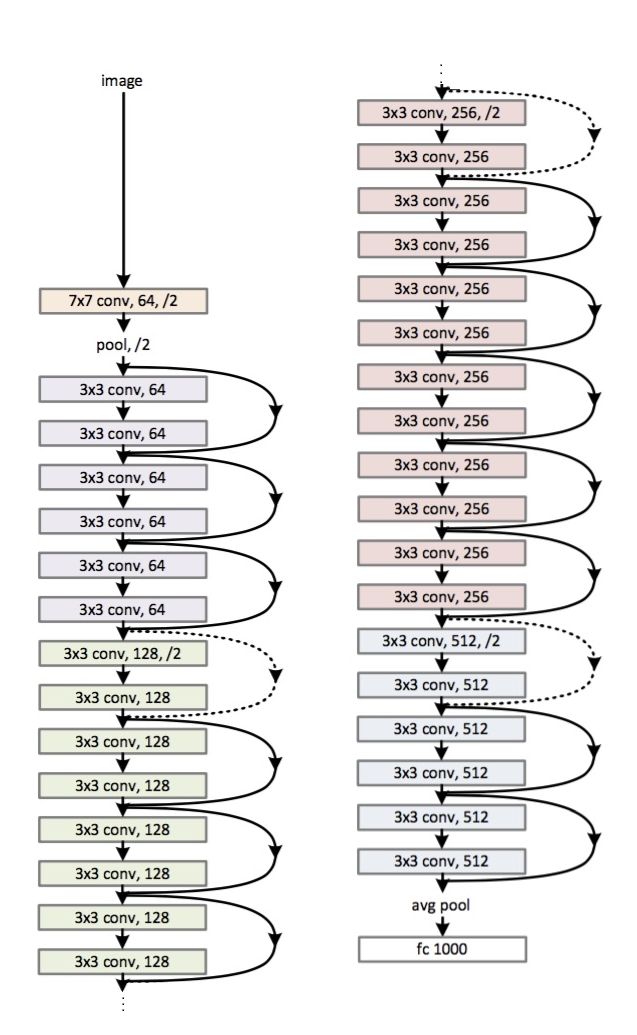


Slika 2.1 VGG16 CNN

* 1. **ResNet**

Rezidualna neuronska mreža je umjetna neuronska mreža koja se temelji na konstrukcijama poznatim iz piramidalnih stanica u moždanoj kori. Preostale neuronske mreže to čine pomoću preskočenih veza ili prečaca za preskakanje pojedinih slojeva. Tipični ResNet modeli implementirani su sa dvo ili troslojnim preskocima koji sadrže nelinearnosti i batch normalizaciju. Kod HighwayNet modela može se koristiti i dodatna matrica težina za učenje težina preskakanja. ResNet je treniran na CIFAR-10 datasetu no ResNet50 koji je korišten u ovom radi je treniran na dijelu ImageNet dataseta.

ResNet arhitektura ima ulazni sloj, konvolucijske slojeve i potpuno povezani sloj, no kod konvolucijskih slojeva uvodi preskakanje slojeva. Ulazni sloj prima RGB sliku veličine 224×224 piksela. Kod konvolucijskih slojeva uvodi rezidualne blokove za preskakanje slojeva. Takav model omogućava izbjegavanje problema nestajanja gradijenta ponovnom uporabom aktivacija iz prethodnih slojeva dok susjedni sloj ne nauči težinu. Tako se tijekom treninga težine prilagođavaju kako bi se ublažile veličine ulaznog sloja i pojačali prethodno preskočeni slojevi. U najjednostavnijem slučaju prilagođavaju se samo težine za vezu susjednog sloja, bez utjecaja težine gornjeg sloja. To najbolje radi kada se prijeđe preko nelinearnog sloja ili kada su međuslojevi linearni. Potpuno povezani sloj se sastoji od tisuću neurona za ImageNet klase(ResNet50), prije kojeg prethodi average pooling sloj.

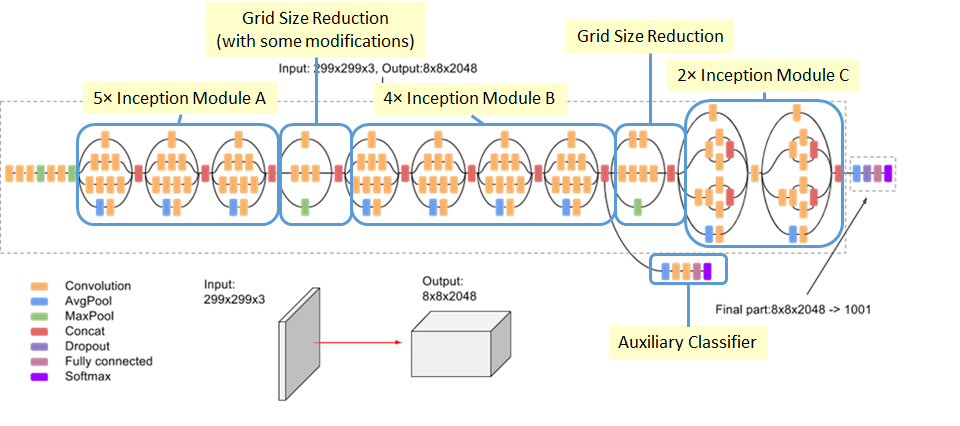


Slika 2.2 ResNet arhitektura, općenito

* 1. **InceptionV3 (GoogLeNet)**

Inception konvolucijske neuronske mreže su Googlove mreže za analizu slika i detekciju objekata. InceptionV3 je treće izdanje Googlove Inception konvolucijske neuronske mreže u kojima je ključna značajka faktoriziranje konvolucije u tzv. Inception slojevima. InceptionV3 je treniran na ImageNet datasetu od tisuću klasa. Arhitektura InceptionV3 modela sastoji se od ulaznog sloja, višestrukih inception sloja, sloja za smanjenje veličine mreže i izlaznog sloja.

Glavna ideja inception sloja je da pokrije što veće područje, ali da zadrži finu različivost za male informacije o slikama. Ideja je paralelno spojiti različite veličine od najmanjih do većih. Također se nastoji da niz filtara različitih veličina odradi bolje skaliranje više objekata. Prednost je što se mogu učiti svi filtri na inception sloju. Time se značajno poboljšavaju performanse na dubokom učenju, no visok broj parametara znači da je model skloniji overfittingu. Kako bi se to spriječilo, koriste se uska grla u inception slojevima. Korištenjem uskih grla može se obnoviti inception sloj s više nelinearnosti i manje parametara. Dodatno se dodaje i max pooling sloj kako bi se sažeo sadržaj prethodnog sloja. Nakon toga se svi rezultati međusobno spajaju i daju sljedećem sloju (nadovezivanje). InceptionV3 za razliku od prethodne dvije mreže u ulaznom sloju prima RGB sliku veličine 299×299 piksela.



Slika 2.3 InceptionV3 arhitektura

1. **RJEŠENJA I PRIMJERI**
   1. **VGG16**

from keras.preprocessing.image import load\_img

from keras.preprocessing.image import img\_to\_array

from keras.applications.vgg16 import preprocess\_input

from keras.applications.vgg16 import decode\_predictions

from keras.applications.vgg16 import VGG16

model = VGG16()

pic = input("Ime slike:")

path = "images\\" + pic

slika = load\_img(path, target\_size=(224, 224))

slika = img\_to\_array(slika)

slika = slika.reshape((1, slika.shape[0], slika.shape[1], slika.shape[2]))

slika = preprocess\_input(slika)

objekt = model.predict(slika)

labels = decode\_predictions(objekt)

labels = labels[0][0]

print('%s (%.2f%%)' % (labels[1], labels[2]\*100))

Kod 3.1 Programsko rješenje VGG16 modela

Kod svakog modela se primjenjuje sličan pristup pri kreiranju modela i testiranju pojedinih slika. Prvo se kreira model mreže koja se koristi, te se učitava slika u dimenzijama kompatibilnim sa modelom. Nakon toga se slika transformira u 3D numpy objekt te potom u tenzor četvrtog reda koji se koristi kao ulaz za model. Slika se dodatno predprocesira u svrhu dobitka boljih rezultata. Nakon toga se evaluira slika sa definiranim modelom. Kod ispisa oznaka i njihovih pripadajućih vjerojatnosti ispisuje se samo oznaka i pripadajuća vjerojatnost sa najvećom vjerojatnošću.



mountain\_bike (89.69%)

Slika 3.1 Izlaz VGG16 modela za sliku brdskog bicikla



church (72.26%)

Slika 3.2 Izlaz VGG16 modela za sliku kuće

Za više primjera i cijeli ispis testiranih slika vidjeti prilog outputs.txt.

* 1. **ResNet50**

import keras

import numpy as np

from keras.applications import resnet50

from keras.preprocessing.image import load\_img

from keras.preprocessing.image import img\_to\_array

from keras.applications.imagenet\_utils import decode\_predictions

import matplotlib.pyplot as plt

model = resnet50.ResNet50()

pic = input("Ime slike:")

path = "images\\" + pic

slika = load\_img(path, target\_size=(224, 224))

npSlika = img\_to\_array(slika)

npOrgSlika = np.expand\_dims(npSlika, axis=0)

ppSlika = resnet50.preprocess\_input(npOrgSlika.copy())

outSlika = model.predict(ppSlika)

labels = decode\_predictions(outSlika)

labels = labels[0][0]

print('%s (%.2f%%)' % (labels[1], labels[2]\*100))

Kod 3.2 Programsko rješenje ResNet50 modela



picket\_fence (32.63%)

Slika 3.3 Izlaz ResNet50 modela za sliku kuće



Leonberg (37.64%)  
Slika 3.4 Izlaz ResNet50 modela za sliku psa

Za više primjera i cijeli ispis testiranih slika vidjeti prilog outputs.txt.

* 1. **InceptionV3**

import keras

import numpy as np

from keras.applications import inception\_v3

from keras.preprocessing.image import load\_img

from keras.preprocessing.image import img\_to\_array

from keras.applications.imagenet\_utils import decode\_predictions

import matplotlib.pyplot as plt

model = inception\_v3.InceptionV3()

pic = input("Ime slike:")

path = "images\\" + pic

slika = load\_img(path, target\_size=(299, 299))

npSlika = img\_to\_array(slika)

npOrgSlika = np.expand\_dims(npSlika, axis=0)

ppSlika = inception\_v3.preprocess\_input(npOrgSlika.copy())

outSlika = model.predict(ppSlika)

labels = decode\_predictions(outSlika)

labels = labels[0][0]

print('%s (%.2f%%)' % (labels[1], labels[2]\*100))

Slika 3.3 Programsko rješenje InceptionV3 modela



Labrador\_retriever (29.65%)

Slika 3.5 Izlaz InceptionV3 modela za sliku psa



notebook (76.66%)  
Slika 3.6 Izlaz InceptionV3 modela za sliku laptopa

Za više primjera i cijeli ispis testiranih slika vidjeti prilog outputs.txt.

1. **ZAKLJUČAK**

Prepoznavanje objekata područje je računalnog vida. Ovisno o tipu mreže, njenim težinama i podacima nad kojim je učena, izlazi se mogu razlikovati ili biti približno jednaki. Npr. korištenjem bilo koje od ove tri mreže za sliku bicikla dobit će se izlaz mountain\_bike dok kod drugih nije takav slučaj, npr. kod slike psa. Ovdje je razlika uglavnom skup podataka nad kojim je mreža trenirana, te zbog toga daje različite izlaze.

1. **LITERATURA**
2. https://engmrk.com/kerasapplication-pre-trained-model/?utm\_campaign=News&utm\_medium=Community&utm\_source=DataCamp.com
3. Raspoznavanje uzoraka i strojno učenje, loomen, LV predlošci