SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 3940

VREDNOVANJE RASPOZNAVANJA KOŠTICA MASLINA KONVOLUCIJSKIM NEURONSKIM MREŽAMA

Dario Pažin

Zagreb, lipanj 2015.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA ZAVRŠNI RAD MODULA

Zagreb, 10. ožujka 2015.

ZAVRŠNI ZADATAK br. 3940

Pristupnik: **Dario Pažin (0036455828)**
Studij: Računarstvo
Modul: Računalno inženjerstvo

Zadatak: **Vrednovanje raspoznavanja koštica maslina konvolucijskim neuronskim mrežama**

Opis zadatka:

Područje rada je klasificiranje izdvojenih okana stvarnih slika u različite vrste objekta ili pozadinu. Pretpostavljamo da izdvojena okna sadrže koštice maslina na temelju kojih treba raspoznati sortu masline. Automatsko pronalaženje okana za klasificiranje u ulaznim slikama nije predmet interesa ovog rada. Glavni zadatci rada su ručno označavanje okana za učenje, validaciju i testiranje te analiza, eksperimentalno vrednovanje i možebitno poboljšanje postojećih izvedbi raspoznavanja temeljenih na konvolucijskim neuronskim mrežama.
U okviru rada, potrebno je iz literature proučiti različite pristupe za ostvarivanje raspoznavanja objekata. Posebnu pažnju posvetiti postupcima koji se temelje na konvolucijskim neuronskim mrežama. Uhodati postupak ručnog označavanja položaja objekata u slikama i zadavanja njihovog točnog razreda. Ručno označiti skupove slika za učenje, validaciju i testiranje. Analizirati prethodno razvijenu izvedbu raspoznavanja konvolucijskim neuronskim mrežama. Validirati hiperparametre i vrednovati generalizacijsku točnost postupka. Prikazati i ocijeniti ostvarene rezultate.
Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Zadatak uručen pristupniku: 13. ožujka 2015.
Rok za predaju rada: 12. lipnja 2015.

Mentor:
Izv. prof. dr. sc. Siniša Šegvić

Djelovođa:
Prof. dr. sc. Danko Basch

Predsjednik odbora za završni rad modula:
Prof. dr. sc. Mario Žagar

Sadržaj

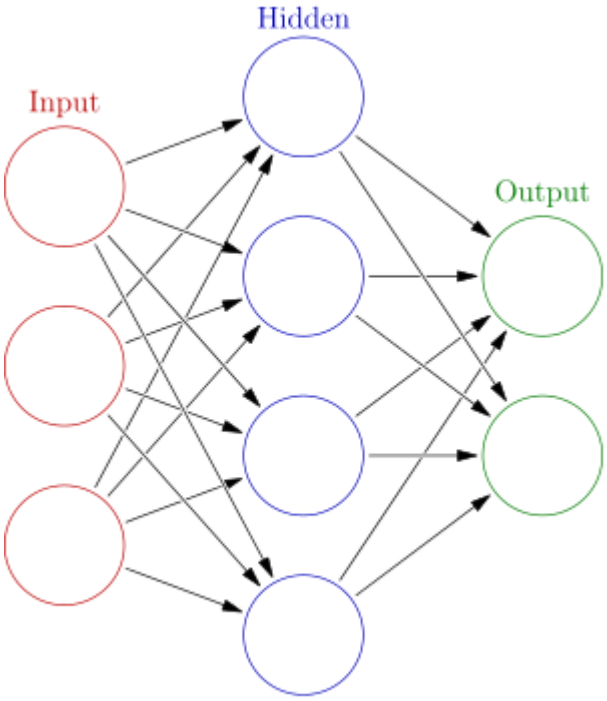
Uvod

Ljudski vid osjetilo je pomoću kojeg čovjek ostvaruje percepciju objekata ili pojava u okolini koja ga okružuje. Računalni vid istražuje percepciju objekata i pokušava što vjernije oponašati ljudski vid i pronaći rješenja koja bi pomogla ljudima u uočavanju stvari iz okoline. Jedan od takvih primjera je raspoznavanje sorti masline pomoću koštica maslina. Promatranjem je uočeno da se koštice različitih sorti maslina, osim po kemijskom sastavu, razlikuju također prema nekim fizikalnim svojstvima. Pronađeno je više karakteristika po kojima se koštice razlikuju, a neke od jednostavnijih su oblik koštice, izgled površine koštice i simetrija same koštice. Postupak promatranja i klasificiranja maslina prema fizikalnim karakteristikama koštice dugotrajan je i naporan postupak. Postoje srodne sorte kod kojih su razlike minimalne i gdje su potrebna kemijska testiranja, što je skup postupak. Ideja ovog rada je napraviti klasifikaciju koštica maslina pomoću konvolucijskih neuronskih mreža kako bi olakšali ljudima taj proces. Vjerojatno je da je broj fizikalnih karakteristika veći od onog što su ljudi uočili promatranjem, a postoji mogućnost da bi ih konvolucijska neuronska mreža mogla uočiti i tako povećati uspješnost klasifikacije.

U ovom radu bit će objašnjene umjetne neuronske mreže s naglaskom na konvolucijske neuronske mreže. Bit će opisan izgled korištene mreže, njezin način rada te predstavljeni dobiveni eksperimentalni rezultati. Poseban dio u radu zauzima *Caffe* koji je trenutno najbolji radni okvir za rad s konvolucijskim neuronskim mrežama i dosta olakšava cijeli postupak rada s konvolucijskim neuronskim mrežama. Predstavljene su upute za instalaciju, opisan način rada i korištene tehnologije kao što su Google Protocol Buffers i Hierarchical Data Format. Opisana je arhitekture mreže, objašnjeni korišteni slojevi i parametri koji su važni za točnost postupka klasifikacije.

1. Umjetne neuronske mreže

Umjetne neuronske mreže skupina su statističkih algoritama učenja koji se koriste za estimiranje i aproksimaciju funkcija koje ovise o velikom broju ulaza i općenito su nepoznate [1]. Umjetne neuronske mreže inspirirane su biološkim neuronskim mrežama, a njihov rad se temelji na radu neurona u mozgu. Mreža stvara konekcije između elemenata koji obrađuju informaciju, analogno neuronima u mozgu. Takvi neuroni mogu biti fizički ili digitalno stvoreni. Svaki neuron prima više ulaznih signala i proizvodi izlazni signal koji prosljeđuje dalje drugim neuronima. Neuroni su međusobno povezani i organizirani u različite slojeve. Ulazni slojevi primaju ulazne podatke, izlazni proizvode krajnje rezultate, a između njih postoje skriveni slojevi koji obrađuju podatke (Slika 1.1). Razvoj umjetnih neuronskih mreža počeo je sredinom 20. stoljeća s otkrivanjem bioloških funkcija neurona i razvojem računala koja su bila sposobna obrađivati i modelirati takvu mrežu. Prvi umjetni neuron baziran na matematici stvorili su 1943. godine neuropsiholog Warren McCulloch i logičar Walter Pist. Tehnologija dostupna u to vrijeme nije im omogućila daljni razvoj, a i njihov pristup rješavanju problema bio je konstrukcija, a ne učenje. Frank Rosenblatt je 1958. godine kreirao perceptron, algoritam za raspoznavanje obrazaca baziran na mreži s dva sloja koja je koristila zbrajanje i oduzimanje. Neuronske mreže se ističu po svojim dobrim svojstvima kao što su otpornost na šum, robusnost na pogreške u podatcima, paralelna obrada podataka i mogućnost učenja. Zbog svojih dobrih svojstava i razvoja tehnologije neuronske mreže se sve više koriste u svakodnevne svrhe. Velike financijske institucije koriste mreže za određivanje kreditnih rejtinga, evaluaciju kredita, ciljani marketing i za praćenje kartičnih transakcija gdje je njihova zadaća otkrivanje mogućih kartičnih manipulacija. Pokazalo se da neuronske mreže postižu za par posto bolje rezultate od dosadašnjih metoda što je veliki profit s obzirom na količinu novca kojom financijske institucije upravljaju. Još jedna od zanimljivih primjena mreža su detektori za bombe na pojedinim aerodromima u Sjedinjenim Američkim Državama gdje se mreže koriste za detekciju tragova eksploziva. Postoji više različitih arhitektura neuronskih mreža od kojih su najbitnije: klasične duboke neuronske mreže, neuronske mreže s povratnim vezama i konvolucijske neuronske mreže. Za probleme vizualne klasifikacije i potrebe ovog rada koristit će se konvolucijske neuronske mreže koje će biti detaljnije objašnjenje.



Slika 1.1. Prikaz višeslojne neuronske mreže [7]

2. Konvolucijske neuronske mreže

Konvolucijske neuronske mreže proširenje su klasičnih višeslojnih unaprijednih neuronskih mreža. Kod klasičnih višeslojnih unaprijednih neuronskih mreža s jednim ulazom, skrivenim slojem i izlazom broj neurona na ulazu će biti jednak dimenzionalnosti uzorka |x|, a broj neurona na izlazu jednak broju klasa |C|. Broj slojeva unutar skrivenog sloja kao i broj neurona u tim slojevima je proizvoljne veličine. Dobar odabir tog broja važan je za postizanje dobrih rezultata klasifikacije [3].

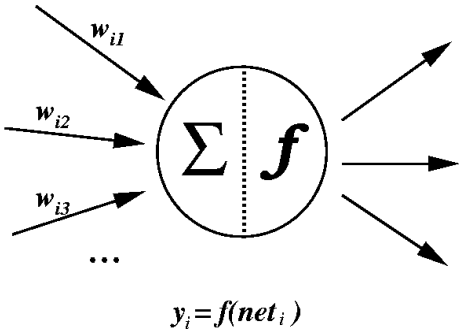
Rad jednog umjetnog neurona osmišljen je prema biološkom neuronu. Ulazi u neuron prikupljaju informacije, skriveni sloj računa sumu ulaza pomnoženu s težinama i formira pobudu koja je skalar i nazivamo je *net*.

$$net_j = \sum_{i=0}^{d} x_i w_{ji} = w_j^t x \tag{2.1.}$$

w_{ji} predstavlja težinu između ulaznog i skrivenog sloja gdje indeks i označava ulazni, a indeks j skriveni sloj. Ulazi u neuron su označeni sa x_i do x_d. U analogiji s neurobiologijom, takve težine i spojevi se zovu sinapse.

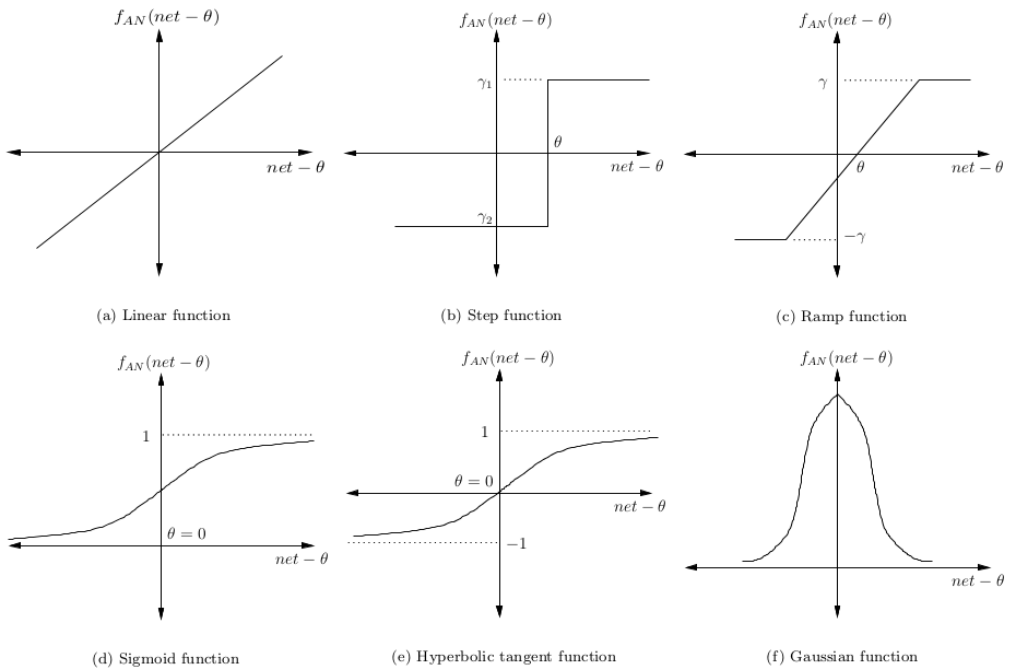
Unutar neurona također se nalazi akivacijska funkcija koja obrađuje *net* i prosljeđuje rezultat na izlaz.

$$y_j = f(net_j) \tag{2.2}$$



Slika 2.1. Prikaz rada neurona [7]

Izbor aktivacijske funkcije je slobodan, algoritam će raditi s gotovo svim funkcijama uz par zahtjeva koje funkcija i njena derivacija moraju ispuniti. Poželjno je da je funkcija nelinearna, da postoji maksimalna i minimalna vrijednost te funkcije da bi težine bile ograničene. U slučajevima kad izlaz mreže treba predstavljati vjerojatnost točne klasifikacije, kao što je u ovom radu slučaj, poželjno je da aktivacijska funkcija bude ograničena [1]. Također, funkcija bi trebala biti kontinuirana i glatka, što znači da su ona i njena derivacija definirane u rasponu njihovih argumenata. Za bilo koji klasifikacijski problem cilj je pronaći aktivacijsku funkciju uz koju ćemo dobiti najbolji rezultat. Neke od najčešćih aktivacijskih funkcija su prikazane na slici 2.2.



Slika 2.2. Primjeri aktivacijskih funkcija

Linearne aktivacijske funkcije koriste se u izlaznim slojevima kada je potreban neograničeni izlaz. Neograničen izlaz često je potreban kod regresijskih problema, no u klasifikacijskim problemima poput ovog je nepoželjan. Funkcija skoka i sigmoidna funkcija mogu se koristiti u klasifikacijskim problemima, ali u problemima gdje se koristi algoritam backpropagation možemo koristiti samo sigmoidalnu funkciju zato jer je ona derivabilna.

Skup sigmoidnih aktivacijskih funkcija zadovoljava sve već navedene kriterije, kao primjer možemo uzeti funkciju tangens hiperbolni. Ona je glatka, nelinearna, diferencijabilna, ograničena i zbog toga je idealan izbor za konvolucijsku neuronsku mrežu.

2.1. Slojevi konvolucijske neuronske mreže

Kod klasičnih višeslojnih unaprijednih neuronskih mreža vrijednosti ulaza, izlaza i težina bile su skalari, dok kod konvolucijskih neuronskih mreža govorimo o matricama. Mreža se sastoji od ulaza, u ovom slučaju možemo govoriti o monokromatskoj slici ili slici u boji. Nakon ulaza slijedi proizvoljan broj konvolucijskih slojeva i slojeva za sažimanje koji na kraju smanjuju dimenziju ulaza na 1x1 te se spajaju na perceptron koji predstavlja nekoliko potpuno povezanih slojeva. Matrice koje predstavljaju težine nazivamo jezgrama (*eng. kernel*), a izlaze nazivamo mapama značajki (*eng. feature maps*).

2.1.1. Konvolucijski sloj

Konvolucijski sloj neuronske mreže uvijek ima vezu jedan-na-više (1-N) s prethodnim slojevima i uči težine u jezgrama s kojima obavlja konvoluciju. Konvolucija se odvija između mapa s ulaza u konvolucijski sloj i jezgra. Neka je M^i mapa ulaz trenutnog sloja, a M^o mapa izlaza trenutnog sloja. Nadalje neka su M_w širina mape, M_h visina mape, K_w širina jezgre, K_h visina jezgre, S_w korak pomaka jezgre po širini prilikom konvolucije te S_h korak pomaka jezgre po visini prilikom konvolucije. Veličine mape značajki se dobiju onda pomoću izraza 2.3 i 2.4.

$$M_w^o = \frac{M_w^i - K_w}{S_w} + 1 \tag{2.3}$$

$$M_h^o = \frac{M_h^i - K_h}{S_h} + 1 \tag{2.4}$$

U ulaznoj mapi uzimamo prozor koji je jednakih dimenzija kao i jezgra. Prozor iz matrice množi se s jezgrom te se dobivene vrijednosti sumiraju s pragom. Nakon toga se izračunava vrijednost aktivacijskom funkcijom te sprema u pripadnu lokaciju na izlaznoj mapi. Prozor sa ulazne mape pomiče se dok se ne dobiju vrijednosti svih neurona u izlaznoj mapi.

2.1.2. Slojevi sažimanja

Postupak sažimanja (*eng. pooling*) smanjuje veličinu mapa značajki te povećava neosjetljivost na manje pomake značajki. Ideja postupka je da se proizvoljna veličina podataka S_w *X* S_h zamijeni jednom vrijednošću. Broj mapa značajki nakon ovog postupka ostaje isti. Nekoliko je metoda za postupak sažimanja, a glavne su opisane u nastavku.

a) Sažimanje usrednjavanjem

Jedan od prvih načina za sažimanje je sažimanje usrednjavanjem (*eng. mean-pooling*). Ideja je da se vrijednosti matrica zamijene srednjim vrijednostima podmatrica veličine S_w *X* S_h.

Na primjer ako imamo matricu $M = \begin{bmatrix} 5 & 2 & 3 & 6 \\ 4 & 1 & 7 & 4 \\ 6 & 1 & 5 & 3 \\ 7 & 2 & 2 & 6 \end{bmatrix}$, a zadane vrijednosti $S_w = S_h = 2$ dobit ćemo novu mapu oblika i vrijednosti $M' = \begin{bmatrix} 3 & 5 \\ 4 & 4 \end{bmatrix}$.

b) Sažimanje maksimalnom vrijednošću

Sažimanje maksimalnom vrijednošću (*eng. max-pooling*) pokazalo se kao bolja opcija od sažimanja usrednjavanjem. Osim što je puno brži način za računanje, ovaj način daje i bolje rezultate na razini čitavog klasifikatora. Ideja postupka je da se vrijednosti matrica zamijene maksimalnom vrijednošću podmatrica veličine S_w *X* S_h. Na primjer, ako imamo matricu

$M = \begin{bmatrix} 2 & 7 & 4 & 2 \\ 4 & 5 & 3 & 3 \\ 5 & 6 & 7 & 1 \\ 3 & 8 & 4 & 5 \end{bmatrix}$, a zadane vrijednosti $S_w = S_h = 2$ dobit ćemo novu mapu oblika i vrijednosti $M' = \begin{bmatrix} 7 & 4 \\ 8 & 7 \end{bmatrix}$.

c) Sažimanje Gaussovim usrednjavanjem

Ovaj postupak sažimanja sličan je sažimanju usrednjavanjem. Razlika između te dvije metode je u načinu izračunavanja sredine. U ovoj metodi sredina se računa pomoću Gaussove matrice G te se veće težine pridodaju elementima na sredini, a manje elementima koji su dalje u podmatrici.

2.2. Učenje konvolucijske neuronske mreže

Algoritam backpropagation jedan je od najjednostavnijih i najčešće upotrebljavanih algoritama za treniranje višeslojnih neuronskih mreža. Pokazao se kao dosta pouzdan i računski isplativ algoritam. Drugi algoritmi za specifične slučajeve mogu biti brži i imati neke druge prednosti, ali malo ih je toliko proučavano kao backpropagation. Kod konvolucijskih neuronskih mreža algoritam treba proširiti u slojevima koji su specifični za takve mreže, a to su konvolucijski sloj i sloj sažimanja.

Osnovni pristup učenju konvolucijske neuronske mreže je da krenemo s neistreniranom mrežom, na ulazni sloj dovedemo podatke za treniranje, pustimo signale da prođu kroz mrežu i odredimo krajnje rezultate na izlaznom sloju. Dobiveni podatci na izlazu uspoređuju se sa željenim rezultatima koje je mreža primila zajedno s trening podatcima. Razlika između dobivenih i željenih rezultata je greška. Greška ovisi o težinama u mreži i umanjuje se kad se izlazi iz mreže približavaju željenim izlazima. Težine u svakom pojedinom sloju se prilagođavaju u svrhu umanjivanja greške. Prvo se određuje greška u izlaznom sloju, a zatim se za prethodni sloj određuje koliko je svaki neuron utjecao na greške u idućem sloju te se računaju njihove greške. Greškom izlaza neurona smatramo sumom kvadriranih razlika dobivenog izlaza z_k i željenog izlaza t_k

$$J(w) = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2 = \frac{1}{2}\|t - z\|^2$$

(2.5)

t i z su dobiveni i željeni izlazni vektori dužine c gdje w predstavlja sve težine u mreži.

Učenje algoritma backpropagation temelji se na gradijentnom spustu, određivanju greške po pojedinim težinama koje povezuju slojeve. Težine su incijalizirane sa slučajnim vrijednostma i mijenjaju se tako da se greška smanjuje.

$$\Delta w = -\eta \frac{\partial J}{\partial w}$$

(2.6)

η predstavlja stopu učenja i samo označava relativnu veličinu promjene.

Iz formule (2.5) očito je da funkcija J nikad ne može biti negativna, što nam garantira da će proces učenja stati u nekom trenutku. Nakon izračuna grešaka trenutnog sloja, greška se propagira na prethodni sloj sumiranjem utjecaja neurona prethodnog sloja j na neurone trenutnog sloja i.

$$\frac{\partial J}{\partial y_i} = \sum_j \frac{\partial J}{\partial net_j}\frac{\partial net_j}{\partial y_i} = \sum_j w_{ij}\frac{\partial J}{\partial net_j}$$

(2.7)

net predstavlja aktivacijsku funkciju izlaza neurona.

Nakon toga se računaju parcijalne derivacije po težinama:

$$\frac{\partial J}{\partial w_{ij}} = \frac{\partial J}{\partial net_j}\frac{\partial net_j}{\partial w_{ij}}$$

(2.8)

Postoje dvije glavne varijante gradijentnog spusta u ovisnosti o trenutku ažuriranja težina: standardni gradijenti spust i stohastički gradijentni skup. Kod standardnog gradijentnog skupa ažuriranje težina se vrši nakon izračuna promjena težina nad svim uzorcima u skupu ili grupi, dok kod stohaističkog gradijentnog spust težine se ažuriraju nakon svakog uzorka. Standardni gradijenti spust je brži i stabilniji, dok je stohaistički gradijentni skup otporniji na zapinjanje u lokalnim optimumima.

3. Radni okvir Caffe

Caffe je radni okvir (engl. *framework)* specijaliziran za duboke neuronske mreže koji je kreirao Yangqing Jia tokom svog doktorskog studija na Berkeleyu. Caffe se i dalje razvija na Berkeley Vision and Learning Centreu uz pomoć izražene zajednice ljudi (engl. *community*) koja aktivno sudjeluje u razvoju i pomaganju ljudima u radu s ovim radnim okvirom. Osim već spomenute jake zajednice, kao prednosti Caffea ističu se [2]:

- **brzina** : Caffe je idealan za istraživanja i industrijski razvoj. Može obraditi preko 60 milijuna slika po danu s pomoću jedne NVIDIA K40 GPU što ga čini najbržim takvim *frameworkom*.
- **proširujući kod** : Caffe se aktivno mijenja i prati razvoj tehnologija
- **ekspresivna arhitektura**: Struktura mreže definira se kroz već napravljene konfiguracijske datoteke. Caffe pruža mogućnost treniranja mreže koristeći GPU.

3.1. Instalacija Caffea

Za potrebe ovog rada izvršen je samo postupak instalacije Caffea za rad na procesorkom računalu bez opcije za izvršavanje na grafičkom procesoru. Prije instalacije Caffea potrebno je instalirati programe i programske pakete bez kojih ne bi mogli pokrenuti niti instalirati Caffe. Upute za instalaciju preuzete su iz [2] i [4].

```
$ sudo apt-get install git
$ mkdir src; cd src/
$ git clone https://github.com/BVLC/caffe.git
$ sudo apt-get install libatlas-base-dev libprotobuf-dev \
  libleveldb-db libsnappy-dev libopencv-dev \
  libboost-all-dev libhdf5-serial-dev
$ sudo apt-get install libgflags-dev libgoogle-glog-dev  \
  liblmdb-dev protobuf-compiler
```

Prikazane su naredbe za instalaciju Caffea za rad na CPU bez sučelja za druge jezike. Caffe nudi sučelje za jezik Python, a upute za instalaciju potrebnih paketa bit će prikazane. Sučelje

za Matlab nije bilo korišteno u sklopu ovog rada i upute za instalaciju potrebnih paketa za rad u Matlabu mogu se pronaći na službenoj stranici[2].

```
$ sudo apt-get  install python-dev python-numpy  \
  python-skimage  python-protobuf  ipython  \
  ipython-notebook  ipython-qtconsole
```

Instalacija Caffea svodi se na kompajliranje izvornog koda.

```
$ cd caffe
$ cp Makefile.config.example Makefile.config
$ sed –i 's/# CPU_ONLY/ CPU_ONLY/'   Makefile.config
$ make –j4 all
$ make –j4 runtest
$ make –j4 pycaffe
$ export PYTHONPATH = $CAFFE_ROOT/python
```

3.2. Google protocol buffers

Caffe koristi Google protocol buffers za definiranje slojeva i parametara konvolucijske neuronske mreže. Google protocol buffers su metoda strukturiranja podataka. Razvijen je od strane Googlea kao interni alat za pohranjivanje i razmjenu podataka. Prvo su napravljeni protokol prevoditelji za C++, Python i Javu, a kasnije je razvijena i implementacija za Javascript, Perl, Go, Php, Scalu i Visual Basic. Cilj razvoja protocol buffersa bio je napraviti manji i brži jezik za označavanje podataka od XML-a. Protocol buffers su jednostavniji, 3-10 puta manji, 20-100 puta brži, puno jasniji i generiraju klase za pristup podatcima koje su jednostavnije za upotrebu. Protocol buffers nisu uvijek bolji izbor od XML-a. Na primjer protocol buffers nisu dobar izbor za modeliranje dokumenta baziranog na tekstu pomoću markera. Također, XML je lako čitljiv ljudima i lagan za prepravljati dok kod protocol buffera je potrebno imati definiranu strukturu podataka (*eng. message*) koja je definirana od strane programera u *.proto* datoteci [9].

Izgled jednog od slojeva je prikazan u sljedećem primjeru:

```
layer {
      name: "ime_sloja"
      type: TIP_SLOJA
      tip_sloja_params {
            source:"lokacija_podataka.txt"
            ...
      }
      Include: {
            phase: Train | Test
      }
}
```

Ime sloja je proizvoljno ime koje odabiremo. Tip slojeva i njegovi pripadajući parametri koje Caffe podržava definirani su datoteci *caffe.proto*. Tipovi slojeva bit će kasnije objašnjeni, oni koji su se koristili u ovom radu su HDF5Data, CONVOLUTION, POOLING, INNER_PRODUCT, SIGMOID, SOFTMAX. Parametri slojeva razlikuju se ovisno o sloju koji je korišten, dok su neki od najčešćih bili *source*, *batch_size*, *kernel_size*, itd. *Include* sa pripadajućim parametrom *phase* pojavljuje se u ulaznom sloju i definira koji se podatci koriste za treniranje, a koji za testiranje mreže.

3.3. Hierarchical Data Format

Hierarchical Data Format je skup formata datotetaka koji su dizajnirani za spremanje i organiziranje velike količine numeričkih podataka. Skup uključuje HDF4 i HDF5. HDF je podržan od strane mnogih programskih jezika uključujući Python. Ulaz u konvolucijsku neuronsku mrežu u ovom radu predstavljali su višedimenzionalni vektori i ovaj format se pokazao prikladnim za tu svrhu. U Caffeu postoji implementacija sloja HDF5Data koji podržava hdf5 format kao ulaz u mrežu.

3.4. Konfiguracijske datoteke

Konfiguracijske datoteke služe za definiranje mreže, njeno treniranje i poslije toga ispitivanje. U njima se definira izgled i količina slojeva koji će se koristiti, njihovi parametri i parametri koji se koriste za treniranje mreže. Tri su kategorije datoteka koje se koriste u ovom radu. Prva datoteka je *ime_train_test* koja služi za definiranje slojeva, njihovih parametara i određivanje koji podatci se koriste za treniranje, a koji za testiranje. Druga datoteka predstavlja strukturu mreže bez parametara za testranje, ulaznih i izlaznih slojeva. U zadnjoj datoteci koju nazivamo *solver* definiraju se parametri algoritma za učenje.

3.4.1. Slojevi i parametri

Slojevi koji su korišteni u radu nabrojani su u poglavlju 3.2. Slijede kratka objašnjenja pojedinih slojeva.

- **HDF5Data** – ulazni sloj koji učitava podatke u mrežu
- **CONVOLUTION** – sloj koji obavlja konvoluciju.
- **POOLING** – sloj koji obavlja sažimanje mapi značajki.
- **INNER_PRODUCT** – predstavlja potpuno povezani sloj, tretira ulaz kao običan vektor i proizvodi izlaz u obliku vektora.
- **SIGMOID** – aktivacijski sloj koji ima ulogu aktivacijske funkcije. Moguće je umjesto SIGMOID koristiti neku drugu aktivacijsku funkciju.
- **SOFTMAXWITHLOSS** – izlazni sloj koji prima dva ulaza, izlaz prethodnog sloja i traženi izlaz.

Parametri koji se pojavljuju u navedenim slojevima:

- **batch_size** – broj podataka koji se skupljaju u jednu grupu.
- **source** – put do *.txt* datoteke koja sadrži put do *.h5* datoteke u kojoj se nalaze podatci.
- **num_output** – broj neurona u sloju.
- **weight_filler** – način inicijalizacije težina na početku treniranja. Primjeri načina inicijalizacije su *constant* koji postavlja težine na zadanu vrijednost 0 i *xavier* algoritam koji određuje koju težinu dodijeliti pomoću broja ulaznih i izlaznih neurona.
- **bias_filler** – način inicijalizacije praga na početku treniranja. Kao i kod weight_fillera moguće je koristiti više načina inicijalizacije.

- **kernel_size** – određuje visinu i širinu jezgre. Moguće je zasebno postaviti visinu i širinu jezgre pomoću kernel_h i kernel_w.
- **stride** – pomak jezgre po visini i širini u mapi značajki. Moguće je zasebno postaviti visinu i širinu jezgre pomoću stride_h i stride_w.
- **pool** – parametar sloja sažimanja koji određuje metodu sažimanja. Mogu se koristiti AVE, MAX, STOHAISTIC.

Parametri algoritma za učenje koji se nalaze u datoteci *solver* su:

- **net** – put do *train_test* datoteke nad kojom se vrši učenje.
- **test_iter** – broj ponavljanja testiranja prilikom faze testiranja.
- **test_interval** – interval iteracija učenja nakon kojih se provodi testiranje.
- **base_lr** – početni koeficijent učenja.
- **momentum** – vrijednost momentuma.
- **weight_decay** – opadanje težina.
- **lr_policy** – način mijenjanja koeficijenta učenja.
- **gamma, power** – parametri za računanje novog koeficijenta učenja.
- **display** – određuje broj iteracija nakon kojih će se ispisivati stanje mreže.
- **snapshot** – snimka stanja mreže nakon određenog broja iteracija koja omogućava opciju prekida i nastavka treniranja nakon određenog trenutka.
- **snapshot_prefix** – prefiks imena snimke stanja mreže.
- **solver_mode** – određuje gdje će se izvršavati treniranje. Instalirana je CPU_ONLY opcija tako da se ovdje mora staviti CPU.

4. Ispitni skup i eksperimentalni rezultati

4.1. Koštice i endokarpi

Endokarp je unutarnji, drveni dio voća koji zatvara sjeme. Naziv koštica najčešće se upotrebljava za sjeme i endokarp zajedno. Endokarpi masline imaju više fizikalnih i vizualnih karakteristika prema kojima se mogu razlikovati. Svaki endokarp se može postaviti u dvije pozicije koje su bitne za razlikovanje, "A" i "B". Pozicija "A" predstavlja poziciju maksimalne asimetrije, a to je slučaj kada je prorez endokarpa okrenut prema promatraču. Pozicija "B" u kojoj je prema okretaču okrenut najrazvijeniji dio endokarpa dobije se okretanjem endokarpa iz pozicije "A" za 90 stupnjeva. Karakteristike koje se koriste za razlikovanje sorti maslina su težina, simetrija u pozicijama A i B, oblik endokarpa u poziciji A, površina endokarpa u poziciji B, broj i raspored brazda na površini endokarpa, itd. [6]

Za potrebe ovog rada bile su dostupne koštice dviju sorti maslina koje su zastupljene u Istri, Buža i Rosinjola. Masline i pripadajuće koštice prikazane su na slici 4.1.

Slika 4.1. Sorte maslina buža i rosinjala s pripadajućim košticama ispod njih

4.2. Vlastiti ispitni skup

Za potrebe ovog rada bilo je potrebno slikati koštice i pripremiti vlastitu bazu slika. Dostupna količina koštica bila je 120 komada svake od već spomenutih sorti. Prvotni plan ovog rada bio je predati konvolucijskoj neuronskoj mreži slike koštica, ali zbog male količine dostupnih koštica odustalo se od te ideje. Slika koštice je imala previše parametara i tako mali skup podataka nije davao zadovoljavajuće rezultate. Druga ideja, koja je provedena i čiji rezulati će biti prezentirani, je fokusiranje na jednu karakteristiku koštice. Karakteristika koja je izabrana je oblik endokarpa. Napravljeno je 240 slika od kojih se 200 koristilo za treniranje mreže, a 40 za testiranje. Iz prikupljenih slika bilo je potrebno pronaći koordinate N točaka na konturi endokarpa. Za N su se koristile razne vrijednosti, a za dobiveni rezultat u poglavlju 4.4. korišten je N = 50. Za svaku sliku dobiven je vektor dimenzija 2X50 koji je bilo potrebno spremiti u datoteku formata hdf5 kojeg podržava Caffe.

4.3. Arhitektura mreže

Tokom eksperimentiranja bile su korištene mreže s različitom kočinom slojeva i s različitim vrijednostima parametara. Arhitektura mreže koja je dala najbolji rezultat slična je arhitekturi *Lenet* koja se koristi u klasifikaciji znamenaka iz skupa MNIST (*engl. Mixed National Institute of Standards and Technology*). Zadatak klasifikacije znamenki iz skupa MNIST je najpoznatiji primjer korištenja konvolucijskih neuronskih mreža za klasifikaciju slika koji ima uspješnost klasifikacije znamenki preko 99%.

- **Ulazni slojevi** – slojevi koji primaju podatke za trenitanje i ispitivanje.
- **Prvi konvolucijski sloj** – sadrži 20 neurona i jezgre veličine 5x2.
- **Prvi sloj sažimanja** – koristi se sažimanje maksimalnom vrijednošću uz korištenje jezgri veličine 2x1.
- **Drugi konvolucijski sloj** – sadrži 50 neurona i jezgre veličine 5x2.
- **Drugi sloj sažimanja** – koristi se sažimanje maksimalnom vrijednošću uz korištenje jezgri veličine 2x2.
- **Prvi potpuno povezani sloj** – sadrži 500 neurona.
- **Sigmoidna aktivacijska funkcija** – sloj aktivacijske funkcije, za ovaj rad je izabrana sigmoidna.

- **Drugi potpuno povezani sloj** – sadrži 2 neurona koji odgovaraju broju klasa koje imamo
- **Izlazni sloj** koji sadrži 2 neurona

4.4. Rezultati

Odabrana arhitektura trenirana je na ukupno 5000 iteracija gdje je točnost mreže testirana na skupu za ispitivanje provjeravana nakon svakih 500 iteracija treniranja. Tako trenirana mreža nakon prođenih svih iteracija ispravno je klasificirana 72.5% uzoraka. Uspješnost klasifikacije mreže prikazana je na slici 4.2 pod Test net output #0.

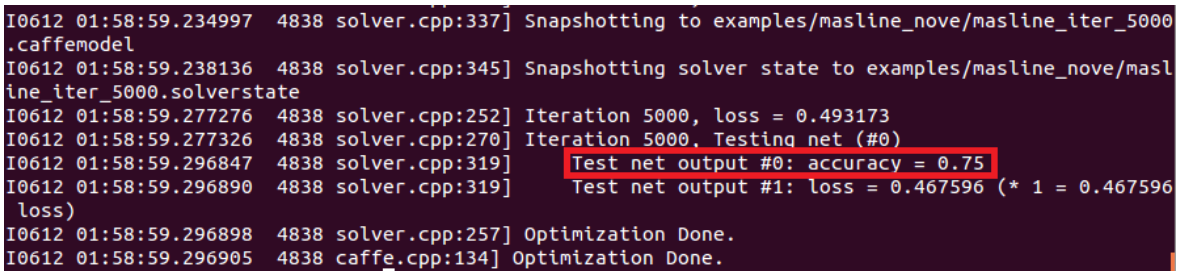
```
I0612 02:10:28.471099  4902 solver.cpp:337] Snapshotting to examples/masline_nove/masline_iter_5000
.caffemodel
I0612 02:10:28.474267  4902 solver.cpp:345] Snapshotting solver state to examples/masline_nove/masl
ine_iter_5000.solverstate
I0612 02:10:28.497370  4902 solver.cpp:252] Iteration 5000, loss = 0.497742
I0612 02:10:28.497413  4902 solver.cpp:270] Iteration 5000, Testing net (#0)
I0612 02:10:28.540419  4902 solver.cpp:319]     Test net output #0: accuracy = 0.725
I0612 02:10:28.540477  4902 solver.cpp:319]     Test net output #1: loss = 0.566178 (* 1 = 0.566178
 loss)
I0612 02:10:28.540487  4902 solver.cpp:257] Optimization Done.
I0612 02:10:28.540493  4902 caffe.cpp:134] Optimization Done.
```

Slika 4.2. Eksperimentalni rezultati

Razlozi za pojavu grešaka mogu biti razni. Mala baza slika za treniranje i ispitivanje, greške u procesu fotografiranja i nalaženja koordinata točaka na konturi. Također, nemamo uvid u način prikupljanja koštica, pojava cijepljenja voćaka može utjecati na miješanje sorti maslina i može dovesti do zabune kod prikupljanja. U skupu za testiranje je bilo 40 slika, 20 slika koštica sorte buža i 20 slika sorte rosinjola. Ispravno je klasificirano 14 od 20 slika sorte buža i 15 od 20 slika sorte rosinjola, a rezultati tih testiranja su prikazani na slikama 4.3. i 4.4. pod Test net output #0.

```
I0612 01:45:59.431502  4768 solver.cpp:337] Snapshotting to examples/masline_nove/masline_iter_5000
.caffemodel
I0612 01:45:59.434720  4768 solver.cpp:345] Snapshotting solver state to examples/masline_nove/masl
ine_iter_5000.solverstate
I0612 01:45:59.472601  4768 solver.cpp:252] Iteration 5000, loss = 0.477397
I0612 01:45:59.472659  4768 solver.cpp:270] Iteration 5000, Testing net (#0)
I0612 01:45:59.494711  4768 solver.cpp:319]     Test net output #0: accuracy = 0.7
I0612 01:45:59.494763  4768 solver.cpp:319]     Test net output #1: loss = 0.676681 (* 1 = 0.676681
 loss)
I0612 01:45:59.494772  4768 solver.cpp:257] Optimization Done.
I0612 01:45:59.494778  4768 caffe.cpp:134] Optimization Done.
```

Slika 4.3. Rezultat testiranja sorte buža



Slika 4.4. Rezultat testiranja sorte rosinjola

Zaključak

Kroz ovaj rad dana su objašnjenja umjetnih neuronskih mreža s posebnim naglaskom na konvolucijske neuronske mreže. Opisani su načini rada takvih mreža, izgled korištene mreže u ovom radu te eksperimentalni rezultati.

Glavni zadatci rada bili su ručno označavanje podataka za učenje, validaciju, testiranje te analiza podataka. Bilo je potrebno eksperimentalno vrednovati rezultate te ih na kraju opisati i prikazati. Korištena je biblioteka Caffe za koju je bilo potrebno proučiti mogućnosti na već pripremljenim primjerima koji se dobiju prilikom instalacije. Prikupljene podatke trebalo je prilagoditi u format za ulaz u konvolucijsku neuronsku mrežu kojeg podržava bibiloteka Caffe te odrediti parametre mreže koji daju najbolje rezultate za vlastiti ispitni skup.

Nakon što je uspostavljeno da je baza slika mala za predavanje cijelih slika kao ulaz u konvolucijsku neuronsku mrežu, izabrana je jedna karakteristika endokarpa koja je obrađena. Konfigurirana mreža za oblik endokarpa u poziciji "A" i odnos točaka na konturi endokarpa na predstavljenom ispitnom skupu postigla je rezultat od 72.5% ispravno klasificiranih uzoraka. Na takav rezultat, izuzev već spomenute male baze slike, mogla je utjecati kvaliteta napravljenih slika, količina i točnost odabranih točaka na konturi. Postupak određivanja parametara mreže za koje bi dobili najbolje rezultate pokazao se jako bitnim i dosta je utjecao na eksperimentalne rezultate.

Za eventualni daljnji rad u poboljšanju prepoznavanja sorti maslina preko slika koštica masline preporučava se povećanje baze slika, uzimanje u obzir više karakteristika koštice te uvođenje novih sorti. Sama ideja raspoznavanja sorti preko endokarpa voćki pokazala se vrlo zanimljivom te je moguća komercijalna iskoristivost te ideje.

Literatura

[1] DUDA, R.O.; HART, P.E.; STORK, D.G. *Pattern Classification, second edition.* SAD: John Wiley and sons, 2001.

[2] JIA, Y. *Caffe: An open source convolutional architecture for fast feature embedding*. 2013. URL http://caffe.berkeleyvision.org/

[3] VUKOTIĆ, V. *Raspoznavanje objekata dubokim neuronskim mrežama*. *Diplomski rad*. Sveučilište u Zagrebu, 2014.

[4] JURIĆ-KAVELJ, M. *Određivanje smjera gledanja konvolucijskim neuronskim mrežama. Diplomski rad*. Sveučilište u Zagrebu, 2015.

[5] HWANG, Y.; LIU, T. . *Contour Detection Using Cost-Sensitive Convolutional Neural Networks*. URL http://arxiv.org/abs/1412.6857

[6] *CHARACTERS OF THE ENDOCARP (STONE)*, URL http://elbereth.zemris.fer.hr/datasets/masline_endokarpi/COI/CHARACTERS%20OF%20THE%20ENDOCARP.pdf

[7] Wikipedia. *Artificial neural network*, 2015, URL http://en.wikipedia.org/wiki/Artificial_neural_network

[8] Wikipedia. *Convolutional neural network*, 2015, URL http://en.wikipedia.org/wiki/Convolutional_neural_network

[9] Google. *Protocol buffers*, 2015, URL https://developers.google.com/protocol-buffers/

Sažetak

Vrednovanje raspoznavanja koštica maslina konvolucijskim neuronskim mrežama

U radu su predstavljene umjetne neuronske mreže i njihovo proširenje konvolucijske neuronske mreže koje su korištene u radu. Opisan je način rada, izgled arhitekture koja je korištena i predstavljeni eksperimentalni rezultati. Cilj ovog rada bila je klasifikacija sorti maslina pomoću pripadajućih koštica. Kao fizikalno svojstvo po kojem se koštice masline razlikuju odabran je oblik masline. Pronađene su točke na konturi masline i predane kao ulaz konvolucijskoj neuronskoj mreži. Za rad s mrežom korištena je biblioteka Caffe, dane su upute za njenu instalaciju, objašnjeno korištenje te opisani njeni slojevi i parametri. Na kraju su predložene ideje za poboljšanje točnosti klasifikacijskog postopka.

Ključne riječi: umjetna neuronska mreža, konvolucijaska neuronska mreža, maslina, koštica, Caffe, višeklasna klasifikacija, Google Protocol Buffer

Summary

Experimental evaluation of recognizing olive pits by convolutional nerual networks

This paper is about artificial neural networks and convolutional neural network that was used in the experimental work. The main goal of this paper was classification of different types of olives using their pits. We can difference olive sort by looking at the shape of the pits, so the plan was to find points on the pit contour. Points became convolutional neural network input. Framework Caffe was used to work with convolutional neural networks. Caffe installation, user instructions and network architecture were presented in this paper. In the end, experimental results of classification were shown and some ideas for classification results improvement were given.

Keywords: artificial neural network, convolutional neural network, olive, pit, Caffe, multiclass classification, Google Protocol Buffer