

Klasifikacija uporabom umjetnih neuronskih mreža

Darijo Brčina

Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva

Završni rad br. 6950

Zagreb, 06.07.2020

Sadržaj

Opis zadatka

Nadzirano učenje

- Problemi nadziranog učenja

 - Regresija

 - Klasifikacija

Umjetne neuronske mreže

- Umjetni neuron

- Aktivacijske funkcije

- Unaprijedna neuronska mreža

- Učenje unaprijedne neuronske mreže

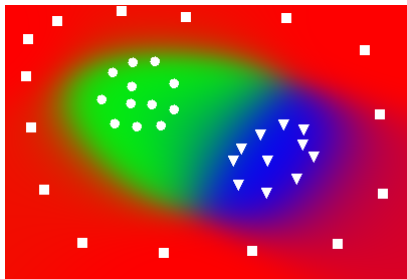
Rezultati

Zaključak

Opis zadatka

Cilj:

- ▶ naučiti neuronsku mrežu da na temelju danih labeliranih podataka jasno može odrediti kojem razredu pripadaju

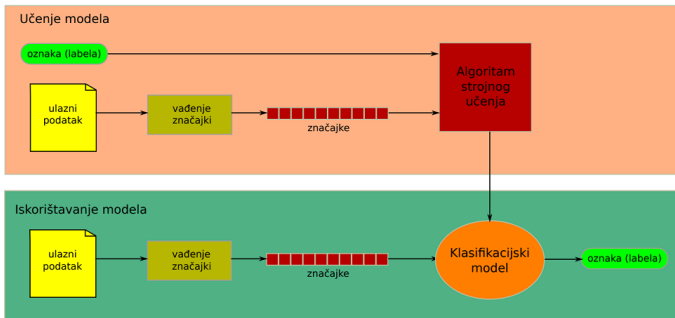


Nadzirano učenje

- ▶ Jedno od glavnih područja strojnog učenja
- ▶ Model nadziranog učenja (engl. *supervised learning*) prolazi kroz dvije faze:

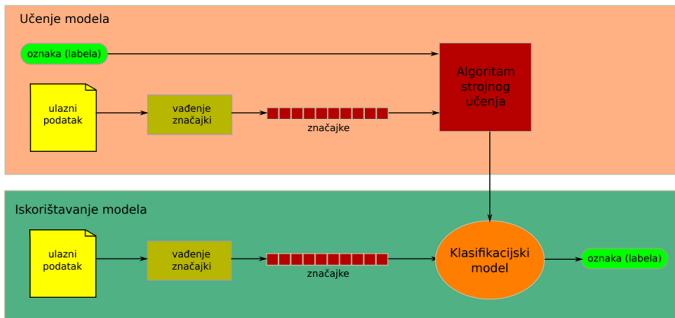
Nadzirano učenje

- ▶ Jedno od glavnih područja strojnog učenja
- ▶ Model nadziranog učenja (engl. *supervised learning*) prolazi kroz dvije faze:



Nadzirano učenje

- ▶ Jedno od glavnih područja strojnog učenja
- ▶ Model nadziranog učenja (engl. *supervised learning*) prolazi kroz dvije faze:



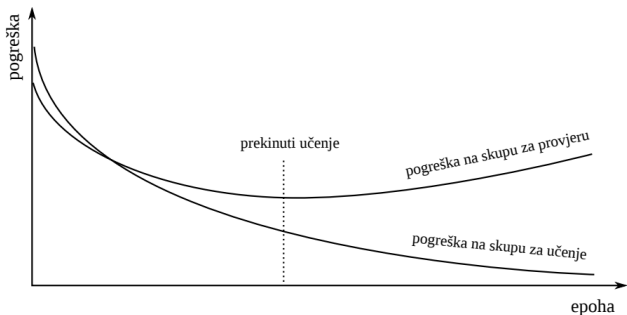
- ▶ Glavni cilj je omogućiti da model razvije svojstvo **generalizacije**
- ▶ Ispostavlja se da je to dosta zahtjevno za izvesti pa često dolazi do problema **prenaučenosti** (engl. *overfitting*)

Lijek za prenaučenosť

- ▶ Prenaučenosť je jedan od glavnih problema strojnog učenja koji model čini beskorisnim zbog loše generalizacije
- ▶ Jedan od načina kako spriječiti prenaučenosť je korištenje metode **unakrsna provjera** (engl. *cross-validation*)
- ▶ Ideja je da se oko 30% uzoraka **skupa za učenje** (engl. *training set*) prebaci u **skup za provjeru** (engl. *validation set*) te prati pogreška nad oba skupa pomoću koje ćemo odrediti kada je učenje modela potrebno zaustaviti:

Lijek za prenaučenost

- ▶ Prenaučenost je jedan od glavnih problema strojnog učenja koji model čini beskorisnim zbog loše generalizacije
- ▶ Jedan od načina kako spriječiti prenaučnost je korištenje metode **unakrsna provjera** (engl. *cross-validation*)
- ▶ Ideja je da se oko 30% uzoraka **skupa za učenje** (engl. *training set*) prebaci u **skup za provjeru** (engl. *validation set*) te prati pogreška nad oba skupa pomoću koje ćemo odrediti kada je učenje modela potrebno zaustaviti:



Skup za testiranje

- ▶ Ponekad su modeli složeni pa je potrebno odrediti njihovu optimalnu složenost te se zbog toga uvodi **skup za testiranje** (engl. *testing set*) koji također sadrži oko 30% uzoraka skupa za učenje i služi za provjeru točnosti jednom naučenog modela
- ▶ Slikovito prikazani skupovi:

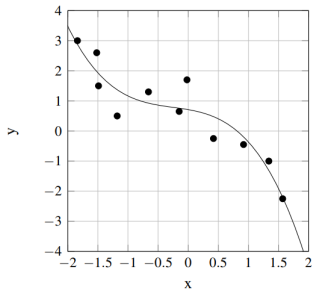
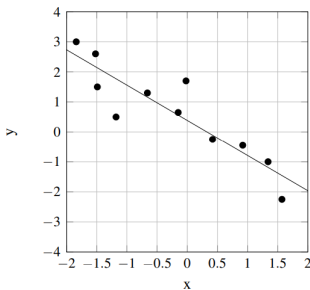


Problemi nadziranog učenja - Regresija

- ▶ Regresija (engl. *regression*) je tehnika kojom se pokušava modelirati odnos između određenog broja značajki i kontinuirane ciljne varijable
- ▶ Razlikujemo više tipova regresija poput linearne i nelinearne regresije kao i regresije s jednom varijablom i regresije s više varijabli

Problemi nadziranog učenja - Regresija

- ▶ Regresija (engl. *regression*) je tehnika kojom se pokušava modelirati odnos između određenog broja značajki i kontinuirane ciljne varijable
- ▶ Razlikujemo više tipova regresija poput linearne i nelinearne regresije kao i regresije s jednom varijablom i regresije s više varijabli

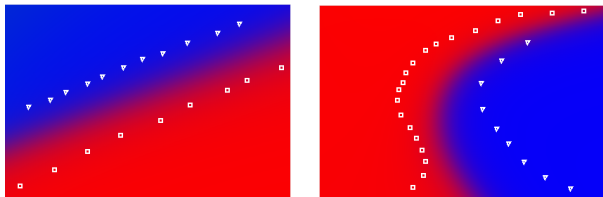


Problemi nadziranog učenja - Klasifikacija

- ▶ Klasifikacija (engl. *classification*) je tehnika kojom se određene oznake pokušavaju kategorizirati diskretnim vrijednostima
- ▶ Kao i kod regresije, razlikujemo linearnu i nelinearnu klasifikaciju
- ▶ Razlikujemo binarnu i višerazrednu klasifikaciju i klasifikaciju više oznaka

Problemi nadziranog učenja - Klasifikacija

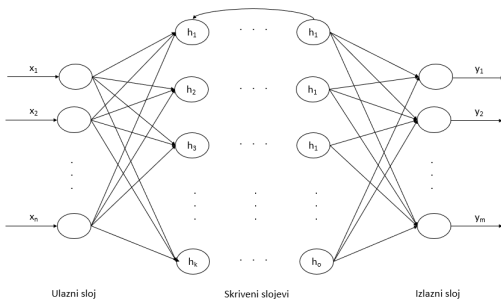
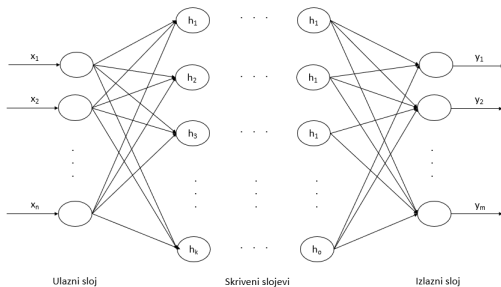
- ▶ Klasifikacija (engl. *classification*) je tehnika kojom se određene oznake pokušavaju kategorizirati diskretnim vrijednostima
- ▶ Kao i kod regresije, razlikujemo linearnu i nelinearnu klasifikaciju
- ▶ Razlikujemo binarnu i višerazrednu klasifikaciju i klasifikaciju više oznaka



Umjetne neuronske mreže

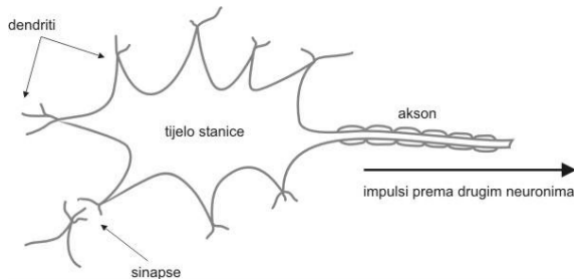
- ▶ Jedan od najkorištenijih algoritama strojnog učenja
- ▶ Motivacija pronađena u građi, povezanosti i brzini obrade raznih podražaja ljudskog mozga te mogućnosti učenja iz prijašnjeg iskustva
- ▶ Danas je poznato da u ljudskom mozgu postoji oko 10^{11} neurona te 10^{15} međusobnih veza, što znači da je svaki neuron u prosjeku povezan s 10^4 različitih veza
- ▶ Primjer konektivističkog pristupa
- ▶ Područje koje se bavi proučavanjem umjetnih neuronskih mreža naziva se neuro računarstvo (engl. *neuro-computing*) koje je jedno od grana mekog računarstva (engl. *soft-computing*)

Umjetne neuronske mreže



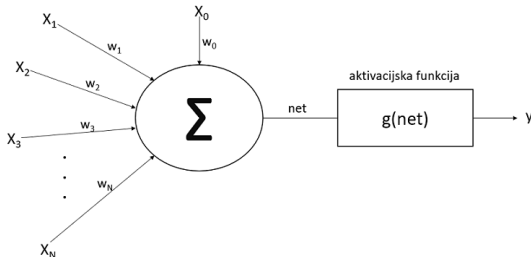
Umjetni neuron

- Motiviran strukturom biološkog neurona:



- Definiran 1943. u radu *A Logical Calculus of Ideas Immanent in Nervous Activity* dvojice znanstvenika Warren S. McCulloch i Walter H. Pitts

Umjetni neuron



- ▶ Sastoji se od $(n + 1)$ različitih ulaza \mathbf{x} i težina \mathbf{w} te jednog izlaza \mathbf{y}
- ▶ Izlaz se dobije tako što se prvo izračuna **net** vrijednost po formuli:

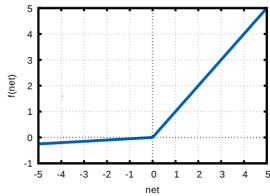
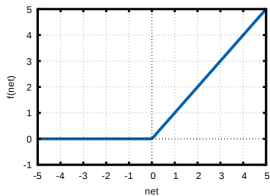
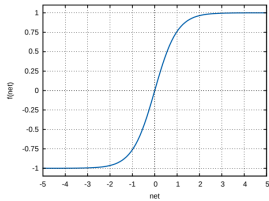
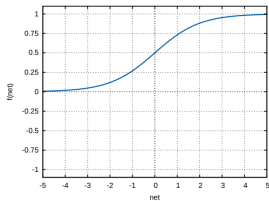
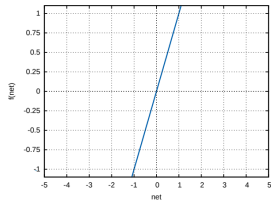
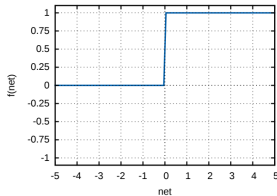
$$net = \sum_{i=0}^N x_i \cdot w_i$$

- ▶ Zatim se dobivena **net** vrijednost provuče kroz neku od aktivacijskih funkcija

Aktivacijske funkcije

- ▶ U starijoj literaturi nazivane i prijenosnim funkcijama
- ▶ Nalaze se tik prije izlaza neurona i određuju hoće li neuron biti aktiviran ili ne, tj. hoće li se "paliti" ili ne
- ▶ Razlikujemo binarnu funkciju skoka (engl. *step function*), linearne aktivacijske funkcije poput funkcije identiteta (engl. *identity function*) i nelinearne aktivacijske funkcije poput sigmoidalne (engl. *sigmoid function*), funkcije zglobnice (engl. *Rectified Linear Unit, ReLU*) i dr.
- ▶ Svaka od njih različito utječe na izgled **decizijskih granica**, granica između razreda nastalih rezultatom klasifikacije
- ▶ Danas se najviše koriste nelinearne aktivacijske funkcije zbog toga što su derivabilne i čije derivacije nisu konstante što uvelike pospješuje učenje mreža
- ▶ Funkcija skoka se koristi kod binarne klasifikacije, linearne funkcije kod linearno interpretabilnih problema, a nelinearne se mogu koristiti gotovo uvijek

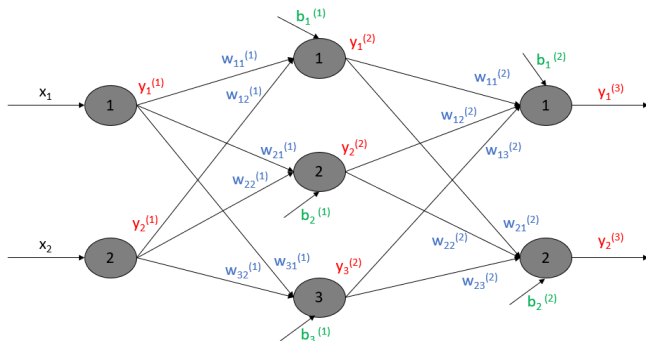
Aktivacijske funkcije



Unaprijedna neuronska mreža

- ▶ Unaprijedna višeslojna potpuno povezana umjetna neuronska mreža (engl. *feedforward multilayered fully connected artificial neural network*)
- ▶ Operacija **mapiranja** ulaza na određene izlaze
- ▶ Često se referira kao **višeslojni perceptron**
- ▶ Naziv **perceptron** prvi je upotrijebio 1958. znanstvenik Frank Rosenblatt u svojem radu *The perceptron: a probabilistic model for information storage and organization in the brain*
- ▶ Ponukan MCP neuronom i Hebbovim principom učenja razvija prvi algoritam učenja **jednoslojnog perceptrona** koji nije upotrebljiv nad višeslojnim perceptronima zbog čega je dosta kritiziran
- ▶ Može sadržavati proizvoljnu arhitekturu, npr. 2x3x2:

Unaprijedna neuronska mreža



- ▶ $w_{jk}^{(l)}$: težina između k-tog neurona $(l-1)$ -vog sloja i j-tog neurona l -tog sloja.
- ▶ $b_j^{(l)}$: bias j-tog neurona l -tog sloja.
- ▶ $y_j^{(l)}$: izlaz j-tog neurona l -tog sloja.

Unaprijedna neuronska mreža

- Provedba algoritma unaprijedne neuronske mreže je računalno dosta prihvatljivo jer se cijeli postupak može prikazati matričnim računom:

$$\vec{y}^{(l+1)} = g(\vec{net}^{(l+1)}) = g(\mathbf{W}^{(l)} \cdot \vec{y}^{(l)} + \vec{b}^{(l)})$$

- $\mathbf{W}^{(l)}$: matrica svih težina l-tog sloja gdje su retci težine iz svih neurona (l-1)-vog sloja u jedan neuron l-tog sloja.
- $\vec{b}^{(l)}$: vektor svih bias vrijednosti l-tog sloja.
- $\vec{net}^{(l)}$: vektor svih net vrijednosti l-tog sloja.
- $\vec{y}^{(l)}$: vektor svih izlaza l-tog sloja.
- Važno je napomenuti da neuronska mreže mora raditi s normaliziranim podacima jer inače neće davati dobre rezultate

Učenje unaprijedne neuronske mreže

- ▶ Učenje algoritmom **gradijentni spust** (engl. *gradient descent*):

$$x \leftarrow x - \alpha \cdot \frac{\partial}{\partial x} f(x)$$

- ▶ Učenje algoritmom **propagacije pogreške unatrag** (engl. *Backpropagation algorithm*) : računanje parcijalnih derivacija za algoritam gradijentni spust
- ▶ 1986. Rumelhart zajedno s dvojicom prijatelja u svojem radu *Parallel Distributed Processing* po prvi put definira termin *Backpropagation* i učenje višeslojnog perceptrona
- ▶ Za učenje je potrebno definirati funkciju kazne koju je potrebno minimizirati, odnosno pronaći one vrijednosti težina za koje je iznos funkcije kazne minimalna
- ▶ Razmatramo funkciju kazne koja je jednaka polovici srednjeg kvadratnog odstupanja između svakog željenog izlaza mreže i stvarne vrijednosti koju mreža generira na izlazu i to kumulativno za sve raspoložive uzorke:

Učenje unaprijedne neuronske mreže

$$E = \frac{1}{2N} \sum_{s=1}^N \sum_{i=1}^m (d_{s,i} - y_{s,i}^{(L)})^2$$

- ▶ N predstavlja broj uzoraka, m predstavlja broj izlaznih neurona, $d_{s,i}$ predstavlja željeni izlaz i -tog neurona s -tog uzorka, a $y_{s,i}^{(L)}$ predstavlja stvarni izlaz i -tog neurona s -tog uzorka
- ▶ Potrebno je ažurirati težine sljedećim pravilom:

$$w_{jk}^{(l)} \leftarrow w_{jk}^{(l)} - \alpha \cdot \frac{\partial E}{\partial w_{jk}^{(l)}}$$

- ▶ Na prvu izgleda kao da je to moguće izvesti klasičnim računom traženja parcijalnih derivacija, no mreže su često jako kompleksnih arhitektura i takav postupak bi resursno i vremenski bio prezahtjevan, stoga tu nastupa algoritam propagacije pogreške unatrag

Učenje unaprijedne neuronske mreže

1. Ciklički prolazi kroz svih N uzoraka za učenje, jedan po jedan.
2. Ponavlja dok nije zadovoljen uvjet zaustavljanja.
3. Za svaki uzorak iz skupa uzoraka za učenje čini:

1. Postavi uzorak na ulaz neuronske mreže i izračunaj izlaze za sve neurone primjenom formule unaprijedne neuronske mreže.
2. Izračunaj pogrešku svakog od neurona izlaznog sloja po formuli:

$$\delta_{s,j}^{(L)} = g'(net_{s,j}^{(L)}) \cdot (d_{s,j} - y_{s,j}^{(L)}).$$

3. Vraćaj se sloj po sloj i izračunaj pogreške svakog neurona po formuli:

$$\delta_{s,j}^{(l)} = g'(net_{s,j}^{(l)}) \cdot \sum_{i=1}^m \delta_{s,i}^{(l+1)} \cdot w_{i,j}^{(l)}.$$

4. Ažuriraj težine po formuli:

$$w_{jk}^{(l)} \leftarrow w_{jk}^{(l)} + \eta \cdot \sum_{s=1}^N \delta_{s,j}^{(l+1)} \cdot y_{s,k}^{(l)}.$$

5. Ažuriraj biase po formuli:

$$b_j^{(l)} \leftarrow b_j^{(l)} + \eta \cdot \sum_{s=1}^N \delta_{s,j}^{(l+1)}.$$

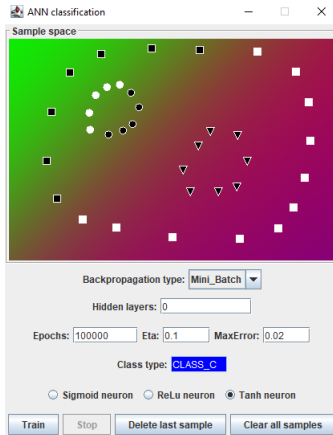
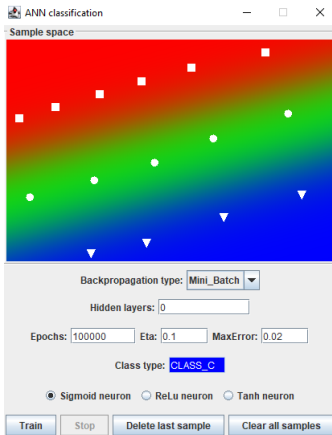
Rezultati

- ▶ U sklopu rada implementirano je grafičko korisničko sučelje kroz koje se interaktivno unose točke u 2D koordinatnom sustavu te se iste predaju neuronskoj mreži na učenje
- ▶ Točke su normalizirane tako da su se stvarne koordinate podijelile sa stvarnom širinom, odnosno visinom prozora kako bi se dobili brojevi iz intervala $[0, 1]$
- ▶ Težine i biasi su inicijalizirane pomoću Xavier inicijalizacije i to tipom normalne razdiobe gdje očekivanje iznosi 0, a standardna devijacija iznosi $\sqrt{\frac{1}{n(l-1)}}$. Nazivnik predstavlja broj neurona u prijašnjem sloju.
- ▶ Moguće je izabrati 3 različita razreda pri unosu točaka gdje svaki razred predstavlja svoju boju iz RGB raspona
- ▶ Interno je svaki razred kodiran sa svojim kodom pa tako točka pripada crvenom razredu ako mreža na izlazu generira $[1, 0, 0]$, zelenom razredu ako generira $[0, 1, 0]$ i plavom razredu ako generira $[0, 0, 1]$

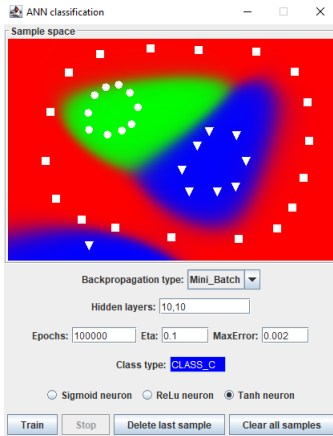
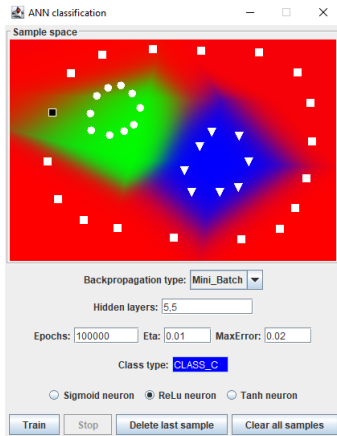
Rezultati

- ▶ Mreža naravno neće davati tako okrugle brojeve pa ih je potrebno zaokružiti sljedećim pravilom: ako je izlaz manji ili jednak 0.5, onda ga postavi na 0, inače na 1. Tako će npr. izlaz $[0.6, 0.3, 0.1]$ biti transformiran u $[1, 0, 0]$ te će se uzorak klasificirati kao da pripada crvenom razredu
- ▶ Boja svakog uzorka grafičkog sučelja se određuje tako da se uzmu onoliki postoci svake od komponenata RGB koji se generiraju na izlazima. Tako će npr. izlaz $[0.6, 0.3, 0.1]$ poprimiti 0.6 crvene boje, 0.3 zelene boje i 0.1 plave boje.
- ▶ Prije učenja je moguće izabrati tri različite aktivacijske funkcije koje djeluju samo na skrivenim slojevima, dok se na izlaznom sloju nalazi funkcija **softmax** koja na temelju svih izlaza generira postotke pripadnosti svakom od razreda gdje vrijedi da je suma svih postotaka potpuna i iznosi 100%
- ▶ U nastavku su dani primjeri linearne i nelinearne klasifikacije kao i primjer prenaučenosti

Rezultati



Rezultati



Zaključak

- ▶ Za rješavanje određenih problema treba jako dobro poznavati teoriju koja se krije iza algoritma umjetnih neuronskih mreža
- ▶ Pametna inicijalizacija težina je jako bitna za postizanje dobrih rezultata unaprijednom neuronskom mrežom kao i odabir arhitekture te aktivacijskih funkcija

Zaključak

- ▶ Za rješavanje određenih problema treba jako dobro poznavati teoriju koja se krije iza algoritma umjetnih neuronskih mreža
- ▶ Pametna inicijalizacija težina je jako bitna za postizanje dobrih rezultata unaprijednom neuronskom mrežom kao i odabir arhitekture te aktivacijskih funkcija

Mogući nastavci na ovaj rad

- ▶ Proučiti inačice algoritma propagacije pogreške unatrag te ih znati primijeniti na probleme kojeg rješavamo
- ▶ Upoznati se s dubokim neuronskim mrežama i algoritmima dubokog učenja