SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 885

RASPOZNAVANJE PROMETNIH ZNAKOVA NEURONSKIM MREŽAMA

Filip Rodik

Zagreb, srpanj 2009.

Sadržaj

Popis oznaka i kratica

OCR	optical character recognition
OO	objektno orijentirani
eng.	engleski
ANN	artificial neural network
RGB	red, green, blue (crveno, zeleno, plavo)
HSV	hue, saturation, value (nijansa, zasićenje i vrijednost boje)
treniranje	učenje neuronske mreže
s_k	ukupna suma ulaza neurona **k**
$\mathcal{F}(x)$	prijenosna funkcija neurona
$w_{j,k}$	jačina veze između neurona **j** i **k**
θ	odmak neurona (eng. offset)
λ	faktor učenja mreže
$d(\boldsymbol{x})$	očekivane vrijednosti izlaza mreže za ulazni vektor $\boldsymbol{x}$
δ	faktor promjene težine veze
y_k	izlaz neurona **k**

1. Uvod

Kako bi se računalo ponašalo „inteligentno“, mora na temelju određenih činjenica donositi razne zaključke, a da bi došlo do činjenica, potrebne su mu ulazne informacije. Čovjek te informacije prima stalno i gotovo bez svijesti o tome. Uočavanje i klasificiranje objekata u okolini, poput drugih radnji koje čovjek izvršava intuitivno, operacije su koja za računala predstavljaju bitnu prepreku. Osnovni korak u savladavanju ove prepreke jest uočavanje objekta na nekom video zapisu, odnosno slici, dakle njegovo izlučivanje od okoline. Nakon toga, potrebno je taj objekt klasificirati kako bi ga računalo povezalo s njegovim značenjem zapisanim u bazi znanja računala. Time dolazimo do problema kojim se bavi ovaj rad, a to je raspoznavanje uzoraka.

Uzorci čijim se raspoznavanjem bavi ovaj rad su prometni znakovi, a metoda kojom se slike znakova klasificiraju je korištenje umjetne neuronske mreže. Kao i kod svih drugih algoritama za prepoznavanje ili klasifikaciju objekata, pri radu s umjetnom neuronskom mrežom cilj je uz što manje resursa i u što kraćem vremenu postići najbolje moguće rezultate, odnosno što veći postotak ispravne klasifikacije.

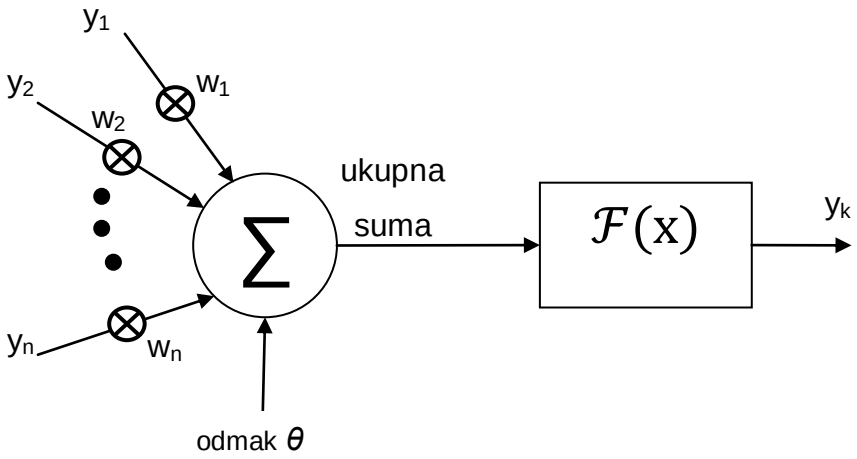
Primjena umjetnih neuronskih mreža danas je vrlo široka, a počiva na ideji ostvarivanja raznih funkcija na temelju analize ulaznih primjera. Dokazano je da se već s vrlo jednostavnim troslojnim mrežama mogu aproksimirati sve nelinearne funkcije. Valja odmah napomenuti da pri korištenju neuronskih mreža za raspoznavanje uzoraka, uspješnost klasifikacije uzoraka strogo ovisi o kvaliteti materijala za učenje mreže. Jedna od najčešćih primjena umjetnih neuronskih mreža za raspoznavanje uzoraka je OCR, optičko prepoznavanje znakova (eng. *optical character recognition*).

U ovom radu opisat će se uporaba umjetnih neuronskih mreža u svrhu raspoznavanja slika prometnih znakova. Prikazat će se osnovni modeli neuronskih mreža, njihova konstrukcija i način korištenja. Opisat će se implementacija mreže i programsko ostvarenje te na kraju analizirati dobivene rezultate i predložiti moguća poboljšanja.

2. Umjetne neuronske mreže

Naziv „umjetne neuronske mreže“ (eng. *artificial neural networks*, ANN) potječe od ideje da se realizira sustav koji bi simulirao ljudsku, pravu, neuronsku mrežu, odnosno ljudski mozak. Poput prave, umjetna mreža također se sastoji od neurona koji propagiraju informacije po mreži. Umjetni neuron je zamišljen kao jednostavna jedinica koja prima informacije, obrađuje ih, i šalje dalje drugim neuronima u mreži. Postoji mnogo različitih vrsta umjetnih neuronskih mreža. Razlikuju se prema svojoj arhitekturi, načinu na koji su neuroni u mreži povezani, načinu na koji „uče“ itd. Svaki tip mreže prikladan je za neki drugi zadatak. Kako je tema ovoga rada klasifikacija uzoraka, u konkretnom slučaju slika, odabrana je i opisana mreža najprikladnija za takav zadatak. Tip odabrane mreže je tzv. „feed-forward“ mreža, odnosno mreža u kojoj su veze između neurona jednosmjerne.

Informacije u umjetnoj neuronskoj mreži predstavljaju realni brojevi, a izvedba primanja informacije nekog neurona je doslovno sumiranje svih ulaznih informacija. Obrada informacije se svodi na uvrštavanje ukupne ulazne sume u proizvoljnu funkciju, a slanje informacije je zapravo omogućavanje drugim neuronima da na svojem ulazu vide informaciju koju im prethodnik šalje. Nužno je još napomenuti da svaka veza između dva neurona ima svoju težinu kojom se izlaz neurona koji šalje informaciju množi. Ideja je ilustrirana na Slikama 1 i 2. Slika 1 predstavlja matematički model neurona, a Slika 2 prikazuje dva povezana neurona unutar same mreže.

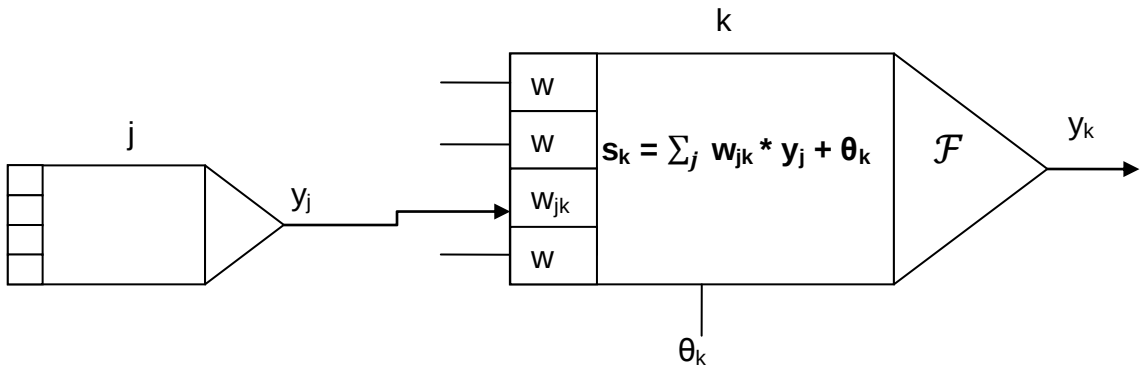


Slika 1. Matematički model neurona - sumiranje ulaza i njihova obrada. [1]

Razlog uvođenja odmaka, θ (eng. bias, offset) kao dodatka sumi ukupnih ulaza, bit će detaljnije objašnjen u sklopu „back-propagation“ algoritma.

Aktivacijska funkcija funkcija, $\mathcal{F}(\mathrm{x})$, je funkcija koja se obavlja nad ukupnom sumom ulaza. Rezultat izvođenja funkcije je upravo izlaz pojedinog neurona.

Sve vrijednosti u umjetnoj neuronskoj mreži, poželjno je ograničiti na neki poznati interval. Tipično su to intervali [-1,1] ili [0,1]. Postoji više razloga zašto je ovo popularna praksa, a jedan od njih je svakako količina informacije koja se može dobiti iz rezultata koji je dio unaprijed poznatog intervala. Pri testiranju, broj neurona može višestruko porasti ili se smanjiti kroz razne inačice mreže. Sumiranje ulaza ovdje može prouzročiti pojavu vrlo velikih brojeva kojima se ne može lako izračunati gornja granica.



Slika 2. Povezanost umjetnih neurona u mreži. [2]

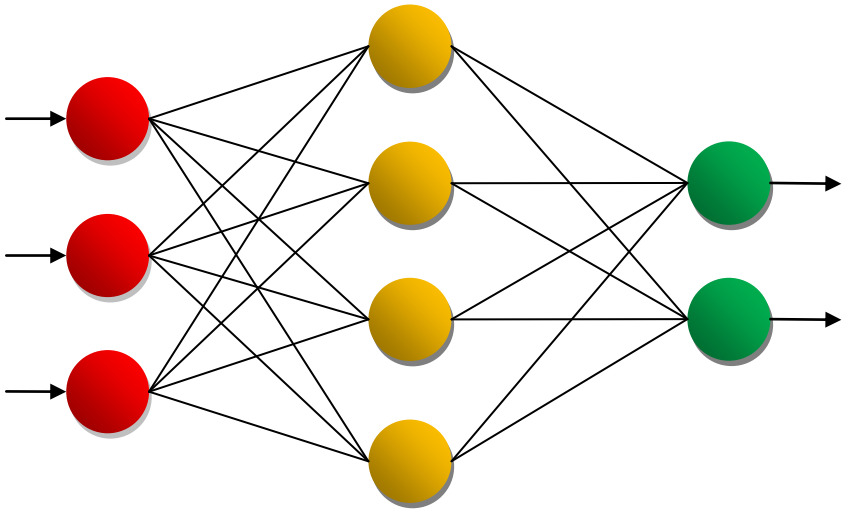
2.1. Tri vrste umjetnih neurona

Neuroni su unutar mreže raspoređeni po slojevima. Unutar jednog sloja nema veza između neurona, one postoje isključivo između neurona susjednih slojeva mreže. Neuron iz sloja N najčešće je vezan sa svim neuronima sloja (N-1) i to na način da svi neuroni iz sloja (N-1) šalju svoj izlaz na ulaz tog neurona. To vrijedi za svaki neuron u sloju N, pa iz toga slijedi da promatrani neuron svoj izlaz šalje svim neuronima u sloju (N+1). Iznimke su jedino prvi i zadnji sloj neurona u mreži, a ti slojevi sadržavaju ulazne, odnosno izlazne neurone. Svi ostali slojevi sadrže tzv. „skrivene“ neurone.

Ulazni neuron se razlikuje od ostala dva tipa neurona po tome što prima isključivo jedan podatak i odmah ga šalje na izlaz bez ikakve obrade. Podatci koji se predaju mreži direktno se upisuju u sloj ulaznih neurona pa se u literaturi često može vidjeti izraz „vektor ulaza“, pri čemu se misli na jednostavnu listu ulaznih vrijednosti za mrežu. Ulazni neuroni nemaju ni odmak, a tokom čitavog procesa „treniranja“ mreže za određeni vektor ulaza, ne mijenjaju svoje vrijednosti ulaza odnosno izlaza. Broj ulaznih neurona ovisi o broju značajki. Za svaku značajku kreira se zaseban ulazni neuron na koji će se poslati vrijednost te značajke za neki uzorak.

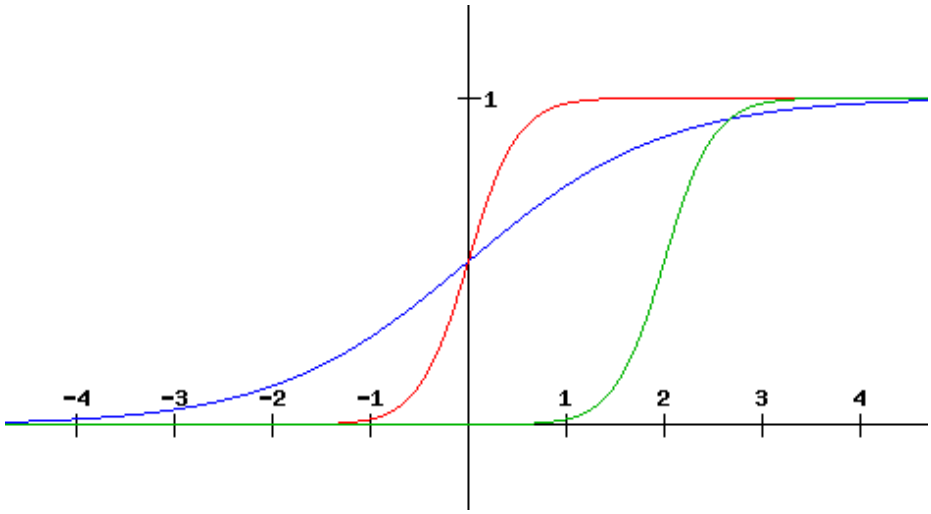
Izlazni neuron odgovara ranije opisanom neuronu sa Slike 1. Kako prije sebe u mreži ima sloj ili ulaznih ili skrivenih neurona, izlazni je neuron u mogućnosti primiti podatke s izlaza svih neurona u prethodnom sloju i obraditi ih. Kao što su ulazni neuroni bili jedna strana komunikacijskog sučelja mreže s ostatkom sustava, tako je sloj izlaznih neurona druga strana tog sučelja. Krajnje vrijednosti izlaznih neurona daju informaciju o tome kako je mreža interpretirala određeni uzorak. U slučaju da se radi raspoznavanje, traži se maksimalna vrijednost nekog izlaznog neurona u sloju. Broj izlaznih neurona ekvivalentan je broju elemenata u domeni nad kojom se radi klasifikacija, kako bi svaki element imao svoj izlaz. Nakon aktivacije čitave mreže za određeni uzorak koji se klasificira, izlazni neuron – povezan s elementom domene za kojeg je mreža izračunala najveću sličnost s promatranim uzorkom – poprimit će najveću vrijednost.

Skriveni neuroni su svi neuroni u slojevima koji nisu ulazni i izlazni. Po načinu rada u samom raspoznavanju, identični su izlaznim neuronima. Razlikuju se po funkciji jedino u postupku učenja mreže. Kada je u pitanju broj skrivenih neurona, pa i broj slojeva skrivenih neurona, ne postoje pravila koja ih definiraju. Aproksimacija linearnih funkcija, odnosno konstrukcija linearnih klasifikatora, moguća je i bez korištenja skrivenih neurona direktnim spajanjem sloja ulaza na sloj izlaza. Već jedan sloj skrivenih neurona omogućuje aproksimaciju bilo koje funkcije s konačnim brojem prekida.



Slika 3. Troslojna neuronska mreža s 3 ulazna, 4 skrivena i 2 izlazna neurona.

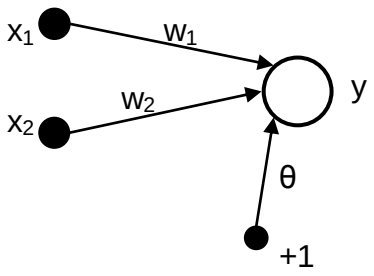
Aktivacijska (prijenosna) funkcija $\mathcal{F}(\mathrm{x})$ sumu ulaza pretvara u izlaznu vrijednost neurona. Ranije neuronske mreže, Perceptroni (Rosenblatt, 1959. [2]), koriste signum funkciju $f(x) = \begin{cases} -1, & x < 0 \\ 1, & x \geq 0 \end{cases}$, a za višeslojne mreže s implementiranim back-propagation algoritmom, obično se koristi sigma (Slika 4). Dodavanjem parametara **a** i **b**, u običnu sigmu, ona postaje $f(x) = \frac{1}{1+\mathrm{e}^{-\mathrm{a}(\mathrm{x}-\mathrm{b})}}$. Tim parametrima regulira se nagib funkcije, odnosno pomak na x-osi koordinatnog sustava. Kod velikog broja neurona u mreži dolazi do velikih suma pri zbrajanju izlaza i zato je ponekad potrebno pomaknuti nagib funkcije u desno na x-osi ili ga samo smanjiti. Ispravnom kombinacijom parametara, odnosno zadavanjem dobrih konstanti pri treniranju mreže, moguće je dobiti kvalitetnije rezultate.



Slika 4. Sigma funkcija: $f(x) = \frac{1}{1+e^{-x}}$, $f(x) = \frac{1}{1+e^{-4x}}$, $f(x) = \frac{1}{1+e^{-4(x-2)}}$.

2.3. Korak rada mreže

Nakon što su prezentirani osnovni elementi mreže i njihova ponašanja, može se opisati čitav korak rada mreže. On se sastoji od čitanja ulaza na ulaznom sloju, i propagiranja informacija sve do izlaznog sloja, pri čemu svaki neuron na tom putu izvršava zbrajanje ulaza, njihovu obradu i postavljanje svojeg izlaza. U trenutku kada svoj izlaz izračunaju izlazni neuroni, potrebno je odrediti točnost rezultata i dojaviti svakom skrivenom neuronu njegov udio u ukupnoj grešci na izlazu. Na temelju te dojave modificiraju se sve veze između neurona u mreži i u tom trenutku može započeti novi korak mreže u kojem se sve opet ponavlja. Ta povratna informacija je ključni dio rada neuronske mreže jer se upravo uz pomoć nje veze „samostalno“ reguliraju i teži se ka ispravnijem klasificiranju uzoraka. Trivijalan primjer modifikacije veza je rad Perceptrona – jednostavne umjetne neuronske mreže s dva sloja.



Slika 5. Perceptron s dva ulazna neurona i jednim izlaznim. [2]

2.4. Rad Perceptrona

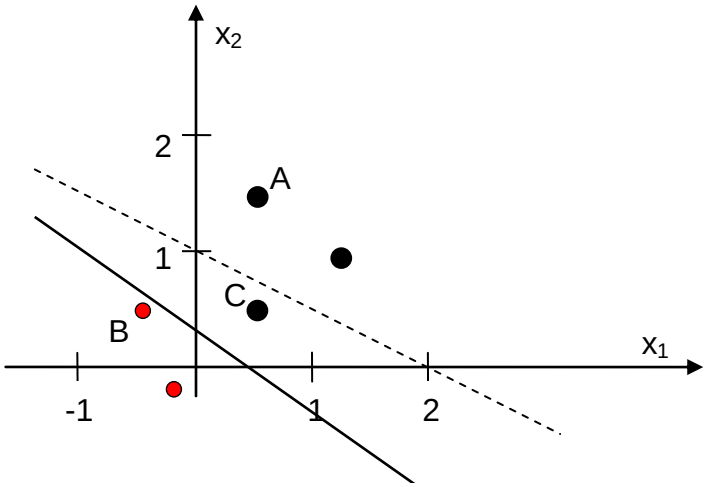
Jednostavni Perceptron sastoji se od dva ulazna neurona i jednog izlaznog (Slika 5). Za aktivacijsku funkciju ($\mathcal{F}$) koristi signum (sgn) funkciju, a izlaz je određen s:

$$\mathrm{y} = \mathcal{F}(\sum_{i=1}^{2} \mathrm{w_i} * \mathrm{x_i} + \theta).$$

Izlaz je, dakle, -1 ili 1, ovisno o predznaku ukupne sume. Na kraju svakog koraka određuju se nove vrijednosti veza u mreži. Kod ovakvog Perceptrona formule za određivanje novih vrijednosti su $w_i(t+1) = w_i(t) + \Delta w_i(t)$ i $\theta(t+1) = \theta(t) + \Delta\theta(t)$. Kako se pri učenju mreže, osim ulzanih vrijednosti, uvijek mora predati i očekivani izlaz izlaznog neurona ($d(\boldsymbol{x})$), vrijednosti $\Delta w_i(t)$ i $\Delta\theta(t)$ iznose:

$$\begin{cases} 0, & y = d(\boldsymbol{x}) \\ d(\boldsymbol{x}) * \mathrm{x_i}, & inače \end{cases}.$$

Iz ovoga se vidi da se jačine veza modificiraju samo u slučaju neispravne klasifikacije uzorka. Rad Perceptrona može se prikazati na jednostavnom primjeru odvajanja točaka u koordinatnom sustavu. Na Slici 6 potrebno je odvojiti crne od crvenih točaka jednim pravcem. Rad mreže započinje na način da se vezama u početku pridjele slučajne vrijednosti npr. $w_1 = 1$, $w_2 = 2$ i $\theta = -2$. Redom se na ulaz mreže šalju sve točke. Za crne točke, očekivani izlaz je 1, a za crvene -1. Za točku A vrijednosti x_1 i x_2 su 0.5 i 1.5. Kaže se da ulazni vektor iznosi $\boldsymbol{x} = (0.5, 1.5)$, a da je očekivani izlaz $d(\boldsymbol{x}) = 1$. Na temelju tih podataka izračuna se izlazna vrijednost y = 1. Za točku B vrijedi: $\boldsymbol{x} = (-0.5, -0.5)$ i $d(\boldsymbol{x}) = -1$. Izlaz će opet biti ispravan i iznositi y = -1. Za točku C vrijedi: $\boldsymbol{x} = (0.5, 0.5)$ i $d(\boldsymbol{x}) = 1$, no izlaz po formuli daje y = -1. Potrebno je, dakle, odrediti nove vrijednosti veza u mreži. Vrijedi da je $\Delta w_1(t) = 0.5$, $\Delta w_2(t) = 0.5$ i $\Delta\theta(t) = 1$. U idućem koraku jačine veza iznose $w_1(t+1) = 1.5$, $w_2(t+1) = 2.5$ i $\theta(t+1) = -1$. Ukoliko je došlo do modifikacije veza za neki uzorak, to je potrebno naznačiti kako bi nakon zadnjeg elementa, perceptron opet započeo s učenjem prvog, odnosno točke A. Ovaj algoritam ponavlja se sve dok se ne prođu svi elementi domene, a da ne dođe do modifikacije veza, odnosno, dok svi elementi nisu ispravno klasificirani.



Slika 6. Odvajanje točaka u 2D prostoru. [2]

- - - - - - - - - diskriminanta prije promjene težina veza

_____ diskriminanta nakon promjene težina veza

2.5. Back-propagation algoritam

Na primjeru Perceptrona prikazan je jedan od mogućih načina povratka informacije o ukupnoj grešci na izlazu pri izradi linearnog klasifikatora. U slučaju mreže s jednim ili više skrivenih slojeva, potreban je algoritam koji će rekurzivno od izlaznog sloja prema ulaznom distribuirati ukupnu grešku i na temelju nje odrediti nove veze u čitavoj mreži. Taj algoritam naziva se „back-propagation“ ili algoritam unazadne propagacije. Ideja algoritma je modificirati veze na taj način da pri idućoj aktivaciji mreže greška na svakom izlazu bude što manja [2]. Svaki izlazni neuron će na temelju usporedbe očekivanog i stvarnog izlaza odrediti ukupnu grešku, i tu informaciju poslati skrivenom sloju. Veze se opet modificiraju na isti način, po formuli $w(t+1) = w(t) + \Delta w(t)$, s time da je sada za dva neurona ***j*** i ***k***:

$$\Delta w_{jk}(t) = \lambda \delta_{\mathrm{k}} y_j ,$$

gdje je λ konstanta koja označava faktor učenja, a za skriveni neuron ***k***:

$$\delta_{\mathrm{k}} = \mathcal{F}'(s_{\mathrm{k}}) \sum_{\mathrm{j}=0}^{\mathrm{N}} \delta_{\mathrm{i}} w_{\mathrm{kj}} .$$

Ova formula kaže da je δ neurona ***k*** jednak umnošku derivacije aktivacijske funkcije i sume umnožaka δ i težine veza svih **N** neurona u idućem sloju.

Vrijednosti će se na ovaj način rekurzivno izračunavati sve do zadnjeg, izlaznog sloja koji određuje δ na način da je za izlazni neuron ***o***:

$$\delta_o = (d_o - y_o)\, \mathcal{F}'(\mathrm{s_o}).$$

Dakle, δ izlaznog neurona je umnožak derivacije aktivacijske funkcije i ukupne greške tj. razlike između očekivanog i stvarnog izlaza neurona.

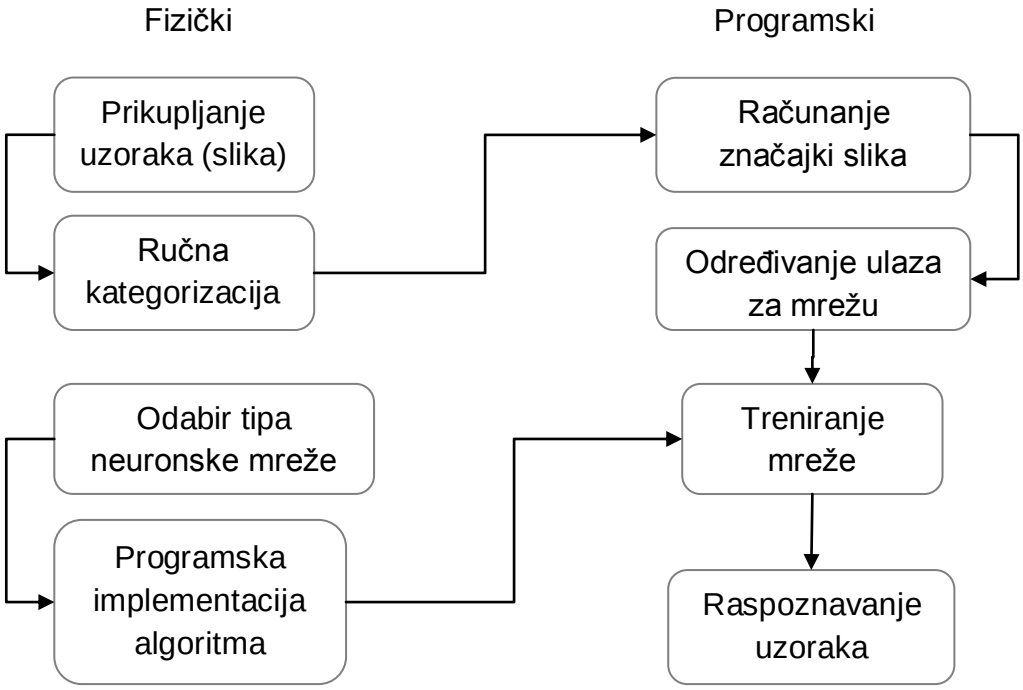
Ukoliko je za aktivacijsku funkciju korištena sigma funkcija, njenu derivaciju je vlo lako izračunati i implementirati, a ona iznosi:

$$\mathcal{F}(x) = \frac{1}{1+\mathrm{e}^{-\mathrm{x}}}\,,\ \mathcal{F}'(x) = \frac{\partial}{\partial x}\frac{1}{1+\mathrm{e}^{-\mathrm{x}}} = \frac{1}{1+\mathrm{e}^{-\mathrm{x}}}\frac{\mathrm{e}^{-\mathrm{x}}}{1+\mathrm{e}^{-\mathrm{x}}} = \mathcal{F}(x)\big(1-\mathcal{F}(x)\big).$$

Zbog jednostavnosti, back-propagation algoritam se izvodi nakon svakog prikazanog uzorka pri učenju mreže. Ovdje do izražaja dolazi potreba uvođenja odmaka, θ. Ukoliko bi se nekim slučajem dogodilo da ukupna suma poprimi vrijednost 0, algoritam nikada ne bi izašao iz te nule. Stoga je potrebno osigurati dodatni odmak za sumu.

3. Konstrukcija i korištenje umjetne neuronske mreže

Sama konstrukcija mreže sastoji se od odabira broja slojeva, broja neurona u određenom sloju mreže i odabira raznih parametara pri samoj izradi. Konstruiranu mrežu potrebno je naučiti kako da raspoznaje znakove. Taj proces zove se „treniranje mreže“. Mreža se trenira na temelju velikog broja uzoraka koje je prethodno potrebno pripremiti. Kako je tema ovog rada raspoznavanje prometnih znakova, opisani postupak pripreme uzoraka odnosit će se isključivo na slike znakova.



Slika 7. Dijagram poslova pri izradi neuronske mreže za raspoznavanje uzoraka.

3.1. Priprema uzoraka – obrada slike i određivanje ulaza

Za razliku od nekih sustava za raspoznavanje uzoraka koji ne posjeduju konkretne statističke informacije o uzorcima, nego uspoređuju objekte direktno na razini slike, umjetne neuronske mreže raspolažu podatcima o samim uzorcima i pomoću njih donose odluku, odnosno klasificiraju uzorak koji se ispituje. Te informacije o uzorcima nazivaju se značajke (eng. *features*), a one mogu biti bilo kakve dosljedne informacije o samom uzorku. Kada su u pitanju slike, značajka

može biti, primjerice, ukupna količina potpuno crnih točaka (piksela) na određenom dijelu slike. Bitna stvar za značajke je da se one mogu izračunati za svaki uzorak iz domene nad kojom se radi klasifikacija. Dakle, prije korištenja same mreže potrebno je i prethodno analizirati uzorke.

Svrha analize slike i računanja značajki je određivanje ulaza za umjetnu neuronsku mrežu. Kvalitetan odabir značajki može višestruko popraviti krajnje rezultate. Kako bi se one kvalitetno odabrale, potrebno je analizirati sve kategorije koje se klasificiraju i pronaći elemente po kojima se one maksimalno razlikuju. Ukoliko su u pitanju boje, teba pronaći područja slike na kojima se boje često razlikuju. U slučaju crno-bijelih slika, potrebno je odrediti lokacije na slici gdje dolazi do velikih razlika u intenzitetu piksela. Bitno je imati na umu da pretjerivanje u broju značajki i detaljima neće nužno rezultirati boljim raspoznavanjem. Velik broj značajki će također uzrokovati i velik broj ulaza mreže što u značajno usporava treniranje mreže.

Osim što ih je potrebno analizirati, slike je nužno razdvojiti u skupine za učenje i testiranje. Ne postoji unaprijed određeni broj uzoraka koji bi se mogao nazvati optimalnim. Taj broj ovisi o kvaliteti uzoraka, broju kategorija i mnogim drugim faktorima. Ono što je sigurno kada je u pitanju raspoznavanje uzoraka, jest to da je pri samom učenju mreže potrebno koristiti podjednak broj uzoraka iz svake kategorije, kako bi mogućnost klasifikacije za svaku kategoriju bila maksimalna.

3.2. Treniranje umjetne neuronske mreže

Umjetnoj neuronskoj mreži, da bi uspješno klasificirala uzorke treba prethodno prezentirati velik broj različitih primjeraka svake kategorije. Budući da se uzorci prethodno moraju i obraditi, proces treniranja može biti vremenski zahtjevan, što ovisi kako i o resursima, tako i o opsegu skupa za treniranje. Cilj treniranja je uzastopnim aktivacijama mreže za poznate uzorke ostvariti optimalne jačine veza u čitavoj mreži. Ukoliko se koristi back-propagation algoritam, potrebno je slučajnim redoslijedom zadavati uzorke mreži. Razlog tomu je što se inače neće postići ispravni rezultati pri klasifikaciji, jer će se uzorci koji su prvi korišteni pri učenju puno teže prepoznavati od uzoraka koji su zadnji prošli kroz mrežu.

Pri treniranju dolazi do izražaja korištenje raznih parametara u mreži. Već najosnovnije mreže imaju značajan broj parametara kojima je potrebno odrediti optimalne vrijednosti. Te vrijednosti nije moguće izračunati, potrebno ih je odrediti testiranjem mreže s različitim vrijednostima. Brojem parametara raste i dimenzija problema odnosno količina testiranja mreže. Trajanje treniranja mreže također predstavlja problem jer može potrajati i do više sati.

Postoji nekoliko različitih metoda treniranja kada je u pitanju završetak postupka treniranja. Najjednostavnija za implementaciju je metoda s unaprijed definiranim brojem epoha. Jednom epohom smatra se prolazak kroz sve uzorke za učenje. Osim broja epoha, potrebno je odrediti i broj koraka mreže za svaki uzorak za učenje. Ukupan broj aktivacija mreže određen je umnoškom ta dva broja. Važno je napomenuti da se, zbog načina rada back-propagation algoritma ulazni skup uzoraka mora permutirati svaki put kako bi se postigli kvalitetniji rezultati i to vrijedi za sve metode učenja mreže. Neke druge metode mogu koristiti drugi skup ulaznih uzoraka nakon svakih N epoha za dobivanje povratne informacije o rezultatima ispravnosti rada. U tom slučaju računa se kvadratna greška za neki uzorak:

$$E = \sum_{o=1}^{N_o} (d_o - y_o)^2.$$

Gdje je d_o očekivani, a y_o stvarni izlaz određenog izlaznog neurona od njih ukupno N_o. U slučaju da je greška nakon zadnjih N epoha manja nego prije, promjene se prihvaćaju i nastavlja se s novih N epoha. Treniranje se zaustavlja kada više nije moguće smanjiti kvadratnu grešku.

Važno je naglasiti da, kod bilo kojeg treniranja, dulje treniranje ne povlači kvalitetnije rezultate. Kao i kod ostalih parametara mreže, potrebno je pronaći optimalnu količinu treniranja mreže jer ukupna greška nakon određene količine treniranja može samo rasti.

3.3. Raspoznavanje umjetnom neuronskom mrežom

Jednom naučena mreža, može se koristiti za raspoznavanje uzoraka. Samo raspoznavanje se ne razlikuje puno od procesa učenja mreže. Jedina promjena je u tome što se nakon raspoznavanja ne modificiraju jačine veza, nego se samo gleda izlazni sloj mreže. Nakon grafičke analize uzorka, on se postavlja na ulaz

mreže i zatim se mreža aktivira. Odluka o kategoriji uzorka donosi se na temelju najvećeg izlaza nekog neurona u izlaznom sloju. Kako je svaka kategorija predstavljena točno jedinim neuronom, a veza između kategorije i neurona je unaprijed poznata, lako je zaključiti na temelju izlaza o kojoj se kategoriji radi. Moguće je uvesti i dodatni prag za svaki izlazni neuron kojim se određuje je li izlaz tog neurona dovoljno velik da se donese zaključak o pripadnosti trenutno promatranog uzorka toj kategoriji. Ukoliko više neurona prelazi prag, opet je moguće tražiti maksimalnu vrijednost među njima.

4. Implementacija

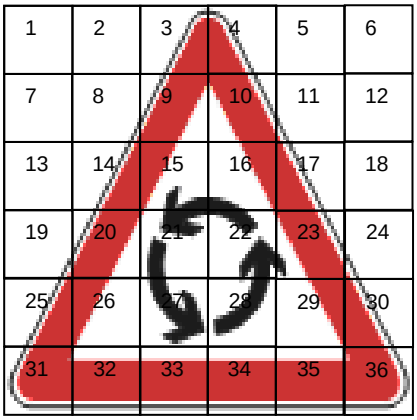
U svrhu usporedbe rezultata, napravljena su dva različita modela mreža. Jedan radi sa slikama u boji i raspoznaje uzorke iz kategorija koje se međusobno razlikuju po obliku i boji, a drugi radi s crno-bijelim slikama i raspoznaje različite trokutaste znakove. Osim korištenja boje, osnovna razlika između ova dva modela jest odabir značajki. Dok je u prvom, koji koristi boje, potrebno ispitati vrijednosti na gotovo čitavoj ulaznoj slici, u drugom je područje po kojem se kategorije razlikuju poprilično smanjeno. Na Slikama 7 i 8 moguće je vidjeti da se bitne informacije, na temelju kojih se može odrediti pripadnost uzorka nekoj kategoriji, nalaze samo u središtu znaka, ujedno i čitave ulazne slike. U nastavku, opisat će se oba modela.

4.1. Korištene značajke

Korišteni skup slika trokutastih znakova sastojao se od cca. 400 slika znakova podijeljenih u 9 kategorija.

Slika 8. Devet kategorija znakova među kojima se radi klasifikacija.

Slike znakova su (u oba slučaja) veličine 24x24 piksela i, za potrebe određivanja značajki, podijeljene u manja kvadratna područja. Napravljeno je više verzija mreža pa tako i više različitih podjela, a u ovom poglavlju opisat će se slika podijeljena u 36 kvadratića veličine 4x4 piksela.



Slika 9. Podjela slike u 36 manjih područja.

Ulazne slike su u boji, pa je potrebno svakom pikselu izračunati vrijednost na sivoj skali (eng. *grayscale*). Za to je korišten najjednostavniji algoritam konverzije iz RGB formata u sivu skalu. **Vrijednost** svakog piksela u sivoj skali dobiva se:

$$sivaV = \frac{crvenaV + zelenaV + plavaV}{3}.$$

Rezultat izvođenja je broj od 0 do 255. U nuli se radi o potpuno crnom pikselu, a maksimalna vrijednost je potpuno bijeli piksel. Skaliranje se vrši na interval [0,1]. Nakon konverzije, potrebno je izračunati prosječnu vrijednost vrijednosti svih piksela nekog područja. Taj iznos određuje jednu značajku i računa se za svih 36 područja slike. Na ulaz mreže moguće je zatim poslati te vrijednosti ili razne njihove kombinacije. Kod trokutastih znakova, moguće je pretpostaviti da će se ključne informacije uvijek nalaziti u područjima 15, 16, 20 do 23 i 26 do 29 (slika 8).

U drugom slučaju radi se s bojama i različitim kategorijama prometnih znakova. Neki od korištenih prikazani su na Slici 9. Osim po bojama, prometni znakovi se razlikuju i po svojem obliku, pa je potrebno odrediti vrijednosti značajki na čitavoj ulaznoj slici.

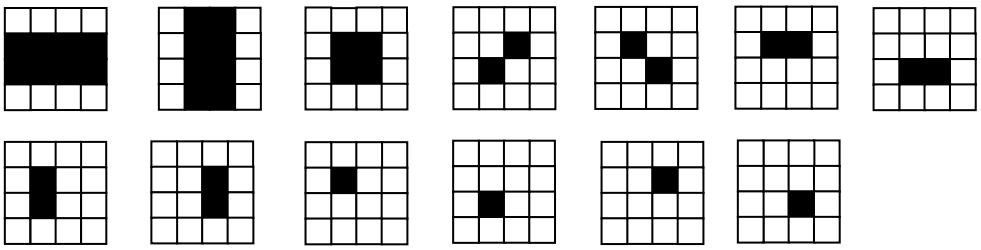
Slika 10. Kategorije znakova u RGB modelu.

Sustav radi u RGB formatu, pa se za svaku boju osnovna značajka definira kao prosječan iznos pojavljivanja boje na skali od 0 do 255 na određenom području. Za potrebe neuronske mreže, ovi iznosi su, prije postavljanja na ulaz mreže skalirani na interval [0,1].

```
for (int i = x1; i <= x2; i++)
{
    for (int j = y1; j <= y2; j++)
    {
        Color piksel = this.slika.GetPixel(i, j);
        iznosR += piksel.R;
        iznosG += piksel.G;
        iznosB += piksel.B;
    }
}
int brojPiksela = (x2 - x1 + 1) * (y2 - y1 + 1);
iznosR /= brojPiksela;
iznosG /= brojPiksela;
iznosB /= brojPiksela;
```

Slika 11. Određivanje intenziteta pojedine boje na isječku slike.

Nakon što su izračunate vrijednosti boja na slici, moguće je kreirati ulazne značajke, odnosno, same ulaze za mrežu. Opet se kombiniraju različita područja slike samo što u ovom slučaju nije moguće odrediti desetak od 36 područja koja sadrže dovoljno informacija za klasifikaciju uzorka. U odluku je potrebno uključiti gotovo svih 36 polja, zasebno ili u raznim kombinacijama. Moguće je izostaviti neka rubna područja, pogotovo ako se radi o kutevima slike. Neke od korištenih značajki prikazane su na smanjenom, 4x4 modelu na Slici 12.



Slika 12. Neke od korištenih značajki RGB modela za 4x4 podjelu slike.

4.2. Specifikacije izrađenih mreža

Iako nema potrebe za više od jednog skrivenog sloja, program je napravljen tako da omogućava izradu mreže s bilo kojim brojem slojeva. Broj ulaznih neurona ovisi o broju značajki i razlikuje se za neke modele napravljenih mreža, a broj izlaznih neurona ovisi o broju odabranih kategorija pri svakom učenju mreže.

Za prijenosnu funkciju u svim neuronima odabrana je sigma funkcija i to oblika:

$$\mathrm{f(x)} = \frac{1}{1 + \mathrm{e}^{-\mathrm{a(x+b)}}}.$$

Dodatnim uvođenjem konstanti **a** i **b** pokušalo se skratiti vrijeme učenja. Konkretno, kod velikog broja ulaza mreže događa se to da u prvih nekoliko stotina, pa čak i tisuća iteracija, sume u neuronima poprimaju velike iznose. Kada se te vrijednosti predaju prijenosnoj funkciji dolazi do toga da većina neurona na izlazu ima broj koji teži u 1. Parametrom **a** moguće je smanjiti nagib izlazne funkcije i time smanjiti same izlaze, a parametrom **b** moguće je pomaknuti nagib prijenosne funkcije u pozitivnom smjeru x osi. Pomakom po x osi moguće je za velike sume u ranijoj fazi učenja vraćati manje iznose, odnosno postavljati izlaze neurona koji neće težiti u maksimum. Derivacija ove funkcije koristi se pri izvođenju back-propagation algoritma i ona iznosi:

$$\mathcal{F}'(x) = \frac{ae^{a(x+b)}}{e^{2a(x+b)} + 2e^{a(x+b)} + 1}.$$

Izračun vrijednosti prijenosne funkcije i njene derivacije ostavljen je na raspolaganje svakom neuronu zasebno, pa je moguće izvesti model u kojem bi se parametar **b** dinamički računao nakon svakih nekoliko ciklusa izvođenja. Krajnji

rezultat bi opet bila sigma funkcija u ishodištu i nešto kraće vrijeme učenja mreže. Ova hipoteza nije u dovoljnoj mjeri testirana da bi donio zaključak o njenoj ispravnosti. Kasnijim testiranjem prikazano je da je jednostavnije i efikasnije rješenje ovog problema dodjeljivanje početnih težina veza iz intervala [-1,1], a ne [0,1].

Odmak pojedinog neurona tretira se kao vanjski neuron sa stalnim izlazom vrijednosti 1. Veza između ta dva neurona je virtualna, odnosno realizirana je na način da se izravno mijenja doprinos odmaka u ukupnoj sumi ulaza neurona.

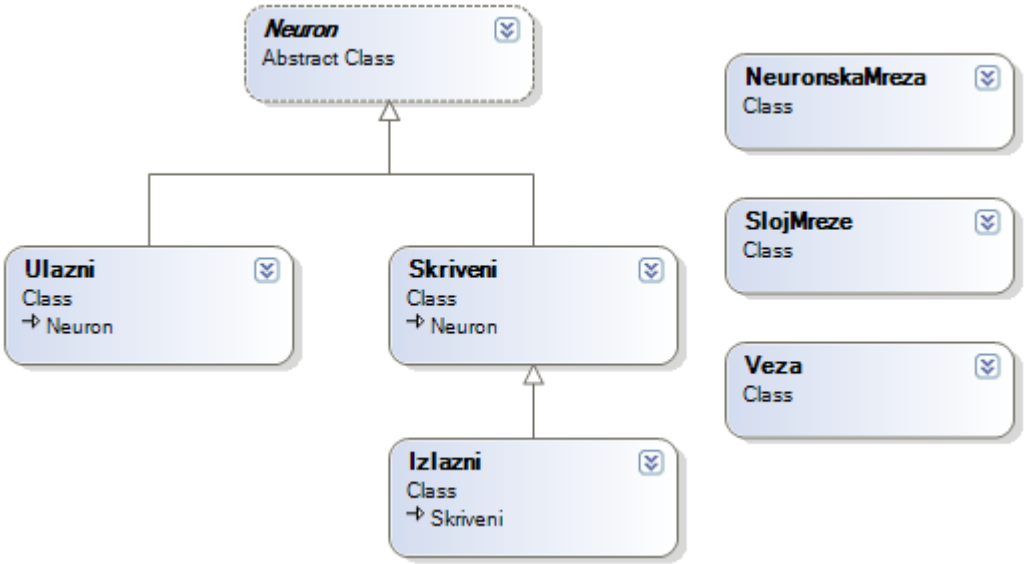
Izvođenje back-propagation algoritma nastupa nakon svake aktivacije mreže, a ne nakon određenog broja aktivacija. Iako je time broj izvođenja naredbi znatno uvećan, jer se modifikacije veza računaju za svaki korak, a ne kumulativno, pretpostavlja se da na taj način veze brže konvergiraju prema optimalnim vrijednostima što u konačnici znači da će se mreža brže naučiti. To znači da ovakav način modifikacije veza ne usporava značajno izvođenje, a jednostavniji je za implementaciju.

Nakon što se svim uzorcima za testiranje izračunaju vrijednosti za ulaze i očekivani izlaz, takve vrijednosti se spremaju u listu i slučajnim odabirom se u svakoj epohi biraju uzorci dok se ne isprazni lista, nakon čega kreće nova epoha.

5. Implementacija u programskom jeziku C#

5.1. OO model neuronske mreže

Osnovna jedinica neuronske mreže koju je potrebno programski ostvariti je, naravno, sami **neuron**. Korištena je apstraktna klasa `Neuron` iz koje su izvedene ostale podvrste neurona (ulazni, skriveni i izlazni). Ona sadrži osnovna svojstva, zajednička svim tipovima neurona. Zbog vrlo sličnog ponašanja skrivenih i izlaznih neurona, klasa `Izlazni` koja predstavlja izlazni neuron izvedena je iz klase `Skriveni`. Izlazni neuroni još proširuju klasu Skriveni nekim svojim funkcijama poput uspoređivanja dobivenog izlaza s očekivanim. Ovaj tip nasljeđivanja nije u skladu s pravilima objektno orijentirane paradigme i ispravnije bi bilo kreirati novu apstraktnu klasu npr. Mrezni koja nasljeđuje Neuron i iz koje bi se pojedinačno izvele klase Izlazni i Skriveni. Klasa `Ulazni` nasljeđuje i neznatno proširuje klasu Neuron. Najbitniji podatak u svakom neuronu koji omogućuje funkcioniranje mreže je izlaz neurona. Ostvaren je kao `double` vrijednost na intervalu [0,1].



Slika 13. Dijagram klasa neuronske mreže.

Veze između neurona su implementirane na način da svaki (skriveni ili izlazni) neuron iz sloja N, sadrži listu veza (`List<Veza>`) koju puni vezama na način da za svaki neuron iz sloja N-1 kreira vezu u koju, kao drugi kraj veze sprema trenutni neuron iz sloja N-1. Sama klasa Veza se koristi na način da je jedan kraj veze objekt (neuron) koji ju je kreirao, a za drugi kraj se taj isti objekt mora pobrinuti da

ga ispravno dodijeli. Veza sadrži i svoju težinu koja se u svakom ciklusu treniranja mreže modificira, a također je realizirana kao `double` vrijednost na intervalu [0,1].

Višestruki konstruktori u gotovo svim klasama posljedica su toga što se mreža može konstruirati na dva načina – treniranjem i učitavanjem postojeće mreže iz tekstualne datoteke. Konkretna razlika u samoj izgradnji je što se, pri izradi entiteta u mreži (neurona i veza), na različit način moraju odrediti vrijednosti odmaka neurona, veza između neurona, sve konstante u mreži itd. Pri kreiranju mreže treniranjem, radi se potpuno nova mreža i vrijednosti poput odmaka pojedinog neurona i veze između dva neurona se određuju slučajnim odabirom (ugrađena klasa `Random`). Pri učitavanju postojeće mreže iz datoteke, sve vrijednosti se moraju predati konstruktorima pri samom pozivu.

Klasa `SlojMreze` nije ništa više od obične liste neurona, preimenovana u SlojMreze ne bi li intuitivnije bilo jasno o čemu se radi pri oblikovanju i analizi programskog kôda.

Sve ranije navedene klase objedinjene su u glavnoj klasi, `NeuronskaMreza`, koja upravlja svim elementima mreže. U njoj su definirane sve funkcije koje predstavljaju neki bitan korak u radu umjetne neuronske mreže, kao i način rada pojedine mreže, odnosno, radi li s crno-bijelim slikama, u koliko fragmenata dijeli sliku itd.

Preostale klase koriste se za obradu slika. Za te potrebe kreiran je iterator za direktorij sa slikama koji olakšava kategorizaciju uzoraka za učenje i rad s njima. Objekt klase `Znacajke` sadrži informacije o slici u obliku polja prosječnih vrijednosti intenziteta piksela na određenim područjima slike. Ovdje je i definirana podjela koja se radi, odnosno, na koliko kvadratića se dijeli originalna slika. Za svaku sliku se zatim kreira objekt klase `TestUzorak` koji od osnovnih značajki kreira ulazne vrijednosti za mrežu raznim kombinacijama osnovnih značajki.

Spremanje kreiranih mreža ostvareno je ispisivanjem parametara postojeće mreže u tekstualnu datoteku. Format zapisa je pojednostavljen kako bi se olakšao postupak parsiranja datoteka pri učitavanju mreže.

5.2. Analiza programskog rješenja

Tokom duljeg rada na programskom rješenju, višestruko su narušeni neki osnovni principi objektno orijentirane paradigme. Razlog tomu je što se, pri izradi, naglasak stavljao na ispravnost rada mreže više nego na ispravnost modela rješenja. Višesatna treniranja mreže dovode u pitanje ispravnost odabira programskog jezika C# za izradu umjetne neuronske mreže. Druga mana jezika C# u ovom slučaju je nedostatak postojećih biblioteka za rad sa slikama. Zbog navedenih razloga, odabir programskog jezika C++ bi svakako bio bolji.

Prednost jezika C# bi svakako bili ugrađeni tipovi poput `List<>`, zbog činjenice što je mrežu lako ostvariti kao listu lista. Zbog čestog rada s raznim listama (neurona i vrijednosti), do izražaja dolazi i ugrađeni iterator `foreach` koji smanjuje broj indeksa u kôdu i olakšava čitljivost.

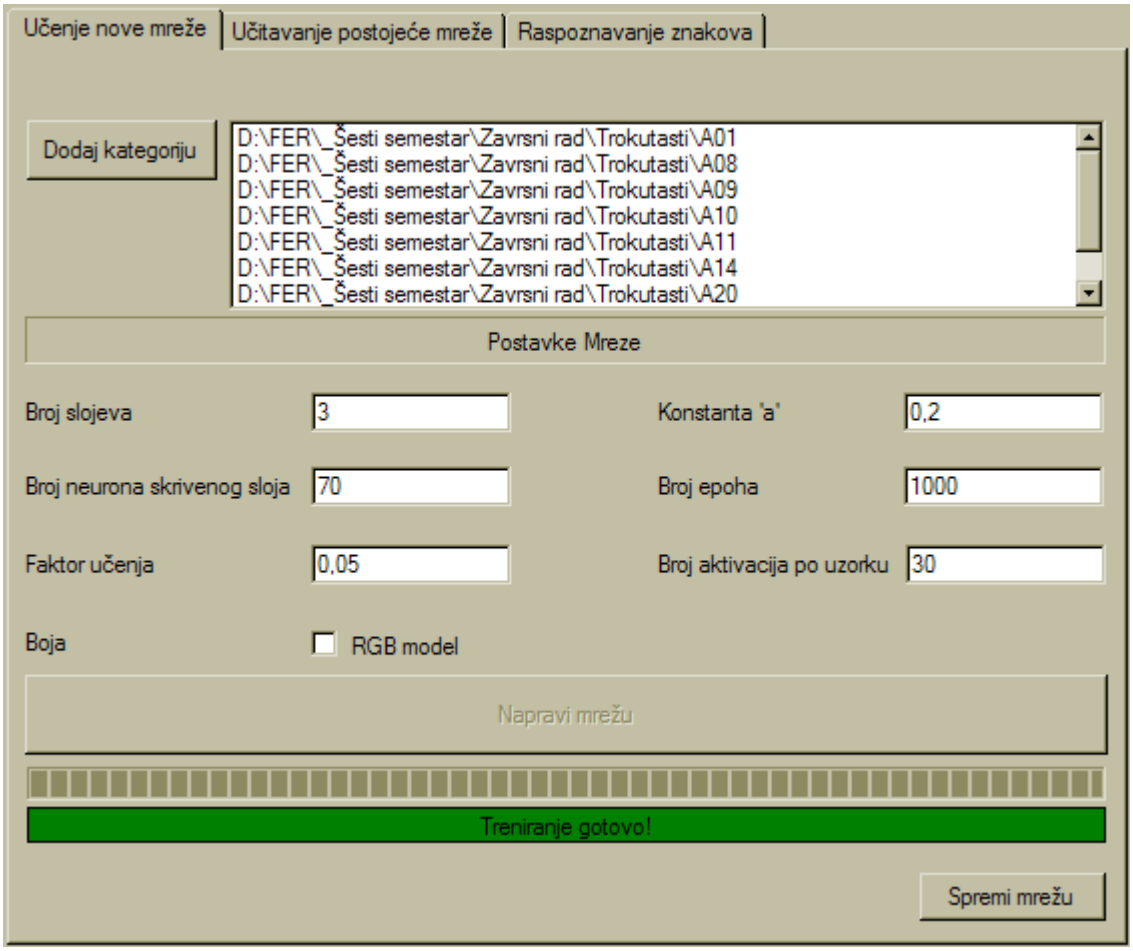
Poduzete su i neke mjere kako bi se, barem malo, smanjilo trajanje učenja mreže. Jedna od njih je, naravno, optimizacija programskog kôda koji se često izvodi. Takvi odsječci koda su npr. propagacija informacija po mreži i modificiranje veza. Pri rekurzivnim akcijama u mreži, neki se iznosi višestruko koriste na raznim mjestima. Pri tome se vodila pažnja da se broj izračuna takvih vrijednosti smanji koliko je god to moguće.

5.3. Grafičko sučelje

U sklopu projekta, napravljeno je i jednostavno grafičko sučelje za izradu umjetnih neuronskih mreža i rad s njima. Sučelje ima tri osnovne funkcije – učenje nove neuronske mreže, učitavanje postojeće neuronske mreže iz tekstualne datoteke i raspoznavanje znakova neuronskom mrežom.

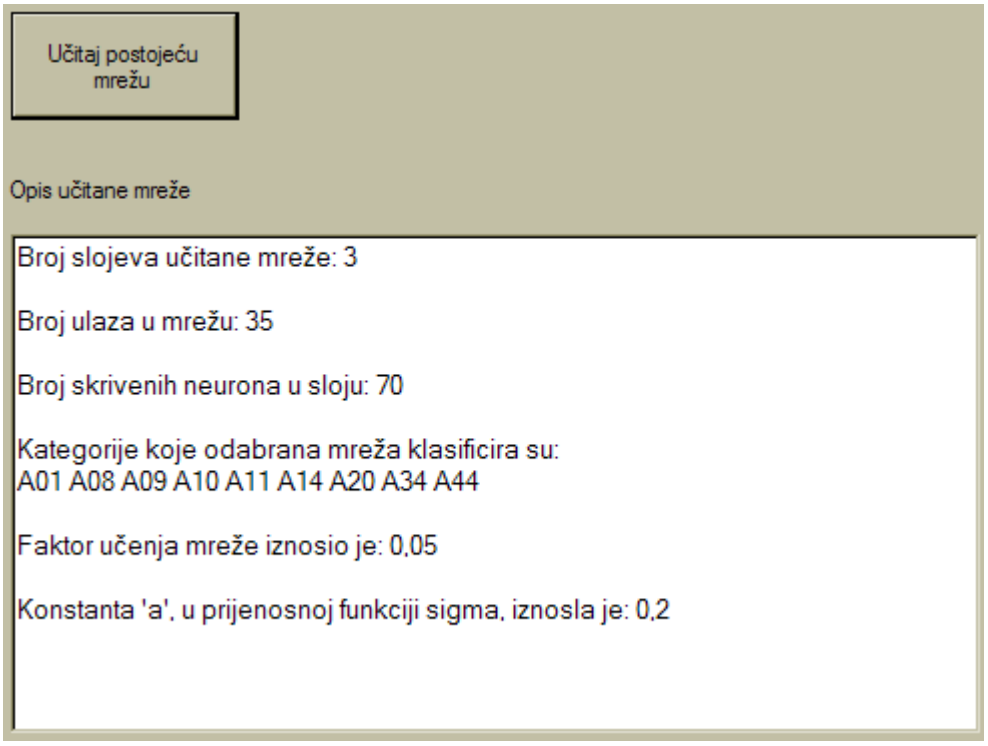
Učenje mreže (Slika 14) započinje dodavanjem kategorija znakova koje se žele raspoznavati. Prethodno je potrebno staviti određen broj uzoraka svake kategorije u zaseban direktorij te ga nazvati imenom kategorije prometnog znaka kojeg predstavlja. Nakon što su dodane kategorije, upisuju se postavke mreže. Broj slojeva, broj neurona skrivenog sloja, boj epoha i broj aktivacija po uzorku su parametri koji primaju cjelobrojne argumente. Faktor učenja i konstanta **a**, su realni brojevi koje je potrebno unijeti s decimalnim zarezom umjesto točke kako ih funkcija parsiranja u `double` vrijednosti ne bi krivo izračunala. Kvačica se stavlja u

polje „RGB model“ ukoliko se radi sa slikama u boji, odnosno znakovima koji nisu trokutasti. Tim odabirom se automatski koristi i podjela slika u 36 fragmenata te značajke u boji. Bez nje se radi podjela u 144 fragmenta i rad s crno-bijelim značajkama. Kada su popunjeni svi parametri, moguće je pokrenuti proces treniranja mreže pritiskom na tipku „Napravi mrežu“. Proces treniranja, ovisno o parametrima, može potrajati i do više sati, a kada je gotov, dobivenu mrežu moguće je spremiti.



Slika 14. Učenje neuronske mreže.

Učitavanje postojeće mreže iz datoteke (Slika 15) vrši se pritiskom na istoimenu tipku. Nakon učitavanja mreže, ispisat će se osnovne informacije o učitanom modelu.



Slika 15. Učitavanje postojeće neuronske mreže iz datoteke.

Raspoznavanje prometnih znakova (Slika 16) vrši se pritiskom na tipku za učitavanje slike prometnog znaka kojeg se želi klasificirati. Nakon odabira, pojavljuju se tri slike. Prva je originalna slika rastegnuta s 24x24 piksela na 100x100, druga je ista ta slika u crno-bijelom, a treća predstavlja rezultat izvođenja, odnosno kategoriju koju je neuronska odredila učitanoj slici.

Slika 16. Raspoznavanje prometnih znakova – primjer uspješnog raspoznavanja.

6. Rezultati

Sam algoritam umjetnih neuronskih mreža relativno je jednostavan za implementaciju i konačni rezultati ne ovise u tolikoj mjeri o samoj mreži koliko o izboru ulaza za mrežu. Iz tog razloga je i napravljeno više različitih modela mreža, odnosno prepušteno je korisniku da pri treniranju odabere željene postavke mreže u smislu načina rada sa slikama i određivanja ulaza.

Parametar	**Testirane vrijednosti**
Broj skrivenih slojeva	1, 2
Broj neurona u skrivenim slojevima	20 – 1000
Konstanta **a** prijenosne sigma funkcije	0.1 – 1
Konstanta **b** prijenosne sigma funkcije	-20 – 0
Faktor učenja λ	0.05 – 1
Broj ulaza neuronske mreže	20 – 100
Veličina slike	24x24 piksela
Fragmenti slike	4x4, 6x6, 12x12
Broj kategorija znakova	3 – 9
Broj epoha	1 – 2000
Broj aktivacija mreže za uzorak u jednoj epohi	5 – 2000
Broj uzoraka za učenje po kategoriji	10 – 40

Tablica 1. Parametri mreže i testirane vrijednosti.

6.1. Optimalne vrijednosti parametara

Najkvalitetniji rezultati dobiveni su za niže vrijednosti konstante **a**, faktora učenja, broja ulaza i broja aktivacija mreže po uzorku. Testiranje je pokazalo da se bolji rezultati dobivaju kada je omjer broja epoha i broja aktivacije mreže po uzorku u jednoj epohi veći u korist broja epoha. U suprotnom se događa to da, pri klasifikaciji, mreža uvijek teži jednoj kategoriji i to najčešće onoj koja je zadnja bila trenirana. Optimalne iznose parametara nije moguće navesti jer se u velikoj mjeri

razlikuju ovisno o tome koja se mreža koristi. Tako je npr. broj ulaza u crno-bijelom modelu koji raspoznaje trokutaste znakove, dvostruko manji od broja ulaza u RGB modelu iz razloga što se u jednom koristi po jedan ulaz za svaki fragment slike, a u drugom svaki fragment predstavljaju tri vrijednosti R, G i B. Broj ulaza također ovisi i o podjeli slike u manje kvadratiće.

Kada su u pitanju skriveni neuroni, testiranje je uglavnom vršeno s jednim slojem skrivenih neurona. Optimalan broj skrivenih neurona u sloju je cca. dva do tri puta veći od broja ulaza mreže. Veći broj skrivenih neurona smanjuje mogućnost upadanja mreže u lokalni minimum iz kojeg ona ne može sama izaći.

Faktor učenja od 0.05 bio je najčešći odabir pri testiranju. Takva niska vrijednost usporava učenje mreže, no s većom vjerojatnošću dolazi do optimalnih veza u mreži.

Vrijeme učenja mreže u prosjeku je trajalo tri do četiri sata.

6.2. Uspješnost raspoznavanja

Ukupan broj uzoraka za testiranje iznosio je, u prosjeku, 10 za svaku kategoriju znakova u boji. Rezultati za znakove u boji prikazani su u Tablici 2.

Kategorija	**B01**	**B02**	**B04**	**B31**	**B39**	**C02**	**C08**
Broj prepoznatih	8/10	8/10	4/7	11/12	11/11	12/12	10/11
Postotak uspješnosti	80%	80%	57%	92%	100%	100%	91%

Tablica 2. Rezultati za RGB model.

Iz prikazanih rezultata vidljivo je da je moguće postići vrlo visoke postotke za pojedine kategorije. Najčešća greška bila je zamjena kategorija B01 i B04 (slike svih kategorija prikazane su na Slici 10), sve ostale greške uglavnom su uzrokovane slabim osvjetljenjem ili vanjskim faktorima (npr. crveno svijetlo na semaforu usmjereno prema bijelom znaku, slikanom pri zalasku sunca, mijenja bijelu boju u crvenu zbog reflektirajućeg svojstva materijala od kojeg su znakovi napravljeni).

Kategorija	A01	A08	A09	A10	A11	A14	A20	A34	A44
Broj prepoznatih	10/10	4/6	7/8	7/10	8/12	0/6	7/7	14/15	7/10
Postotak uspješnosti	100%	67%	88%	70%	67%	0%	100%	93%	70%

Tablica 3. Rezultati za trokutaste znakove.

Ukupan postotak ispravne klasifikacije kreće se oko 70%. Jedna od češćih grešaka prikazana je na Slici 17. Unutrašnjost ovih znakova preslična je po svojem obliku da bi ih algoritam uspješno razlikovao na temelju ovih značajki. Kod ovakvih slučajeva, potrebna je detaljnija analiza slike, odnosno njene unutrašnjosti.

Slika 17. Primjer neispravne klasifikacije.

Izrazito slab postotak raspoznavanja kategorije A14 uzrokovan je time što je mreža tu kategoriju učila na temelju tri različite kategorije (Slika 18). Početna pretpostavka, da neće biti velikih utjecaja na rezultate ukoliko se to učini, pokazala se neispravnom. Ovakvo učenje posljedica je nedovoljnog broja slika znakova kategorije A14.

Slika 18. Kategorije korištene za učenje kategorije A14.

6.3. Moguća poboljšanja

Programska izvedba u jeziku C++ bi sigurno skratila vrijeme izvođenja programa odnosno učenja mreže. Od samih poboljšanja na mreži, sigurno dolazi u obzir korištenje nekih naprednijih i kompleksnijih oblika mreža. Vrsta umjetnih

neuronskih mreža danas je mnogo, no većina ih nije u dovoljnoj mjeri testirana da bi se koristile često poput ovakve feed-forward mreže s back-propagation algoritmom. Konkretno na ovoj mreži, moguće je još uvesti i parametar momenta učenja koji neznatno ubrzava proces učenja, a dio je implementacije samog back-propagation algoritma. Također, odabir prijenosne funkcije bi mogao utjecati na kvalitetu rada i brzinu učenja. Iako najčešće korištena, nije još dokazano da je sigma funkcija bolji izbor prijenosne funkcije od neke druge.

Najbitnije promjene odnosile bi se na materijale za učenje mreže. Razlog zašto napravljena neuronska mreža radi sa slikama veličine 24x24 piksela leži u tome što su slike, odnosno materijali za učenje, preuzete iz ranijih radova s ovog područja tj. rada na predmetu Projekt na kojem se generirala baza od cca 400 slika znakova navedene veličine. Problem kod tih slika je taj što su izvađene iz video zapisa i često su vrlo nejasne. Kod raspoznavanja trokutastih znakova, gdje je u strogoj sredini znaka kategorija određena s nekoliko desetaka piksela, svaka smetnja ili zamućenje poprilično otežava rad algoritmu. Sustav koji bi radio s većim slikama (50x50 ili 100x100 piksela) bolje kvalitete, sigurno bi ostvario daleko kvalitetnije rezultate.

HSV sustav boja češće je bolji izbor, pri radu sa slikama u boji, nego RGB sustav. Omogućava jednostavnu diskretizaciju boja ograničavanjem pojedinih vrijednosti sustava. Time je moguće izbjeći smetnje na slici uzrokovane različitim vremenskim uvjetima, različitim osvjetljenjem uzoraka ili slikanjem u pokretu.

Na kraju, kada bi se radio sustav raspoznavanja nad velikim brojem kategorija znakova, moguće je napraviti „stablo“ mreža koje će na svoje grane propuštati znakove određenih karakteristika. S dubinom stabla, rasla bi i razina detalja koja se promatra u određenoj mreži. Tip mreže koji bi se nalazio pri vrhu takvog stabla je RGB model napravljen u ovom radu, a tip mreže koji bi bio pri dnu je model raspoznavanja trokutastih znakova.

7. Zaključak

Na temelju provedenih testiranja moguće je zaključiti da, s kvalitetnim materijalima za učenje i dovoljnom količinom ispitivanja optimalnih parametara, umjetna neuronska mreža može poslužiti kao pouzdan klasifikator prometnih znakova. Slike u pokretu, odnosno isječci iz video zapisa, mogu sadržavati preveliku količinu smetnji na sebi i time otežavati klasifikaciju neuronskim mrežama.

Najvažniji faktor, pri izradi sustava za raspoznavanje koji koristi algoritam umjetne neuronske mreže, jest odabir ulaza za mrežu, odnosno vađenje značajki iz uzoraka. Potrebno je odrediti skup značajki po kojem se uzorci maksimalno razlikuju, a da je taj skup nije prevelik.

Daljnjim testiranjem dodatnih parametara, vjerojatno je moguće dodatno podići postotak ispravne klasifikacije neuronske mreže. Osim toga, danas je prisutan i velik broj raznih algoritama učenja za neuronskih mreža, pa i samih tipova mreža, koji još nisu u dovoljnoj mjeri testirani, a neki od njih imaju potencijala da prikažu bolje rezultate od ovakve tipične feed-forward mreže.

Literatura

1. Dalbelo Bašić B; Čupić M; Šnajder J: *Umjetne neuronske mreže*, Zagreb, Fakultet elektrotehnike i računarstva, 2008.
2. Kröse B; Van der Smagt P: *An introduction to Neural Networks*. Osmo izdanje. The University of Amsterdam, studeni 1996.
3. Staudacher L: *Detekcija lica primjenom neuronskih mreža*, Zagreb, Fakultet elektrotehnike i računarstva, 2006.
4. Krnjić F: *Raspoznavanje prometnih znakova neuronskim mrežama*, Zagreb, Fakultet elektrotehnike i računarstva, 2009.
5. Ruta A: Traffic sign recognition using discriminitive local features, s Interneta, http://videolectures.net/ida07_ruta_tsr/, 2007.
6. Skupina autora, *Neuronske mreže* – Predavanje, s Interneta, http://eris.foi.hr/11neuronske/nn-predavanje.html#, ožujak 2003.
7. Frölich J: *Neural networks with Java*, s Interneta, http://www.nnwj.de/backpropagation.html, 2009.

Sažetak/Summary

Raspoznavanje prometnih znakova neuronskim mrežama

Rad analizira korištenje umjetnih neuronskih mreža u svrhu raspoznavanja slika prometnih znakova. Opisuju se osnovni principi rada umjetnih neuronskih mreža i način konstrukcije mreže za potrebe klasifikacije uzoraka. Dodatno se opisuje i način vađenja značajki iz slika, odnosno veza između optimalnijeg rada umjetne neuronske mreže i kvalitetno određenih značajki. U sklopu ovog rada, programskim jezikom C# napravljena je i programska implementacija algoritma feed-forward umjetne neuronske mreže s back-propagation algoritmom.

Ključne riječi: umjetna inteligencija, umjetna neuronska mreža, neuron, raspoznavanje uzoraka, raspoznavanje prometnih znakova, feed-forward, back-propagation, perceptron, prijenosna funkcija, značajke, vađenje značajki.

Traffic signs recognition using neural networks

This paper analyzes the use of artificial neural networks for traffic sign recognition. The basic principles of how artificial neural networks work are described as well as the procedure of constructing a network for pattern recognition. Additionally, feature extraction methods and the connection between a higher recognition rate of a neural network and feature extraction are described in this paper. An implementation of a feed-forward neural network using back-propagation was created using C# programming language.

Key words: artificial intelligence, artificial neural network, neuron, pattern recognition, traffic sign recognition, feed-forward, back-propagation, perceptron, activation function, features, feature extraction.