Brojanje zaraženih stanica koristeći konvolucijsku neuronsku mrežu

Al Depope, Marta Han, Ivan Lazarić i Ivan Miošić, *PMF–MO*

Sažetak—Konstruirali smo konvolucijsku neuronsku mrežu koja algoritmom stohastičkog gradijentnog spusta uči aproksimirati broj zaraženih stanica na danoj slici.

Index Terms—Strojno učenje, detekcija broja zaraženih stanica, konvolucijska neuronska mreža, regresija.

1 Uvod

Biolozi i medicinari se neprestano susreću s jednom vrstom problema: kako brzo i precizno izvući što točniju informaciju iz slike mikroskopskih organizama? Stoga postoji potreba da se takvi manualni procesi automatizaraju, posebno uzimajuću u obzir sve veću snagu i primjenjivost računala na raznovrsne probleme. U tom kontekstu prirodno je pokušati primjeniti metode strojnog učenja.

U našem projektu usredotočili smo se na jedan konkretan potproblem, a to je efikasno određivanje broja zaraženih stanica na danoj slici. U ovom radu predstavit ćemo naš pristup rješenju, koji obećava daljnji napredak u ovom smjeru istraživanja.

1.1 Motivacija

U sklopu istraživanja na temu utjecaja alkohola na rizik od srčanog udara, tim Medicinskog fakulteta Sveučilišta u Splitu suočio se s problemom detekcije stanica u kojima se odvijaju upalni procesi na slikama dobivenim mikroskopiranjem tkiva. Na slikama su prikazane stanice čije su jezgre plave boje, a signal koji reprezentira upalni proces označen je rozom bojom. Zadatak je locirati jezgre stanica koje se preklapaju s rozim signalom i pobrojati takvu stanicu kao zaraženu (dakle, ne brojimo rozi signal na poziciji gdje nema stanice te ako ih je više na području jedne stanice, to brojimo kao samo jednu zaraženu stanicu). Pritom nam lokacija zaražene jezgre nije zapravo ključna, već je od interesa samo ukupan broj takvih stanica na danoj slici.

1.2 Podatci

Skup podataka sastoji se od 141 slike dobivene iz različitih skupina testiranih životinja, a na kojima su izbrojane zaražene stanice i označene približne lokacije njihovih centara. Slike su dimenzija 1360×1024, a jezgre stanica razlikuju se oblikom i veličinom te je na značajnom broju slika granica između njih nejasna. Zbog takvih područja, važno je pokušati izbjeći da se nakupina nekoliko, pa čak i više desetaka stanica, broji kao samo jedna stanica.

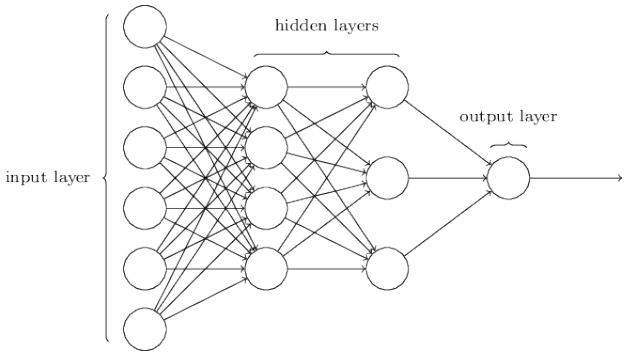
Budući da su dimenzije slika velike, a veličina dataseta relativno mala, prirodna je ideja podijeliti svaku sliku na manje dijelove koji će sadržavati nekoliko zaraženih stanica. Analizom broja upaljenih stanica na slikama zaključili smo kako je optimalno podijeliti slike na $16 \times 16 = 256$ manjih slika, dimenzija 80×80. Inicijalni dataset podijelili smo na *training set* veličine 119 i *test set* koji se sastoji od 22 slike.

2 Konvolucijske neuronske mreže

2.1 Neuronske mreže

U projektu smo koristili konvolucijske neuronske mreže. Stoga iznosimo pregled potrebnih činjenica o njima.

Kao prvo, ponovimo letimično gradivo o (klasičnim) neuronskim mrežama (NN, neural network), ne samo zbog toga što ćemo ih koristiti kao dio veće mreže, već i da utvrdimo terminologiju. NN se sastoji od nekoliko slojeva elemenata koje po uzoru na biologiju nazivamo *neuroni*. Prvi sloj mreže nazivamo *ulazni sloj*, završni *izlazni sloj*, dok se ostali slojevi (između ulaznog i izlaznog) nazivaju *skriveni* slojevi (slika 1). Neuroni su povezani s neuronima iz susjednih slojeva vezama koje mogu biti jače ili slabije. „Jakost" ove konekcije nazivamo *težina* i mjerimo je realnim brojem. Osim toga, svaki neuron ima *aktivaciju* koju mjerimo realnim brojem iz segmenta $[0, 1]$.



Slika 1. Model jednostavne neuronske mreže

Na početku, težine mreže su postavljene slučajno, bilo kojom tehnikom generiranja slučajnih brojeva (često se koristi normalna razdioba). Ulaz za mrežu se zadaje putem aktivacije neurona u ulaznom sloju mreže. Dakle, aktivacije prvog sloja su zadane, pitanje je kako sada odrediti aktivaciju neurona u ostalim slojevima. U tu svrhu koristimo **feed-forward algoritam**. Najprije, za neuron j iz l-tog sloja prvo izračunamo produkt vektor-retka w_j^l koji sadrži težine

svih veza koje ulaze u promatrani neuron, te vektor stupca a^{l-1} aktivacije neurona u prethodnom sloju. Tom iznosu dodamo konstantan član b_j^l koji nazivamo *pristranost* (eng. *bias*) neurona. Na kraju, aktivacija neurona se dobije primjenom aktivacijske sigmoidne funkcije $\sigma(z) = \dfrac{1}{1+e^{-z}}$, tako da završna formula glasi:

$$a_j^l = \sigma(w_j^l \cdot a^{l-1} + b_j^l).$$

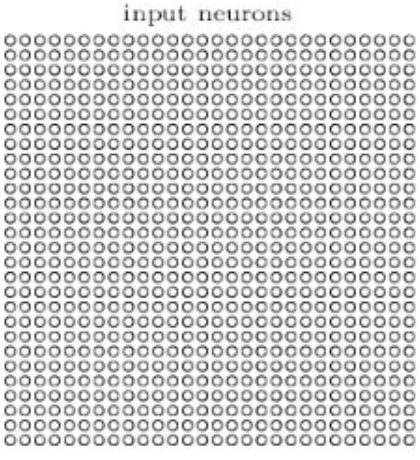
Odavde vidimo zašto se algoritam naziva feed-forward. Primjenom sigmoidne funkcije želi se postići nelinearnost modela (u obzir dolaze i druge mogućnosti za aktivacijsku funkciju, npr. $\tanh$). Kad uzmemo u obzir sve neurone u jednom sloju te pogodno zapisemo indekse, zadnji izraz postaje ugodniji za čitanje:

$$a^l = \sigma(w^l a^{l-1} + b^l).$$

Ključni dio neuronske mreže je način „učenja", tj. postavljanja težina veza. U većini slučajeva koristi se *backpropagation* metoda uporabom algoritma **stohastičkog gradijentnog spusta**. To je klasični optimizacijski algoritam, s time da se u svakom koraku spusta razmatra samo podskup dimenzija. Detalji se mogu pronaći u [1].

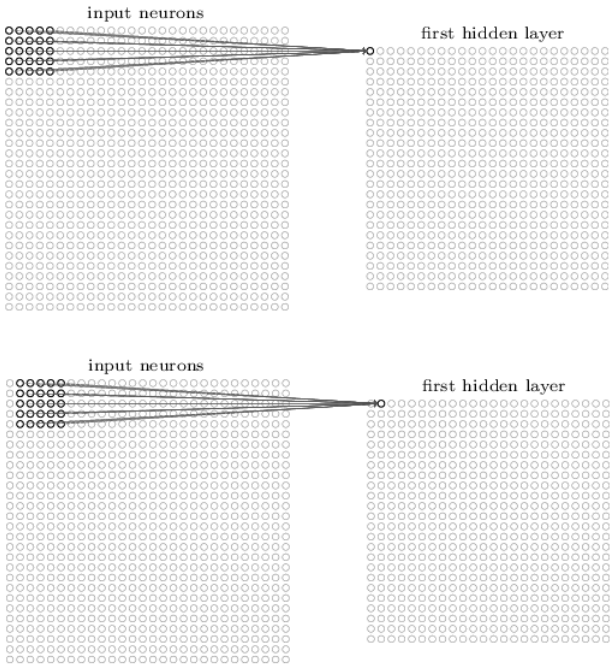
2.2 Konvolucija

Klasične neuronske mreže kronično pate od problema ogromne dimenzije prostora parametara. Uzrok tome je gusta mreža veza između neurona u slojevima. Kad malo promislimo, takva mreža čak nije ni prirodno prikladna za probleme koji se tiču procesuiranja slika, jer ne uzimaju u obzir prostorni položaj elemenata slike. Naime, RGB slike su prezentirane u računalima kao 3 matrice intenziteta pojedine boje, dok je ulazni sloj neuronske mreže običan vektor-stupac. Stoga je prirodno uzeti za ulazni sloj neurona tri 2-D matrice neurona, pri čemu je aktivacija pojedinog neurona proporcionalna intenzitetu odgovarajuće boje. Time dolazimo do prve definirajuće razlike između klasične neuronske mreže i konvolucijske neuronske mreže (CNN).



Slika 2. 2-D ulazni sloj neurona CNN-a, prilagođeno za analizu slike

Sada rješavamo i drugi navedeni problem, a to je prevelik broj veza između neurona. Ulazni sloj u 2-D obliku pogodan je za izvlačenje svojstava (eng. *feature*-a) iz malog



Slika 3. Primjena jezgre na prvi sloj CNN-a

lokalnog dijela slike. To postižemo spajanjem malog povezanog podskupa (recimo 5×5 podslike) sa samo jednim neuronom iz idućeg sloja. U procesu spajanja koristimo *konvoluciju* između aktivacija neurona i težina mreže, kroz standardnu konvolucijsku formulu (preuzeto iz [1]):

$$\sigma\left(b + \sum_{l=0}^{n-1}\sum_{m=0}^{n-1} w_{l,m} a_{j+l,k+m}\right),$$

gdje broj n predstavlja veličinu promatranog područja.

Širinu i visinu podslike te težine s kojima se konvoluira određuje *jezgra* konvolucije (eng. *kernel*).

Sada je svaka takva podslika spojena s novim neuronom (slika 3). Skup tako dobivenih neurona naziva se *mapa svojstva* (eng. *feature map*). Karakteristično svojstvo jedne mape je da su težine dijeljene, odnosno jednake za svaki spoj podslike i neurona. Naime, ideja feature map-a je ekstrakcija/detekcija određenog svojstva slike (možda boja, debljina linije, vertikalna crta, zaobljena crta, ...). Jasno je da postoji više feature map-ova koji traže razna svojstva na istoj podslici.

Spomenimo još samo dvije često korištene metode kod CNN-a, *pooling* i *dropout*. Pooling (hrv. glasanje) je jednostavan postupak redukcije dimenzionalnosti. Promatra se mala podmatrica feature map-a (2×2 do 4×4), te se odabire samo jedna vrijednost koja se šalje dalje kroz mrežu umjesto ovog skupa. Mogući izbori su max-pooling (izbor najveće vrijednosti) ili mean-pooling (izbor neke srednje vrijednosti). Ideja pooling-a je sljedeća: ako je određeno svojstvo detektirano, nije toliko bitno gdje se točno nalazi, već je bitnija njegova relativna pozicija, obzirom na druge uočene karakteristike. S druge strane, za drouput možemo reći da je gruba metoda poticanja generalizacije modela. Tijekom procesa treniranja, neki podskup veza između neurona se privremeno zanema-

ruje. Time se idejno povećava redundantnost mreže te time smanjuje ovisnost od prevelike prilagodbe mreže ulaznim podacima.

Konvolucijska mreža se dakle sastoji od nekoliko naizmjeničnih konvolucijskih slojeva za ekstrakciju feature mapova i pooling slojeva, kod kojih se kao aktivacijska funkcija koristi *ReLu* funkcija $\phi(x) = \max(0, x)$, te „repa" koji je standardna neuronska mreža.

Povijesno je zanimljivo istaknuti veliki uspjeh CNN-a na natjecanju ImageNet (Alex Krizhevsky, 2012.), od kada su konvolucijske neuronske mreže standardni alat za strojnu obradu slika.

3 PROVEDBA PROJEKTA

Implementacija koda napisana je u Pythonu, koristeći biblioteke *Keras* i *TensorFlow*. Izvorni kod može se naći u javnom GitHub repozitoriju na adresi:
https://github.com/IvanMiosic/Danica-ML.

3.1 Softverski paketi za implementaciju CNN-a

Koristili smo sljedeće Python module za implementaciju našeg koda: numpy, pandas, keras, tensorflow, PIL, te još par paketa za tehničke stvari: os, sys, glob, time.

Paketi numpy i pandas spadaju pod klasične module potrebne u bilo kakvoj obradi podataka. Obrađeni su na vježbama, stoga nećemo trošiti riječi ovdje. PIL je još jedan konvencionalan paket za procesuiranje podataka u slikovnom formatu (.jpg, .png, .tif). Koristili smo ga na vrlo uobičajen način, za otvaranje slika te njihovo rezanje i rotiranje po potrebi i želji.

Paketi tensorflow i keras nam omogućuju brzo i jednostavno eksperimentiranje sa neuronskim mrežama. Koristili smo konvolucijske slojeve jer su nam ulazni podaci slike. Naime, očekujemo da su puno bitniji odnosi bliskih naspram dalekih piksela pa konvolucijskim slojevima smanjujemo složenost mreže i vjerojatnost overfittanja. Nadalje, tensorflow i keras su izvrsno dokumentirani i često su korišteni u amaterskom i profesionalnom svijetu strojnog učenja. Službena dokumentacija paketa Keras bila je vrlo pristupačna, a može se pronaći u [3].

3.2 Obrada podataka

Ukratko ćemo opisati postupak pripreme ulaznih podataka za našu konvolucijsku mrežu, za detalje implementacije vidjeti izvorni kod i prateću dokumentaciju.

Ulazni podatci u izvornom obliku sastoje se od slika veličine 1360×1024 spremljenih u formatu `.tiff` te pridruženih tablica koje sadrže označene koordinate lokacija zaraženih stanica. Prvi korak u procesu pripreme podataka je učitavanje tih slika i konverzija u tenzorski oblik. Kako smo odlučili primjenjivati našu mrežu na podslike dimenzija 80×80, sljedeći korak je rezanje slika. Pritom je važno paziti koliko zaraženih stanica je unutar određene regije i pridružiti taj broj. Podatci su sada u pogodnom obliku za daljnju obradu putem neuronske mreže.

3.3 Arhitektura mreže

Našu mrežu dizajnirali smo po uzoru na primjer 3 iz knjige [1]. Glavna razlika, osim nužnog drukčijeg izbora hiperparametara je to što naš problem promatramo kao **regresijski**, a ne kao klasifikacijski. To povlači da je izlaz naše mreže jedan neuron koji aproksimira broj zaraženih stanica na danoj slici.

Mreža se sastoji od 3 konvolucijska sloja koja filtriraju područje 3×3, 5×5 i 6×6 te dva gusto povezana sloja (s izlaznim slojem od jednog neurona) od redom 16 i 8 neurona.

Aktivacijska funkcija koju koristimo je ReLu. Max pooling djeluje na 2×2 regiji.

3.4 Treniranje mreže

Proces treniranja mreže sastoji se od učitavanja slika opisanog u dijelu 3.2, koje se pretvaraju u ulaz neuronske mreže, odnosno trening skup. Samo treniranje odvija se kroz 10 epoha korištenjem stohastičkog gradijentnog spusta, na *mini-batch*-u veličine 32. *Learning rate* iznosi 0.001 (određen uz pomoć paketa Keras). U postupku učenja koristimo i dropout.

Nakon završetka treniranja, uspješnost mreže se testira na testnom skupu slika, usporedbom predviđenog broja sa stvarnom vrijednošću (apsolutna i relativna razlika).

4 ANALIZA REZULTATA

S obzirom da problem nismo rješavali klasifikacijski, ne možemo koristiti mjere evaluacije dobivene iz *multiclass* konfuzijske matrice. Stoga koristimo klasične evaluacijske metrike za problem linearne regresije opisane u literaturi [2]; srednju apsolutnu pogrešku (MAE), srednje kvadratnu pogrešku (MSE) te koeficijent determinacije (R^2).

4.1 MAE (Mean Absolute Error)

Srednja apsolutna pogreška predstavlja prosjek apsolutnih vrijednosti razlika između predviđenih i stvarnih vrijednosti varijable odziva (u našem slučaju, broj detektiranih stanica s upalnim procesima). Svojstvo ove mjere je da linearno penalizira odstupanje od stvarne vrijednosti, za razliku od kvadratne greške koja s većom težinom kažnjava veća odstupanja. Ako s n označimo ukupan broj varijabli, s y_i stvarnu vrijednost, a s $\hat{y}_i$ procijenjenu vrijednost i-te varijable za $i \in \{1, \ldots, n\}$, MAE računamo kao

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

4.2 MSE (Mean Square Error)

Srednje kvadratna greška predstavlja uzoračku varijancu varijable y, a računa se kao prosjek kvadrata odstupanja predviđene vrijednosti od starne vrijednosti varijable odziva. Osim što omogućava lakše određivanje gradijenta, prednost ove metrike je što naglašava velika odstupanja, tj. daje veću težinu velikim greškama u računanju ukupne pogreške. Uz iste oznake kao u prethodnom paragrafu, MSE određujemo kao

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Također, kao mjera evaluacije koristi se i korijen ove vrijednosti, tzv. $RMSE$ (Root Mean Square Error), koji predstavlja uzoračku standardnu devijaciju.

4.3 $\mathbf{R^2}$ (koeficijent determinacije)

Koeficijent determinacije je statistika koja opisuje omjer varijabilnosti zavisne varijable (varijabla odziva, y - u našem slučaju, broj zaraženih stanica) objašnjene modelom i ukupne varijabilnosti iste. Na neki način, ovu veličinu interpretiramo kao mjeru koja nam govori koliko dobro naš model predviđa stvarne vrijednosti zavisne varijable. Vrijednost koeficijenta determinacije predstavlja udio, pa očekujemo da ova vrijednost upada u interval $[0, 1]$.

R^2 računamo pomoću omjera MSE i ukupne varijance od y, gdje MSE predstavlja varijabilnost zavisne varijable koja nije objašnjena modelom. Stoga velika vrijednost kvadratnog odstupanja predviđanja od stvarne vrijednosti rezultira malom vrijednošću statistike R^2.

$$R^2 = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

4.4 Evaluacijske mjere na dobivenim rezultatima

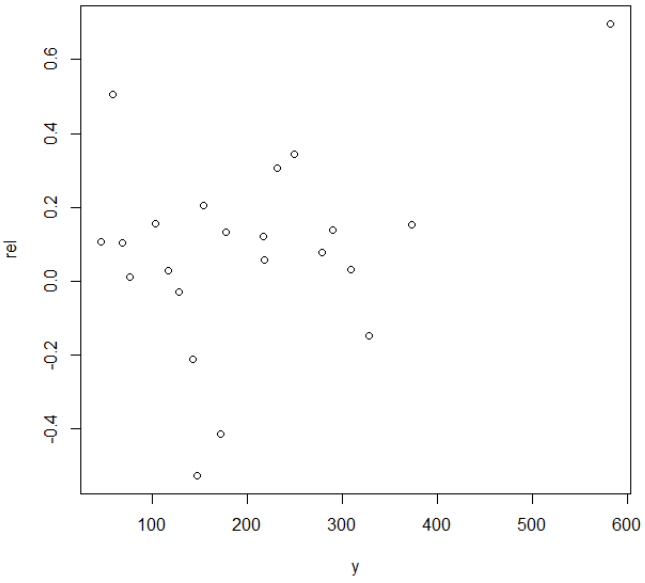
Vrijednost srednjeg apsolutnog odstupanja (broja zaraženih stanica na slici dobivenog procjenom modela od stvarnog broja) na podatcima iz *test seta* je $MAE = 49.04$, dok je srednje kvadratno odstupanje $MSE = 9090.09$, a korijen srednjeg kvadratnog odstupanja $RMSE = 95.34$. Koeficijent determinacije, koji daje interpretabilnost ovim vrijednostima iznosi $R^2 = 0.39$, što znači da model objašnjava oko 40% ukupne varijabilnosti zavisne varijable. Međutim, valja imati na umu da relativno nezadovoljavajuća vrijednost ove statistike nije argument za odbacivanje hipoteze da je model dobar jer statistička inferencija na temelju ove vrijednosti koristi dodatne hipoteze o linearnosti i normalnosti, što na ovom primjeru nije slučaj, ali donekle je robusna na odstupanje od tih pretpostavki pa možemo reći da ukazuje na odnos dviju spomenutih varijablinosti, ali ne daje pritom isključiv *da ili ne* odgovor.

Promotrimo sada relativna odstupanja procjena zavisne varijable od stvarnih vrijednosti. Prosjek relativnih odstupanja je oko 20%, ali na grafičkom prikazu istih vidimo da većina procjena odstupa za oko 10%, dok na prosjek negativno utječe nekoliko *outliera* koji većinom dolaze od slika slabije oštrine na kojima je teže dobiti dobru procjenu.

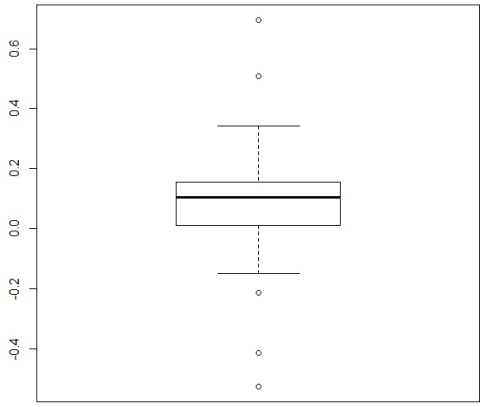
U nastavku donosimo i boxplot kako bi se bolje uočila statistička obilježja relativnih odstupanja.

5 Zaključak

Naš projekt imao je dvostruku svrhu: služiti kao projekt za polaganje kolegija, te možda važnije i plemenitije, pomoći kolegama u obavljanju posla. Također, smatrali smo da je područje (konvolucijskih) neuronskih mreža trenutno najzanimljiviji dio strojnog učenja. Stoga smo se gotovo isključivo fokusirali na ovakav pristup rješavanju. S te strane zadovoljni smo postignutim rezultatima i još više onim što smo u procesu naučili, teorijski i praktično. S druge strane, činjenica je da ćemo trebati još razvijati naše rješenje kako bi postigli efikasnost i pouzdanost potrebnu za praktičnu primjenu našeg modela.



Slika 4. Grafički prikaz relativnih odstupanja



Slika 5. Dijagram pravokutnika za relativna odstupanja

Literatura

[1] Michael Nielsen: *Neural Networks and Deep Learning*, http://neuralnetworksanddeeplearning.com/
[2] Post discussing metrics for evaluating ML models: https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4
[3] Keras documentation homepage: https://keras.io/