Umjetna inteligencija

11. Umjetne neuronske mreže

Bojana Dalbelo Bašić, Marko Čupić, Jan Šnajder

Sveučilište u Zagrebu
Fakultet elektrotehike i računarstva

Ak. god. 2019./2020.

v3.10

Motivacija

- Automatiziranu obradu podataka danas uglavnom rade digitalna računala.
- Ipak, još je uvijek daleko više podataka čija obrada nije automatizirana. Te podatke obrađuju živčani sustavi živih organizama!
- Razvoj jedne grane računarstva motiviran je razmatranjem prevladavajućeg načina obrade podataka u svijetu u kojem živimo.
- Tražimo drugačiji koncept obrade podataka koji bi bio sličniji funkcioniranju biološkog mozga.
- A.I.-sustav koji uspješno oponaša rad mozga bio bi inteligentan.

Sadržaj

Sadržaj

Motivacija razvoja neuro-računarstva

- Poznato je da se mozak sastoji od velikog broja neurona koji rade paralelno.
- Znamo sljedeće:
  - Postoji više od 100 vrsta različitih neurona
  - Svaka vrsta radi vrlo jednostavnu obradu
  - Vrijeme obrade u neuronu: ≈ 2 milisekunde
  - Broj neurona u ljudskom mozgu: 10^{11}
  - Svaki neuron u prosjeku dobiva informacije od 10^3 do 10^4 drugih neurona
  - Informacije se obrađuju i serijski i paralelno
  - Informacije su analogne
  - Obrada je tolerantna na pogreške

Motivacija razvoja neuro-računarstva

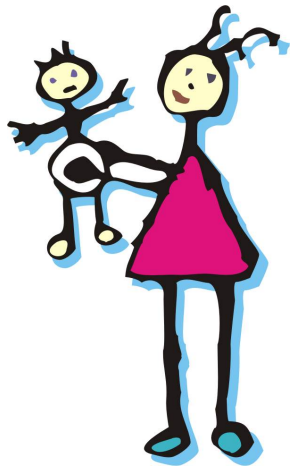
- Želimo izgraditi računalni sustav koji bi podatke obrađivao na jednak način!
- Nova paradigma:
  umjetne neuronske mreže (engl. *artificial neural networks*)
- Područje koje se bavi tim aspektom obrade: neuro-računarstvo
  - Jedna od grana računarstva iz skupine mekog računarstva (engl. *soft-computing*)

Pravci razvoja umjetne inteligencije

- Od prvih dana razvoja umjetne inteligencije (rane '50) postoje dva pristupa razvoju inteligentnih sustava:
  - Simbolički pristup: znanje iz neke domene nastoji se obuhvatiti skupom atomičkih semantičkih objekata (simbola) i zatim manipulirati tim simbolima pomoću algoritamskih pravila
  - Konektivistički pristup: temelji se na izgradnji sustava arhitekture slične arhitekturi mozga koji, umjesto da ga se programira, **uči samostalno na temelju iskustva**
- Simbolički pristup je dobar u mnogim područjima (osobito isplativ postao je razvojem ekspertnih sustava), ali nije ispunio rana ekstravagantna obećanja.
- Neuspjeh leži u pogrešnoj pretpostavci da je svako znanje moguće formalizirati i da je mozak stroj koji podatke obrađuje formalnim pravilima.

Konektivistički pristup

- Mnogi su svakodnevni zadaci previše složeni za simboličko predočavanje, npr. raspoznavanje uzoraka...
- Majku možemo prepoznati u 0.1 s
- Neuroni u mozgu pale poprilici svake milisekunde
- Slijedi da u tom vremenu u seriji može paliti najviše 100 neurona
- Očigledno paralelna obrada!



Umjetna neuronska mreža - definicija

U širem smislu: umjetna replika ljudskog mozga kojom se nastoji simulirati postupak učenja i obrade podataka.

Umjetna neuronska mreža

Umjetna neuronska mreža je skup međusobno povezanih jednostavnih procesnih elemenata (neurona) čija se funkcionalnost temelji na biološkom neuronu i koji služe distribuiranoj paralelnoj obradi podataka.

- Omogućavaju robusnu obradu podataka.
- Može ih se koristiti za probleme klasifikacije te za probleme regresije.
- Sposobne su učiti iz podataka.

Učenje umjetne neuronske mreže

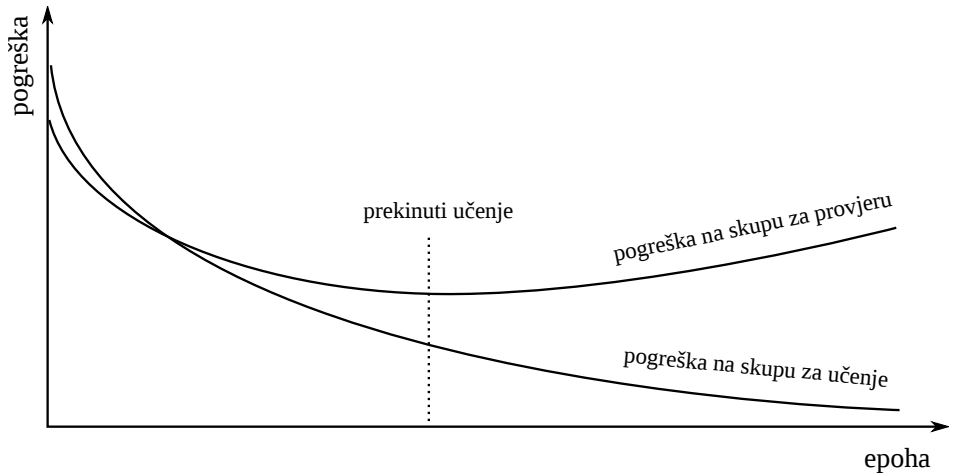
- Dvije faze rada s ANN:
  1. Faza učenja (treniranja) i
  2. Faza obrade podataka (iskorištavanja, eksploatacije).
- Učenje je iterativan postupak predočavanja ulaznih primjera (uzoraka, iskustva) i eventualno očekivanog izlaza pri čemu dolazi do postupnog prilagođavanja težina veza između neurona.
- Jedno predočavanje svih uzoraka naziva se epohom
- Razlikujemo:
  1. Pojedinačno učenje (engl. *on-line*): učenje se događa nakon svakog predočenog uzorka
  2. Učenje s minigrupama (engl. *mini-batches*): učenje se događa nakon više predočenih uzoraka
  3. Grupno učenje (engl. *batch*): učenje se događa tek nakon svih predočenih uzoraka
- Znanje o izlazu kao funkciji ulaza pohranjeno je implicitno u težinama veza neurona

Učenje umjetne neuronske mreže

- Učenje neuronskih mreža može biti:
  1. nadzirano (učenje s učiteljem)
  2. nenadzirano (učenje bez učitelja)
  3. podržano
- Skup primjera za učenje često dijelimo na:
  1. Skup za učenje (engl. *training set*):
     npr. 70% dostupnih uzoraka,
     služi za iterativno podešavanje težina
  2. Skup za provjeru (engl. *validation set*):
     npr. 15% dostupnih uzoraka,
     provjeravamo generalizacijska svojstva mreže
  3. Skup za testiranje (engl. *test set*):
     npr. 15% dostupnih uzoraka,
     konačna provjera mreže, usporedba s drugim modelima

Učenje umjetne neuronske mreže

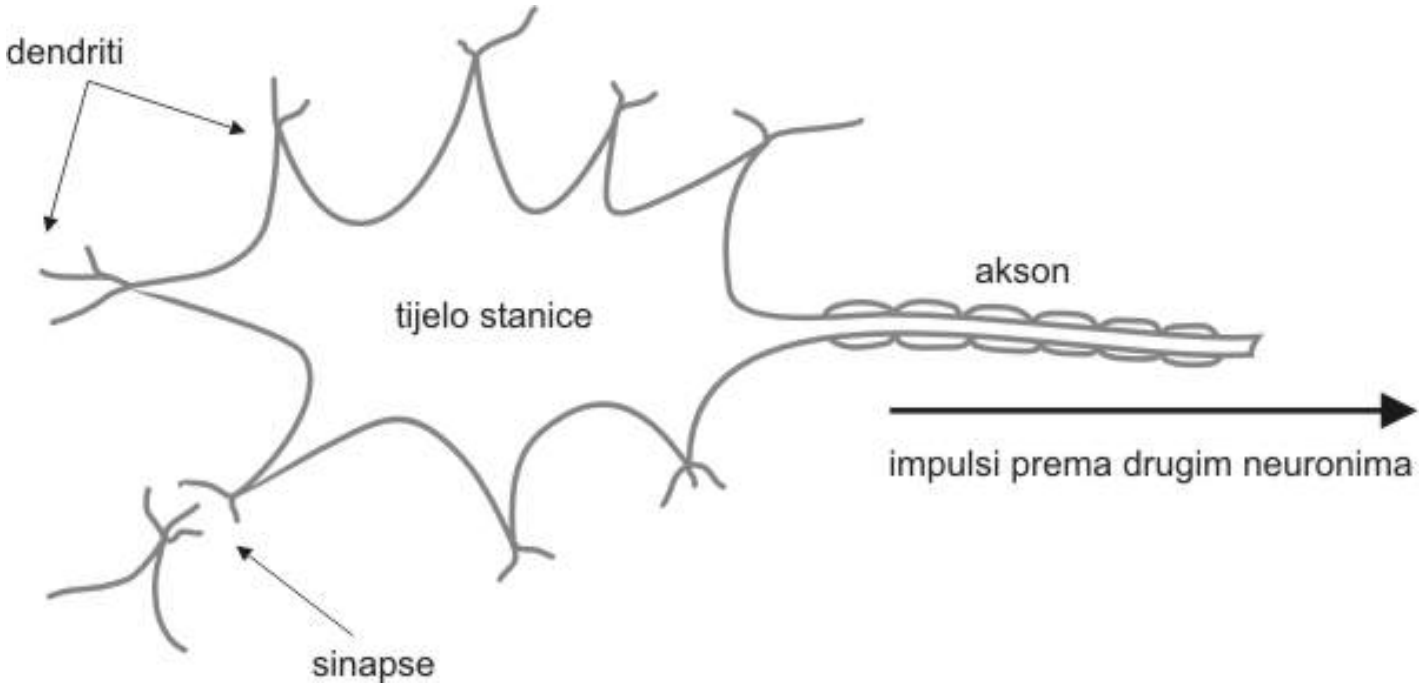
- Učenje se provodi na skupu za učenje i prati se točnost obrade podataka (uvodi se mjera pogreške).
- Točnost se prati i na skupu na provjeru (ali mreža nad tim skupom ne uči).
- Pretreniranost: mreža gubi poželjno svojstvo generalizacije i postaje stručnjak za podatke iz skupa za učenje (štreber).
- Učenje prekidamo kada pogreška počne rasti na skupu za provjeru.



Sadržaj

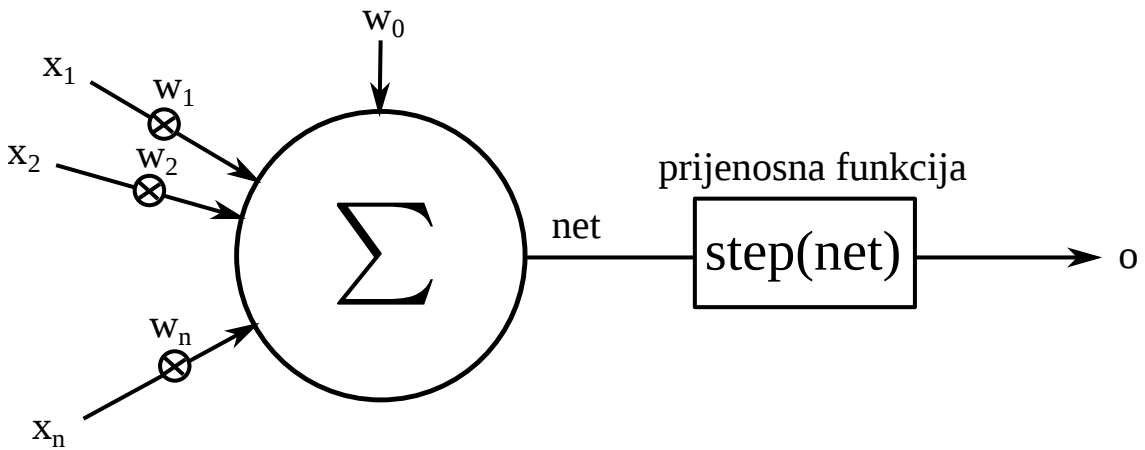
Biološki neuron

- Biološki neuron sastoji se od tijela (some), dendrita, aksona te završnih članaka.
- U ljudskom mozgu svaki je neuron u prosjeku povezan s 1000 do 10000 drugih neurona



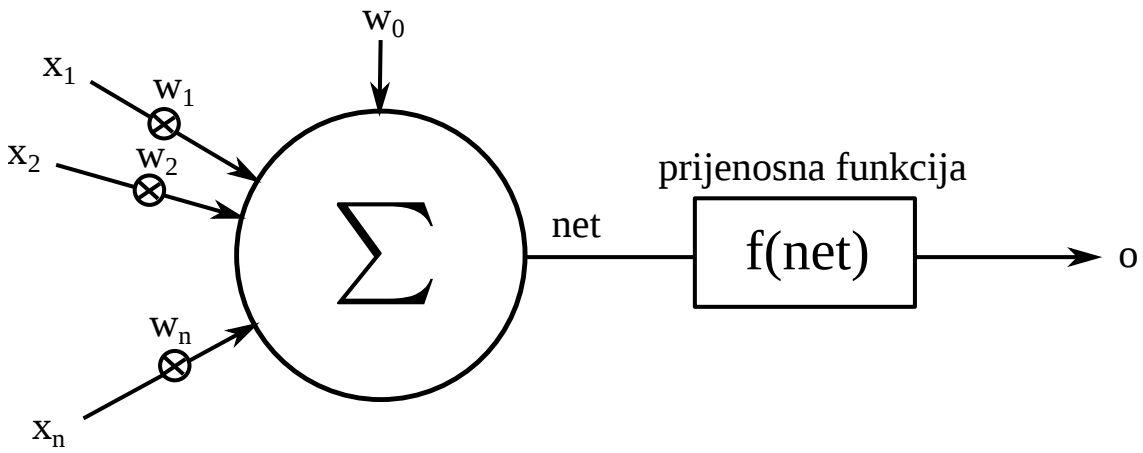
Umjetni neuron

- McCulloch-Pitts definiraju jednostavan model biološkog neurona (1943.): TLU-perceptron (engl. *Threshold Logic Unit*)
  - Vrijednost sa svakog ulaza x_i množi se osjetljivošću tog ulaza w_i i akumulira u tijelu.
  - Ukupnoj sumi dodaje se pomak w_0 (još se označava i b od engl. *bias*).
    Time je definirana akumulirana vrijednost $net = \left(\sum_{i=1}^{n} x_i \cdot w_i\right) + w_0$.
  - Ta se vrijednost propušta kroz prijenosnu funkciju čime nastaje izlazna vrijednost: $o = step(net)$.



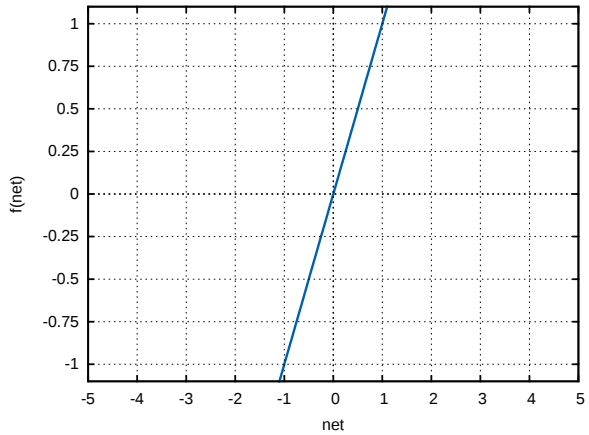
Umjetni neuron

- Općenito, model umjetnog neurona omogućava da se akumulirana vrijednost propusti kroz neku prijenosnu funkciju f

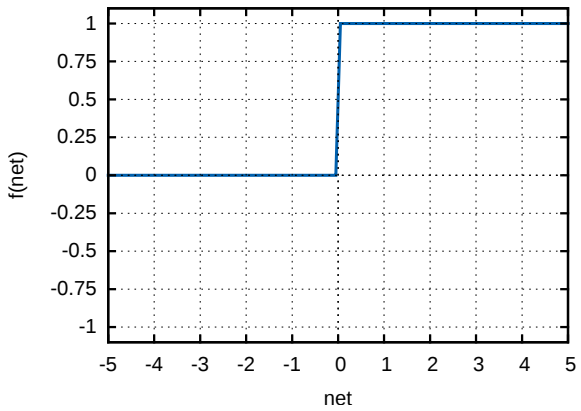


- Često korištene prijenosne funkcije su:
  - Identitet (ADALINE-neuron)
  - Funkcija skoka (TLU-perceptron)
  - Sigmoidalna funkcija (sigmoidalni neuron)
  - Tangens hiperbolni
  - Zglobnica (*Rectified Linear Unit*, ReLU)
  - Propusna zglobnica (*Leaky Rectified Linear Unit*, LReLU)

Prijenosne funkcije



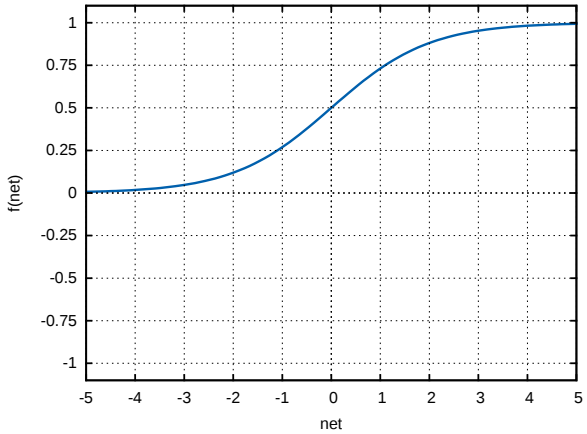
(a) Funkcija identiteta



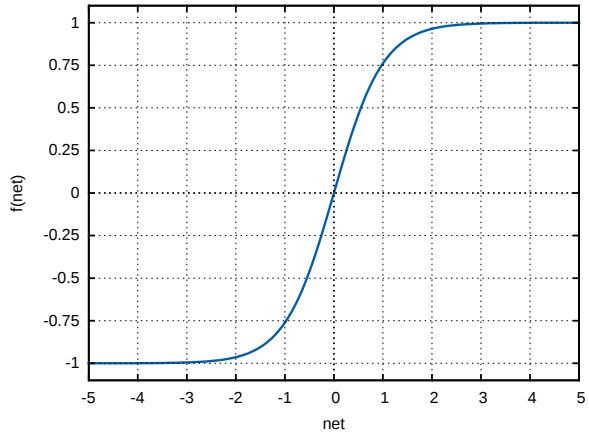
(b) Funkcija skoka

- Identitet
  $f(net) = net$
- Funkcija skoka
  $$f(net) = step(net) = \begin{cases} 0 & net < 0 \\ 1 & \text{inače} \end{cases}$$

Prijenosne funkcije



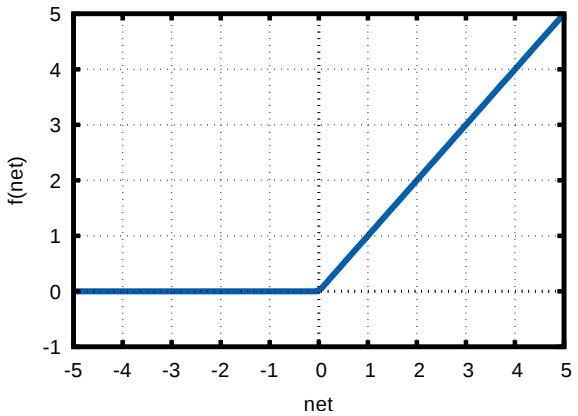
(c) Sigmoidalna funkcija



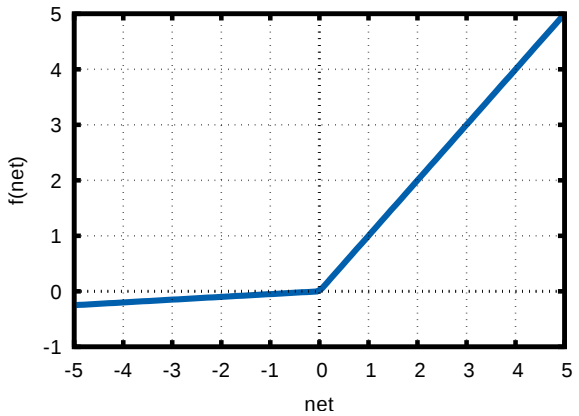
(d) Tangens hiperbolni

- Sigmoidalna funkcija
  $f(net) = sigm(net) = \frac{1}{1+e^{-net}}$
- Tangens hiperbolni
  $f(net) = tanh(net) = \frac{2}{1+e^{-2 \cdot net}} - 1 = 2 \cdot sigm(2x) - 1$

Prijenosne funkcije



(e) Zglobnica



(f) Propusna zglobnica uz $\alpha = 0.05$

- Zglobnica
  $f(net) = \max(0, net)$
- Propusna zglobnica
  $$f(net) = \begin{cases} net & net >= 0 \\ \alpha \cdot net & \text{inače} \end{cases}$$

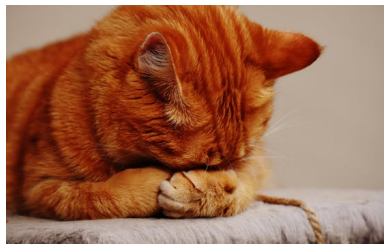
Klasifikacija

- Klasifikacija je postupak pridjeljivanja oznaka (labela) uzorcima na temelju značajki uzoraka (primjerice, boja, oblik, težina, ...).
- Želimo izgraditi klasifikatorski sustav koji na temelju uzoraka iz skupa za učenje može korektno klasificirati nove uzorke
- Postoje li samo dva razreda, govorimo o binarnoj klasifikaciji.
  - Možemo koristiti sustav koji će imati samo jedan izlazu i na njemu dva jasno razlučiva stanja (primjerice: 0 i 1; alternativno -1 i 1) čime uzorak pridjeljuje jednom ili drugom razredu
- Imamo li više razreda (a svaki uzorak pripada samo jednom od njih), uobičajeno je koristiti jednojediničnо kodiranje: izlaza ima koliko i razreda pri čemu je i-ti izlaz jednak 1 ako uzorak pripada i-tom razredu, inače je 0.

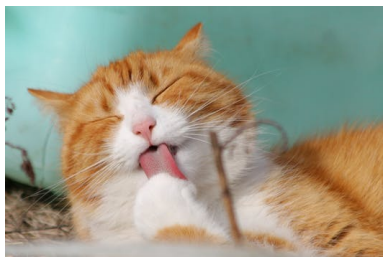
Binarna klasifikacija

Pretpostavimo da gradimo binarni klasifikator koji slike razvrstava u slike pasa i slike mačaka. Neka smo razred "psi" kodirali s 0 a razred "mačke" s 1. Tada bismo za slike u nastavku očekivali vrijednost izlaza koja je prikazana ispod svake slike.

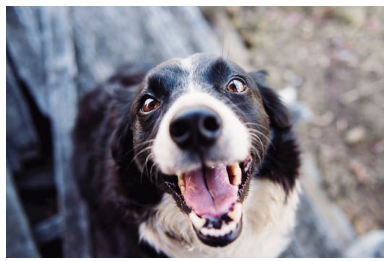
(a) 0



(b) 1



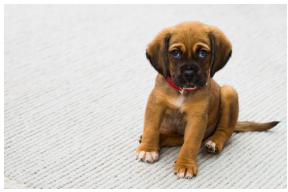
(c) 1



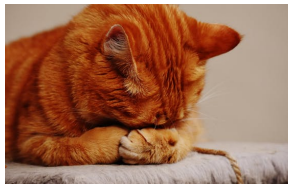
(d) 0

Klasifikacija u više razreda

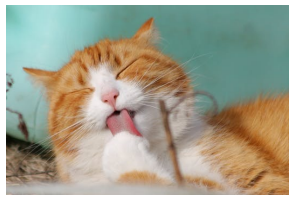
Razmotrimo sada sustav koji bi slike razvrstavo u jedan od tri razreda: slike pasa, slike mačaka te slike papiga. Mogli bismo tražiti da izlaz sustava bude kodna riječ iz jednojediničnog koda: 100 - ako se radi o psu, 010 ako se radi o mački te 001 ako se radi o papigi.



(a) 100

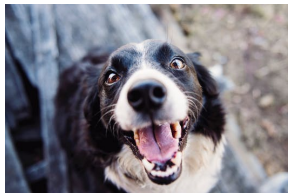


(b) 010



(c) 010

(d) 001



(e) 100

(f) 001

Binarna klasifikacija: primjer

Tijekom postupka zapošljavanja u tvrtki ACME:

- svaki kandidat prolazi kroz dva nezavisna kruga ispitivanja te za svako dobiva ocjenu između 1 i 5
- na temelju tih dviju ocjena, voditelj službe za zapošljavanje daje pozitivno ili negativno mišljenje
- kandidati koji su dobili pozitivno mišljenje prosljeđuju se na daljnje razmatranje

U okviru racionalizacije poslovanja, dio poslova koje su radili ljudi želi se automatizirati - konkretno, razmatra se izgradnja računalnog sustava koji bi davao pozitivno ili negativno mišljenje.
Kako bi se to omogućilo, prikupljeni su podatci o prethodnim odlukama voditelja službe.

Binarna klasifikacija: primjer

- Označimo s x_1 ocjenu koju je kandidat dobio u prvom krugu te s x_2 ocjenu koju je dobio u drugom krugu
- U arhivama tvrtke pronađeni su dokumenti za četiri kandidata i donesene odluke:
  - $(x_2, x_1) = (2, 5)$: mišljenje je bilo pozitivno
  - $(x_2, x_1) = (5, 2)$: mišljenje je bilo pozitivno
  - $(x_2, x_1) = (1, 5)$: mišljenje je bilo negativno
  - $(x_2, x_1) = (5, 1)$: mišljenje je bilo negativno
- Na temelju ovih podataka želimo naučiti klasifikator koji bi dalje samostalno donosio odluke.

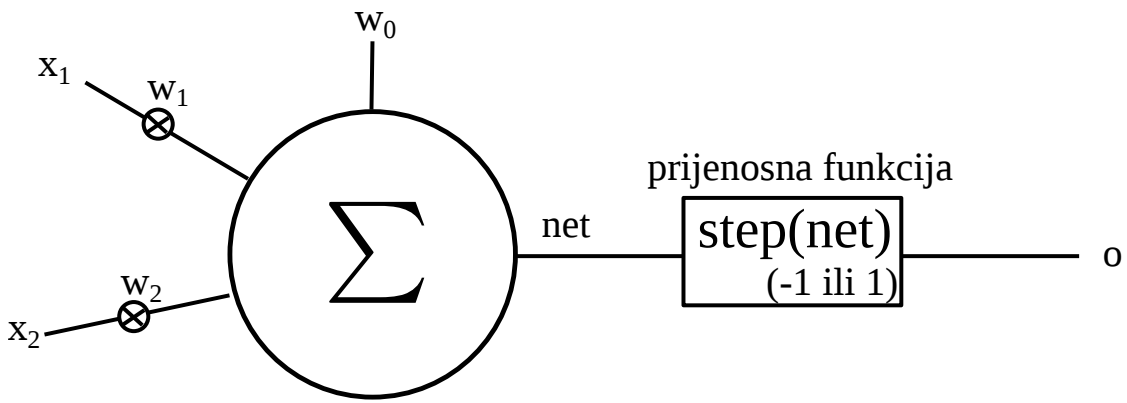
Binarna klasifikacija: primjer

- Kako imamo samo dva razreda, kao klasifikatorski sustav razmotrit ćemo neuron s prijenosnom funkcijom skoka čiji je izlaz, ovisno o ulazu, -1 ili 1
  - razred *negativno mišljenje* kodirat ćemo s -1
  - razred *pozitivno mišljenje* kodirat ćemo s 1
- Time ćemo dobiti sljedeći skup uzoraka za učenje:

x_2	x_1	Razred	t
2	5	pozitivno mišljenje	1
5	2	pozitivno mišljenje	1
1	5	negativno mišljenje	-1
5	1	negativno mišljenje	-1

Binarna klasifikacija: primjer

Korišteni neuron:



- Na ulaz x_1 dovodi se ocjena iz prvog kruga
- Na ulaz x_2 dovodi se ocjena iz drugog kruga
- Računamo: akumulirana suma $net = x_2 \cdot w_2 + x_1 \cdot w_1 + w_0$
- Računamo: izlaz $o = step(net)$

Binarna klasifikacija: primjer

Pretpostavimo sada da smo težine odabrali nasumično, te da one iznose $w_2 = 1$, $w_1 = 1.3$ i $w_0 = -5.85$. Kako će promatrani neuron klasificirati uzorke iz skupa za učenje?

(x_2, x_1, x_0)	t	(w_2, w_1, w_0)	***net***	o	**Ispravno**
$(2, 5, 1)$	1	$(1, 1.3, -5.85)$	2.65	1	da
$(5, 2, 1)$	1	$(1, 1.3, -5.85)$	1.75	1	da
$(1, 5, 1)$	-1	$(1, 1.3, -5.85)$	1.65	1	ne
$(5, 1, 1)$	-1	$(1, 1.3, -5.85)$	0.01	1	ne

Binarna klasifikacija: primjer

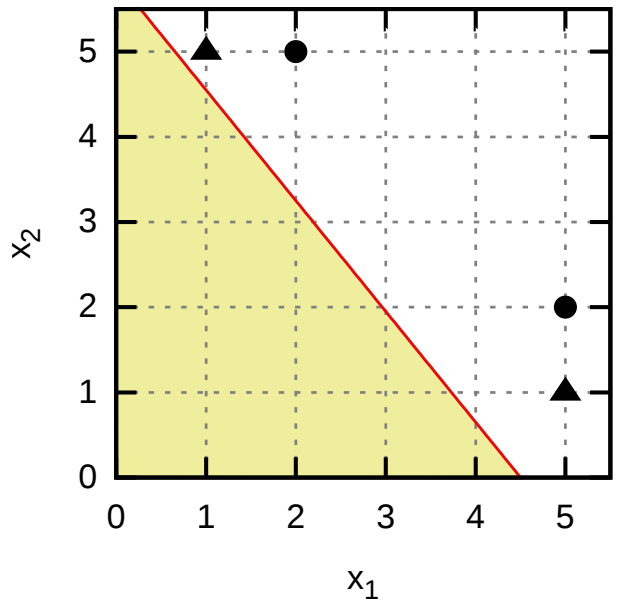
Decizijska granica kod TLU neurona je linearna:

- $o = step(net)$ mijenja vrijednost kada se mijenja predznak od net
- stoga nas zanima kada je $net = 0$, što je:
  $x_2 \cdot w_2 + x_1 \cdot w_1 + w_0 = 0$
- to je pravac u dvodimenzijskom prostoru s osima x_1 i x_2:
  $x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_0}{w_2}$ gdje je $-\frac{w_1}{w_2}$ nagib tog pravca
- Ovaj pravac dvodimenzijski prostor dijeli u dva podprostora: u jednom su svi uzorci kojima klasifikator pridjeljuje vrijednost -1 a u drugom svi uzorci kojima klasifikator pridjeljuje vrijednost 1
- U 3 dimenzije decizijska granica je ravnina a u višedimenzijskim prostorima hiperravnina

Binarna klasifikacija: primjer

Pogledajmo naš konkretan primjer gdje računamo $net = 1 \cdot x_2 + 1.3 \cdot x_1 - 5.85$.

- Decizijska granica prikazana je na slici.
- Uzorci kojima klasifikator pridjeljuje oznaku -1 su u žutom podprostoru (npr. uzorak (1,1) mu pripada).
- Uzorci kojima klasifikator pridjeljuje oznaku 1 su u bijelom podprostoru (npr. uzorak (5,5) mu pripada).
- Sa slike sada vidimo i da klasifikator griješi: svim uzorcima iz skupa za učenje pridjeljuje oznaku 1.



Učenje iz podataka

- 1949. godine Hebb dolazi do spoznaje na temelju proučavanja bioloških neurona: *učiti zanči mijenjati jakosti veza*
- 1958. godine Rosenblatt spaja Hebbovu ideju i McCulloch-Pitts model te definira *pravilo učenja perceptrona*

Pravilo učenja perceptrona

1. Ciklički prolazi kroz svih N uzoraka za učenje, jedan po jedan.
2. Klasificiraj trenutni uzorak.
   1. Ako se klasificira korektno, ne mijenjaj težine i
      1. ako je to N-ti uzastopni uzorak klasificiran korektno, prekini učenje,
      2. inače prijeđi na sljedeći uzorak.
   2. Ako se ne klasificira korektno, korigiraj težine perceptrona prema sljedećem izrazu:
      $w_i(k+1) \leftarrow w_i(k) + \eta \cdot (t - o) \cdot x_i$

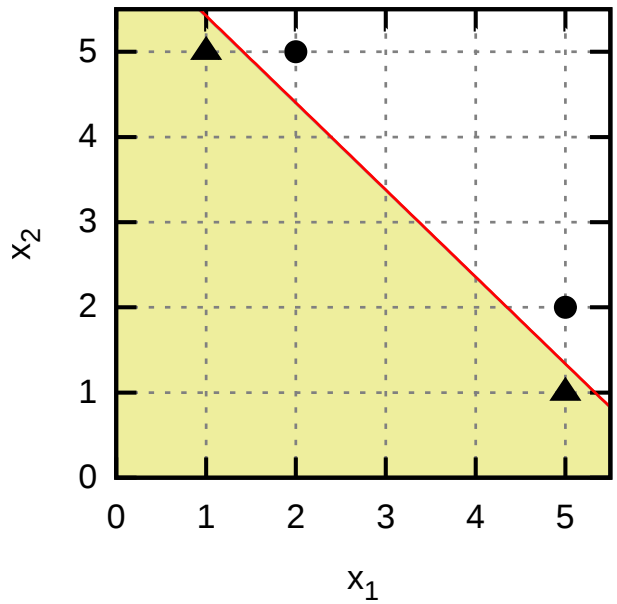
Učenje iz podataka

- Parametar η (eta) naziva se stopa učenja
  - To je pozitivan broj malog iznosa (primjerice, između 0.001 i 0.5) koji regulira u kojoj će se mjeri ažurirati trenutne vrijednosti težina
  - Ako je η premali, postupak učenja će napredovati vrlo sporo
  - Ako je η preveliki, postupak može divergirati
- Provedimo postupak učenja uz $\eta = 0.02$
  (vidi skriptu "Umjetne neuronske mreže" na
  `http://java.zemris.fer.hr/nastava/ui/`,
  poglavlje 2 za čitav primjer).
- Postupak završava s težinama:
  $w_2 = 0.92,\ w_1 = 0.94,\ w_0 = -5.93.$

Binarna klasifikacija: primjer

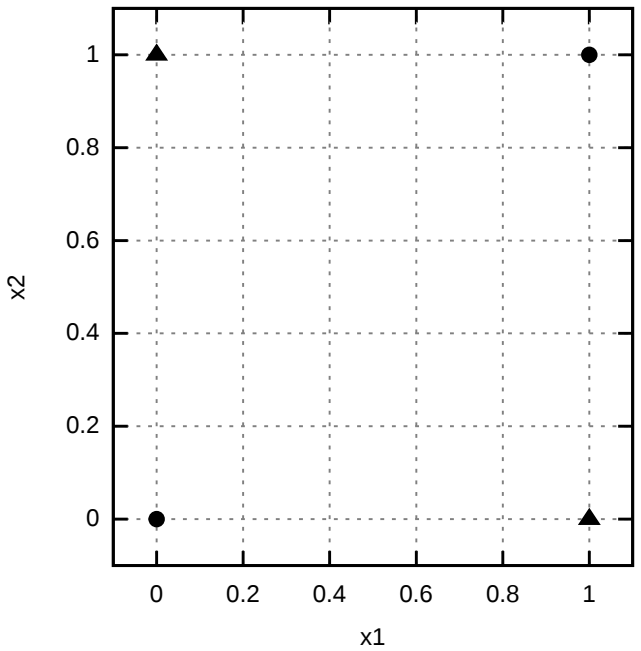
Sada TLU-perceptron računa $net = 0.92 \cdot x_2 + 0.94 \cdot x_1 - 5.93$.

- Svi su uzorci dobro klasificirani.
- Sustav je naučio generalizirati: uzorcima poput (1,1) pridjeljuje oznaku -1 a uzorcima poput (5,5) pridjeljuje oznaku 1, baš kao što bi to učinio i čovjek



Ograničenja TLU-perceptrona

- TLU-perceptron ima linearnu decizijsku granicu
- Takvim neuronom ne možemo riješiti problem klasifikacije linearno-nerazdvojivih razreda
- Primjer linearno-nerazdvojivih razreda (funkcija XOR) prikazan je na slici desno
- Za rješavanje ovakvih složenijih problema razmotrit ćemo sustave koji su izgrađeni od više neurona: umjetne neuronske mreže



Matrični prikaz

Implementacijski, a posebice kada krenemo na neuronske mreže, prikladno je koristiti matrični prikaz.

- Držat ćemo se sljedeće konvencije:
  - vektor ćemo poistovjećivati s jednostupčanom matricom
  - zapis $\vec{y} = f(\vec{x})$ gdje je f skalarna funkcija jedne varijable označavat će novi vektor kod kojeg je f primijenjana na svaki element od $\vec{x}$, tj. $y(i) = f(x(i))$.
- Neka neuron ima n ulaza; iste ćemo označiti s $\vec{x} = (x_1, x_2, \ldots, x_n)$
- Tada ima i n težina $\vec{w} = (w_1, w_2, \ldots, w_n)$ i jedan prag b (tj. w_0)
- Tada vrijedi:

$$o = f(net) = f(\vec{w}^T \cdot \vec{x} + b)$$

- Imamo li na raspolaganju prikladnu biblioteku za rad s matricama i vektorima, kod može biti vrlo kratak i učinkovit

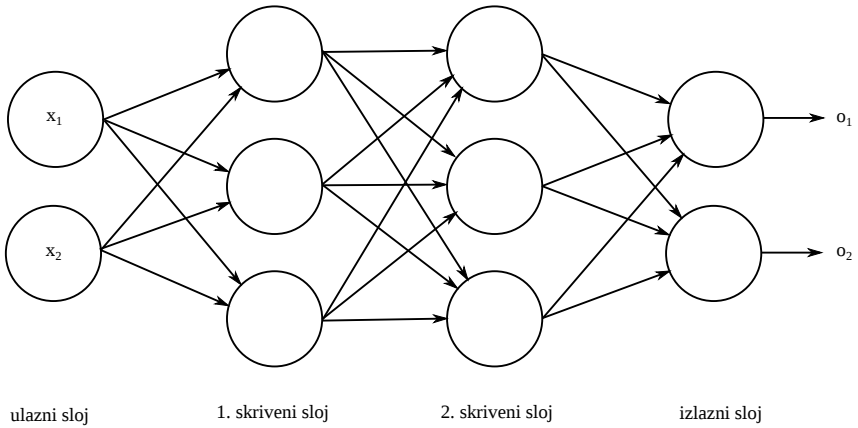
Sadržaj

Arhitekture

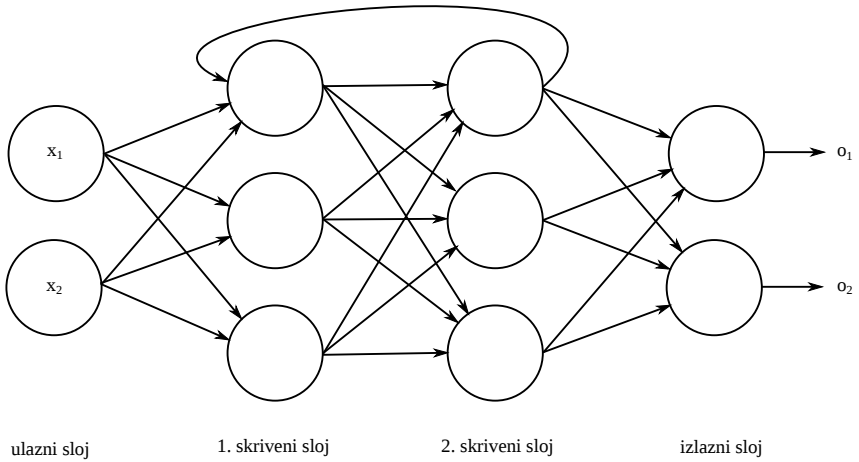
Kako bismo omogućili modeliranje složenijih odnosa u postupcima klasificiranja te regresije, koristit ćemo više neurona

- Pojam *arhitektura* neuronske mreže govori nam kako su neuroni međusobno povezani i koliko ih ima

Slojevita neuronska mreža $2 \times 3 \times 3 \times 2$



Neslojevita neuronska mreža

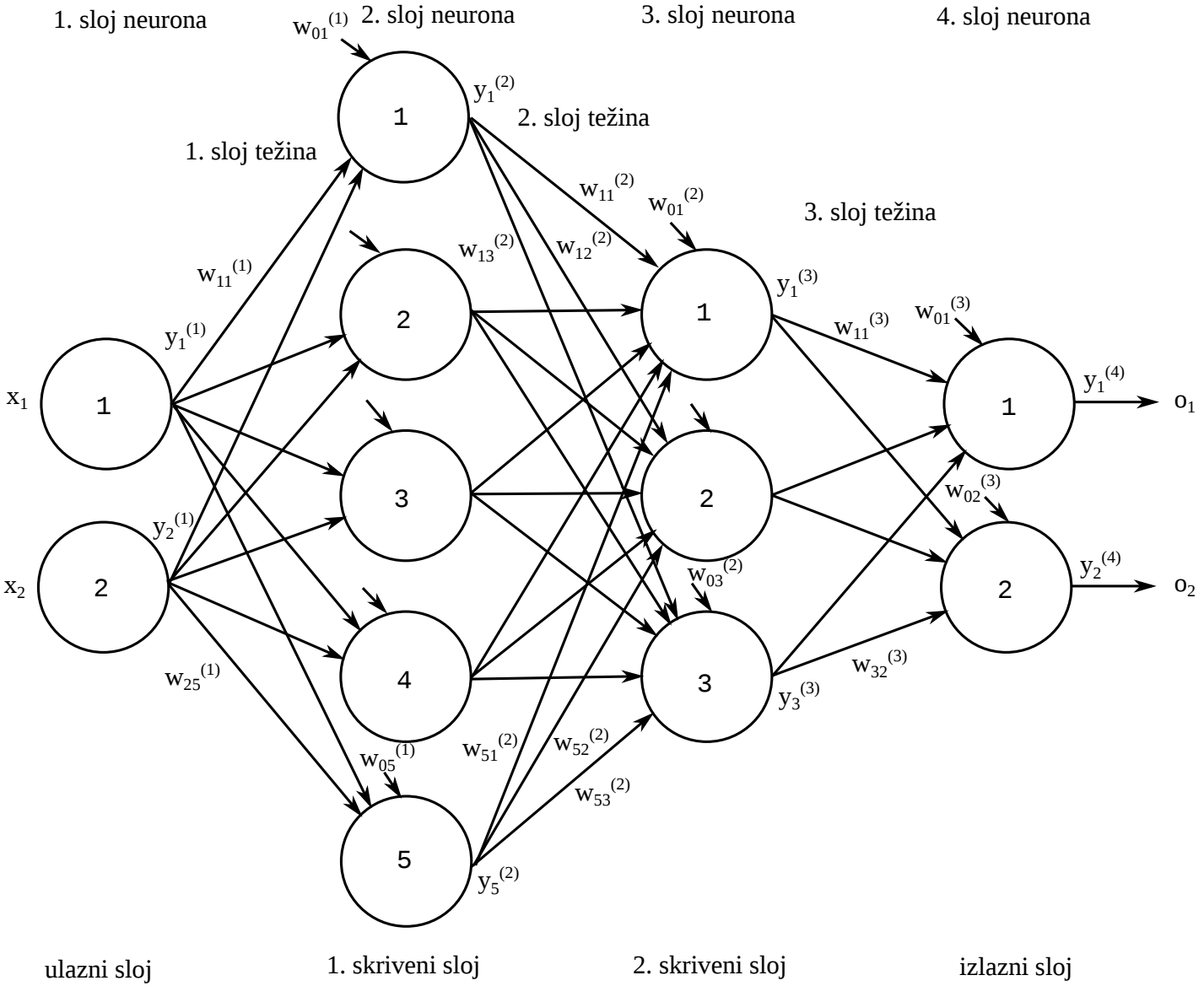


- Kod desne mreže problem je povratna veza zbog koje mreža više nije unaprijedna

Potreba za nelinarnim prijenosnim funkcijama

- Pretpostavimo da smo izgradili višeslojnu neuronsku mrežu u kojoj svi neuroni kao prijenosnu funkciju imaju funkciju identiteta.
- Ta čitava mreža jednako je ekspresivna kao i jedan neuron istog tipa: linearna kombinacija linearnih kombinacija opet je linearna kombinacija.
- Da bismo povećali ekspresivnost mreže i omogućili joj modeliranje nelinearnih odnosa, korišteni neuroni moraju imati prijenosne funkcije koje su nelinearne.
- U unaprijednim neuronskim mrežama:
  - do nedavno je bilo uobičajeno koristiti sigmoidalne prijenosne funkcije
  - danas se za to preferira uporaba ReLU jer omogućava treniranje dubljih mreža (mreža s većim brojem slojeva)

Unaprijedna slojevita neuronska mreža: primjer



Slika: Unaprijedna slojevita potpuno-povezana neuronska mreža

Unaprijedna slojevita neuronska mreža: primjer

- prethodna mreža ima dva skrivena sloja i jedan izlazni
- obavlja preslikavanje $\mathbb{R} \times \mathbb{R} \to \mathbb{R} \times \mathbb{R}$
- konvencija:
  - ulaz u mrežu čemo označiti dvokomponentnim vektorom $\vec{x}$
  - izlaz svakog sljedećeg sloja također će biti jedan vektor dimenzionalnosti koja je jednaka broju neurona u sloju
  - izlaze skrivenih slojeva obično označavamo slovom h (od *hidden*), pa ćemo imati $\vec{h}_1$ (dim=5), $\vec{h}_2$ (dim=3) te $\vec{y} = \vec{h}_3$ (dim=2)

Unaprijedna slojevita neuronska mreža: primjer

- Pogledajmo sada prvi neuron u prvom skrivenom sloju. On računa:

$$y_1^{(2)} = f\left(\begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + w_{01}^{(1)}\right)$$

- Drugi neuron u prvom skrivenom sloju računa:

$$y_2^{(2)} = f\left(\begin{bmatrix} w_{12}^{(1)} & w_{22}^{(1)} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + w_{02}^{(1)}\right)$$

- Poslužimo li se matričnim zapisom, kompletan izlaz sloja možemo zapisati kao:

$$\begin{bmatrix} y_1^{(2)} \\ y_2^{(2)} \\ y_3^{(2)} \\ y_4^{(2)} \\ y_5^{(2)} \end{bmatrix} = f\left(\begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} \\ w_{12}^{(1)} & w_{22}^{(1)} \\ w_{13}^{(1)} & w_{23}^{(1)} \\ w_{14}^{(1)} & w_{24}^{(1)} \\ w_{15}^{(1)} & w_{25}^{(1)} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} w_{01}^{(1)} \\ w_{02}^{(1)} \\ w_{03}^{(1)} \\ w_{04}^{(1)} \\ w_{05}^{(1)} \end{bmatrix}\right)$$

Unaprijedna slojevita neuronska mreža: primjer

- Prethodno konciznije možemo zapisati kao:

$$\vec{h}_1 = f(\mathbf{W}_1 \cdot \vec{x} + \vec{b}_1)$$

gdje je $\vec{h}_1$ vektor izlaza sloja, $\mathbf{W}_1$ matrica težina (jedan neuron, jedan redak), $\vec{x}$ vektor ulaza te $\vec{b}_1$ vektor pragova

- uz dodatno poopćenje dolazimo do izraza:

$$\vec{h}_i = f(\mathbf{W}_i \cdot \vec{h}_{i-1} + \vec{b}_i)$$

uz $\vec{h}_0 = \vec{x}$ i $\vec{y} = \vec{h}_3$, što se opet lagano implementira

- primijetimo: za svaki sloj trebamo znati matricu težina, vektor pragova te prijenosnu funkciju

Učenje: podešavanje težina

- U koje će vrijednosti neuronska mreža preslikati ulazni uzorak određeno je težinama i pragovima neuronske mreže
- Prilikom nadziranog učenja neuronske mreže, dostupan nam je skup uzoraka za učenje koji se sastoji od parova (ulazni uzorak, željeni izlaz): $\{(x_{1,1}, \ldots, x_{1,N_i}) \rightarrow (t_{1,1}, \ldots, t_{1,N_o}), \ldots, (x_{N,1}, \ldots, x_{N,N_i}) \rightarrow (t_{N,1}, \ldots, t_{N,N_o})\}$ gdje je N_i dimenzionalnost ulaza a N_o dimenzionalnost izlaza
- Kako bismo došli do algoritma za prilagođavanje težina, moramo najprije definirati funkciju pogreške E. Često korištena funkcija je polovična suma srednjih kvadratnih odstupanja:

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

Učenje: podešavanje težina

- Jednom kada je definirana funkcija pogreške, ako ista zadovoljava svojstvo derivabilnosti, moguće je napisati optimizacijski postupak koji se temelji na izračunu gradijenata (parcijalnih derivacija funkcije pogreške s obzirom na svaku težinu i prag)
- parcijalna derivacija nam govori kako će se funkcija pogreške promijeniti ako malo povećamo težinu
- tu informaciju možemo iskorisiti kako bismo ciljano povećavali ili smanjivali težine s ciljem smanjivanja vrijednosti funkcije pogreške E

Postupak propagacije pogreške unatrag

Postupak učenja neuronskih mreža koji se temelji na učinkovitom izračunu svih parcijalnih derivacija i njihovoj primjeni na određivanje iznosa kojim korigiramo svaku od težina zove se Postupak propagacije pogreške unatrag (engl. *Error Backpropagation*)

Nećemo raditi njegov izvod, ali ćemo dati konačni rezultat.

Postupak propagacije pogreške unatrag: algoritam

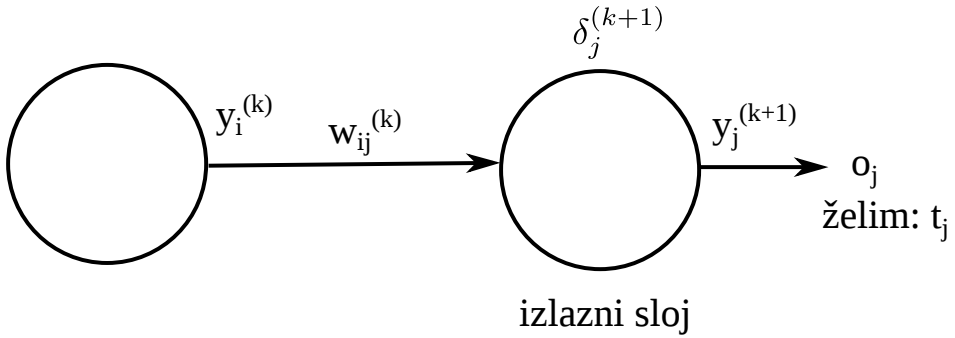
1. Sve težine neuronske mreže postavi na slučajne vrijednosti.
2. Ponavljaj dok nije zadovoljen uvjet zaustavljanja
   1. Za svaki uzorak $s : (x_{s,1}, \ldots, x_{s,N_i}) \rightarrow (t_{s,1}, \ldots, t_{s,N_o})$ čini:
      1. Postavi podatak $(x_{s,1}, \ldots, x_{s,N_i})$ na ulaz mreže.
      2. Izračunaj izlaze svih neurona u svim slojevima, od prvog prema posljednjem; posljednji označimo: $(o_{s,1}, \ldots, o_{s,N_o})$.
      3. Odredi pogreške neurona *izlaznog* sloja
      $$\delta_i^K = o_{s,i} \cdot (1 - o_{s,i}) \cdot (t_{s,i} - o_{s,i}).$$
      4. Vraćaj se sloj po sloj prema početku mreže. Za i-ti neuron koji se nalazi u k-tom sloju pogreška je:
      $$\delta_i^{(k)} = y_i^{(k)} \cdot (1 - y_i^{(k)}) \cdot \sum_{d \in \text{Downstream}} w_{i,d} \cdot \delta_d^{(k+1)}$$
      5. Napravi korekciju svih težina. Težinu $w_{i,j}^{(k)}$ korigiraj prema izrazu:
      $$w_{i,j}^{(k)} \leftarrow w_{i,j}^{(k)} + \eta \cdot y_i^{(k)} \cdot \delta_j^{(k+1)}$$
      a pragove prema izrazu:
      $$w_{0,j}^{(k)} \leftarrow w_{0,j}^{(k)} + \eta \cdot \delta_j^{(k+1)}.$$

Postupak propagacije pogreške unatrag: algoritam

- Na ovom kolegiju ne radimo izvod algoritma propagacije pogreške unatrag
- Detaljniji opis algoritma dan je u skripti
- Izrazi u danom algoritmu podrazumijevaju da su u neuronima korištene sigmoidalne prijenosne funkcije
- U slučaju drugih prijenosnih funkcija dijelove izraza koji su se pojavili kao posljedica deriviranja sigmoidalne prijenosne funkcije $(f \cdot (1 - f))$ potrebno je zamijeniti derivacijom korištene prijenosne funkcije
- Izrazi za korigiranje mogu se lagano zapamtiti "vizualno": sljedeća dva slidea

Postupak propagacije pogreške unatrag: algoritam

Izračun pogreške u neuronu izlaznog sloja.

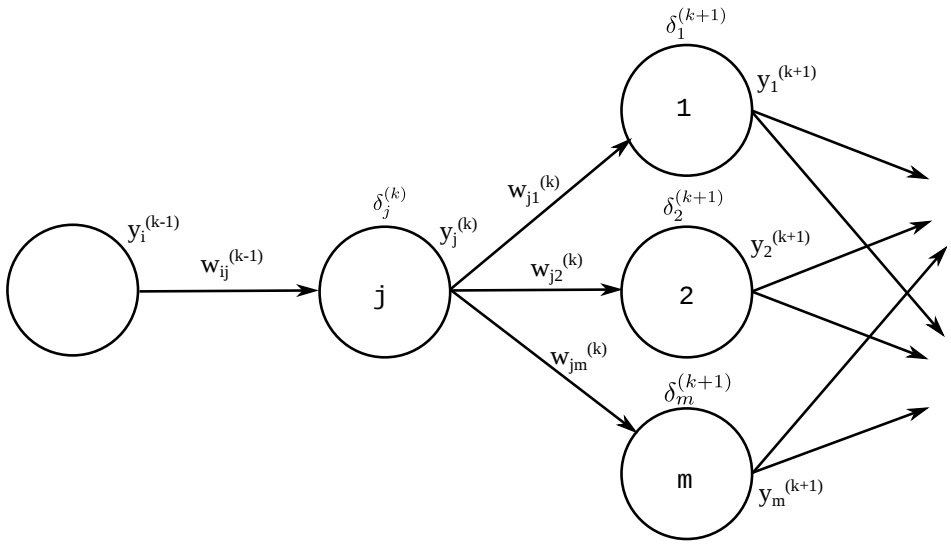


Pogreška: umnožak derivacije prijenosne funkcije neurona $(y_j^{(k+1)} \cdot (1 - y_j^{(k+1)}))$ i stvarne pogreške $(t_j - o_j)$.

$$\delta_j^{k+1} = o_{s,j} \cdot (1 - o_{s,j}) \cdot (t_{s,j} - o_{s,j}).$$

Postupak propagacije pogreške unatrag: algoritam

Izračun pogreške u neuronu skrivenog sloja: $\delta_j^{(k)}$.

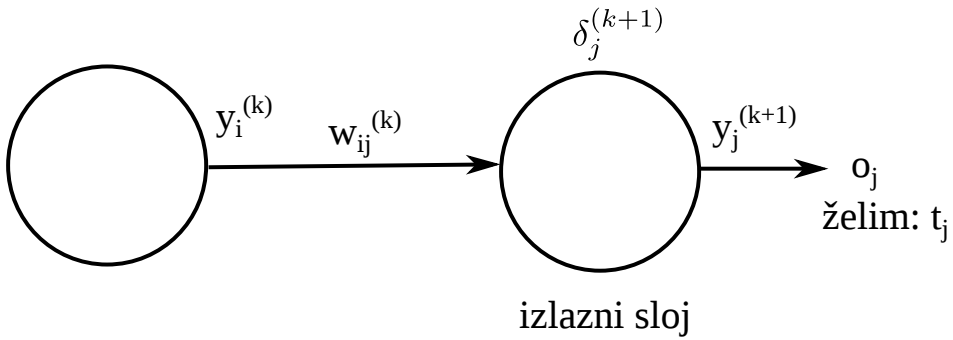


Pogreška: umnožak derivacije prijenosne funkcije promatranog neurona i težinske sume pogrešaka neurona kojima on šalje svoj izlaz:

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

Postupak propagacije pogreške unatrag: algoritam

Korekcija: proporcionalno umnošku stope učenja η, izlaza neurona lijevo od težine i pogreške neurona desno od težine:



$$\Delta w_{ij}^k = \eta \cdot y_i^{(k)} \cdot \delta_j^{(k+1)}$$

$$w_{ij}^k \leftarrow w_{ij}^k + \Delta w_{ij}^k$$

Za pragove "lijevo" je konstanta 1, pa imamo:

$$\Delta w_{0j}^k = \eta \cdot \delta_j^{(k+1)}$$

Primjeri

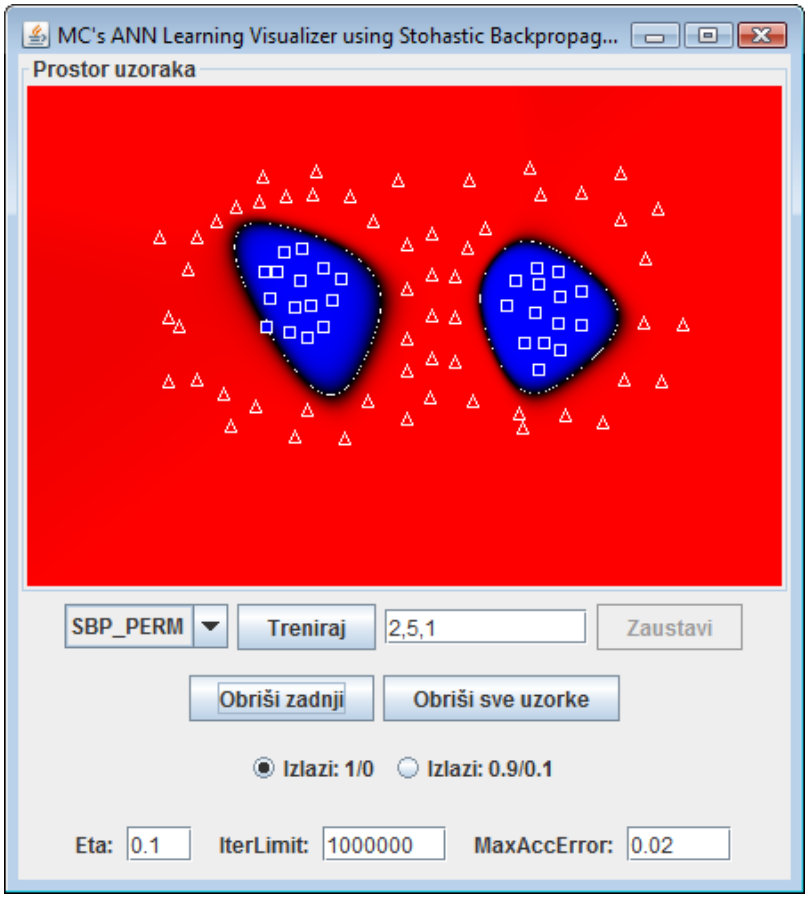
Na adresi `http://java.zemris.fer.hr/nastava/ui/` pod sekcijom *Umjetne neuronske mreže* dostupne su implementacije i opisi niza primjera:

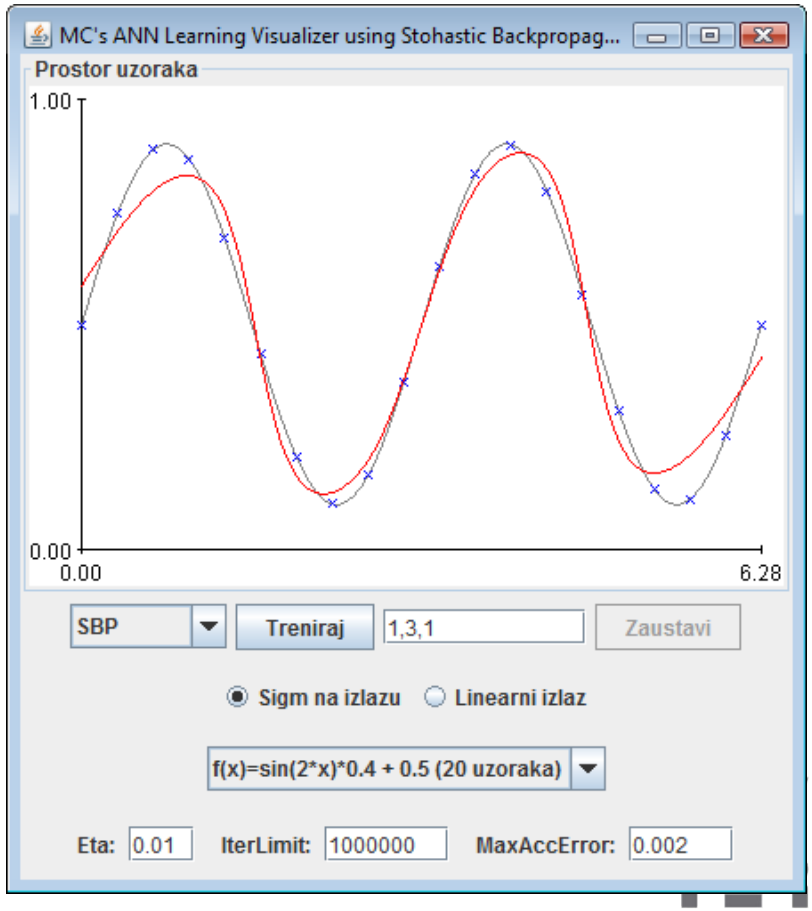
- Klasifikacija uzoraka u 2D
- Funkcijska regresija
- Klasifikacija gesti - kako od pokreta miša doći do razreda geste

Preporuka: svakako samostalno isprobajte.

Primjer: klasifikacija i regresija

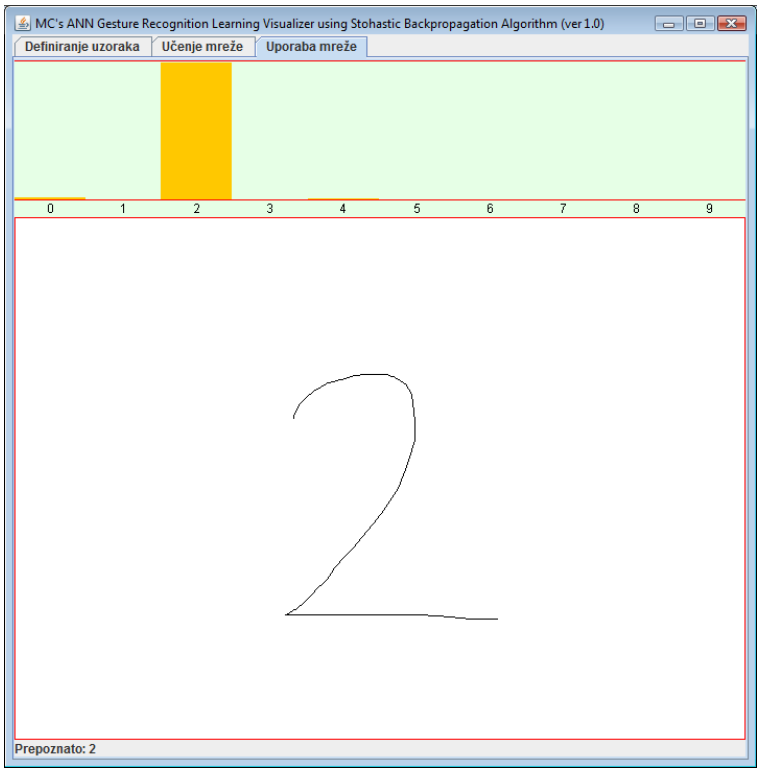
- Razmatramo uporabu neuronske mreže za klasifikaciju 2D uzoraka te regresiju





Primjer: prepoznavanje znamenaka putem gesti

- Zadatak: napraviti program koji neuronsku mrežu koristi za prepoznavanje znamenaka (0-9) na temelju geste koju korisnik u jednom potezu radi mišom
- Program omogućava prikupljanje skupa uzoraka za učenje, provođenje postupka učenja te uporabu naučene mreže

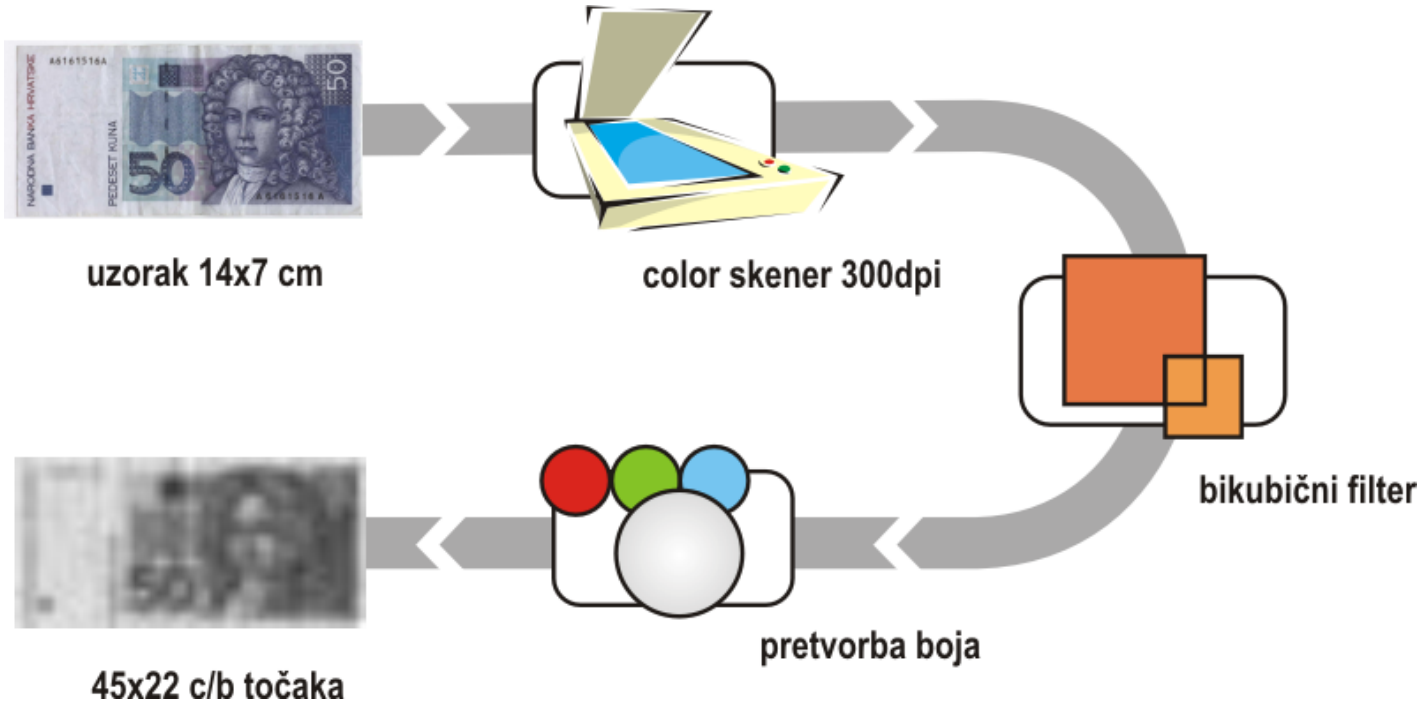


Primjer: klasifikacija novčanica

- Zadatak: klasificirati četiri vrste papirnatih novčanica neovisno o orijentaciji
- Skup primjera za učenje sačinjava 16 različitih uzoraka uzorkovanih s 45x22 slikovna elementa

Primjer: klasifikacija novčanica

- Uzorci se prije dovođenja na ulaze ANN predprocesiraju

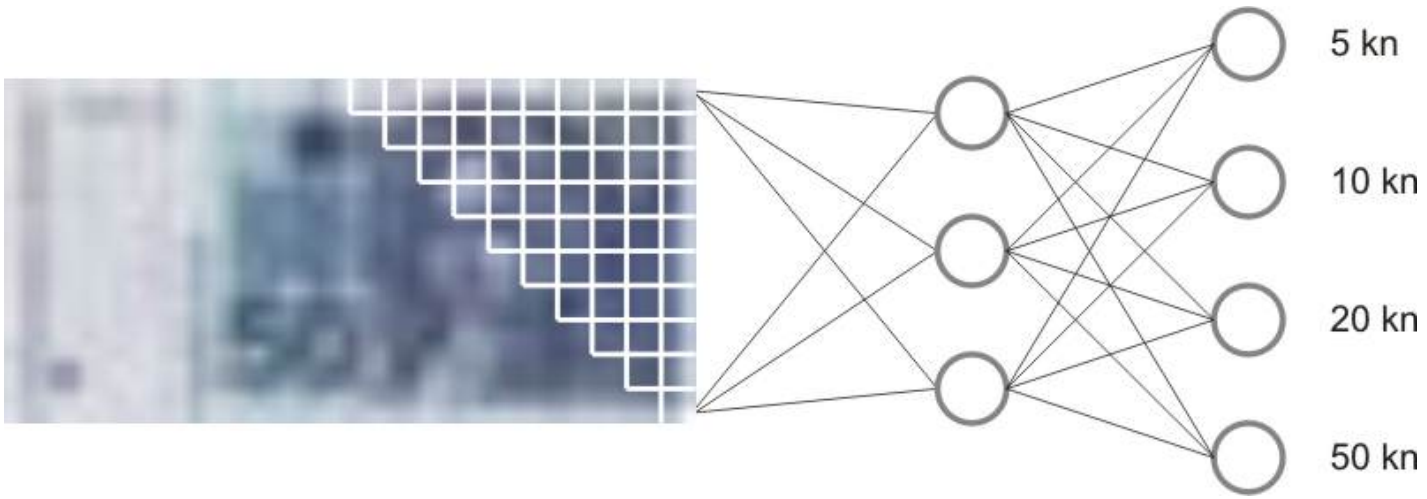


Primjer: klasifikacija novčanica

- Uzorci su u stvari digitalizirane slike, pa će se međusobno razlikovati u intenzitetu odgovarajućih slikovnih elemenata (posljedica razlike u stupnju istrošenosti novčanice, izgužvanosti papira, oštećenosti).
- U primjeru je korišten generator umjetnih uzoraka koji generira uzorke različitog stupnja oštećenja u svrhu provjere rada mreže i ugađanja njezinih parametara.

Primjer: klasifikacija novčanica

- Parametri mreže: aciklička potpuno povezana višeslojna neuronska mreža $990 \times 3 \times 4$
- Učenje: algoritam propagacije pogreške unatrag, stopa učenja=0.02, moment=0.02, provjera nad skupom za validaciju svakih 2500 epoha



Dalje?

- Umjetne neuronske mreže danas imaju mnoge primjene (obrada slike, zvuka, teksta)
- Nekoliko vrsta koje nismo spominjali danas se često koriste: konvolucijske neuronske mreže, povratne neuronske mreže
- Više informacija moguće je dobiti na diplomskom studiju
  - Strojno učenje
  - Neizrazito, evolucijsko i neuroračunarstvo
  - Duboko učenje