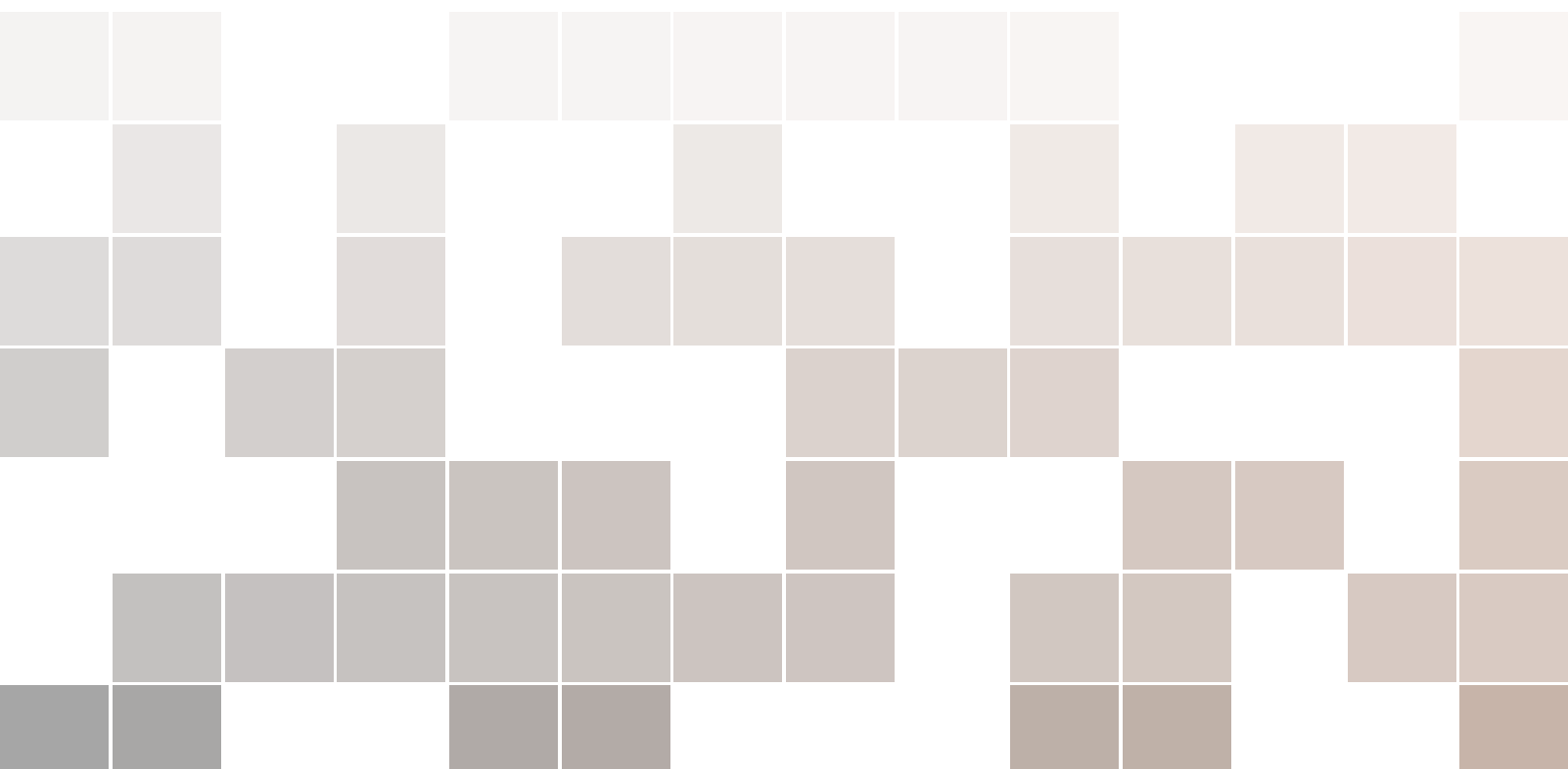


# Umjetna inteligencija

Umjetne neuronske mreže

doc. dr. sc. Marko Čupić



Copyright © 2016.-2017. Marko Čupić, v0.1.2

IZDAVAČ

JAVNO DOSTUPNO NA WEB STRANICI [JAVA.ZEMRIS.FER.HR/NASTAVA/UI](http://JAVA.ZEMRIS.FER.HR/NASTAVA/UI)

Ovaj materijal nastao je na temelju prezentacije o Umjetnim neuronskim mrežama na kolegiju Umjetna inteligencija (autori Bojana Dalbelo Bašić, Marko Čupić i Jan Šnajder) te na temelju dokumenta "Umjetne neuronske mreže" istih autora, kao proširenje istih.

Licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License (the "License"). You may not use this file except in compliance with the License. You may obtain a copy of the License at <http://creativecommons.org/licenses/by-nc/3.0>. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

*Prvo izdanje, svibanj 2016.*

# Sadržaj

<b>1</b>	<b>Motivacija i povijesni pregled</b> .....	<b>5</b>
1.1	Motivacija	5
1.2	Umjetna neuronska mreža i Von Neumannova arhitektura	6
1.3	Primjena umjetnih neuronskih mreža	7
1.4	Učenje neuronske mreže	9
1.5	Povijesni razvoj	11
<b>2</b>	<b>Umjetni neuron</b> .....	<b>17</b>
2.1	TLU-perceptron	17
<b>3</b>	<b>Višeslojna umjetna neuronska mreža</b> .....	<b>25</b>
3.1	Algoritam propagacije pogreške unatrag	26
3.2	Primjer učenja umjetne neuronske mreže	29
3.3	Drugi primjer	34
<b>4</b>	<b>Daljnji razvoj</b> .....	<b>37</b>
	<b>Bibliografija</b> .....	<b>39</b>
	Knjige	39
	Članci	39
	<b>Indeks</b> .....	<b>41</b>





# 1. Motivacija i povijesni pregled

Svijet koji nas okružuje neiscrpan je izvor inspiracije za rješavanje mnogih otvorenih pitanja koja nas još uvijek muče u tehničkim sustavima koje gradimo. Kada smo govorili o načinima kako ostvariti da se tehnički sustav ponaša inteligentno, prvi pristup koji smo obradili oslanjao se na manipulaciju simbolima vođenu definiranim pravila koja govore što je dozvoljeno a što ne. Tada smo govorili o različitim vrstama *logika* poput *propozicijske logike* te *predikatne logike*. Osvrnuli smo se na izgradnju tehničkih sustava koji su temeljeni na takvim formalnim logikama te na ograničenja koja iz toga slijede. Takav pristup izgradnji tehničkih sustava odnosno modeliranju znanja poznat je kao *simbolički pristup*.

Međutim, zapitamo li se tko (ili što) čak i u ovo doba brzog tehnološkog razvoja obrađuje najveću količinu podataka, odgovor nas ne bi trebao iznenaditi: živčani sustavi živih organizama. Mi sami (ljudi) živi smo organizmi i inteligencija o kojoj upravo govorimo nešto je što kao živo biće posjedujemo.

## 1.1 Motivacija

U svijetu u kojem živimo i djelujemo, svakodnevno se susrećemo s inteligentnim ponašanjem: od inteligentnog ponašanja koje je rezultat suradnje mnoštva jednostavnih bioloških jedinki (mravlje kolonije, kolonije pčela, jata ptica, rojevi riba) do inteligentnog ponašanja koje je utjelovljeno u svakoj pojedinoj biološkoj jedinki (poput čovjeka). Ljude kao vrstu oduvijek je fascinirao pojam inteligencije, pitanje možemo li razumjeti kako inteligencija nastaje te možemo li izgraditi inteligentan umjetni sustav (tehnički ili neke druge vrste).

Znanosti poput neurofiziologije i kognitivne znanosti posebnu pažnju posvetile su odgovaraju na takva pitanja u kontekstu čovjekove inteligencije. Kroz provedena istraživanja spoznali smo da se mozak (i ljudi i životinja) sastoji od izuzetno velikog broja *neurona* (živčanih stanica; gradbenih stanica živčanog sustava) koji obradu podataka rade paralelno. Da je tome tako, jednostavno se možemo uvjeriti razmotrimo li sljedeće. Prema provedenim istraživanjima, brzina rada biološkog neurona je poprilično na skali jedne milisekunde ( $10^{-3}$  s) - svake milisekunde neuron "pali" odnosno naboj prikupljen u tijelu stanice prenosi se dalje prema neuronima na koje je povezan. Pogledajmo sada jedan, za čovjeka vrlo jednostavan, zadatak: da bismo s fotografije prepoznali našu mamu,

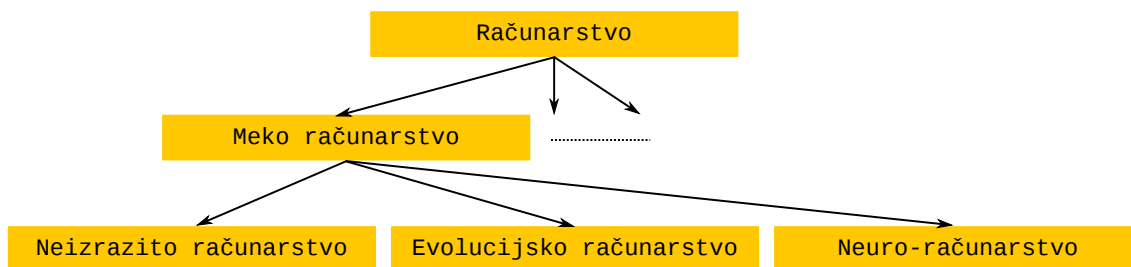
potrebno nam je oko desetinke sekunde ( $10^{-1}$  s). Pretpostavimo li da se postupak prepoznavanja u mozgu odvija serijski - dugačkim nizom paljenja međusobno povezanih neurona, imamo problem: u jednoj desetinki milisekunde, najduži lanac povezanih neurona ne može biti dulji od 100 neurona ( $10^{-1}/10^{-3} = 10^2 = 100$ ) što bi značilo da je u 100 jednostavnih koraka moguće riješiti taj problem. No kada bi to bilo tako, tada bismo već odavno znali napisati odgovarajući računalni program. Ono što se u stvarnosti događa jest da ljudsko oko (kao "senzor") u mozak šalje mnoštvo informacija koje predstavljaju "sliku" naše mame; to mnoštvo informacija dalje se masovno paralelno obrađuje prije no što *spoznamo* da se radi o našoj mami.

Ova spoznaja poslužila je kao motivacija za razvoj potpuno drugačijeg pristupa izgradnji tehničkih sustava i modeliranju znanja u odnosu na simbolički pristup. Taj novi pristup tehnički sustav gradi oponašajući građu bioloških sustava. Definiraju se i grade jednostavne procesne jedinice (*umjetni neuroni*) koje se potom povezuju u paralelne strukture različitih arhitektura poznate kao *umjetne neuronske mreže* (engl. *Artificial Neural Networks*). Ovaj pristup poznat je kao *konektivistički pristup*. U nastavku teksta, umjesto punog naziva "umjetna neuronska mreža" najčešće ćemo koristiti pokratu "neuronska mreža" ili čak samo "mreža" podrazumijevajući da se radi o umjetnoj neuronskoj mreži.

**R** *Simbolički pristup* je pristup kod kojega se znanje iz neke domene nastoji obuhvatiti skupom atomičkih semantičkih objekata (simbola) i zatim činiti niz manipulacija tih simbola pomoću algoritamskih pravila.

*Konektivistički pristup* temelji se na izgradnji sustava čija je arhitektura slična arhitekturi mozga i koji, umjesto da ga se programira, uči samostalno na temelju iskustva.

Područje koje se bavi izučavanjem izgradnje, učenja i uporabe umjetnih neuronskih mreža naziva se *neuro-računarstvo* i čini jednu od grana *računarstva* iz skupine *mekog računarstva* (engl. *Soft Computing*; vidi sliku 1.1). Skupina mekog računarstva objedinjuje niz pristupa koji su fokusirani na izgradnju tehničkih sustava koji rješavaju teške probleme i to često u okruženju u kojem postoji nepouzdanost i šum u ulaznim podacima. Glavne grane mekog računarstva su *neizrazito računarstvo* koje obuhvaća primjerice uporabu neizrazite logike za oblikovanje tehničkih sustava, *evolucijsko računarstvo* koje inspiraciju povlači iz teorije *evolucije* pa razmatra evoluciju kao mehanizam izgradnje sve boljih i boljih sustava te *neuro-računarstvo* koje inspiraciju povlači iz spoznaja o građi mozga pa gradi tehničke sustave konceptualno slične arhitekture.



Slika 1.1: Neuro-računarstvo i njegov položaj unutar računarstva.

## 1.2 Umjetna neuronska mreža i Von Neumannova arhitektura

Simbolički pristup obradi podataka često podrazumijeva slijednu obradu podataka. Stoga je upravo Von Neumannova arhitektura računala primjerena za takav pristup: problem se na Von Neumannovoj arhitekturi rješava algoritamski na sekvencijskom stroju. Konektivistički pristup odnosno umjetne neuronske mreže s druge je pak strane primjer paradigme koja je prirodna za

masovno raspodijeljeno i paralelno računanje. Pojednostavljena usporedba bitnijih karakteristika ovih dviju paradigmi dana je u tablici 1.1.

Tablica 1.1: Usporedba Von Neumannove arhitekture te umjetnih neuronskih mreža

Von Neumannova arhitektura	Umjetna neuronska mreža
Unaprijed detaljno opisujemo algoritam kroz korake	Uči samostalno ili s učiteljem
Samo se precizni podatci primjereno obrađuju	Podatci mogu biti nejasni (šum) ili neizraziti
Funkcionalnost ovisi o svakom elementu	Obrada i rezultat ne ovise mnogo o jednom elementu
Eksplisitna veza: semantički objekt - sklopovi računala	Implicitno znanje (teška interpretacija)

Pri radu na Von Neumannovoj arhitekturi, način rješavanja problema moramo definirati unaprijed i to tako da ga razložimo na niz koraka: specificiramo algoritam rješavanja problema. Pri tome se podrazumijeva da su podatci savršeno točni: situaciju u kojoj u neku varijablu zapišemo primjerice vrijednost `true` a nakon toga pročitamo vrijednost `false` okarakterizirat ćemo kao pogrešan rad sustava. U takvoj situaciji algoritam neće na ispravan način obrađivati podatke. Isto tako, podatci moraju biti precizni: "malo ću kasniti" ne znamo koristiti kao ulaz algoritma. Za ispravno izvođenje algoritma od presudne je važnosti da svi relevantni dijelovi računala rade ispravno. Dogodi li se da se jedan bit u aritmetičko-logičkoj jedinici središnjeg procesora vraća pogrešno, rezultat obrade će biti pogrešan; štoviše, ako se rezultat koristi u nekoj petlji kao uvjet zaustavljanja, moguće je da izračun zaglavi u beskonačnoj petlji. Program koji se izvodi djeluje nad varijablama: za svaku varijablu točno znamo što ona u programu predstavlja, gdje je smještena u memoriji, kada se učitava u registre procesora i slično.

Umjetna neuronska mreža predstavnik je sasvim drugačijeg načina obrade podataka. Mreži se ne definira način na koji treba obrađivati podatke već mrežu učimo na temelju samih podataka. Ovisno o vrsti podataka, mreža može učiti samostalno (bez učitelja) ili pak pod nadzorom (s učiteljem). Podatci koji se dovode neuronskoj mreži na ulaz ne moraju biti savršeno precizni niti točni: mreža će tijekom učenja razviti sposobnost generalizacije pa će dobro raditi i u situacijama kada su ulazni podatci zagađeni šumom. S obzirom da se obrada podataka odvija masovno raspodijeljeno te uz mnogo redundancija, točan rezultat obrade ne ovisi mnogo o jednom konkretnom neuronu; stoga će neuronska mreža čak i uz umjerenu količinu kvarova podatke obrađivati prihvatljivo dobro. Uz sve prethodno navedene prednosti, umjetne neuronske mreže imaju i jedan nedostatak: znanje koje je naučeno iz podataka, neuronska mreža interno pohranjuje u način povezivanja neurona od kojih se sastoji te moduliranjem utjecaja koji jedan neuron ima na drugi. Stoga kažemo da je u neuronskoj mreži znanje pohranjeno implicitno i za čovjeka je ono često neinterpretabilno.

### 1.3 Primjena umjetnih neuronskih mreža

Umjetne neuronske mreže danas imaju široku primjenu u području *klasifikacije* te *funkcijske regresije*. Prisjetimo se, klasifikacija je postupak određivanja razreda (klase) kojoj ulazni uzorak pripada. Primjeri klasifikacijskih zadataka su odrediti na temelju krvnog tlaka, broja otkucaja srca i još niza drugih značajki treba li pacijenta smjestiti na intenzivnu skrb ili ne; na temelju mjesečne plaće, povijesti prihoda u prethodne tri godine, vlasništva nekretnina i pokretnina te drugih značajki odrediti želimo li osobi dati novi kredit i pod kojim uvjetima (redovni kredit, rizičan kredit, odbiti

davanje kredita); te mnogi drugi. Funkcijska regresija je postupak pri kojem se na temelju danih podataka pokušava napraviti aproksimacija funkcije: izgraditi model na temelju kojega će sustav koji obavlja regresiju čak i za prethodno neviđene ulazne uzorke na izlazu dati vrijednost koja je "u skladu" s danim podatcima. Evo jednostavnog primjera.

■ **Primjer 1.1** Pretpostavimo da smo mjerili ulaz i izlaz nekog sustava koji ima jedan ulaz i jedan izlaz i na oba mjerimo napon. Ako smo mjerenjem utvrdili da kada je ulaz 1V, izlaz je bio 2V, za ulaz 2V izlaz je bio 3V a za ulaz 3V izlaz je bio 4V, što ćemo zapisati kao skup uzoraka za učenje oblika  $(x, f(x))$ , dakle  $\{(1, 2), (2, 2.99), (3, 4.01)\}$ , možemo se zapitati što bi bio izlaz sustava za ulaz 1.5V ili pak za ulaz 4V. Ulaz sustava označit ćemo s  $x$  a izlaz  $f(x)$  i pretpostavit ćemo da postoji funkcijska ovisnost izlaza o ulazu. Ako je skup uzoraka za učenje jedino što znamo, tada nam ne preostaje ništa dugo nego da pretpostavimo kakav je oblik funkcije  $f$  (odnosno kakav je *model sustava*). Primjerice, mogli bismo pretpostaviti da su ona sitna odstupanja izlaza oko cijelobrojnih vrijednosti rezultat djelovanja šuma i da su podatci zapravo trebali biti  $\{(1, 2), (2, 3), (3, 4)\}$  te da stoga vrijedi  $f(x) = x + 1$ . Uz takav model (a to je rezultat *učenja* provedenog u svrhu izgradnje sustava koji radi funkcijsku aproksimaciju), lagano je odgovoriti na postavljena pitanja:  $f(1.5V) = 2.5V$  te  $f(4V) = 5V$ . Možemo li biti sigurni da smo izgradili korektan model? Uz ovako malo podataka, odgovor je ne. Uz puno veći skup podataka mogli bismo već s više sigurnosti tvrditi da je pronađeni model dobar model sustava (barem u onom području ulazne varijable koje smo uzorkovali provedenim mjerenjima). ■

Konkretni primjeri uporabe umjetnih neuronskih mreža u različitim područjima navedeni su u nastavku.

- **Obrada jezika.** Primjena uključuje sustave za pretvorbu teksta u govor, sustave kojima se može upravljati govorom, sigurnosne sustave koji na temelju govora prepoznaju govornika, sustave za automatsku transkripciju (pretvaranje govora u tekst) i slično.
- **Prepoznavanje pisanih znakova.** Primjena uključuje sustave koji na temelju skenirane stranice rade detekciju teksta, segmentaciju u retke, riječi i pojedina slova te prepoznavanje slova. Tekst pri tome može biti ili štampan ili pisan rukom. U današnje doba umjetne neuronske mreže zadatak prepoznavanja znamenaka rješavaju s točnošću od preko 99.7%<sup>1</sup> (ova točnost postignuta je *dubokim konvolucijskim neuronskim mrežama*).
- **Kompresija slika.** Uključuje sustave koji u stvarnom vremenu obavljaju kompresiju i dekompresiju podataka.
- **Prepoznavanje uzoraka.** Primjena uključuje sustave koji se koriste u zračnim lukama i koji na temelju snimaka prtljage detektiraju bombe i druge opasne predmete, kao i niz primjena u medicini.
- **Obrada signala.** Primjeri uključuju sustave za uklanjanje šuma iz signala (filtriranje), sustave za detekciju signala, sustave za uklanjanje interferencije signala, sustave za detekciju i ispravljanje pogrešaka, sustave za identifikaciju izvora signala i druge.
- **Financije.** U financijskom sektoru (primjerice banke i kreditne kuće) često se donosi niz odluka koje zahtjevaju pažljivo razmatranje od strane čovjeka. Stoga su vrlo interesantni sustavi koji mogu učiti na temelju povijesnih podataka, generalizirati i potom biti korišteni za automatsko donošenje odluka (ili barem savjetovanje) što može bitno ubrzati postupak odlučivanja.
- **Upravljanje.** Postoji niz primjena, od jednostavnih (upravljanje inverznim njihalom) pa do složenih poput upravljanja Shuttleom i upravljanja vozilom. Primjer ovog posljednjeg je troslojna neuronska mreža ALVINN<sup>2</sup> razvijena još davne 1989. godine, koja je kao ulaz

<sup>1</sup>Za sve željne izazova, javno dostupna baza uzoraka rukom pisanih znamenki poznata pod nazivom MNIST sadrži preko 60000 uzoraka za učenje a dostupna je na adresi <http://yann.lecun.com/exdb/mnist/>

<sup>2</sup>Više informacija dostupno je na <http://repository.cmu.edu/cgi/viewcontent.cgi?article=2874&context=compsci> te [http://ftp.utcluj.ro/pub/docs/imaging/Autonomous\\_driving/Article%](http://ftp.utcluj.ro/pub/docs/imaging/Autonomous_driving/Article%2874&context=compsci)



dobivala video rezolucije  $30 \times 32$  slikovna elementa (samo plavu komponentu boje) i  $8 \times 32$  polje informacija o udaljenostima dobivenih mjerenjem laserom. Ova mreža uspješno je upravljala vozilom na autocesti sjeverno od grada Pittsburgh vozeći pri tome brzinom do 113 km/h (70 mph) kroz dionicu od preko 144 km (90 milja).

## 1.4 Učenje neuronske mreže

Rad s umjetnim neuronskim mrežama možemo podijeliti u dvije faze: fazu učenja (treniranja, engl. *learning, training*) te fazu iskorištavanja (eksploatacije, engl. *exploitation*).

**R** Neuronske mreže prolaze kroz dvije faze: fazu učenja te fazu iskorištavanja.

U fazi učenja, neuronskoj se mreži predočavaju uzorci iz skupa uzoraka za učenje. Uslijed toga, dolazi do promjena u jakosti veza između neurona koji čine neuronsku mrežu čime se mreža prilagođava viđenim podacima. Jakost veza još ćemo zvati i težinama. Ovisno o načinu učenja, neuronska mreža težine može podešavati nakon svakog viđenog uzorka ili tek nakon što vidi sve uzorke iz skupa za učenje. Ovaj prvi slučaj u kojem neuronska mreža uči nakon svakog predočenog uzorka naziva se *pojedinačno učenje* (engl. *on-line learning*). Drugi slučaj u kojem se neuronska mreža ne podešava sve dok ne vidi čitav skup uzoraka za učenje naziva se *grupno učenje* (engl. *batch learning*). Predočavanje jednog uzorka neuronskoj mreži zvat ćemo jednom *iteracijom* a predočavanje čitavog skupa uzoraka za učenje jednom *epohom*.

**R** Učenje neuronske mreže može biti pojedinačno ili grupno. Postoje i izvedbe koje su negdje između a poznate su kao mini-grupe (engl. *mini-batches*).

Konačno, spomenimo još jednu podjelu načina učenja neuronskih mreža. Neuronska mreža može učiti *s učiteljem* (nadzirano učenje, engl. *supervised learning*), *učiti podržano* (engl. *reinforcement learning*) te *učiti bez učitelja* (engl. *unsupervised learning*). Kod učenja s učiteljem, neuronskoj se mreži predočavaju uzorci oblika  $\{(ulaz, \text{željeni izlaz}), \dots, (ulaz, \text{željeni izlaz})\}$  a zadaća mreže je naučiti klasifikaciju ili funkcijsku regresiju (ovisno o tome što su izlazi). Kod podržanog učenja, neuronska se mreža može koristiti primjerice kao model za aproksimaciju Q-vrijednosti stanja i akcija<sup>3</sup>. Kod učenja bez učitelja, uzorci koje dobiva neuronska mreža oblika su  $\{(ulaz), \dots, (ulaz)\}$ . Zadaća koju neuronska mreža pri tome obavlja jest grupiranje podataka (engl. *clustering*).

**R** Neuronska mreža može:

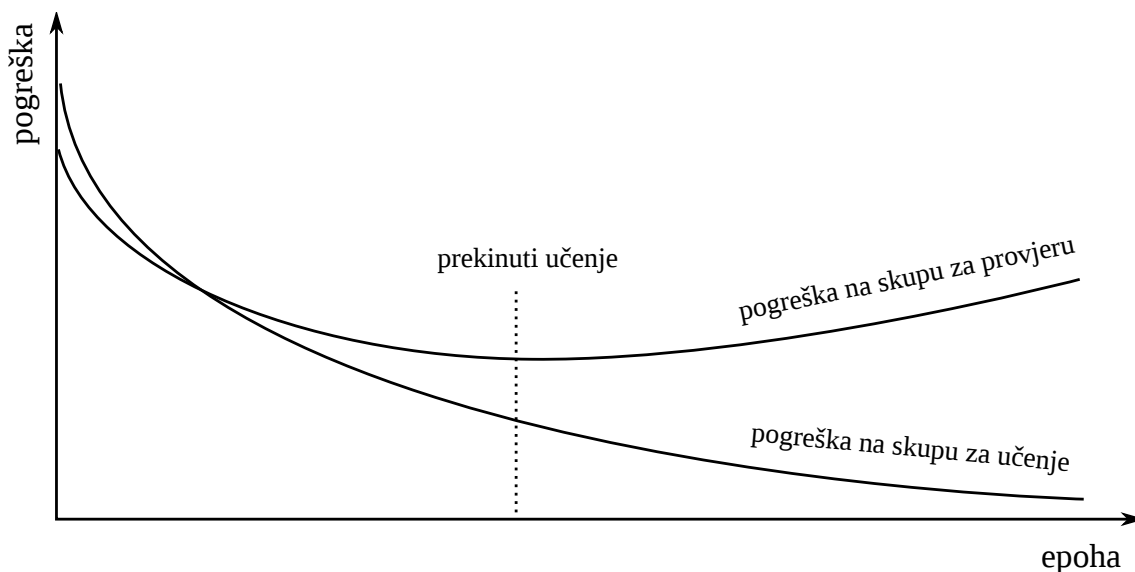
- učiti s učiteljem,
- učiti podržano te
- učiti bez učitelja.

Prisjetite se primjera u kojem smo uz zadani skup podataka tražili prikladan model funkcije. U tom primjeru, kao prikladan model smo odabrali jednostavan aditivni model. Kada radimo s neuronskim mrežama, složenost modela (tj. njegova ekspresivnost) određena je brojem neurona od kojih se mreža sastoji, njihovom funkcijom te načinom na koji su povezani u mreži. Sadrže li ulazni podatci šum, dovoljno ekspresivna neuronska mreža mogla bi tijekom učenja umjesto prilagodbe općim trendovima prisutnim u podacima početi se prilagođavati tako da napamet uči i prisutan šum i time gubi sposobnost generalizacije. Stoga nam je od posebnog interesa biti u stanju pratiti

20sortate/CThorpe/ALVINN%20Project%20Home%20Page.htm

<sup>3</sup>Za one koji žele znati više, zgodna prezentacija dostupna je na adresi [http://web.mst.edu/~gosavia/neural\\_networks\\_RL.pdf](http://web.mst.edu/~gosavia/neural_networks_RL.pdf)

postupak učenja, pustiti mrežu da uči tako dugo dok joj se sposobnost generalizacije popravlja te biti u stanju detektirati trenutak kada mreža počne gubiti sposobnost generalizacije i kada se počne prilagođavati prisutnom šumu u podacima, kako bismo tada prekinuli učenje.



Slika 1.2: Kretanje pogrešaka pri učenju neuronske mreže

U tu svrhu, čitav skup uzoraka s koji raspolažemo uobičajeno dijelimo u tri podskupa: *skup za učenje* (engl. *training set*, primjerice 60% uzoraka), *skup za provjeru* (engl. *validation set*, primjerice 20% uzoraka) te *skup za testiranje* (engl. *testing set*, primjerice 20% uzoraka). Tijekom učenja neuronske mreže, mrežu učimo na skupu za učenje - predočavamo joj te uzorke i dopuštamo joj da se prilagođava. Rad mreže povremeno kontroliramo nad skupom za provjeru. Primjerice, nakon svake epohe učenja provedene nad skupom za učenje, rad takve mreže provjerimo nad skupom za provjeru. Pri toj provjeri, neuronsku mrežu držimo fiksnom odnosno ne dopuštamo joj da se prilagođava tim uzorcima. Ideja ovakve izvedbe učenja je sljedeća. U ranim fazama učenja, neuronska će se mreža prilagođavati globalnim trendovima prisutnim u podacima pa će tako pogreška koju mreža ostvaruje i nad skupom za učenje i nad skupom za provjeru padati (vidi sliku 1.2). Naime, kako su isti globalni trendovi prisutni u oba skupa, prilagođavanjem skupu za učenje, neuronska će mreža kao posljedicu imati popravljavanje performansa i na skupu za provjeru (iako na njemu ne uči). Međutim, u jednom će se trenutku neuronska mreža početi prilagođavati specifičnim podacima i šumu prisutnom u skupu na učenje zbog čega će početi odstupati od globalnih trendova koji su prisutni u podacima. Kao posljedicu toga, pogreška koju neuronska mreža radi na skupu za učenje nastavit će padati (mreža će sve bolje i bolje oponašati šum u podacima jer ga uči napamet). Međutim, istovremeno će pogreška koju neuronska mreža radi na skupu za provjeru početi rasti - u tom skupu šum nije na istim mjestima i što mreža više odstupa na globalnih trendova prisutnih u podacima, to će lošije raditi na ovom skupu. Praćenjem krivulja kretanja pogrešaka na skupu za učenje te na skupu za provjeru moguće je detektirati taj trenutak kada pogreška na skupu za provjeru počne rasti i tada prekinuti učenje. Jednom kad je postupak učenja gotov, konačnu kvalitetu rada mreže provjeravamo na skupu za testiranje. ta se provjera radi samo jednom, kad je učenje gotovo, i mjeri konačnu kvalitetu mreže.

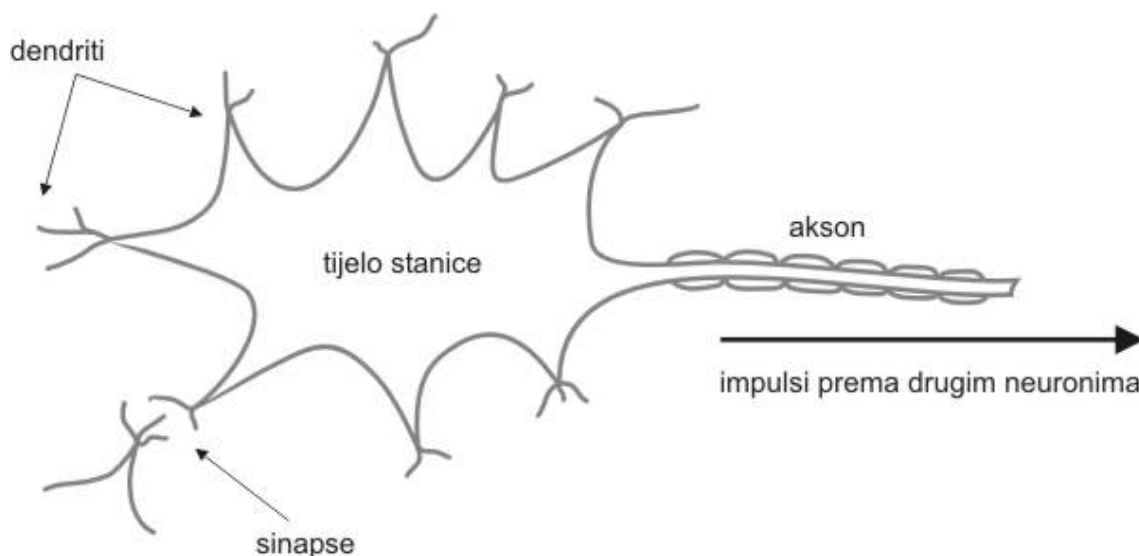
**R** Skup uzoraka za učenje neuronske mreže uobičajeno dijelimo na:

- skup uzoraka za učenje,
- skup uzoraka za provjeru tijekom učenja te
- skup uzoraka za završno testiranje.

*Pretrreniranost* je stanje neuronske mreže u kojem je ista izgubila sposobnost generalizacije te je postala stručnjak za podatke iz skupa za učenje (kažemo da je mreža postala *štreber*).

## 1.5 Povijesni razvoj

Prema trenutno dostupnim informacijama, ljudski se mozak sastoji od 100 milijardi neurona ( $10^{11}$ ) pri čemu postoji oko 100 različitih vrsta neurona. Neuroni su pri tome povezani prema određenim pravilima i vrlo gusto. Svaki je neuron u prosjeku povezan (odnosno komunicira) s deset tisuća ( $10^4$ ) drugih neurona. Time se broj veza koje neuroni razmjenjuju informacije penje na teško pojmljivih bilijardu ( $10^{15}$ ). Građa biološkog neurona prikazana je na slici 1.3.



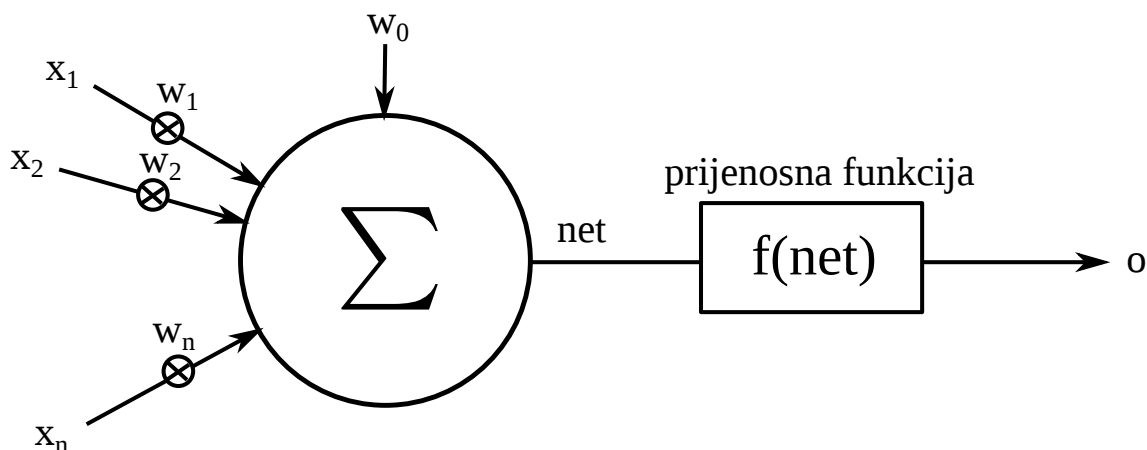
Slika 1.3: Biološki neuron.

*Tijelo stanice* je dio neurona u kojem se nalazi jezgra. Na rubovima tijela stanice nalaze se *dendriti*: tanki krakovi koji se granaju u dendritska stabla. Na slici 1.3 prikazano prema desno širi se *akson*: jedan dugačak krak čija duljina može biti i tisuću puta veća od duljine tijela stanice. U odnosu na okolinu, tijelo stanice kroz kemijske procese koji uključuju ione klora, kalcije i druge održava određenu razinu električkog potencijala. Dođe li do promjene tog potencijala, nastaje elektrokemijski impuls koji se brzo širi kroz akson koji je sinaptičkim vezama povezan s dendritima do 10000 drugih neurona. Preko dendrita ti neuroni preuzimaju poslani impuls koji kao posljedicu u određenoj mjeri modificira potencijal njihova tijela. Efikasnost prijenosa impulsa s aksona na dendrite određena je jakošću sinaptičkih veza<sup>4</sup>. Veze između neurona nisu statičke - što češće neuroni komuniciraju, to veza između njih postaje efikasnija pa poslana pobuda jače djeluje na sljedeći neuron.

Proučavajući strukturu biološkog neurona, tim znanstvenika Warren McCulloch i Walter Pitts definirali su 1943. godine model umjetnog neurona kakav je prikazan na slici 1.4.

*Dendriti* umjetnog neurona modelirani su ulazima  $x_1, x_2, \dots, x_n$ . Preko tih varijabli umjetni neuron prima pobudu od prethodnih neurona (ili vanjskog svijeta). U kojoj mjeri signal primljen kroz ulaz  $x_i$  utječe na neuron, modelirano je težinom  $w_i$ . Signal koji dolazi na ulaz  $x_i$  množi se težinom  $w_i$  te je njegovo djelovanje određeno umnoškom  $x_i \cdot w_i$ . Pozitivne težine pojačavaju djelovanje ulaza dok negativne težine inhibiraju djelovanje.

<sup>4</sup>Interesantna crtica: između aksona i dendrita nalazi se prazan prostor (engl. *synaptic cleft*) duljine oko 20 nanometara. Impuls na kraju aksona otpušta molekule poznate kao neurotransmiteri koje se otpuštaju u taj prazan prostor i prelaze do dendrita sljedećeg neurona prenoseći na taj način podražaj.



Slika 1.4: Model umjetnog neurona.

Tijelo stanice umjetnog neurona modelira integriranje svih primljenih pobuda u jedan zajednički potencijal stanice. Na slici 1.4 to je označeno znakom za sumaciju. Tijelo stanice umjetnog neurona na svojem izlazu (što je ulazni dio aksona) generira sumu koja je označena kao  $net$ , a računa se prema izrazu:

$$net = \sum_{i=1}^n w_i \cdot x_i + w_0. \quad (1.1)$$

U daljnjem tekstu definirat ćemo da svaki neuron ima još jedan (pomoćni) ulaz koji ćemo označiti s  $x_0$  i čija će vrijednost uvijek biti 1. Uz takav dogovor, izraz (1.1) kraće ćemo pisati na sljedeći način:

$$net = \sum_{i=0}^n w_i \cdot x_i = \vec{w} \cdot \vec{x} \quad (1.2)$$

gdje su sada  $\vec{w}$  i  $\vec{x}$  oba  $(n+1)$ -dimenzijski vektori a  $net$  njihov skalarni produkt.

Akson umjetnog neurona prikazan je kao prijenosna funkcija (engl. *transfer function*) koja modelira prolaz signala kroz akson biološkog neurona i koja određuje konačni izlaz  $o$  neurona prema izrazu:

$$o = f(net). \quad (1.3)$$

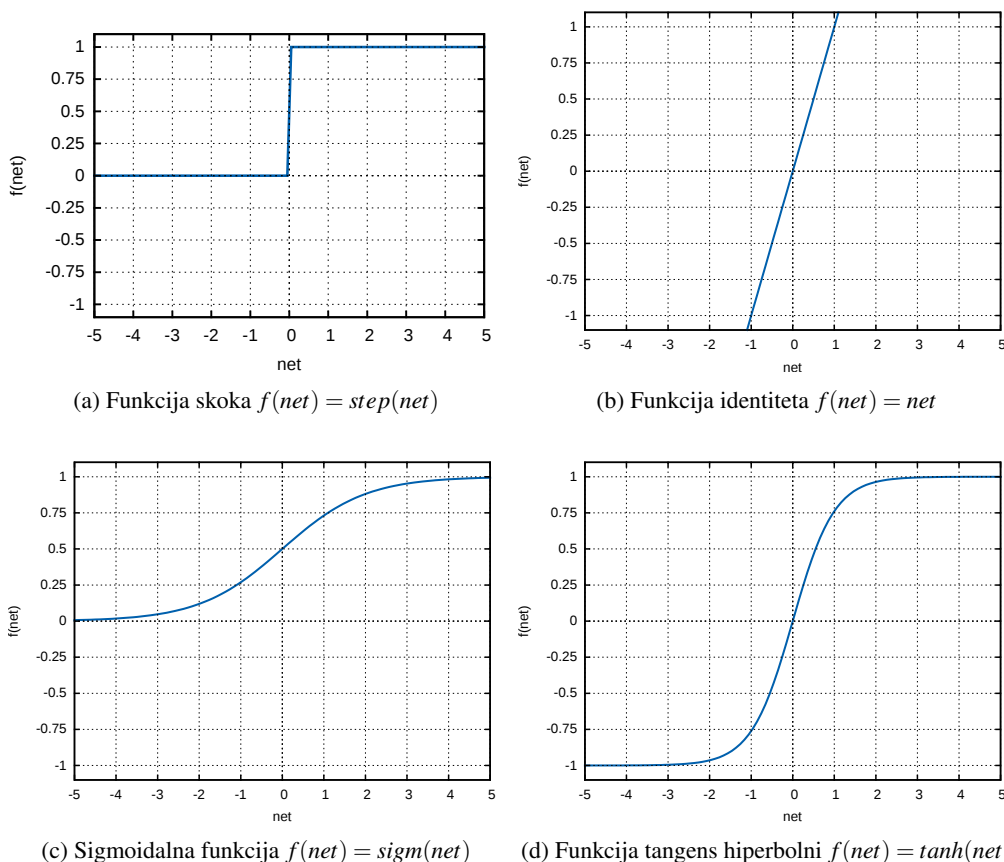
Do današnjeg dana umjetne su neuronske mreže građene uz niz različitih prijenosnih funkcija poput funkcije identiteta, funkcije skoka, sigmoidalne funkcije i funkcije tangens hiperbolni, dok se u posljednjih nekoliko godina razvojem dubokih neuronskih mreža sve više razmatraju ispravljena linearna funkcija (engl. *Rectified Linear Unit*), slabo-ispravljena linearna funkcija (engl. *Leaky Rectified Linear Unit*), *max-out* neuroni, i druge.

Ako se kao prijenosna funkcija koristi *funkcija skoka*, neuron koji se dobiva poznat je kao *TLU-perceptron* (engl. *Threshold Logic Unit*, kratica TLU). Funkcija skoka prikazana je na slici 1.5a i definirana izrazom:

$$step(net) = \begin{cases} 0, & net < 0 \\ 1, & net \geq 0. \end{cases} \quad (1.4)$$

U praksi je moguće pronaći i alternativnu definiciju koja umjesto između 0 i 1 izlaz mijenja između -1 i 1:

$$step(net) = \begin{cases} -1, & net < 0 \\ 1, & net \geq 0. \end{cases} \quad (1.5)$$



Slika 1.5: Različite prijenosne funkcije.

Sigmoidalna prijenosna funkcija (još poznata i kao logistička prijenosna funkcija) prikazana je na slici 1.5c a definirana je sljedećim izrazom:

$$sigm(net) = \frac{1}{1 + e^{-net}}. \quad (1.6)$$

Ovu funkciju možemo promatrati kao poopćenje funkcije skoka definirane izrazom (1.4) čija se vrijednost postupno mijenja od 0 do 1. Njezina prednost je činjenica da je derivabilna što će omogućiti da se neuronske mreže izgrađene od neurona s ovom prijenosnom funkcijom mogu učiti postupcima koji su temeljeni na gradijentnom spustu. Interesantno svojstvo ove funkcije je da se njezina derivacija može jednostavno izraziti opet preko nje same, kako je prikazano u izrazu:

$$\frac{d \, sigm(net)}{d \, net} = sigm(net) \cdot (1 - sigm(net)). \quad (1.7)$$

Funkcija tangens hiperbolni prikazana je na slici 1.5d i možemo je smatrati poopćenjem funkcije skoka definirane izrazom (1.5) čija se vrijednost postupno mijenja od -1 do 1. Baš kao i sigmoidalna prijenosna funkcija, i ona je derivabilna. Štoviše, funkciju tangens hiperbolni moguće je dobiti vrlo jednostavno izravno iz sigmoidalne prijenosne funkcije, što je vidljivo u nastavku.

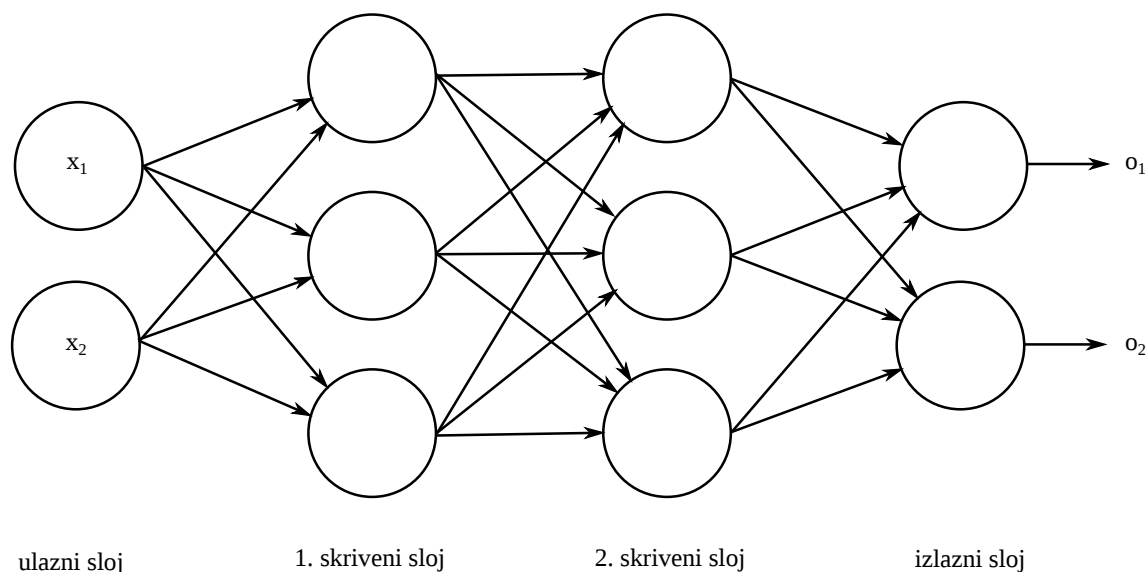
$$tanh(net) = 2 \cdot sigm(2 \cdot net) - 1 \quad (1.8)$$

Razvoj umjetnih neuronskih mreža moguće je kroz povijest pratiti od ranih 1940-ih pa do danas. Imao je svojih uspona i padova, a jedno od značajnijih razdoblja događa se tijekom *zime umjetne*

*inteligencije* (nazvano po analogiji na nuklearnu zimu - razdoblje smanjene aktivnosti). Krajem 50-ih godina prošlog stoljeća pada interes za konektivističkim pristupom a gotovo do potpunog zastoja dolazi 1969. godine kada znanstvenici Marvin Minsky i Seymour Papert objavljuju knjigu *Perceptrons* koja analizira mogućnosti i ističe ograničenja jednog perceptrona. Velik problem u to doba predstavljala je činjenica da se u složenim neuronskim mrežama nije znalo kako odrediti koji je neuron koliko odgovoran za neispravan rad mreže (netočnu klasifikaciju ili pogrešnu vrijednost izlaza pri funkcijskoj regresiji, ovisno o tome što mreža uči). Taj je problem dobio ime *problem dodjele zasluga* (engl. *Credit Assignment Problem*). Sve to zajedno dovelo je do značajne stagnacije na području istraživanja umjetnih neuronskih mreža koje je ponovno dobilo zalet tek u drugoj polovici 80-ih godina otkrićem algoritma propagacije pogreške unatrag (engl. *backpropagation*) koji je riješio problem dodjele zasluga. Ovaj algoritam otkriven je i primjenjen na umjetne neuronske mreže u razdoblju od 1982. (Werbos), 1985. (Parker, LeCun) pa sve do 1989. (Rumelhart i drugi). Treba napomenuti da je algoritam primjenjiv samo na jednu specifičnu porodicu umjetnih neuronskih mreža koje koriste derivabilne prijenosne funkcije – od tuda dolazi značaj prijenosnih funkcija poput sigmoidalne.

Od različitih vrsta umjetnih neuronskih mreža, mi ćemo se fokusirati na *potpuno povezane unaprijedne slojevite umjetne neuronske mreže* (engl. *Fully-connected Feed-forward Multi-layered*). Pojam "unaprijedna" mreža govori nam da u mreži ne postoje ciklusi u obradi podataka. Pojam "slojevita" dodatna je ograničenje koje govori da su neuroni grupirani u jasno definirane slojeve, te da neuroni smješteni u sloju  $i$  dobivaju pobudu isključivo od neurona u sloju  $i - 1$ . Pojam "potpuno povezana" pojašnjava da svaki neuron iz sloja  $i$  na ulaz dobiva pobude od svih neurona iz sloja  $i - 1$ .

Primjer potpuno povezane slojevite unaprijedne umjetne neuronske mreže prikazan je na slici 1.6.



Slika 1.6: Slojevita mreža arhitekture  $2 \times 3 \times 3 \times 2$ .

Prikazana mreža sastoji se od četiri sloja. Ulazni sloj čine dva neurona i kroz njih mreža dobiva ulazne podatke. Ti neuroni ne obavljaju nikakvu funkciju već je njihov izlaz direktno jednak podatku koji je dobiven izvana i koji mreža treba obraditi.

Drugi sloj (odnosno prvi skriveni sloj) sastoji se od 3 neurona. Svaki od neurona ulaze dobiva od svih neurona ulaznog sloja. Neuroni u 1. skrivenom sloju (kao i u svih sljedećim slojevima) obavljaju obradu podataka na prethodno opisani način: računaju težinsku sumu uzlaznih podataka, propuštaju je kroz prijenosnu funkciju i tako obrađen podatak postavljaju kao svoj izlaz. Svaki

od neurona u prvom skrivenom sloju ima po 3 težine: jednu koja određuje utjecaj prvog ulaznog neurona, druga koja određuje utjecaj drugog ulaznog neurona te težinu koja predstavlja konstantan pomak (engleski termin je *bias*, a u izrazima smo je označavali s  $w_0$ ). Te težine se ne dijele između neurona već svaki ima svoje vlastite tri težine čime drugi sloj mreže (1. skriveni sloj) raspolaže s  $3 \cdot 3 = 9$  težina koje postupak učenja može podešavati.

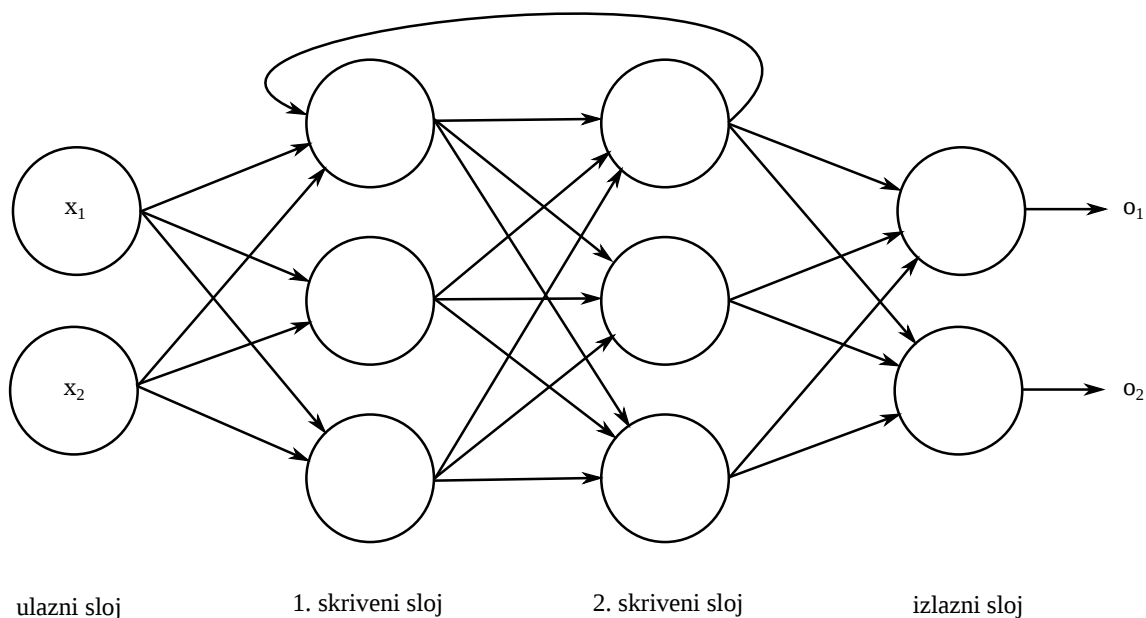
Treći sloj (odnosno drugi skriveni sloj) također se sastoji od 3 neurona koji su spojeni na izlaze 1. skrivenog sloja. U ovom sloju svaki od neurona ima po četiri težine (po jednu za svaki neuron iz 1. skrivenog sloja te jednu koja određuje konstantni pomak). Time 2. skriveni sloj raspolaže s  $3 \cdot 4 = 12$  težina.

Konačno, četvrti sloj je izlazni sloj. U njemu se nalaze dva neurona koja su spojena na neurone drugog skrivenog sloja. Ovdje svaki neuron ima po četiri težine čime je ukupni broj težina u ovom sloju  $2 \cdot 4 = 8$ .

Prikazana neuronska mreža ukupno raspolaže s  $9 + 12 + 8 = 29$  težina, što predstavlja broj parametara koji algoritam učenja mreže može podešavati kako bi funkciju mreže prilagodio podacima iz skupa za učenje.

Arhitekturu slojevitih mreža kraće zapisujemo navođenjem broja neurona u svakom od slojeva. Arhitekturu neuronske mreže prikazane na slici 1.6 označit ćemo s  $2 \times 3 \times 3 \times 2$ . Prikazana neuronska mreža može obavljati zadaću funkcijske regresije dviju funkcija ovisnih o dvije varijable:  $f_1(x_1, x_2)$  i  $f_2(x_1, x_2)$  pri čemu bismo vrijednosti funkcija za narinuti ulazni podatak čitali na izlazima  $o_1$  i  $o_2$ . S druge strane, ovakva neuronska mreža mogla bi obavljati i zadaću klasifikacije podataka u primjerice dva razreda (gdje bismo htjeli na izlazima  $(o_1, o_2)$  dobiti vrijednosti  $(1, 0)$  ako narinuti uzorak pripada prvom razredu, a  $(0, 1)$  ako pripada drugom). Ovakva mreža mogla bi raditi i klasifikaciju u četiri razreda, ako se dogovorimo o drugačijem načinu kodiranja pripadnosti; primjerice, mogli bismo tražiti da se na izlazima  $(o_1, o_2)$  dobije vrijednost  $(0, 0)$  ako uzorak pripada prvom razredu,  $(1, 0)$  ako pripada drugom razredu,  $(0, 1)$  ako pripada trećem razredu, odnosno  $(1, 1)$  ako pripada četvrtom razredu.

Primjer arhitekture neuronske mreže koja nije unaprijedna prikazan je na slici 1.7.



Slika 1.7: Slojevita mreža arhitekture  $2 \times 3 \times 3 \times 2$ .

U odnosu na mrežu prikazanu na slici 1.6, napravljena je samo jedna izmjena: dodana je veza kojom prvi neuron 1. skrivenog sloja kao ulaz dobiva izlaz prvog neurona 2. skrivenog

sloja. Razmotrite što smo time napravili. Da bismo odredili izlaz prvog neurona u 1. skrivenom sloju, trebamo izlaz prvog neurona iz 2. skrivenog sloja; pretpostavimo da smo pročitali njegovu vrijednost i potom izračunali *net* i konačni izlaz prvog neurona iz 1. skrivenog sloja (te ostalih iz tog istog sloja). Sada na red dolazi određivanje izlaza neurona iz 2. skrivenog sloja. Nakon što napravimo izračun, moguće je da će izlaz prvog neurona iz tog sloja sada promijeniti vrijednost. No kako prvi neuron iz 1. skrivenog sloja to dobiva kao ulaz, on bi mogao promijeniti svoju vrijednost čime bi opet mogao uzrokovati promjenu izlaza prvog neurona u 2. skrivenom sloju čime ... imamo *ciklus*. Prikazana neuronska mreža nije unaprijedna jer u njoj postoji ciklus, a postojanje ciklusa može dovesti do vremenski promjenjivog ponašanja neuronske mreže: nešto što se kod unaprijednih mreža ne može dogoditi.

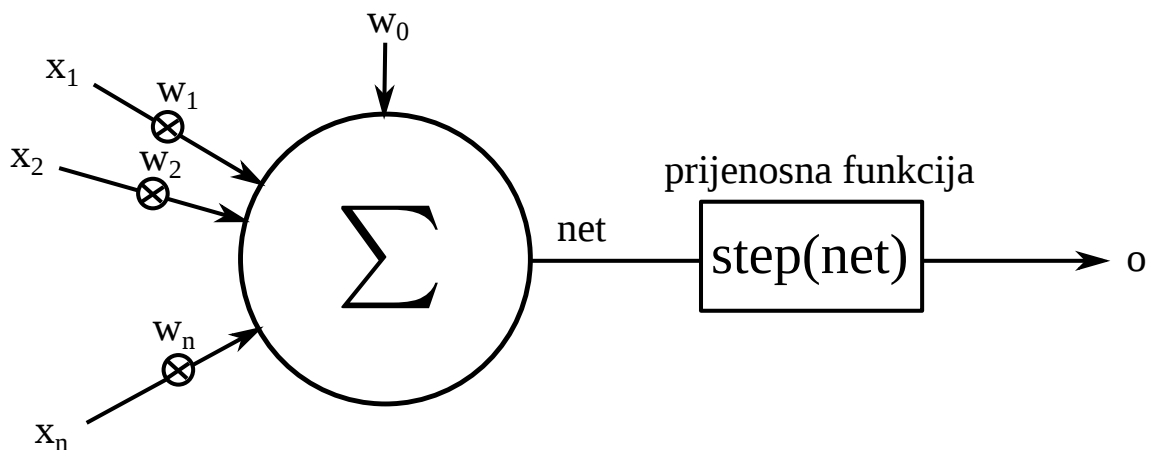


## 2. Umjetni neuron

Na putu prema razumijevanju složenijih neuronskih mreža, razmatranje ćemo započeti proučavanjem jednog neurona.

### 2.1 TLU-perceptron

Umjetni neuron poznat pod nazivom *TLU-perceptron* (engl. *Threshold Logic Unit*) je neuron koji težinsku sumu propušta kroz funkciju skoka i time određuje izlaznu vrijednost. Ovaj model prikazan je na slici 2.1.



Slika 2.1: TLU-perceptron kako su ga 1943. definirali Warren McCulloch i Walter Pitts.

Prisjetimo se, utjecaj svih ulaza neurona označili smo s *net* i računali kao težinsku sumu prema izrazu (1.2) koji ovdje ponavljamo:

$$net = \sum_{i=0}^n w_i \cdot x_i = \vec{w} \cdot \vec{x}.$$

Izlaz ovog neurona određen je izrazom (1.4) koji također ponavljamo:

$$\text{step}(\text{net}) = \begin{cases} 0, & \text{net} < 0 \\ 1, & \text{net} \geq 0. \end{cases}$$

Da bismo stekli predodžbu o zadaći koju ovakav neuron obavlja, razmotrimo situaciju u kojoj neuron ima samo dva ulaza:  $x_1$  i  $x_2$ . Tada je  $\text{net}$  određen izrazom:

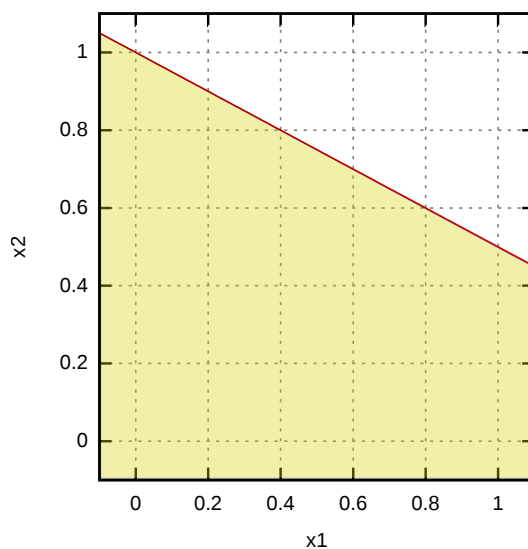
$$\text{net} = \sum_{i=0}^2 w_i \cdot x_i = w_2 \cdot x_2 + w_1 \cdot x_1 + w_0$$

(uvažavajući činjenicu da je po dogovoru  $x_0 = 1$  pa ga nećemo pisati). Funkcija skoka gleda predznak težinske sume: sve dok je težinska suma negativna, izlaz je 0 a kada težinska suma postane 0 ili pozitivna, izlaz postaje 1. Promjena izlaza ove vrste neurona stoga se događa upravo u okolini  $\text{net} = 0$ .

Pogledajmo konkretan slučaj: neka su težine TLU-perceptrona sljedeće:  $w_2 = 1$ ,  $w_1 = 0.5$ ,  $w_0 = -1$ . Pripadna težinska suma tada je:

$$\text{net} = 1 \cdot x_2 + 0.5 \cdot x_1 - 1.$$

U dvodimenzijском prostoru koji razapinju  $x_1$  i  $x_2$  sada možemo pogledati koji sve parovi točaka  $(x_1, x_2)$  zadovoljavaju uvjet  $\text{net} = 0$  jer se u okolini tih točaka mijenja izlaz neurona. Točke koje čine granicu oko koje neuron mijenja izlaz (a kako je on ovdje diskretan, još ćemo reći i da neuron mijenja odluku) nazivaju se *decizijska granica*. Pripadna decizijska granica za razmatrani neuron prikazana je na slici 2.2 gdje je crvena linija decizijska granica.



Slika 2.2: TLU-perceptron ulazni prostor dijeli u dva podprostora.

Sa slike možemo uočiti da je decizijska granica ovog neurona *linearna*; kako se radi o dvodimenzijском slučaju, ona je pravac. Za sve točke koje se nalaze u donjem poluprostoru (koji je na slici obojan žuto), neuron će na izlazu dati vrijednost 0. U to se lako uvjerimo ako uočimo da je primjerice ishodište u tom poluprostoru. Uvrštavanjem  $x_1 = 0$  i  $x_2 = 0$  u izraz  $1 \cdot x_2 + 0.5 \cdot x_1 - 1$  dobivamo da je  $\text{net} = -1$  pa neuron zbog funkcije skoka na izlazu daje 0. Za sve točke koje su u gornjem poluprostoru (na slici prikazan bijelo), neuron na izlazu daje vrijednost 1. Uvrstimo primjerice  $x_1 = 1$  i  $x_2 = 1$  u izraz  $1 \cdot x_2 + 0.5 \cdot x_1 - 1$ : dobivamo 0.5 zbog čega je izlaz neurona 1.

Kako se približavamo decizijskoj granici, magnituda težinske sume sve se više smanjuje i prilazi nuli. Dolaskom na decizijsku granicu, težinska suma postaje nula i prelaskom na drugu stranu dolazi do promjene izlaza neurona.

Iz izraza za *net* lagano je provjeriti da je u ovom slučaju decizijska granica upravo pravac. Evo koraka.

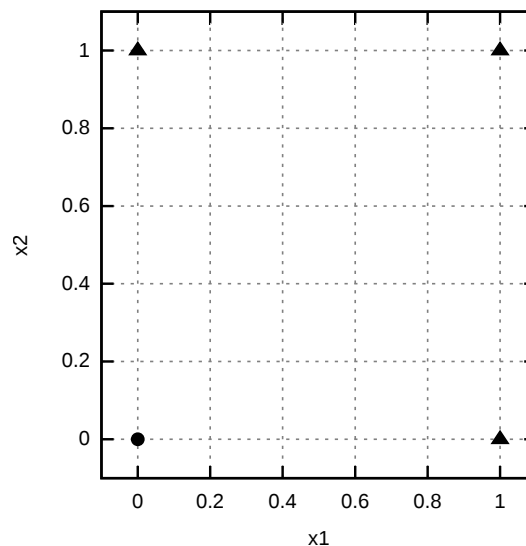
$$\begin{aligned} w_2 \cdot x_2 + w_1 \cdot x_1 + w_0 &= 0 \\ w_2 \cdot x_2 &= -w_1 \cdot x_1 - w_0 \\ x_2 &= -\frac{w_1}{w_2} \cdot x_1 - \frac{w_0}{w_2} \end{aligned}$$

što je školski primjer jednadžbe pravca ( $y = k \cdot x + l$ ).

Decizijska granica TLU-perceptrona koji ima tri ulaza je ravnina razapeta u trodimenzijskom prostoru ( $x_1, x_2, x_3$ ). Decizijske granice neurona koji imaju više od tri ulaza su hiperravnine. Zajedničko svojstvo svima njima je da su *linearne*.

Gdje se koristi TLU-perceptron? S obzirom na njegov binarni izlaz, uobičajena primjena je u klasifikaciji. U najjednostavnijem slučaju, ulazne uzorke potrebno je razvrstati u dva razreda:  $R_1$  i  $R_2$ . To možemo napraviti tako da svaki uzorak za koji je TLU-perceptron na izlazu dao vrijednost 0 smjestimo u jedan od ta dva razreda, a svaki uzorak za koji je TLU-perceptron na izlazu dao vrijednost 1 smjestimo u preostali razred.

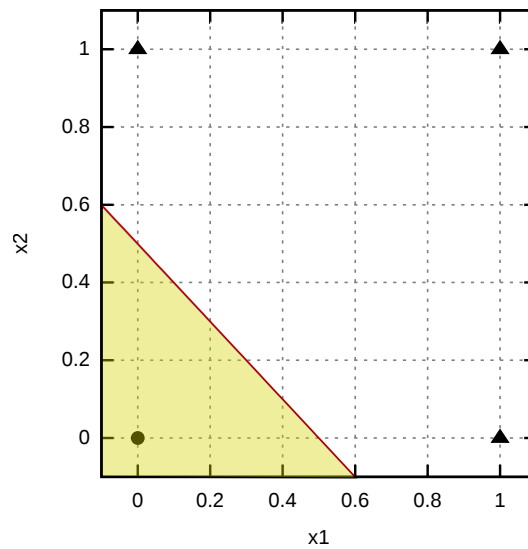
■ **Primjer 2.1** Razmatramo klasifikaciju u dvodimenzijskom prostoru ( $x_1, x_2$ ). U razred  $R_1$  pripada sljedeći skup uzoraka:  $\{(0,0)\}$  a u razred  $R_2$  pripada sljedeći skup uzoraka:  $\{(0,1), (1,0), (1,1)\}$ . Za klasifikaciju želimo koristiti jedan TLU-perceptron čiji ćemo izlaz 0 tumačiti kao znak da uzorak pripada razredu  $R_1$  a izlaz 1 kao znak da pripada razredu  $R_2$ . Situacija je ilustrirana na slici u nastavku. Uzorci razreda  $R_1$  prikazani su kružićima a uzorci razreda  $R_2$  trokutićima.



Odgovarajuća bi bila bilo koja decizijska granica (pravac) koja korektno dijeli ove uzorke. Jedna mogućnost bi bila:

$$1 \cdot x_2 + 1 \cdot x_1 - 0.5 = 0$$

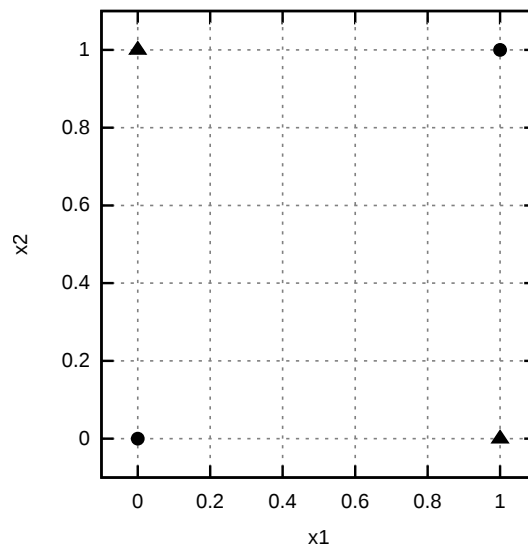
odnosno TLU-perceptron koji ima  $w_2 = 1$ ,  $w_1 = 1$ ,  $w_0 = -0.5$ . Slika u nastavku ilustrira ovu decizijsku granicu.



Uočite da ovo nije jedino moguće rješenje: mogli bismo pronaći beskonačno mnogo linearnih decizijskih granica uz koje bi TLU-perceptron korektno klasificirao zadane uzorke. ■

Pogledajmo još jedan primjer.

■ **Primjer 2.2** Razmatramo klasifikaciju u dvodimenzijском prostoru  $(x_1, x_2)$ . U razred  $R_1$  pripada sljedeći skup uzoraka:  $\{(0, 0), (1, 1)\}$  a u razred  $R_2$  pripada sljedeći skup uzoraka:  $\{(0, 1), (1, 0)\}$ . Za klasifikaciju želimo koristiti jedan TLU-perceptron čiji ćemo izlaz 0 tumačiti kao znak da uzorak pripada razredu  $R_1$  a izlaz 1 kao znak da pripada razredu  $R_2$ . Situacija je ilustrirana na slici 2.3.



Slika 2.3: TLU-perceptron ulazni prostor dijeli u dva podprostora.

Možemo li ovaj zadatak riješiti TLU-perceptronom? Odgovor je negativan: pogledate li malo bolje sliku, uočit ćete da ne postoji pravac (linearna decizijska granica) koji bi mogao korektno odraditi klasifikaciju odnosno koji bi s jedne strane imao kružice a s druge trokutiće. Primjer koji je ovdje prikazan ilustrira *linearno nerazdvojive razrede*. ■

Ostalo nam je odgovoriti na još jedno pitanje: moramo li težine TLU-perceptrona uvijek ručno namještati ili postoji mogućnost da ih sam perceptron nauči iz podataka? Odgovor na ovo pitanje

dao je Frank Rosenblatt, inspiriran opažanjem kanadskog znanstvenika Donalda Hebba koji je 1949. godine objavio djelo *The Organization of Behavior* u kojem primjećuje da se biološki neuroni koji međusobno komuniciraju mijenjaju na način da dolazi do promjena u jakosti sinaptičkih veza među njima. Rosenblatt 1958. godine spaja McCulloch-Pitts model neurona i Hebbovo opažanje te formulira *algoritam učenja perceptrona* koji dajemo u nastavku.

1. Ciklički prolazi kroz svih  $N$  uzoraka za učenje, jedan po jedan.
2. Klasificiraj trenutni uzorak.
  - (a) Ako se klasificira korektno, ne mijenjaj težine i
    - i. ako je to  $N$ -ti uzastopni uzorak klasificiran korektno, prekini učenje jer perceptron uz trenutni skup težina sve klasificira korektno,
    - ii. inače prijeđi na sljedeći uzorak.
  - (b) Ako se ne klasificira korektno, korigiraj težine perceptrona prema sljedećem izrazu:

$$w_i(k+1) \leftarrow w_i(k) + \eta \cdot (t - o) \cdot x_i$$

**R** Naglasimo da postupak definiran ovim algoritmom staje onog trenutka kada su zaredom točno klasificirani svi uzorci iz skupa uzoraka za učenje. Primjerice, pretpostavimo da radimo sa skupom za učenje veličine 10 uzoraka, te da je nakon nekoliko epoha algoritam korigirao težine nakon predočenog uzorka 3. Predočavanje 4. uzorka rezultiralo je ispravnom klasifikacijom, 5. također, 6. također i tako dalje. Nakon prelaska u novu epohu, nakon predočavanja uzoraka 1, 2 i 3 (uz pretpostavku da su svi do tada također bili korektno klasificirani), algoritam prestaje - nema potrebe ponovno provjeravati uzorke od 4 do posljednjeg.

Rad algoritma ilustrirat ćemo na jednostavnom primjeru. Neka imamo skup uzoraka za učenje prikazan tablicom 2.1. Uzorci su dani u dvodimenzijском prostoru kako bismo lakše mogli vizualizirati decizijsku granicu koju ostvaruje neuron. Stoga je svaki uzorak oblika  $(x_2, x_1) \rightarrow t$ . Slovo  $t$  označava željeni odnosno "pravi" izlaz (engl. *target-output* ili *true-output*).

Tablica 2.1: Skup uzoraka za učenje klasifikacijskog problema. Za uzorke koji pripadaju razredu  $R_1$  želimo da neuron na izlazu generira vrijednost 1 a za uzorke koji pripadaju razredu  $R_2$  želimo da neuron na izlazu generira vrijednost  $-1$ .

Redni broj uzorka	Ulaz $(x_2, x_1)$	Željeni izlaz $t$
1	(2, 5)	1
2	(5, 2)	1
3	(1, 5)	-1
4	(5, 1)	-1

Pretpostavit ćemo da je početni skup težina  $(w_2, w_1, w_0)$  jednak  $(1, 1.3, -5.85)$  te da se koristi stopa učenja iznosa  $\eta = 0.02$ . Tablica 2.2 prikazuje provedbu postupka učenja. Stupci tablice najprije prikazuju uzorak koji se predočava neuronu, potom aktivne vrijednosti težina koje se koriste na izračun vrijednosti *net*, potom izračunatu vrijednost *net* i pripadni izlaz neurona koji se dobije propuštanjem vrijednosti *net* kroz funkciju skoka koja skače između  $-1$  i  $1$ . Posljednja dva stupca tablice sadrže informaciju je li potrebna korekcija ('da' ako je dobiveni izlaz neurona različit od željenog izlaza neurona tj. ako je  $t \neq o$ , 'ne' inače), te ako je, koliki je iznos  $\eta \cdot (t - o)$ . Prisjetimo se, kod korekcije težina, svakoj se težini pridodaje upravo taj broj pomnožen vrijednošću ulaza koji ta težina množi.

Svi uzorci u tablici prošireni su ulazom  $x_0$  koji je fiksno postavljen na 1 kako bismo mogli kratko pisati  $net = \sum_{i=0}^n w_i \cdot x_i$ .

Postupak započinje prvom epohom i predočavanjem prvog uzorka. Kao vrijednost *net* dobiva se 2.65 i posljedično izlaz  $o = 1$ . Kako je time željena vrijednost na izlazu  $t = 1$  jednaka dobivenoj vrijednosti na izlazu, korekcija nije potrebna, težine se ne mijenjaju i prelazi se na sljedeći uzorak. Iz tog razloga, težine koje se koriste u drugom retku tablice jednake su onima koje se koriste u prvom retku tablice.

Predočavanjem drugog uzorka (drugi redak tablice) *net* je opet pozitivan, dobiva se izlaz  $o = 1$  što je u skladu s željenim pa se ne radi korekcija težina koje se prenose u treći redak tablice.

U trećem retku predočava se treći uzorak. Željena vrijednost na izlazu je  $t = -1$ ; međutim, neuron opet generira vrijednost  $o = 1$ . Stoga je potrebno provesti korekciju. Računamo  $\eta \cdot (t - o) = 0.02 \cdot (-1 - 1) = 0.02 \cdot -2 = -0.04$ . Potom računamo nove vrijednosti težina:

$$w_2 \leftarrow w_2 + \eta \cdot (t - o) \cdot x_2 = 1 + (-0.04) \cdot 1 = 0.96,$$

$$w_1 \leftarrow w_1 + \eta \cdot (t - o) \cdot x_1 = 1.3 + (-0.04) \cdot 5 = 1.1,$$

$$w_0 \leftarrow w_0 + \eta \cdot (t - o) \cdot x_0 = -5.85 + (-0.04) \cdot 1 = -5.89.$$

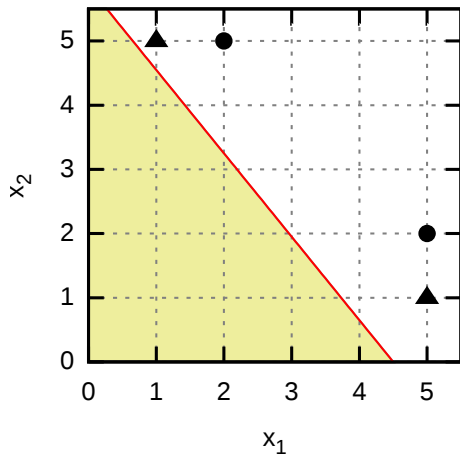
Tako korigirane vrijednosti težina prenose se dalje u četvrti redak tablice u kojem se neuronu predočava i posljednji četvrti uzorak. Dobiva se *net* koji je pozitivan i izlaz  $o = 1$  što opet nije u skladu sa željenim  $t = -1$ . Stoga se težine opet korigiraju i dobiva se novi skup težina  $(w_2, w_1, w_0) = (0.76, 1.06, -5.93)$ . Te se težine prenose u sljedeći korak kojim započinje nova epoha - predočavanje uzoraka opet kreće od početka.

Tablica 2.2: Provedba postupka učenja

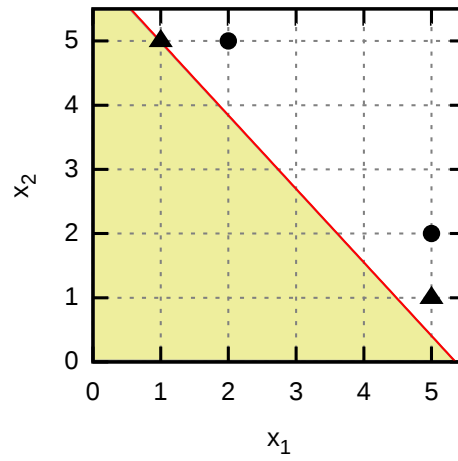
Epoha	Uzorak	$(x_2, x_1, x_0)$	$t$	$(w_2, w_1, w_0)$	<i>net</i>	$o$	Korekcija	$\eta \cdot (t - o)$
1	1	(2, 5, 1)	1	(1, 1.3, -5.85)	2.65	1	ne	×
1	2	(5, 2, 1)	1	(1, 1.3, -5.85)	1.75	1	ne	×
1	3	(1, 5, 1)	-1	(1, 1.3, -5.85)	1.65	1	da	-0.04
1	4	(5, 1, 1)	-1	(0.96, 1.1, -5.89)	0.01	1	da	-0.04
2	1	(2, 5, 1)	1	(0.76, 1.06, -5.93)	0.89	1	ne	×
2	2	(5, 2, 1)	1	(0.76, 1.06, -5.93)	-0.01	-1	da	0.04
2	3	(1, 5, 1)	-1	(0.96, 1.14, -5.89)	0.77	1	da	-0.04
2	4	(5, 1, 1)	-1	(0.92, 0.94, -5.93)	-0.39	-1	ne	×
3	1	(2, 5, 1)	1	(0.92, 0.94, -5.93)	0.61	1	ne	×
3	2	(5, 2, 1)	1	(0.92, 0.94, -5.93)	0.55	1	ne	×
3	3	(5, 1, 1)	-1	(0.92, 0.94, -5.93)	-0.31	-1	ne	×

Ovaj se postupak nastavlja tako dugo dok se za redom ne klasificira ispravno onoliko uzoraka kolika je veličina skupa uzoraka za učenje; u tom trenutku postupak se zaustavlja i to se u ovom slučaju događa u trećoj epohi nakon predočenog trećeg uzorka.

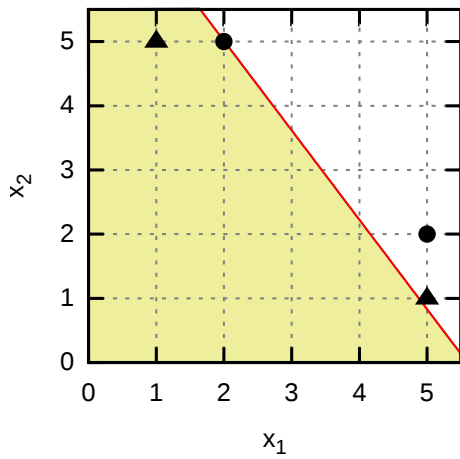
Promjene koje radi ovaj postupak grafički su prikazane na slici 2.4. Slika 2.4a prikazuje decizijsku granicu uz početne vrijednosti težina. Vidimo da će uz takvu decizijsku granicu neuron za sva četiri razreda na izlazu dodijeliti vrijednost 1. To se slaže i s rezultatima iz postupka učenja: predočavanjem prva tri uzorka, neuron je na izlazu generirao 1. Za prva dva, to je bila i željena vrijednost pa ništa nismo korigirali. Međutim, u koraku 3, željeni izlaz je bio  $-1$  pa je napravljena korekcija. Nakon tog trećeg koraka, decizijska granica je promijenjena u prikazana je na slici 2.4b.



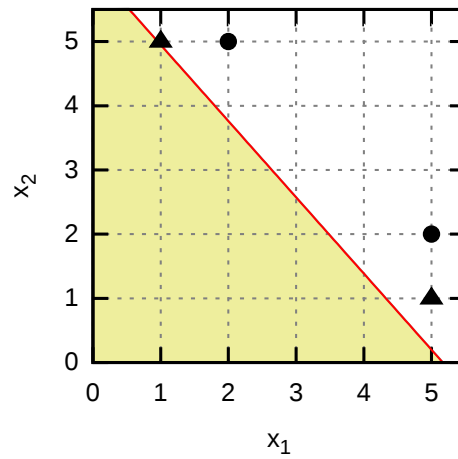
(a) Početno stanje, tj. prije koraka 1



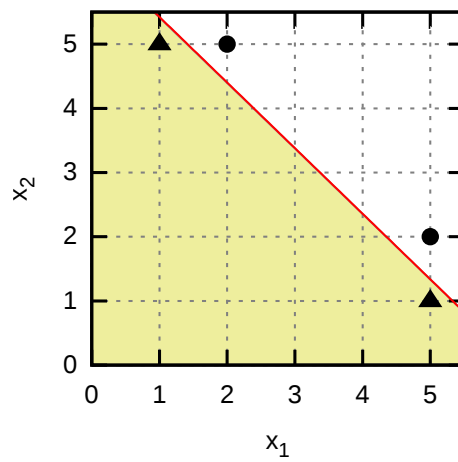
(b) Nakon koraka 3



(c) Nakon koraka 4



(d) Nakon koraka 6



(e) Nakon koraka 7

Slika 2.4: Promjene decizijske granice neurona tijekom postupka učenja

Uočimo da, iako nova decizijska granica nije korektna, njezin položaj bliži je položaju uz koji bismo imali korektnu klasifikaciju. Nova granica i dalje sva četiri uzorka klasificira u razred  $R_1$  (iako je četvrti uzorak - trokutić u gornjem lijevom kutu - gotovo upao u ispravan razred  $R_2$ ).

U četvrtom koraku razmatra se upravo taj četvrti uzorak i zbog njega se opet radi korekcija težina. Uslijed te korekcije mijenjaju se težine a time i decizijska granica koja je nakon ovog koraka prikazana na slici 2.4c. Ova decizijska granica ispravno klasificira uzorak 1 u  $R_1$  i uzorak 4 u  $R_2$ ; neispravno klasificira uzorke 2 i 3. Stoga prelaskom u novu epohu (korak 5) utvrđujemo da je klasifikacija uzorka 1 ispravna i ne mijenjamo težine, a u koraku 6 utvrđujemo da je klasifikacija uzorka 2 neispravna i ponovno mijenjamo težine. Situaciju nakon koraka 6 prikazuje slika 2.4d. Ovom korekcijom uzorci 1 i 2 ispravno se klasificiraju a uzorci 3 i 4 neispravno (iako, uzorak 4 je vrlo blizu korektnoj klasifikaciji: *net* za taj uzorak iznosi svega 0.05).

U koraku 7 razmatra se uzorak 3 koji granica neispravno klasificira i po posljednji se puta radi korekcija težina. Decizijska granica nakon koraka 7 prikazana je na slici 2.4e. Uz takvu decizijsku granicu uzorci 1 i 2 smještaju se u razred  $R_1$  a uzorci 3 i 4 u razred  $R_2$  što je i željeno ponašanje. Ispitivanjem uzorka 4 u koraku 8 te u sljedećoj epohi uzoraka 1, 2 i 3 u koracima 9, 10 i 11 postupak učenja to utvrđuje i obustavlja postupak učenja.

- R** Algoritam učenja Perceptrona sigurno konvergira ka pravoj decizijskoj granici ako su uzorci iz skupa za učenje linearno odvojivi te ako odabrana stopa učenja nije prevelika. Ovaj se algoritam, međutim, ne može koristiti kao test za provjeru jesu li uzorci linearno odvojivi. Naime, ako odradimo određen broj iteracija i postupak još nije konvergirao, ne znamo što je uzrok tome: je li razlog taj što su uzorci linearno neodvojivi (pa sve i da nastavimo još beskonačno mnogo koraka, postupak nikada ne bi konvergirao) ili taj što koristimo neprikladnu stopu učenja pa postupak konvergira ali sporo (i da smo mu dali još određen broj iteracija, konvergirao bi ka korektnoj granici).



### 3. Višeslojna umjetna neuronska mreža

U ovom poglavlju osvrnut ćemo se na osnovni algoritam učenja unaprijednih neuronskih mreža: *algoritam propagacije pogreške unatrag*. Algoritam je otkriven sredinom 80-ih godina prošlog stoljeća. Do danas je napravljen niz njegovih modifikacija koje su isprobane u praksi, kao i druge metode učenja poput evolucijskih algoritama koji su primjenjivi i na širi skup neuronskih mreža.

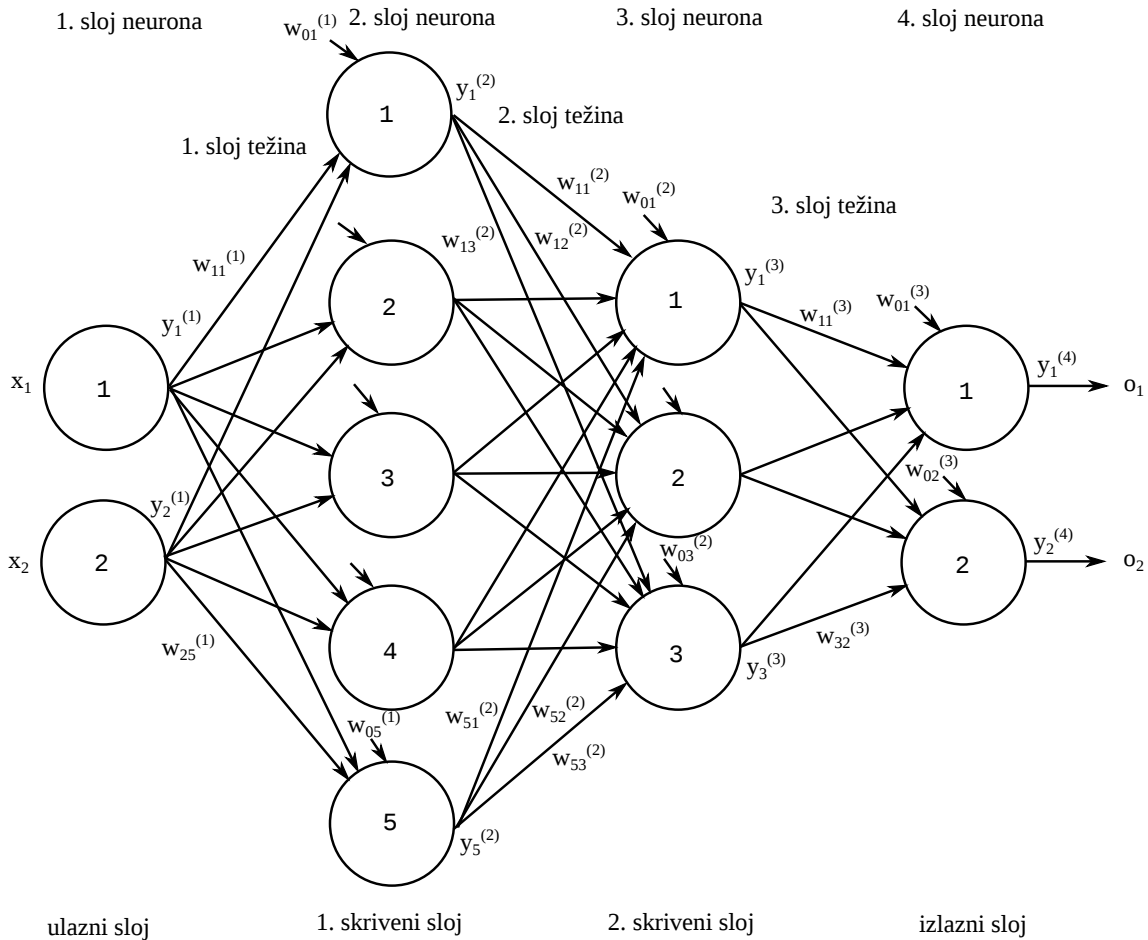
Što znače pojmovi unaprijedna mreža, slojevita mreža te potpuno povezana mreža, već smo upoznali. Sada ćemo se malo detaljnije pozabaviti strukturom same mreže i oznakama koje ćemo koristiti. Slika 3.1 prikazuje jednu četveroslojnu unaprijednu potpuno-povezanu umjetnu neuronsku mrežu arhitekture  $2 \times 5 \times 3 \times 2$ . Ovakva mreža u općem slučaju računa preslikavanje  $\vec{f}(x_1, x_2) = [f_1(x_1, x_2), f_2(x_1, x_2)] = [o_1, o_2]$  odnosno  $\vec{f}: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ .

Slojevite mreže sastoje se od određenog broja slojeva neurona (u primjeru na slici 3.1, broj slojeva neurona je 4). Prvi sloj je ulazni i sastoji se od neurona koji ne obavljaju nikakvu dodatnu funkciju već narinuti podatak nude neuronima drugog sloja. Posljednji sloj je ujedno i izlazni.

Unutar svakog sloja, neurone ćemo numerirati počev od 1. Izlaz  $i$ -tog neurona u  $k$ -tom sloju neurona označit ćemo s  $y_i^{(k)}$ . Izlazi neurona prvog sloja ujedno odgovaraju narinutim podacima pa vrijedi  $y_i^{(1)} = x_i$ . Izlaze neurona posljednjeg sloja još ćemo kraće označiti i s  $o_i$ , pa će tako vrijediti  $o_i = y_i^{(K)}$ , gdje smo velikim slovom  $K$  označili broj slojeva mreže (u ovom primjeru,  $K = 4$ ).

Slojeve težina također numeriramo od 1 na više. Prvi sloj težina čine veze između neurona prvog i drugog sloja neurona, drugi sloj težina čine veze između neurona u drugom i trećem sloju, itd. Težinu koja se nalazi između izlaza neurona  $i$  u sloju  $k$  i ulaza neurona  $j$  u sloju  $(k + 1)$  označit ćemo s  $w_{ij}^{(k)}$ . Svaki neuron uz te težine ima i težinu koja modelira pomak (engl. *bias*) i koju smo u prethodnom poglavlju označavali s  $w_0$ . U neuronskoj mreži, ta će težina  $i$ -tog neurona koji se nalazi u sloju  $(k + 1)$  neurona imati oznaku  $w_{0i}^{(k)}$ .

Oznake jednog dijela težina u skladu s ovom konvencijom prikazane su na slici 3.1. Oznake za sve težine nisu prikazane jer mreža ukupno ima  $5 \times 3 + 3 \times 6 + 2 \times 4 = 15 + 18 + 8 = 41$  težinu, pa bi slika bila vrlo nepregledna.



Slika 3.1: Skup uzoraka za učenje

### 3.1 Algoritam propagacije pogreške unatrag

Razmatranje ćemo ograničiti isključivo na unaprijedne neuronske mreže izgrađene od neurona sa sigmoidalnom prijenosnom funkcijom.

Neka je na raspolaganju skup uzoraka za učenje koji se sastoji od  $N$  uzoraka. Neka su uzorci specificirani ulazom i željenim izlazom koji mreža mora naučiti. Neka je dimenzionalnost ulaznog prostora  $N_i$  a izlaznog prostora  $N_o$ . Primjerice, uz  $N_i = 2$  i  $N_o = 3$ , jedan mogući uzorak bio bi  $(0.3, 0.5) \rightarrow (1.0, 0.0, 0.5)$ , drugim riječima, kada se na ulaz mreže dovede par brojeva  $(0.3, 0.5)$ , mreža na svoja tri izlaza mora generirati  $(1.0, 0.0, 0.5)$ . Skup uzoraka za učenje tada će biti oblika  $\{(x_{1,1}, \dots, x_{1,N_i}) \rightarrow (t_{1,1}, \dots, t_{1,N_o}), \dots, (x_{N,1}, \dots, x_{N,N_i}) \rightarrow (t_{N,1}, \dots, t_{N,N_o})\}$ .

Stavljanjem podatka iz  $s$ -tog uzorka na ulaz mreže, mreža će na izlazima generirati izlaz u skladu s naučenim težinama. Te ćemo izlaze označiti s  $(o_{s,1}, \dots, o_{s,N_o})$  gdje prvi broj u indeksu označava da se radi o izlazu dobivenom za  $s$ -ti uzorak. Što je mreža trebala dati na izlazima, zapisano je u uzorku za učenje:  $(t_{s,1}, \dots, t_{s,N_o})$ . Stoga ćemo kvadratno odstupanje (pogrešku) za  $s$ -ti uzorak izračunati kao sumu po svakom izlaznom neuronu od kvadrata razlike željene vrijednosti i vrijednosti koju je neuron izračunao, te ćemo sve podijeliti s 2:

$$E(s) = \frac{1}{2} \sum_{i=1}^{N_o} (t_{s,i} - o_{s,i})^2. \quad (3.1)$$

Ukupnu prosječnu pogrešku koju neuronska mreža radi nad čitavim skupom uzoraka za učenje označit ćemo s  $E$  i računati kao sumu pogrešaka definiranih izrazom (3.1) nad svim uzorcima, i

zatim sve podijeliti s brojem uzoraka:

$$E = \frac{1}{N} \sum_{s=1}^N E(s) = \frac{1}{2 \cdot N} \sum_{s=1}^N \sum_{i=1}^{N_o} (t_{s,i} - o_{s,i})^2. \quad (3.2)$$

Algoritam propagacije pogreške unatrag dobiva se primjenom algoritma gradijentnog spusta na problem minimizacije funkcije pogreške. Taj izvod u okviru ovog uvodnog kolegija nećemo raditi<sup>1</sup>. Međutim, upoznat ćemo se s rezultatom. Algoritam propagacije pogreške unatrag prikazan je u nastavku.

1. Sve težine neuronske mreže postavi na slučajne vrijednosti.
2. Ponavljaj dok nije zadovoljen uvjet zaustavljanja
  - (a) Za svaki uzorak  $s : (x_{s,1}, \dots, x_{s,N_i}) \rightarrow (t_{s,1}, \dots, t_{s,N_o})$  iz skupa uzoraka za učenje čini:
    - i. Postavi podatak  $(x_{s,1}, \dots, x_{s,N_i})$  na ulaz mreže.
    - ii. Izračunaj izlaze svih neurona u svim slojevima, od prvog prema posljednjem. Označimo izlaze neurona posljednjeg sloja s  $(o_{s,1}, \dots, o_{s,N_o})$ .
    - iii. Odredi pogrešku svakog od neurona *izlaznog* sloja. Pogrešku  $i$ -tog izlaznog neurona označit ćemo s  $\delta_i^K$  i računati prema izrazu:

$$\delta_i^K = o_{s,i} \cdot (1 - o_{s,i}) \cdot (t_{s,i} - o_{s,i}).$$

gdje je  $(t_{s,i} - o_{s,i})$  učinjeno odstupanje između željene vrijednosti i dobivene vrijednosti a član  $o_{s,i} \cdot (1 - o_{s,i})$  predstavlja derivaciju prijenosne funkcije tog neurona (prisjetite se, prijenosna funkcija je sigmoidalna i njezinu derivaciju smo već izveli i pokazali da je ovog oblika).

- iv. Potom se vraćaj sloj po sloj prema početku mreže. Za  $i$ -ti neuron koji se nalazi u  $k$ -tom sloju (gledamo samo skrivene slojeve), pogrešku neurona računamo kao težinsku sumu pogrešaka neurona kojima on šalje svoj izlaz, pomnoženu derivacijom prijenosne funkcije tog neurona:

$$\delta_i^{(k)} = y_i^{(k)} \cdot (1 - y_i^{(k)}) \cdot \sum_{d \in \text{Downstream}} w_{i,d} \cdot \delta_d^{(k+1)}$$

- v. Napravi korekciju svih težina. Težinu  $w_{i,j}^{(k)}$  korigiraj prema izrazu:

$$w_{i,j}^{(k)} \leftarrow w_{i,j}^{(k)} + \eta \cdot y_i^{(k)} \cdot \delta_j^{(k+1)}$$

Prisjetite se, ako je  $i = 0$  (gledamo težinu koja određuje pomak), pravimo se da je na njezin ulaz konstantno dovedena vrijednost 1 pa je prethodni izraz izravno iskoristiv. Možemo napisati i skraćeni izraz koji se dobije kada tu vrijednost 1 uvrstimo, pa tako težine koje definiraju pomak možemo korigirati prema izrazu:

$$w_{0,j}^{(k)} \leftarrow w_{0,j}^{(k)} + \eta \cdot \delta_j^{(k+1)}.$$

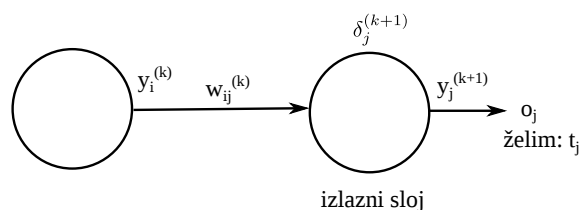
Naziv *algoritam propagacije pogreške unatrag* sada bi morao biti jasan. Najprije je potrebno odrediti izlaz mreže. Zatim računamo pogrešku neurona posljednjeg (izlaznog) sloja. Potom temeljem tih pogrešaka računamo pogreške neurona u predzadnjem sloju. Potom temeljem pogrešaka neurona u predzadnjem sloju računamo pogreške neurona u predpredzadnjem sloju. I tako redom sve do prvog skrivenog sloja - pogrešku propagiramo od kraja mreže prema ulazu mreže.

<sup>1</sup>Zainteresiranog čitatelja upućujemo na literaturu <http://java.zemris.fer.hr/nastava/nenr/knjiga-0.1.2013-08-12.pdf>

- R** Broj koji smo u izrazima označili s  $\eta$  naziva se *stopa učenja* i određuje iznos korekcije težine. Radi se o pozitivnom broju koji je često dosta malog iznosa (npr. 0.1, ali ovisno o situaciji moguće je da bude i bitno manji). Ako se koristi prevelika stopa učenja, algoritam će u svakom koraku raditi prevelike izmjene, pa se može dogoditi da postupak ne konvergira (ili da čak divergira).
- R** Pri postupku inicijalizacije težina, preporuka je da se postave na broj u okolici nule koji nije prevelik. Jedno od pravila koje daje dobre rezultate je za neuron koji dobiva ulaze od  $m$  drugih neurona svaku težinu postaviti na slučajno generirani broj iz intervala  $[-2.4/m, +2.4/m]$ .
- R** Ako se težine korigiraju nakon svakog predočenog uzorka, kako je opisano u prethodnom algoritmu, dobiva se inačica poznata kao *Stohastički algoritam propagacije pogreške unatrag* (engl. *Stochastic Backpropagation*). Ako se umjesto toga korekcija za svaku težinu akumulira kroz jednu čitavu epohu te se korekcija napravi tek na kraju epohe (u tom slučaju trebamo još onoliko pomoćnih varijabli - akumulatora - koliko imamo težina za pamćenje izračunatih korekcija kroz iteracije jedne epohe), dobivamo *klasični algoritam propagacije pogreške unatrag* (još ga zovemo i *grupni*; engl. *Batch Backpropagation*). Ova inačica brže konvergira ali uz veću opasnost zaglavljivanja u lošim lokalnim optimumima. Stoga uz dovoljno vremena stohastička inačica često daje bolje rezultate.

Postoji i rješenje koje je negdje na pola puta a naziva se *algoritam propagacije pogreške unatrag s mini-grupama* (engl. *Mini-batch Backpropagation*). Ideja je da se korekcije akumuliraju ali samo u trajanju od  $m < N$  uzoraka i tada provedu. Uz dobro odabranu veličinu grupe, ova implementacija može dati najbolje ponašanje.

Način izračuna pogreške uz prikladne sličice jednostavno je zapamtiti. Slučaj izlaznog neurona prikazan je na slici 3.2.



Slika 3.2: Izlazni neuron: definicija pogreške i korekcija težina

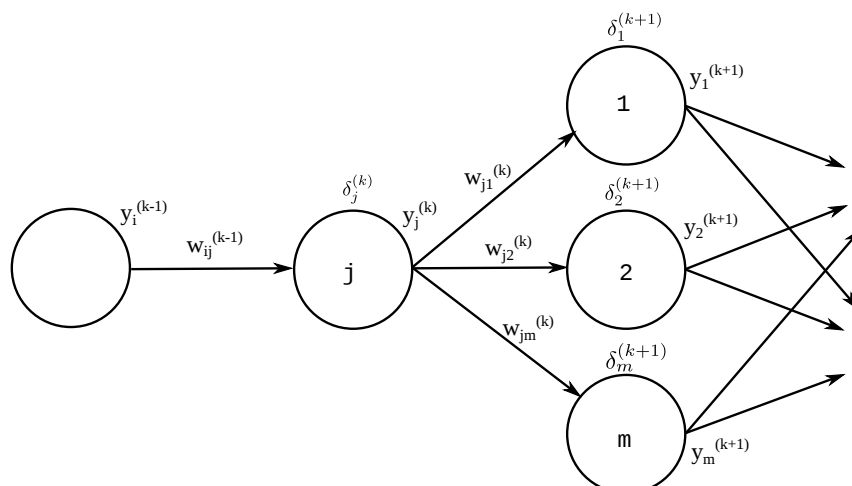
Pogreška je u tom slučaju jednaka umnošku derivacije prijenosne funkcije neurona, što daje:  $y_j^{(k+1)} \cdot (1 - y_j^{(k+1)})$  i stvarne pogreške koja je razlika između željene ( $t_j$ ) i generirane ( $o_j$ ) vrijednosti:  $(t_j - o_j)$ . Korekcija težine neurona radi se proporcionalno umnošku izlaza neurona ( $y_i^k$ ) koji težina dovodi u sljedeći sloj i pogreške neurona u tom sljedećem sloju:  $\delta_j^{(k+1)}$ : to su na slici dva neurona koje težina spaja.

Slika 3.3 ilustrira izračun pogreške u neuronu skrivenog sloja. Razmatramo izračun pogreške  $j$ -tog neurona u sloju  $k$  (na slici prikazan u sredini), koju označavamo s  $\delta_j^{(k)}$ .

Ta će pogreška biti jednaka umnošku derivacije prijenosne funkcije promatranog neurona i težinske sume pogrešaka svih neurona kojima promatrani neuron šalje svoj izlaz. Na prikazanoj slici to su neuroni u sloju  $(k + 1)$ , pa pogrešku računamo kao:

$$\delta_j^{(k)} = y_j^{(k)} \cdot (1 - y_j^{(k)}) \cdot (w_{j,1}^{(k)} \cdot \delta_1^{(k+1)} + \dots + w_{j,m}^{(k)} \cdot \delta_m^{(k+1)}).$$

Jednom kada je pogreška na taj način određena, korekcija težina tog neurona rade se na potpuno isti način kao i u prethodnom slučaju: proporcionalno izlazu neurona s kojeg težina polazi i pogrešci neurona na koji težina dolazi.



Slika 3.3: Izlazni neuron: definicija pogreške i korekcija težina

### 3.2 Primjer učenja umjetne neuronske mreže

Kako bismo lakše razumjeli specijalizaciju neurona koju obavlja algoritam propagacije pogreške unatrag, pogledat ćemo jednostavan primjer. Zadatak neuronske mreže izgrađene od neurona sa sigmoidalnom prijenosnom funkcijom je obaviti klasifikaciju uzoraka prema tablici 3.1.

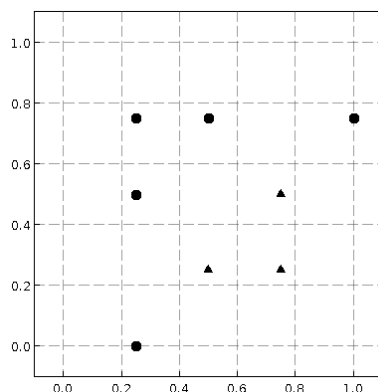
Tablica 3.1: Skup uzoraka za učenje

Uzorak $(x_1, x_2)$	Razred	Željeni izlaz mreže $(o_1)$
0.25, 0.5	$R_1$	0.0
0.5, 0.75	$R_1$	0.0
0.25, 0.75	$R_1$	0.0
0.25, 0.0	$R_1$	0.0
1.0, 0.75	$R_1$	0.0
0.5, 0.25	$R_2$	1.0
0.75, 0.5	$R_2$	1.0
0.75, 0.25	$R_2$	1.0

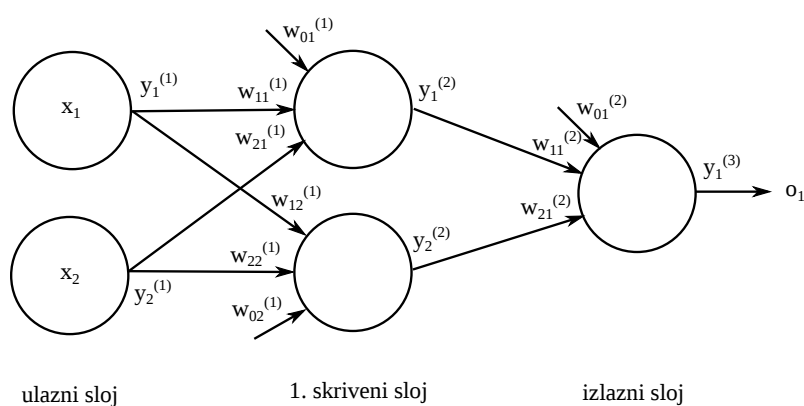
Ovaj skup prikazan je na slici 3.4. Kružićem su označeni uzorci koji pripadaju razredu  $R_1$  a trokutićem uzorci koji pripadaju razredu  $R_2$ . Kako je već sa slike vidljivo da uzorci nisu linearno razdvojni, neuronska mreža koja bi bila arhitekture  $2 \times 1$ , s obzirom da raspolaže samo jednim neuronom koji nešto doista i računa, ne bi mogla riješiti ovaj zadatak. Naime, unatoč tome što taj jedan neuron izračunatu težinsku sumu propušta kroz sigmoidalnu prijenosnu funkciju, kako je to jedino što se u mreži računa, decizijska granica koju takva mreža može naučiti je i dalje linearna.

Uzimajući prethodno u obzir, radit ćemo s umjetnom neuronskom mrežom koja raspolaže jednim skrivenim slojem i u njega ćemo staviti 2 neurona. Arhitektura takve mreže je  $2 \times 2 \times 1$ , i prikazana je na slici 3.5. Mreža raspolaže s ukupno 9 težina koje je moguće prilagođavati. Ulazima i izlazima mreže dali smo i dodatne oznake oblika  $x_i$  i  $o_i$ , pa vrijedi  $y_i^1 = x_i$  odnosno  $o_1 = y_1^{(3)}$ , što nam omogućava da funkciju mreže jasnije zapišemo kao

$$o_1 = f(x_1, x_2).$$

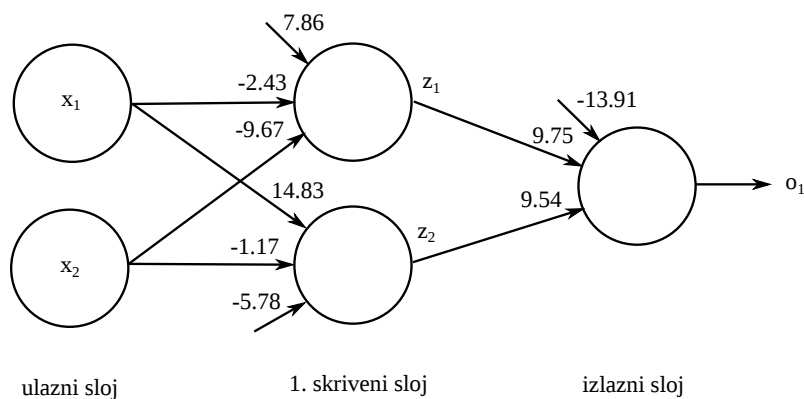


Slika 3.4: Skup uzoraka za učenje



Slika 3.5: Neuronska mreža koja se koristi za postupak klasifikacije

Proveden je postupak učenja algoritmom propagacije pogreške unatrag. Korištena je stohastička inačica uz uvjet zaustavljanja da srednje kvadratno odstupanje padne ispod  $10^{-3}$  ili da broj epoha dođe do 5000 (gdje je epoha u skladu s prethodnom definicijom jedno predočavanje svakog od uzoraka iz skupa uzoraka za učenje). Postupak učenja u ovom je slučaju stao jer je pogreška pala ispod zadane. Naučeni skup težina prikazan je na slici 3.6.

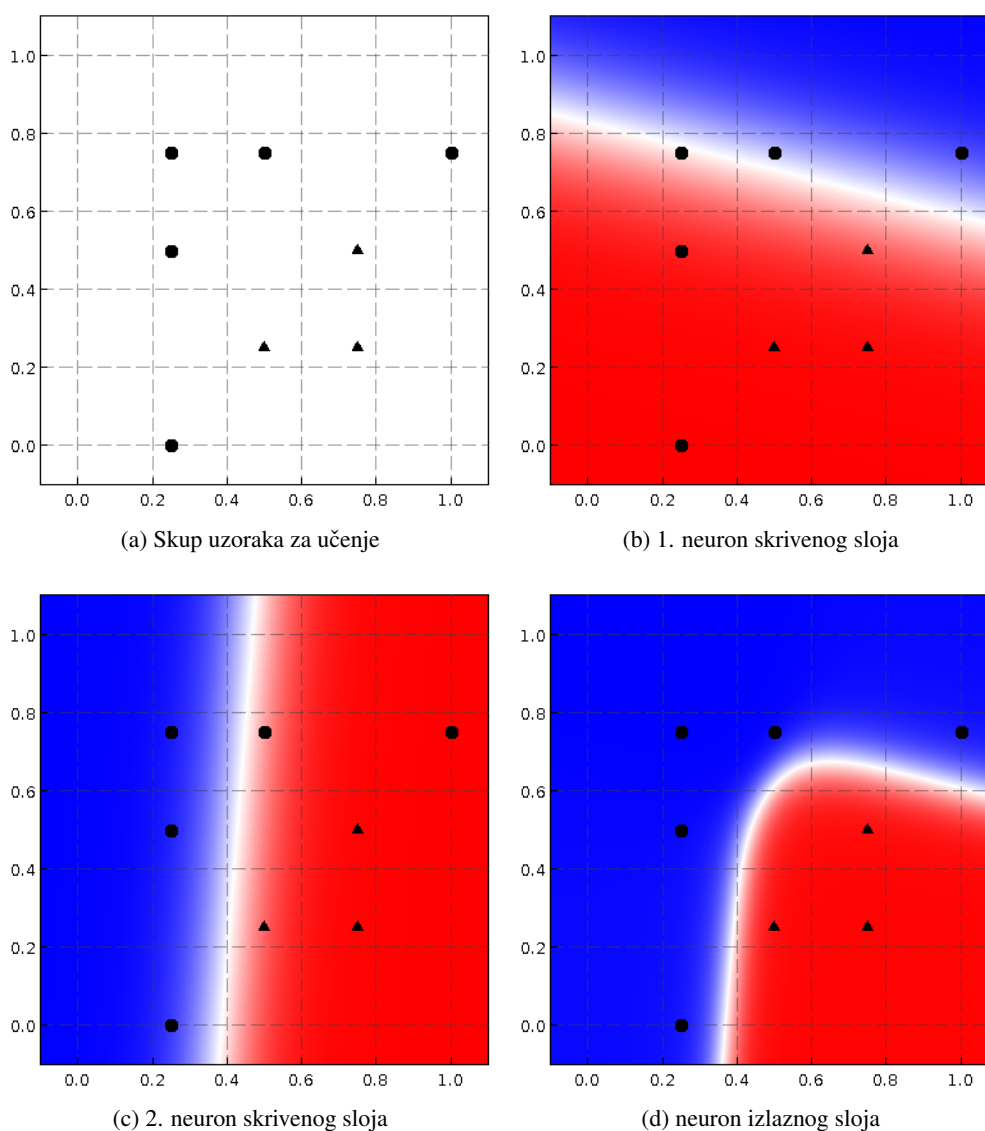


Slika 3.6: Naučena neuronska mreža

Pri uporabi mreže, klasifikaciju ćemo raditi na sljedeći način. Na ulaz mreže postavimo novi uzorak koji nas zanima. Potom izračunamo izlaz mreže. Ako je izlaz mreže manji od 0.5, uzorak smještamo u razred  $R_1$  a inače u razred  $R_2$ . Takav način u skladu je s posljednjim stupcem tablice

3.1 gdje je željeni izlaz mreže za sve uzorke koji pripadaju razredu  $R_1$  postavljen na 0.0 a za sve koji pripadaju razredu  $R_2$  na 1.0.

Sada možemo i malo detaljnije pogledati što je svaki od neurona naučio. Slike 3.7b, 3.7c i 3.7d prikazuju kako se mijenjaju izlazi prvog neurona u prvom skrivenom sloju (gornji), drugog neurona u prvom skrivenom sloju (donji) te izlaznog neurona, tim redosljedom. Slike su izrađene na sljedeći način. Generirani su ulazni uzorci  $(x_1, x_2)$  gdje su  $x_1$  i  $x_2$  birani iz intervala  $[-0.1, 1.1]$ . Takav uzorak doveden je na ulaz naučene mreže i određen je izlaz mreže (a time i izlazi svih neurona u skrivenom sloju). Za svaki je neuron zabilježen njegov izlaz uz narinuti ulaz mreže. Tri slike prikazuju izlaze za tri neurona mreže, pri čemu je izlaz vizualno kodiran bojom. Ako je izlaz bio 0, prikazan je crvenom bojom. Što se je više izlaz od nule približavao vrijednosti 0.5, to je crvena boja sve više prelazila u bijelu. Izlaz vrijednosti 0.5 prikazan je bijelom bojom. Što je izlaz više rastao od 0.5 prema 1, prikazivan je sve plavijom bojom. Mjesta gdje je izlaz bio upravo 1 prikazana su čistom plavom bojom.



Slika 3.7: Ilustracija specijalizacije neurona. Na apscisi su vrijednosti od  $x_1$ , na ordinati vrijednosti od  $x_2$ .

Proučimo izlaz gornjeg neurona u prvom skrivenom sloju (slika 3.7b). Taj je neuron od uzoraka razreda  $R_1$  naučio korektno klasificirati samo dva uzorka:  $(0.5, 0.75)$  i  $(1.0, 0.75)$ . Sve ostale smješta u razred  $R_2$  (crveno područje). Njegova decizijska granica određena je izrazom:

$$-9.67 \cdot x_2 - 2.43 \cdot x_1 + 7.86 = 0$$

što poprilici odgovara pravcu:

$$x_2 = -0.25 \cdot x_1 + 0.81.$$

To je pravac čiji je nagib lagano prema dolje (koeficijent smjera je negativan) a ordinatu siječe na visini  $x_2 \approx 0.81$ . Provjerite to na slici 3.7b.

Drugi neuron skrivenog sloja od svih uzoraka razreda  $R_1$  naučio je korektno klasificirati preostala tri uzorka:  $(0.25, 0.0)$ ,  $(0.25, 0.5)$  i  $(0.25, 0.75)$  dok sve ostale uzorke smješta u razred  $R_2$ . Njegova decizijska granica određena je izrazom:

$$-1.17 \cdot x_2 + 14.83 \cdot x_1 - 5.78 = 0$$

što poprilici odgovara pravcu:

$$x_2 = 12.67 \cdot x_1 - 4.94.$$

Radi se o pravcu koji raste vrlo strmo (koeficijent smjera mu je  $\approx 12.67$ ) i koji apscisu siječe na  $x_1 \approx 0.39$ . Provjerite to na slici 3.7c.

Svaki od ova dva neurona *specijalizirao se* za jedan dio ulaznog prostora u kojem dobro obavlja klasifikaciju. U kontekstu neuronskih mreža reći ćemo je svaki od tih neurona na temelju podataka iz ulaznog prostora izračunao po jednu novu značajku. Označimo li izlaz prvog neurona u skrivenom sloju sa  $z_1$  te drugog neurona sa  $z_2$ , prvi skriveni sloj mreže ulazni prostor  $(x_1, x_2)$  prevodi u novi prostor značajki  $(z_1, z_2)$ . Uočite da ovaj novi prostor može biti i veće dimenzionalnosti od ulaznog prostora: da smo u skrivenom sloju imali tri neurona, ovaj novi prostor značajki bio bi trodimenzijski.

Zadaća sljedećeg sloja (koji je u našem slučaju i posljednji, izlazni) jest kombiniranjem ovih značajki niske razine složenosti odrediti nove složenije značajke (ako sloj nije izlazni) odnosno obaviti klasifikaciju (ako mrežu učimo klasificirati).

U našem konkretnom slučaju, sljedeći sloj sadrži samo jedan neuron i on na ulaze dobiva značajke  $(z_1, z_2)$  te temeljem njih određuje novu decizijsku granicu. Mreža prikazana na slici 3.6 naučila je sljedeću decizijsku granicu:

$$9.54 \cdot z_2 + 9.75 \cdot z_1 - 13.91 = 0$$

što je u prostoru značajki  $(z_1, z_2)$  pravac:

$$z_2 = -1.02 \cdot z_1 + 1.46.$$

U tom prostoru, to je pravac čiji je nagib poprilici 45 stupnjeva prema dolje, i on nije prikazan na slici 3.7d - ta slika prikazuje kako se mijenja izlaz izlaznog neurona kada mijenjamo ulaz mreže. Stoga analizu treba napraviti samo malo opreznije.

Pogledajmo najprije sliku 3.7c. Uočite da je na desnoj polovici te slike izlaz drugog neurona (koji računa značajku  $z_2$ ) vrlo blizu vrijednosti 1 (jarka crvena boja na slici). Uvrstimo li tu činjenicu u izraz za decizijsku granicu izlaznog neurona, imamo:

$$9.54 \cdot 1 + 9.75 \cdot z_1 - 13.91 = 0$$

$$9.75 \cdot z_1 = -9.54 + 13.91$$

$$9.75 \cdot z_1 = 4.37$$

$$z_1 = 0.45$$



Dakle, tako dugo dok je značajka  $z_2$  bliska 1, decizijska granica bit će na mjestima na kojima je  $z_1 = 0.45$ . Ali kako je  $z_1$  zapravo izlaz gornjeg neurona, možemo to dalje raspisati:

$$\begin{aligned}\frac{1}{1 + e^{-net_1}} &= 0.45 \\ 1 &= 0.45 + 0.45e^{-net_1} \\ 0.55 &= 0.45e^{-net_1} \\ 1.222 &= e^{-net_1} \\ net_1 &= -0.2 \\ -9.67 \cdot x_2 - 2.43 \cdot x_1 + 7.86 &= -0.2 \\ -9.67 \cdot x_2 - 2.43 \cdot x_1 + 8.06 &= 0\end{aligned}$$

čime smo dobili da je u ulaznom prostoru decizijska granica praktički ona koju određuje prvi neuron (uz zanemarivu translaciju), i to se lijepo vidi na slici 3.7d: u desnom dijelu slike, poprili za  $x_1 > 0.6$ , decizijska granica se poklapa s onom prikazanom na slici 3.7b.

Pogledajmo sada situaciju u kojoj je  $z_1$  postavljen na 1: na slici 3.7b to je gotovo čitava donja polovica prostora (gdje je  $x_2 < 0.5$ ). Uvrštavanjem u izraz za decizijsku granicu izlaznog neurona imamo:

$$\begin{aligned}9.54 \cdot z_2 + 9.75 \cdot 1 - 13.91 &= 0 \\ 9.54 \cdot z_2 &= -9.75 + 13.91 \\ 9.54 \cdot z_2 &= 4.16 \\ z_2 &= 0.436\end{aligned}$$

No kako je  $z_2$  izlaz drugog neurona skrivenog sloja, lagano možemo uvrštavanjem provjeriti što to znači u ulaznom prostoru. Evo izračuna.

$$\begin{aligned}\frac{1}{1 + e^{-net_2}} &= 0.436 \\ 1 &= 0.436 + 0.436e^{-net_2} \\ 0.564 &= 0.436e^{-net_2} \\ 1.294 &= e^{-net_2} \\ net_2 &= -0.257 \\ -1.17 \cdot x_2 + 14.83 \cdot x_1 - 5.78 &= -0.257 \\ -1.17 \cdot x_2 + 14.83 \cdot x_1 - 5.523 &= 0\end{aligned}$$

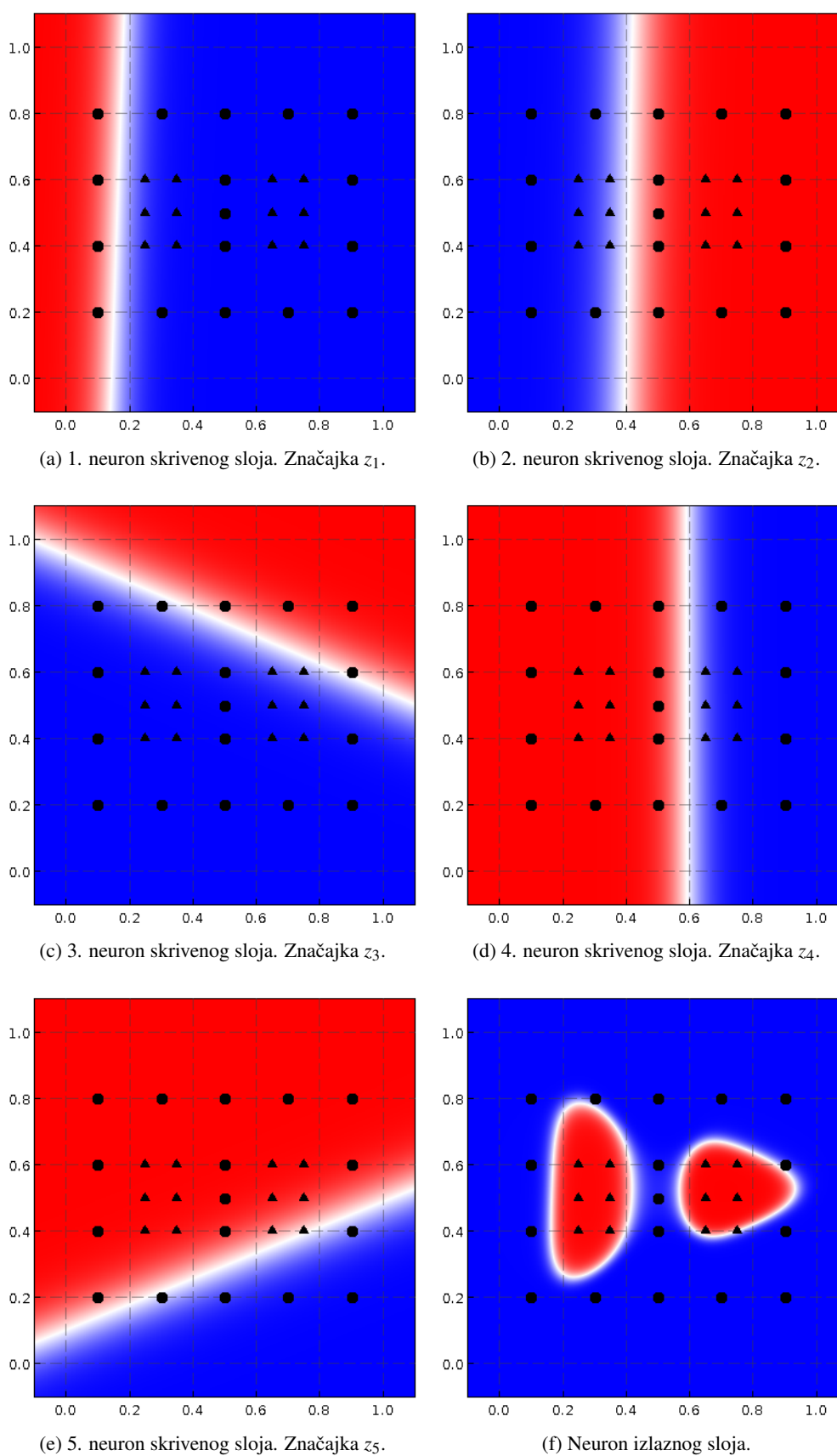
Ovime smo dobili da je sada u ulaznom prostoru decizijska granica praktički ona koju određuje drugi neuron (uz zanemarivu translaciju), što se također lijepo vidi na slici 3.7d: u donjoj polovici slike, poprili za  $x_2 < 0.5$ , decizijska granica se poklapa s onom prikazanom na slici 3.7c.

Kako se od točke (0.5,0.5) iz ulaznog prostora pomičemo prema gore i prema lijevo,  $z_1$  i  $z_2$  se smanjuju, a kako za decizijsku granicu vrijedi da se nalazi na mjestu gdje je težinska suma  $z_1$  i  $z_2$  jednaka 0 (uz prethodno izračunate težine), decizijska granica se polako pretapa iz one koju određuje prvi neuron skrivenog sloja u onu koju određuje drugi neuron skrivenog sloja.

### 3.3 Drugi primjer

Pogledajmo još jedan, nešto složeniji primjer. Odabran je skup uzoraka za učenje koji uzorke jednog razreda ugrađuje u dva razdvojena područja drugog razreda. Trenirana mreža je arhitekture  $2 \times 5 \times 1$ . Što je koji neuron naučio, prikazano je na slici 3.8.

S obzirom da je u skrivenom sloju dostupno samo 5 neurona, mreža je morala pronaći 5 značajki na temelju kojih može odraditi korektnu klasifikaciju. Pogledajte slike 3.8a do 3.8e koje prikazuju naučene značajke: svaki od neurona pronašao je jednu smislenu granicu kojom razdvaja jedan dio uzoraka. Izlazni neuron potom te granice kombinira u konačnu granicu. U granici lijevog skupa uzoraka možemo prepoznati elemente značajki  $z_1, z_2, z_3$  i  $z_5$ . Granica desnog skupa dobivena je kombiniranjem značajke  $z_4$  (lijevi rub) te  $z_3$  i  $z_5$  - stoga desni rub izgleda gotovo kao trokut.



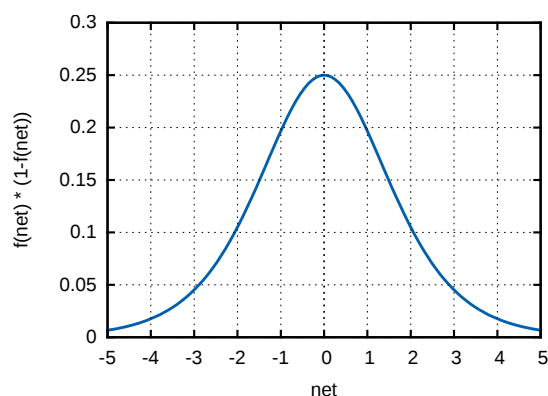
Slika 3.8: Ilustracija specijalizacije neurona uz drugačiji skup uzoraka za učenje. Na apscisi su vrijednosti od  $x_1$ , na ordinati vrijednosti od  $x_2$ .



## 4. Daljnji razvoj

Mreže temeljene na sigmoidalnim prijenosnih funkcijama imaju problem poznat kao *problem nestajućeg gradijenta* (engl. *vanishing gradient problem*). Pojam dolazi od načina na koji je izveden algoritam propagacije unatrag, a koji se temelji na izračunu gradijenta funkcije pogreške. Rezultat toga je član oblika "izlaz puta jedan minus izlaz" koji množi pogrešku svakog od neurona (prisjetite se izraza po kojima se računaju pogreške), a iznos te pogreške proporcionalno modificira težine neurona.

Uzimajući u obzir da je izlaz sigmoidalnog neurona funkcija od težinske sume, ovisnost umnoška  $f(net) \cdot (1 - f(net))$  o  $net$  prikazana je na slici 4.1.




Slika 4.1: Ovisnost derivacije prijenosne funkcije o  $net$

Kada je  $net$  blizak nuli, ovaj umnožak ima maksimalnu vrijednost i iznosi  $1/4$ . Što  $net$  postaje veći (u pozitivnom ili negativnom smjeru), to ovaj umnožak više teži k nuli. kod neuronske mreže koja ima vrlo malo slojeva, ovo nije (prevelik) problem. Međutim, uslijed ovog prigušivanja, u mrežama koje imaju više slojeva, do ranijih slojeva gotovo pa ne dolazi nikakva korekcija (jer je iznos konstantno eksponencijalno prigušivan kako se pogreška vraćala od izlaznog sloja unatrag). Zbog toga pak algoritam propagacije pogreške unatrag na takvim mrežama postaje gotovo pa

neupotrebiv.

Ova činjenica uzrokovala je od sredine devedesetih godina ponovno zastoj u razvoju neuronskih mreža. Isti je nastavljen tek kroz posljednjih nekoliko godina uvođenjem novih tipova prijenosnih funkcija, poput *ReLU* i *max-out* te novim tehnikama učenja poput tehnike *dropout* koja su zajedno konačno omogućile treniranje neuronskih mreža s većim brojem slojeva koje su postale poznate kao *duboke neuronske mreže*.

U obradi slike, posebno se dobrim pokazao spoj dubokih arhitektura i konvolucijske inačice neuronskih mreža koje danas postižu zavidne rezultate.



## Bibliografija

Knjige

Članci





# Kazalo

## A

Algoritam propagacije pogreške unatrag . . .	26
algoritam učenja perceptrona . . . . .	21
ALVINN . . . . .	8
arhitekture	
potpuno povezana . . . . .	14
slojevita . . . . .	14
unaprijedna . . . . .	14

## B

biološki neuron . . . . .	5
---------------------------	---

## C

cikličke veze . . . . .	16
-------------------------	----

## D

decizijska granica . . . . .	18
------------------------------	----

## E

epoha . . . . .	9
-----------------	---

## G

grupno učenje . . . . .	9
-------------------------	---

## I

iteracija . . . . .	9
---------------------	---

## K

klasifikacija	
decizijska granica . . . . .	18
konektivistički pristup . . . . .	6

## L

linearno nerazdvojni razredi . . . . .	20
--	----

## N

neuron	
biološki . . . . .	5
umjetni . . . . .	6
neuronska mreža . . . . .	6
arhitektura . . . . .	14
pretreniranost . . . . .	11
primjene . . . . .	8
učenje . . . . .	9

## P

pojedinačno učenje . . . . .	9
pretreniranost . . . . .	11
prijenosna funkcija . . . . .	12
funkcija identiteta . . . . .	12

funkcija skoka . . . . .	12
funkcija tangens hiperbolni . . . . .	12
ispravljena linearna funkcija . . . . .	12
max-out . . . . .	12
ReLU . . . . .	12
sigmoidalna funkcija . . . . .	12
slabo-ispravljena linearna funkcija . . . . .	12
problem ciklusa . . . . .	16

## S

simbolički pristup . . . . .	5
------------------------------	---

## T

TLU-perceptron . . . . .	17
--------------------------	----

## U

### učenje

bez učitelja . . . . .	9
grupno . . . . .	9
podržano . . . . .	9
pojedinačno . . . . .	9
pretreniranost . . . . .	11
s učiteljem . . . . .	9
skup za provjeru . . . . .	10
skup za testiranje . . . . .	10
skup za učenje . . . . .	10
umjetna neuronska mreža . . . . .	6
umjetni neuron . . . . .	6