SVEUČILIŠTE U ZAGREBU
FAKULTET KEMIJSKOG INŽENJERSTVA I TEHNOLOGIJE
SVEUČILIŠNI DIPLOMSKI STUDIJ
PRIMIJENJENA KEMIJA

Seminarski rad iz kolegija Uvod u matematičke metode u inženjerstvu

Studenti: Jelena Popčević, Ines Varga, Petar Žuvela
Mentor: dr. sc. Ivica Gusić, red. prof.

Tema: *Izračunavanje temperature vrelišta uz pomoć umjetnih neuronskih mreža*

Zagreb, srpanj, 2012.

Sadržaj

1. Uvod

Mozak kao središte živčanog sustava predstavlja najvažniji organ za čovjeka jer mu omogućuje da razmišlja. Živčane stanice ili neuroni igraju važnu ulogu jer je mozak s ostatkom tijela povezan živcima. Njihova uloga je da prihvaćaju, obrađuju i odašilju podatke. Umjetne neuronske mreže u širem smislu predstavljaju umjetnu repliku ljudskog mozga kojom se nastoji simulirati postupak učenja i obrade podataka. Neuronska mreža predstavlja skup međusobno povezanih jednostavnih procesnih elemenata (jedinica, čvorova) čija se funkcionalnost temelji na biološkom neuronu. Interes za umjetne neuronske mreže javio se nakon što su 1943. Pitts i Mulock prvi dokazali da neuroni mogu imati dva stanja. [5]

Umjetne neuronske mreže odlično se primjenjuju u rješavanju problema klasifikacije i predviđanja, kod rješavanja svih problema kod kojih postoji složena (nelinearna) veza ulaza i izlaza. Dobre strane umjetnih neuronskih mreža nalazimo u procjeni nelinearnosti, mogućnosti rada s nejasnim ili manjkavim podacima, rad s velikim brojem varijabli i parametara,te njihove sposobnosti učenja.

Umjetne neurosnke mreže primjenu pronalaze u raspoznavanju uzoraka, obradi slika i govora, problemima optimizacije, simulacija i slično. Strukturu umjetne neuronske mreže upoznat ćemo pobliže u ovom radu.

1.1. Uvod u QSAR

Quantitative Structure-Activity Relationships (QSAR) novija je metoda praktično razvijena početkom šezdesetih godina prošlog stoljeća uvođenjem dvaju ekstratermodinamičkih metoda. Navedene ekstratermodinamičke metode su: LFER (linear free energy relationships) metoda između bioloških aktivnosti strukturalno sličnih lijekova i kemijskih supstituenata na zajedničkoj molekuli te Free-Wilson analiza temeljena na zbrajanju doprinosa različitih supstitucijskih grupa i višestrukih položaja supstituenata na biološku aktivnost. QSAR metoda se općenito temelji na određivanju relacija između specifične promjenjive veličine (aktivnosti) i fizikalno-kemijskih i/ili strukturalnih svojstava određenih vrijednostima koje nazivamo deskriptori. Pripadajući matematički oblik opisan je jednadžbom (1).

$$Veličina = f(\text{fizikalnokemijska/strukturalna svojstva}) \quad (1)$$

Prva primjena *QSAR* metode pripisana je Corwin Herman Hansch-u koji je 1969. godine predstavio jednadžbu koja je povezala biološku aktivnost s određenim elektroničkim svojstvima i hidrofobnosti seta struktura. [1]

$$\log \frac{1}{C} = k_1 \log P - k_2 (\log P)^2 + k_3 s + k_4 \quad (2)$$

Pri tome C predstavlja koncentraciju – specifičnu promjenjivu veličinu, dok su P oktanol/voda koeficijent particije, te P i s *farmakokinetički* utjecaj na aktivnost (log P – dolazi li spoj gdje bi trebao; s – inducira li elektronska priroda spoja aktivnost) fizikalno – kemijska svojstva.

1.2. Primjena QSAR metode

QSAR nalazi primjenu unutar mnoštva znanstvenih disciplina i područja poput farmaceutske industrije pri određivanju učinkovitosti lijekova[2], zaštite okoliša određivanjem akutnih toksičnosti (koncentracije dovoljne za pojavu pomora riba) [3], analitičke kemije pri određivanju ovisnosti između fizikalno-kemijskih i/ili strukturalnih svojstava i vremena zadržavanja unutar tekućinske kromatografije [4] i sl.

2. Povijesni razvoj neuronskih mreža

Pedesetih godina prošlog stoljeća na konferenciji *Dartmouth Summer Research Project,* kao jedna od grana umjetne inteligencije, profiliralo se neuronsko računarstvo. Na toj konferenciji je najavljeno ostvarivanje vizije računalnog modela koji bi mogao u nekim temeljnim načelima oponašati funkcioniranje mozga.

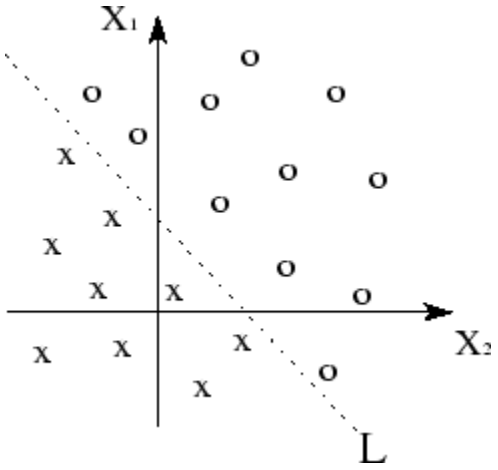
Neki vrlo važni događaji za razvoj neuronskih mreža:

- 1943. Pitts i Mulock su prvi dokazali da neuroni mogu imati dva stanja (umirujuće i pobuđujuće) te da njihova aktivnost ovisi o određenoj vrijednosnoj granici. Ti dokazi su bili temelj za razvoj neuronskih mreža.

- 1949. Hebb je dao prijedlog za pravilo kojim se opisuje proces učenja (Hebbovo pravilo).

- 1956. Rochester i skupina autora predstavljaju prvu simulaciju Hebbovog modela na *Dartmouth Summer Conference,* koji je kolijevka modela neuronskih mreža.

- 1958. prva neuronska mreža *Perceptron* koju je razvio Frank Rosenblatt, u kojoj se učenje razvija u dva sloja, nije mogla rješavati probleme klasifikacije koji nisu bili linearno djeljivi. Linearna djeljivost (Slika 1) je pojam pri kojem se klase uzoraka s n – dimenzijskim vektorom ($x = (x_1, x_2, ..., x_n)$) mogu odijeliti s jednom površinom odluke, koju na slici 1 predstavlja linija L.



Slika 1. Shematski prikaz linearno djeljivog uzorka.

- 1969. Minsky i Papert objavljuju rad u kojem oštro kritiziraju nedostatke *Perceptrona* što dovodi do prekida ulaganja u razvoj neuronskih mreža.

- 1974. unatoč slabim ulaganjima Paul Werbos je razvio višeslojnu *Perceptron* mrežu – MLP, prvu verziju *Backpropagation* mreže koja prevladava nedostatak *Perceptrona* uvođenjem učenja u „skrivenom sloju“.

- 1986. Rumelhart, Hinton i Williams usavršavaju *Backpropagation* mrežu koja vraća ugled neuronskim mrežama jer omogućuje aproksimiranje gotovo svih funkcija i rješavanje praktičnih problema. [5]

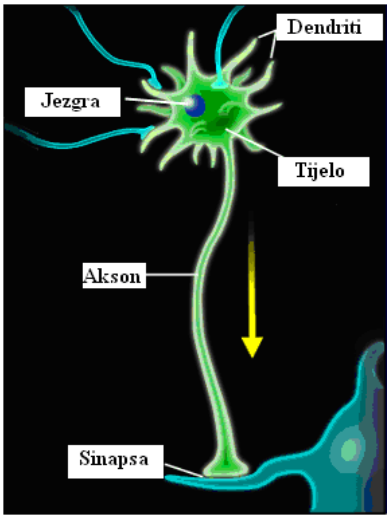
3. Neuronska mreža

Umjetna inteligencija je naziv koji pridajemo svakom neživom sustavu koji pokazuje sposobnost snalaženja u novim situacijama (inteligenciju). Uobičajeno je da se to ime pridaje računalnim sustavima.

Umjetne neuronske mreže su metoda umjetne inteligencije inspirirane i strukturirane prema ljudskom mozgu. [7]

3.1. Biološke neuronske mreže

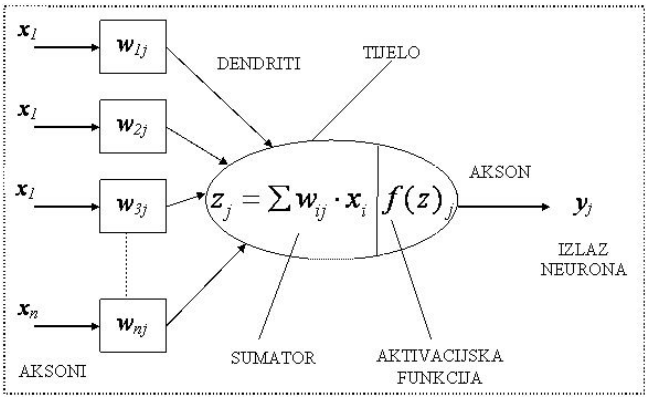
Biološka neuronska mreža je vrlo složena. Sastoji se od tijela, aksona i mnoštva dendrita koji okružuju tijelo neurona (Slika 2). Akson je tanka cjevčica koja s jedne strane sadrži tijelo neurona, a s druge se strane dijeli na niz grana. Na krajevima tih grana nalaze se zadebljanja koja dodiruju dendrite, a ponekad i tijelo drugog neurona. Sinapsa je razmak između završetka aksona prethodnog neurona i dendrita ili tijela sljedećeg neurona. Dendriti su produžeci staničnog tijela, vrlo su tanki u odnosu na veličinu tijela stanice i funkcija im je prikupljanje informacija. Biološke neuronske mreže sastavni su dio ljudskog tijela čiji se mozak sastoji od 100 milijardi neurona, a zaslužne su za izvođenje vrlo važnih funkcija kao što su razmišljanje, učenje, emocije, percepcija, motorika itd. Iako se funkcije neuronskih mreža istražuju već dugi niz godina, brojni procesi, kao i njihov način rada, ljudskom umu još uvijek nisu u potpunosti razumljivi. [5]



Slika 2. Prikaz biološke neuronske mreže.[8]

3.2. Umjetne neuronske mreže

Umjetne neuronske mreže dizajnirane su tako da oponašaju osnovne funkcije bioloških neuronskih mreža. Tijelo biološkog neurona zamjenjuje se sumatorom, ulogu dendrita preuzimaju ulazi (ulazne varijable) u sumator, izlaz (izlazna varijabla) sumatora je akson umjetnog neurona, a uloga praga osjetljivosti bioloških neurona preslikava se iz tzv. aktivacijske funkcije (Slika 3). Veza umjetnog neurona s okolinom ostvaruje se pomoću funkcijske sinaptičke veze biološkog neurona. Težinski faktori mogu biti pozitivan ili negativan broj, a imaju istu funkciju kao i sinapse kod biološkog neurona: povezuju izlaze (izlazne varijable) iz okoline neurona tj. izlaze drugih neurona (aksone) s ulazima sumatora (dendriti). Intenzitet te veze ovisi o iznosu (modulu), a vrsta veze o predznaku težinskog faktora. [5]



Slika 3. Shematski prikaz umjetne neuronske mreže.[9]

Matematički model obrade informacija u računalnom neuronu je sljedeći. Ulazi u neurone indeksirani s i = 1, ..., n primaju ulazne vrijednosti x_i koji su realni brojevi. Svaka ulazna vrijednost x_i množi se s težinskom vrijednošću w_{ji} i zatim se zbrajaju – jednadžba (3).

$$z_j = \sum_{i=1}^{n} w_{ji} \cdot x_i \tag{3}$$

Tako dobiven zbroj z_j obrađuje se pomoću funkcije obrade ili aktivacijske funkcije $f(z)_j$. Izlaz iz neuronske mreže jednak je y_j – jednadžba (4). [10]

$$y_j = f(z_j) \tag{4}$$

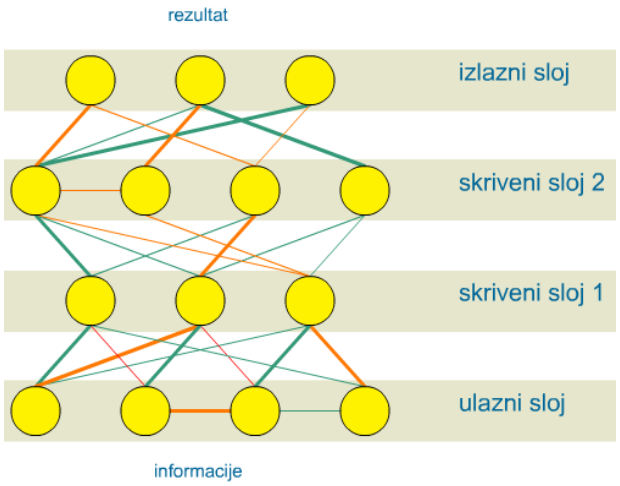
3.2.1. Arhitektura neuronske mreže

Neuroni su obično u umjetnoj neuronskoj mreži organizirani u grupe ili slojeve u kojima se informacije paralelno obrađuju. Tipična neuronska mreža sastoji se od nekoliko slojeva, najčešće dva vanjska, te od jednog ili više međuslojeva ili tzv. skrivenih slojeva (Slika 4).

Vanjski slojevi su:

- ulazni sloj koji učitava podatke iz okoline
- izlazni sloj koji prikazuje rezultat mreže za zadani ulaz

Upravo je skriveni sloj onaj u kojem se uče međuzavisnosti u modelu, informacije neurona se ovdje obrađuju i šalju u neurone izlaznog sloja. [10]



Slika 4. Arhitektura neuronske mreže po slojevima. [12]

Kada ulazni sloj šalje podatke u prvi skriveni sloj, svaka jedinica skrivenog sloja prima vagani ulaz iz ulaznog sloja (početne težine su postavljene slučajno, često u intervalu od -0.1 do 0.1) prema jednadžbi (5).

$$z_j^{[s]} = \sum_i w_{ji}^{[s]} \cdot x_i^{[s-1]} \tag{5}$$

Pri tome je $z_j^{[s]}$ ulaz u neuron *j* u sloju s, $w_{ji}^{[s]}$ je težina veze od neurona *j* (u sloju s) prema neuronu *i* (u sloju *s-1*), a $x_i^{[s-1]}$ je ulazna vrijednost koja se šalje iz prethodnog sloja s-1. Npr. ako je skriveni sloj označen sa s, tada je ulazni sloj označen sa s-1. Jedinice u skrivenom sloju prenose svoje ulaze prema jednadžbi (6).

$$x_j^{[s]} = f\left(\sum_i w_{ji}^{[s]} \cdot x_i^{[s-1]}\right) = f(z_j^{[s]}) \tag{6}$$

Pri tome je $x_j^{[s]}$ izlaz neurona *j* u sloj s, a *f* je prijenosna funkcija (sigmoida, tangens-hiperbolna ili neka druga funkcija).

Ako postoji više od jednog skrivenog sloja, gore navedena prijenosna funkcija upotrebljava se kroz sve skrivene slojeve sve dok se ne dostigne izlazni sloj. [10]

3.2.2. Aktivacijske funkcije

Aktivacijske funkcije dijele se na: ***linearne i nelinearne***.

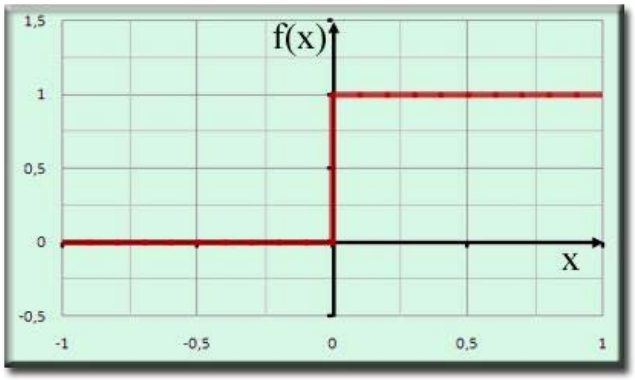
- **Linearne aktivacijske funkcije –** izlaz sumatora množi se nekim faktorom (pojačanjem) i tako dobiva izlaz neurona.
- **Nelinearne aktivacijske funkcije –** mogu poprimiti različite oblike, a najrasprostranjenije su: funkcije praga, sigmoidne, hiperbolične i harmoničke funkcije.

Nelinearne aktivacijske funkcije prevode izlaz sumatora na izlaz neurona preko nelinearnog pojačanja. Na taj način funkcija praga osjetljivosti daje na izlazu neurona 1, ako je izlaz sumatora veći od zadanog broja (prag osjetljivosti), što odgovara ispaljivanju impulsa kod biološkog neurona. Ukoliko neuron nije aktivan, onda je na izlazu neurona 0.

3.2.2.1. Funkcija tipa praga

Funkcija praga (engl. *step function*) je korištena u modelu perceptrona. Poopćena funkcija praga preslikava sve vrijednosti iznad određene granice (praga) u neku fiksnu vrijednost. Sve vrijednosti ispod te iste granice preslikavaju se u nulu. Uobičajeno se za prag koristi vrijednost 0, a sve vrijednosti iznad praga se preslikavaju u vrijednost 1. Korištenjem tako definirane aktivacijske funkcije dobiva se binarni izlaz iz neurona (Slika 5). Izlaz se računa prema jednadžbi (7). [11]

$$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases} \tag{7}$$

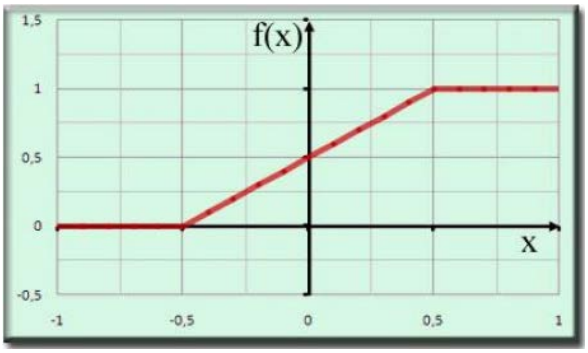


Slika 5. Aktivacijska funkcija tipa praga.

3.2.2.2. Funkcija linearna po dijelovima

Aktivacijska funkcija koja je linearna po dijelovima (engl. *piecewise linear function*) dobiva se ako je izlaz *ADALINE* inačice ograničen na određeni interval. Tako je samo jedan dio preslikavanja linearan, dok periodi ispod i iznad zadane granice prelaze u minimalne, odnosno maksimalne vrijednosti. Poopćena funkcija ima dvije granice za ulaz i dvije fiksne vrijednosti za izlaz izvan granica. Kod spomenute funkcije se uobičajeno uzimaju -0,5 i +0,5 kao granice ulaza te 0 i 1 za vrijednosti izlaza. Takva se funkcija može vidjeti na slici 5 i zadaje se prema jednadžbi (8). [11]

$$f(x) = \begin{cases} 1, & x \geq 0.5 \\ x + 0.5, & -0.5 < x < 0.5 \\ 0, & x \leq -0.5 \end{cases} \tag{8}$$



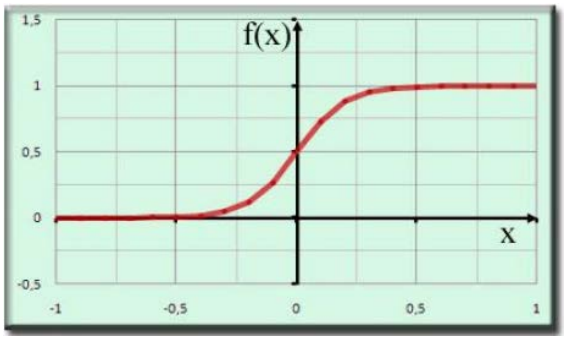
Slika 6. Aktivacijska funkcija linearna po dijelovima

3.2.2.3. Sigmoidna funkcija

Sigmoidna funkcija definirana je jednadžbom **Error! Reference source not found.**. [11]

$$f(x) = \frac{1}{1+e^{-ax}} \tag{9}$$

Sigmoidna funkcija (engl. *sigmoid function*, Slika 7) ima parametar nagiba "a". Povećanjem nagiba smanjuje se prostor linearnosti pa valja biti oprezan s podešavanjem tog parametra. Sigmoidna funkcija je zapravo vrlo slična funkciji praga, ali se dozvoljava područje nesigurnosti unutar određenog intervala i upravo se zato takva funkcija najčešće koristi.

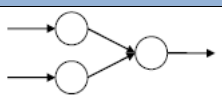
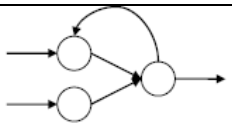
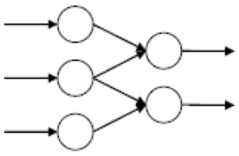
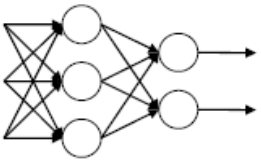
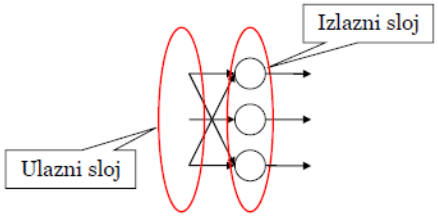
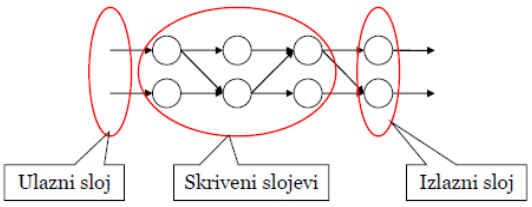


Slika 7. Sigmoidna aktivacijska funkcija

3.2.3. Podjela neuronskih mreža prema topologiji

Građu, odnosno arhitekturu ili topologiju neuronske mreže, možemo razlikovati s obzirom na način povezivanja neurona u mrežu.

Tablica 1. Podjela neuronskih mreža prema topologiji.

TIP MREŽE	OPIS	SHEMATSKI PRIKAZ
Acikličke	Ne sadrže povratne veze.	
Cikličke	Sadrže povratne veze.	
Djelomično povezane	Svaki neuron u prvom sloju ne mora nužno biti povezan sa svakim neuronom u drugom sloju.	
Potpuno povezane	Svaki neuron prethodnog sloja povezan je sa svakim neuronom sljedećeg sloja.	
Jednoslojne	Sastoji se od jednog sloja neurona tzv. izlaznog sloja. Ali osim izlaznog sloja neurona, jednoslojna mreža sadrži i ulazni sloj, koji sadrži ulazne podatke. On se ne broji kao sloj neurona, jer u njemu nema nikakvog računanja. Ulazi u mrežu su spojeni na ulaze neurona izlaznog sloja, a izlazi neurona predstavljaju i izlaz mreže. Nema povratnih veza s izlaza na ulaz.	
Višeslojne	Razlikuju se od jednoslojnih u tome što imaju jedan ili više skrivenih slojeva između ulaznog i izlaznog sloja. Upravo je skriveni sloj onaj u kojem se uče međuzavisnosti u modelu, informacije neurona se ovdje obrađuju i šalju u neurone izlaznog sloja. Izlazi neurona iz s-tog sloja predstavljaju ulaze u neurone iz	

	s+1-og sloja. Kod mreže s propagacijom unaprijed nema povratnih veza među neuronima, tj. nema veza iz neurona jednog sloja (s-tog) prema prethodnim slojevima (1, 2, …, s-1). To kretanje veza samo prema sljedećim slojevima (prema izlazu mreže) zovemo propagacija unaprijed.	

3.2.4. Na koji način se učenjem stvara neuronska mreža?

Rad umjetne neuronske mreže odvija se u dvije osnovne faze:

1. faza učenja ili treniranja mreže

2. faza testiranja

3.2.4.1. Faza učenja ili treniranje mreže

Učenje je proces mijenjanja težina u mreži, a odvija se kao odgovor na podatke izvana koji su predstavljeni ulaznom sloju i u nekim mrežama izlaznom sloju. Podaci koji se predstavljaju izlaznom sloju su željene vrijednosti izlaznih varijabli. Ukoliko su one poznate, radi se o tzv. *nadgledanom učenju* (npr. mreža širenje unatrag). Ukoliko je ulazni vektor jednak izlaznom vektoru, radi se o *autoasocijativnim mrežama*, a ukoliko je različit, radi se o *heteroasocijativnim mrežama*. Kod nekih mreža željeni izlaz ne mora biti predstavljen mreži. U tom slučaju radi se o tzv. *nenadgledanom učenju* (npr. Kohonenova mreža). [13]

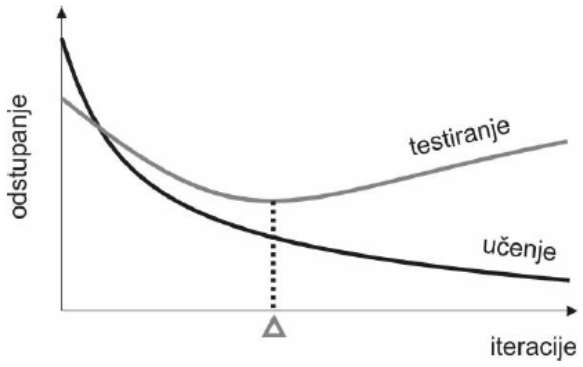
Kod *učenja podrškom* ne postoji učitelj koji određuje kolika je pogreška za određeni par ulaz-izlaz, nego sudac koji samo kaže koliko je određeni korak uučenju dobar (daje ocjenu ili podršku).[12]

Prije samog učenja potrebno je definirati model (ulazne i izlazne varijable), te prikupiti podatke iz prošlosti na kojima će se primijeniti mreža. Prikupljene podatke treba podijeliti u dva poduzorka (uzorak za treniranje i uzorak za testiranje), a ukoliko se za vrijeme učenja planiraju koristiti optimizacijske tehnike za optimiranje duljine učenja i strukture mreže, potrebno je ukupan uzorak podijeliti na tri poduzorka (za treniranje, testiranje i konačnu validaciju). Pravila za ovu podjelu nema, osim što se preporuča najveći dio podataka ostaviti

za treniranje mreže, a manji dio podataka za testiranje i validaciju (npr. 70% za treniranje, 10% za testiranje i 20% za validaciju). Podaci se raspoređuju u poduzorke slučajno, osim kod vremenskih serija gdje treba poštovati vremenski slijed nastajanja promatranja, tj. trenirati mrežu na starijim, a testirati na novijim podacima.

Nakon što je definiran model, pripremljeni ulazni podaci i izabran algoritam neuronske mreže, te pravilo učenja i potrebne funkcije, mrežu treba učiti ili trenirati na pripremljenim podacima iz prošlosti, kako bi ona prepoznala vezu između podataka i bila u mogućnosti na osnovu ulaznih vrijednosti predviđati izlaze. Sama faza učenja je proces podešavanja težina u mreži, koje se odvija u više iteracija ili prolaza kroz mrežu. Jedna iteracija predstavlja učitavanje jednog promatranja iz podataka (jednog ulaznog i izlaznog vektora), ali se zbog povećanja brzine učenja ponekad preporuča učitati više promatranja odjednom, pri čemu se broj promatranja koji se obrađuju u jednoj iteraciji zove epoha. U svakoj iteraciji računaju se nove težine, a kod nadgledanih algoritama i nova greška. Obično se mreža trenira u nekoliko tisuća iteracija. [13]

Umjetnu neuronsku mrežu moguće je, naime, pretrenirati - nakon određenog broja iteracija mreža gubi svojstvo generalizacije (svojstvo dobre klasifikacije za nepoznate ulaze) i postaje stručnjak za obradu podataka iz skupa primjera za učenje dok preostale podatke obrađuje loše. Stalnim praćenjem izlaza iz mreže dobivenog pomoću primjera iz skupa za testiranje moguće je otkriti iteraciju u kojoj dobiveni izlaz najmanje odstupa od željenog odziva (Slika 8). Točnost i preciznost obrade podataka moguće je naposljetku provjeriti nad trećim skupom primjera – skupom za validaciju. [14]



Slika 8. Odstupanje stvarnog izlaza kroz iteracije. [14]

Najvažnije pitanje u ovoj fazi jest koliko dugo treba trenirati mrežu kako bi ona dala što bolji rezultat, odnosno najmanju grešku. Ne postoje egzaktna pravila za dužinu treniranja, te odgovor na ovo pitanje treba potražiti vlastitim eksperimentiranjem ili primjenom optimizacijskih tehnika kao npr. tehnika unakrsnog testiranja. Ova se tehnika može opisati u nekoliko koraka:

- mreža se najprije trenira na određenom broju iteracija (npr. 10 000),
- tako naučena mreža se testira na uzorku za testiranje te pohrani dobiveni rezultat i mreža.
- mreža se zatim nastavlja trenirati na još tolikom broju iteracija (npr. još 10 000), te se dobiveni rezultat uspoređuje s prethodno pohranjenim. Ukoliko je u ponovnom učenju dobiven bolji rezultat, pohranjuje se novi rezultat i nova mreža.
- postupak se ponavlja sve dok se rezultat prestane poboljšavati, a najbolja pohranjena mreža ulazi u daljnji postupak validacije.[13]

3.2.4.2. Testiranje mreže

Testiranje mreže je druga faza rada neuronske mreže, i ona je odlučujuća za ocjenjivanje mreže. Razlika između faze učenja i faze testiranja je u tome što u ovoj drugoj fazi mreža više ne uči, a to znači da su težine fiksne na vrijednostima koje su dobivene kao rezultat prethodne faze učenja. Takvoj mreži se predstavljaju novi ulazni vektori koji nisu sudjelovali u procesu učenja, a od mreže se očekuje da za predstavljen novi ulazni vektor proizvede izlaz. Ocjenjivanje mreže obavlja se izračunavanjem greške ili nekog drugog mjerila točnosti (npr. stope točnosti), na način da se izlaz mreže uspoređuje sa stvarnim izlazima.

Dobivena greška mreže na uzorku za validaciju je rezultat kojim se tumači uspješnost ili neuspješnost neuronske mreže i njezina korisnost u primjeni za predviđanje na budućim podacima. Najčešća greška koja se računa kod neuronskih mreža je srednja kvadratna greška (Root mean square error), prema jednadžbi **Error! Reference source not found.**.

$$RMS\ ERR = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(t_i - o_i)^2} \tag{10}$$

Pri tome je t_i izračunati izlaz koji daje mreža, o_i je željeni (stvarni) izlaz za slučaj (ulazni vektor) *i*, a *n* je broj slučajeva u uzorku. Greška se uprosječuje prema broju izlaznih varijabli i u odnosu na broj slučajeva u uzorku na kojem se računa. [13]

3.3. Algoritmi učenja

Neuronska je mreža u potpunosti određena tek kada je uz njezinu strukturu definiran i algoritam učenja. Algoritam učenja podešava parametre mreže s ciljem postizanja njezinog željenog vladanja. [15]

Osnovna podjela algoritma učenja :

- Učenje korekcijom pogreške
    - ✓ Back propagation
    - ✓ Gauss Newton
    - ✓ Levenberg Marquardt
- Kompetitivno učenje
- Hebbovo učenje
- Boltzmannovo učenje
- Thorndikeovo učenje

3.3.1. Učenje korekcijom pogreške

3.3.1.1. Back propagation

Učenje *back propagation* algoritmom je optimizacijski problem koji se može dobro okarakterizirati nazivom metoda silaznog gradijenta. Pomoću pogreške na izlazu podešavaju se težine od skrivenog do izlaznog sloja i tako redom do ulaznog sloja s ciljem minimiziranja izlazne pogreške. Pogreška se tako propagira unatrag i istovremeno se podešavaju težine. Kod ovakvog učenje problem je što su se težine, koje su se namjestile za prvi primjer, na kraju promijenile u sasvim drugom smjeru zbog čega je potrebno izvesti više prolaska kroz sve primjere. Takvim algoritmom radimo gradijentni spust u lokalni minimum.

Što treba poduzeti s algoritmom učenja kako bi se iz polazne točke postupkom učenja moglo doći do globalnog minimuma?

Problem lokalnog minimuma može se pokazati i u dvodimenzionalnoj ravnini (Slika 10).

Kretanjem korak po korak u smjeru $E(w)$ koja označava niže vrijednosti pogreške daje nam naziv silazni gradijent. Međutim značajna je razlika ako krenemo iz polazne točka 1 ili iz polazne točke 2. Ulazom u lokalni minimum algoritam detektira za porast pogreške ako se nastavimo kretati u bilo kojem smjeru, i *ne zna* izaći iz lokalnog minimuma. Za rješenje ovog problema *backpropagation* algoritam se proširuje momentnim članom (*backpropagation* s momentom) koji unosi inerciju za smjer kretanja što je prethodno pridonosio smanjenju pogreške.[**Error! Reference source not found.**]

Slika 9. Grafički prikaz algoritma učenja Back propagation - lokalni i globalni minimumi.

Pogreška je razlika između željenog i dobivenog odziva neurona *j* u koraku *n*, a zadana je jednadžbom – relacijom (11).

$$e_j(n) = d_j(n) - y_j(n) \tag{11}$$

Pri tome d_j (n) predstavlja željeni odziv neurona j u koraku n, a y_j (n) dobiveni odziv neurona j u koraku n.

Cilj učenja korekcijom pogreške je da se minimizira funkcija pogreške temeljena na pogreškama $e_j(n)$ tako da se dobiveni odziv svih neurona približava željenom odzivu u nekom statističkom smislu. Najčešće se koristi srednja kvadratna pogreška kao funkcija pogreške određena jednadžbom (12).

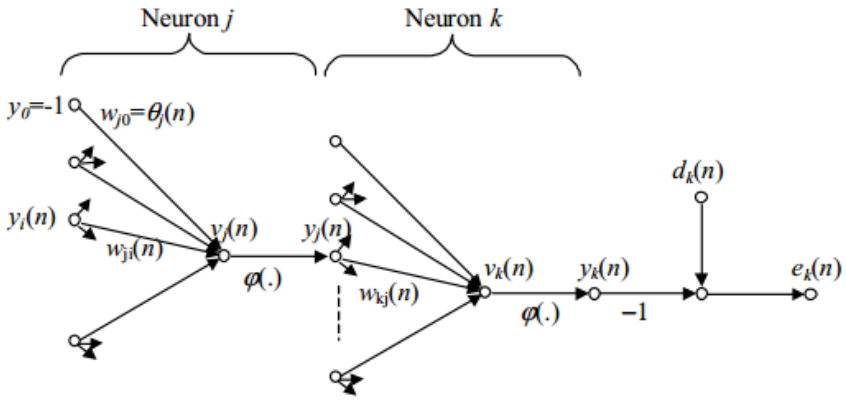
$$J = E\left[\frac{1}{2}\sum_j e_j^{\,2}(n)\right] \tag{12}$$

Pri tome E[.] predstavlja statistički operator očekivanja, a sumacija se radi za sve neurone u izlaznom sloju mreže

Problem s minimizacijom funkcije J je da je potrebno znati statistička svojstva slučajnog procesa $e_j(n)$. Zbog toga se kao procjena pogreške koristi trenutna vrijednost pogreške u nekom koraku n kao funkcija koja se minimizira i dana je jednadžbom **Error! Reference source not found.**.

$$E(n) = \frac{1}{2}\sum_j e_j^{\,2}(n) \tag{13}$$

Ovakav pristup daje približno rješenje.



Slika 10. Prikaz neurona sa skrivenim slojevima [16]

Ako se uzme da je aktivacija na ulazu u nelinearni blok neurona j jednaka:

$$v_j(n) = \sum_{i=0}^{p} w_{ji}(n)\, y_i(n) \tag{14}$$

gdje je p ukupni broj ulaza u neuron j

Izlaz neurona j jednak je:

$$y_i(n) = \varphi_j\left(v_j(n)\right) \tag{15}$$

gdje je φ_j aktivacijska funkcija koja se označava još i $f(z)_j$.

Korekcija težine izvodi se u smjeru negativnog gradijenta. Gradijent se može izračunati na sljedeći način: (lančano pravilo za deriviranje)

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)}\frac{\partial e_j(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)}\frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{16}$$

$$\frac{\partial E(n)}{\partial e_j(n)} = e_j(n) \tag{17}$$

gdje su e_j nezavisni

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi'_j(v_j(n)) \tag{18}$$

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \tag{19}$$

slijedi iz (11) jer su d_j konstante

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n) \tag{20}$$

i tu se koristi da su w_{ji} nezavisni

Na temelju prethodnih izraza dobivamo:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi_j'(v_j(n))y_i(n) \tag{21}$$

Korekcija težina definirana je delta pravilom (metoda najbržeg spusta):

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} \tag{22}$$

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} \tag{22}$$

Dakle dobiveni izraz može se pisati kao:

$$\Delta w_{ji}(n) = \eta e_j(n)\varphi_j'(v_j(n))y_i(n) = \eta \delta_j(n) y_i(n) \tag{23}$$

gdje je $\delta_j(n)$ tzv. lokalni gradijent:

$$\delta_j(n) = e_j(n)\varphi_j'(v_j(n)) \tag{24}$$

Konačna jednadžba za korekciju Δw_{ji} (n) težina od neurona i na neuron j dana je delta pravilom:

(korekcija) = (parametar) × (lokalni gradijent neurona j) × (i-ti ulaz u neuron j) [17]

$$\Delta w_{ji}(n) = \eta\ \delta_j(n)\ y_i(n) \tag{25}$$

Pri tome je η pozitivna konstanta koja određuje brzinu učenja. Promjena težine proporcionalna je pogrešci na izlazu i iznosu pripadnog ulaza. Konstanta η mora se pažljivo odabrati

- mali η daje stabilnost ali je učenje sporo

- veliki η ubrzava učenje ali donosi rizik nestabilnosti [18]

3.3.1.2. Kompetitivno učenje

Kompetitivno učenje je učenje bez nadzora. Neuroni se natječu za priliku da se aktiviraju, a samo jedan neuron može biti aktivan u bilo kojem trenutku. Kompetitivno učenje se primjenjuje za probleme klasifikacije.

Tri osnovna elementa kompetitivnog učenja jesu:

- Skup jednakih neurona koji imaju slučajno raspoređene težine
- Ograničenje "jačine" svakog neurona tj. težina u mreži
- Mehanizam koji omogućuje natjecanje neurona za pravo na odziv na danu pobudu, tako da je samo jedan neuron (ili jedan po grupi) aktivan u jednom momentu [19]

3.4. Primjena neuronskih mreža

Neuronske se mreže primjenjuju za rješavanje onih problema čija rješenja nisu u potpunosti određena, te nije nužno da rješenje bude ispravno u 100% slučajeva. Unatoč tome što neuronske mreže ne daju rezultate sa 100%-tnom točnošću, one ipak pronalaze široku primjenu u mnogim područjima ljudske aktivnosti.

3.4.1. Prepoznavanje uzoraka

Ovdje je potrebno istaknuti područja primjene poput procesiranja slike dobivene putem senzora (upotrebljava se u vojne svrhe za prepoznavanje meta, te u medicinske svrhe za analizu razmaza), procjenjivanja rijetkih novčića, kontrole kvalitete, detekcije bombi u prtljazi i sl.

3.4.1.1. Prepoznavanje znakova

Ovo područje primjene je već uvelike komercijalizirano. U praksi se pokazalo da jednom uvježbana mreža pokazuje visoku razinu ispravnosti u pogledu prepoznavanja čak i onih znakova koji nisu služili kao primjeri za uvježbavanje. Mnoga istraživanja na ovom području usmjerena su na prepoznavanje istočnjačkih pisama.

3.4.2. Obrada govora

Neuronske se mreže mogu primijeniti za obradu prirodnog govora, prevođenje teksta u govor, davanje uputa strojevima glasovnim naredbama, automatsko prevođenje, sigurnosne sustave sa glasovnom identifikacijom, kao pomoć gluhim osobama i osobama koje su fizički nepokretne. Problem koji se ovdje javlja je taj što većina neuronskih mreža može adekvatno prepoznavati govor samo one osobe koja ju je uvježbavala, a i tada je količina riječi koju može prepoznati ograničena.

3.4.3. Obrada signala

Neuronske su se mreže pokazale pogodnima za smanjenje šuma u izobličenim električnim signalima, kao i u razdvajanju signala iz višesignalnog prijenosa. Prva takva mreža je bila MADALINE koju je izradio Widrow, a koja uklanja šum u telefonskim linijama. Još jedna primjena je otkrivanje neuspjelog paljenja motora i to na temelju šuma.

3.4.4. Kompresija slike

Slike možemo kompresirati ili dekompresirati u stvarnome vremenu pomoću auto asocijativnih neuronskih mreža. Auto asocijativne mreže omogućavaju svođenje zapisa o slici na manje podataka, pomoću kojih će mreža asocijacijom rekonstruirati ostale podatke. No kako mreža ipak ne može rekonstruirati podatke bez gubitaka, ovoj se primjeni ne posvećuje velika pažnja, odnosno i dalje se više koriste standardni načini kompresiranja slika.

3.4.5. Kontrola procesa

Za područje kontroliranja procesa u složenim sustavima pomoću neuronskih mreža, postoji veliki interes. Prednost neuronskih mreža je, u ovom slučaju, njihova fleksibilnost, odnosno mogućnost pronalaženja adekvatnih rješenja na osnovu nepotpunih podataka. Mreže se uvježbavaju tako što im se omogući da «promatraju» rad sustava. Jednom uvježbana mreža može uspješno kontrolirati sustav čak i u uvjetima kada postoje određeni problemi. Ovu vrstu primjene neuronske mreže možemo pronaći u naftnoj industriji, distribuciji električne energije, a čak se testira i za kontrolu aviona.

3.4.6. Optimizacija procesa

Kod optimizacije procesa, neuronsku mrežu također uvježbavamo na način da joj omogućimo da promatra rad sustava. Zatim mreži zadajemo željeno završno stanje, te ona optimizira odvijanje procesa kako bi se postiglo željeno stanje. Primjer je optimizacija rada motora kako bi se uštedijelo gorivo.

3.4.7. Poslovna primjena

Neuronskim se mrežama koriste financijske institucije, a također se koriste nespretno i u marketinške svrhe. Financijske ustanove poput banaka ili kartičnih kuća, koriste neuronske mreže za procjenu rizika pružanja usluga svojim mušterijama. U marketinške svrhe se npr. koriste za procjenu u koje doba dana treba zvati koje brojeve telefona kako bi se reklamirani proizvod najbolje prodao.

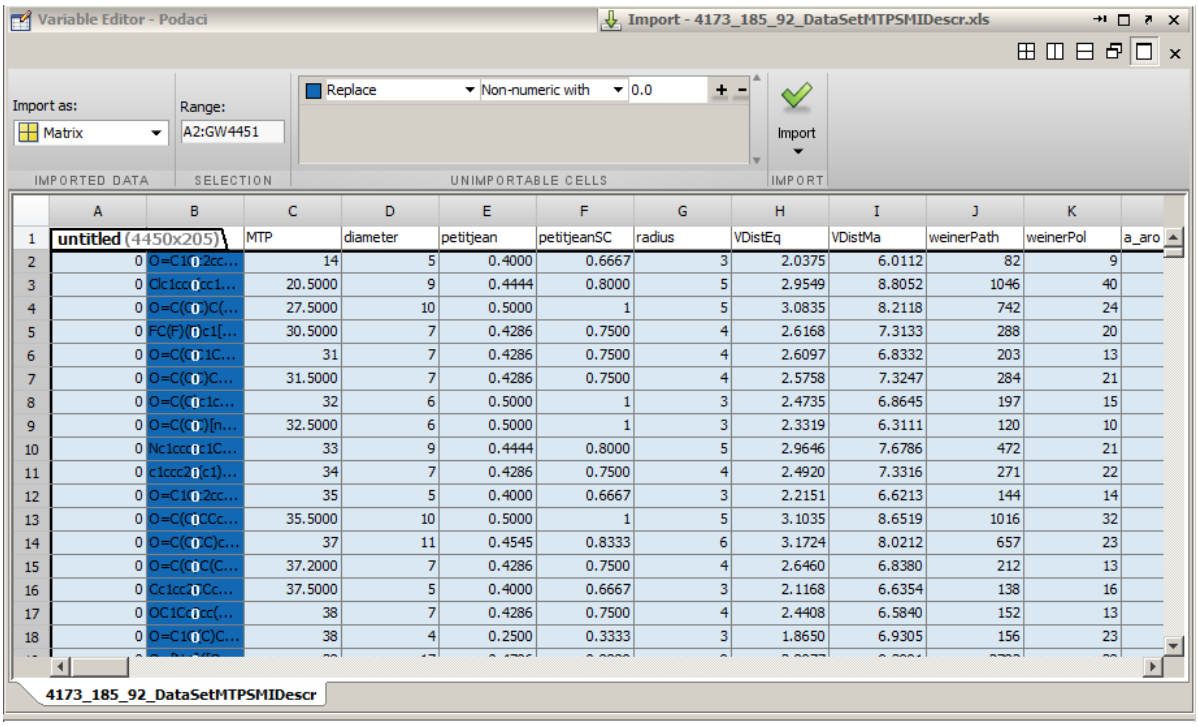
3.4.8. Primjena u medicini

Pomoću neuronskih mreža nastoje se modelirati dijelovi ljudskog tijela. Primjer toga je modeliranje ljudskog kardiovaskularnog sustava. Model se izrađuje prema pojedincu, te se na osnovu njega može odrediti tijek razvoja bolesti. Neuronske se mreže u medicini također koriste za detekciju bolesti na osnovu analize rezultata dobivenih pomoću raznih biomedicinskih senzora. Mrežu se može poučiti i uspostavljanju dijagnoze na osnovu simptoma, a još jedna od vrlo zanimljivih primjena je i razvoj elektroničkog nosa. Elektronički nos se koristi u telemedicini, tj. pomoću njega se detektiraju mirisi za vrijeme operacije, a podaci o mirisu se zatim šalju stručnjacima koji telekomunikacijskim putem učestvuju u operaciji. [19]

4. Pregledni dio

Unutar preglednog dijela korišteni su eksperimentalni podaci 4400 spojeva i 202 deskriptora pri treniranju mreže, te 50 spojeva i 202 deskriptora za testiranje mreže. [20] Navedeni *dataset* preuzet je sa mrežnih stranica *ChemoInformatics*. Za treniranje mreže i testiranje korišten je programski paket Matlab 2012a i *Neural Network Fitting Toolbox*.

Prije svega potrebno je unijeti podatke iz Excel tablice unutar samog Matlaba i raspodijeliti podatke u pripadajuće matrice. Podatke ćemo unijeti odabravši iz menija *Import Data* te na taj način uvesti Excel tablicu. Nakon toga dobijemo prozor prikazan na sljedećoj slici (Slika 11).

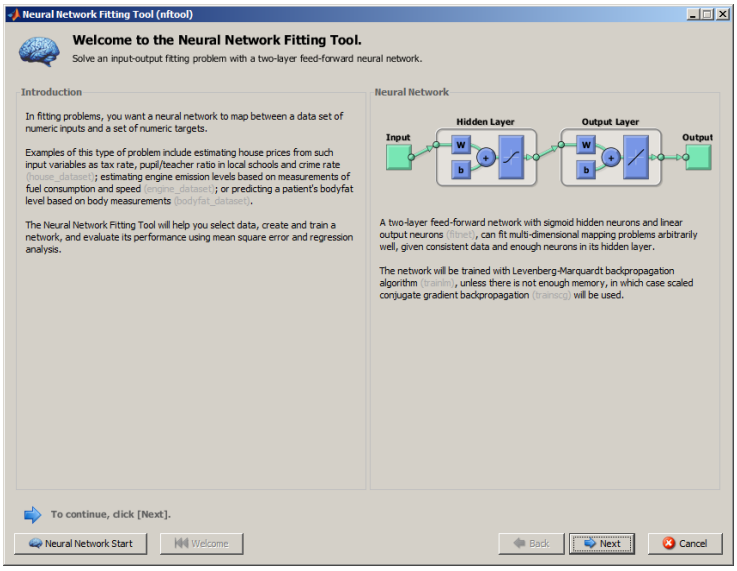


Slika 11. Prikaz uvoza podataka iz Excel tablice.

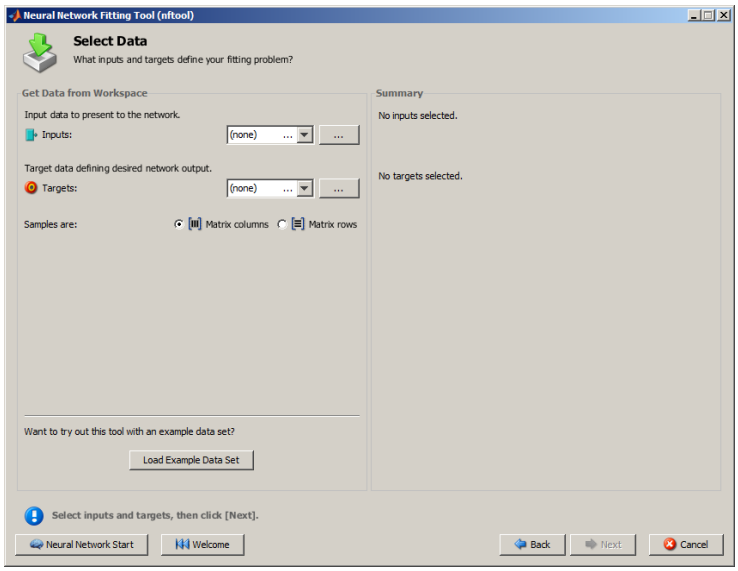
Zatim je potrebno uvesti podatke bez naslova i naziva spojeva unutar matrice Podaci, te matricu Podaci raspodijeliti u zasebne matrice: Tv – temperatura vrelišta za treniranje, Des – deskriptori za treniranje, Tv_test – temperatura vrelišta za testiranje i Des_test – deskriptori za testiranje. Valja napomenuti kako unutar toolboxa postoji opcija za transponiranje matrice, te zato matrice za treniranje nije potrebno transponirati, no pri testiranju te opcije nema, te je matrice Tv_test i Des_test potrebno transponirati u oblik povoljan za korištenje s neuronskom mrežom. (Prilog 1)

Nakon uvođenja podataka i raspodijeljivanja u matrice pokrećemo *toolbox* iz menija *Start* u donjem lijevom kutu Matlab radnog okruženja. Otvori se uvodni prozor *Neural Network*

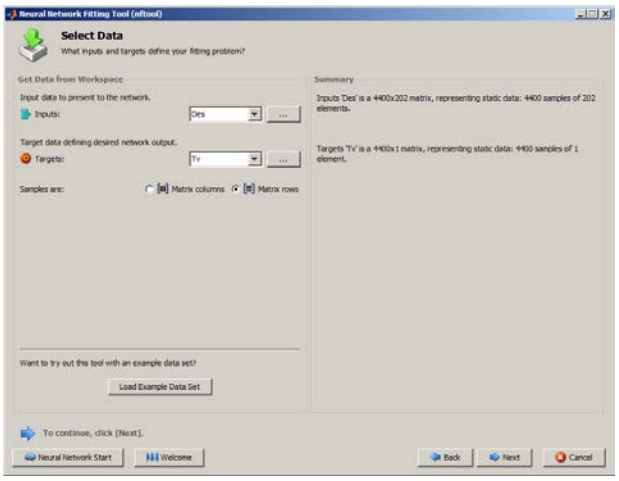
Fitting Tool-a (Slika 12) te pritiskom na aktivno polje *Next* prelazimo na prozor za unos podataka (Slika 13). Odaberemo ulazne i željene vrijednosti te odabirom *Matrix Rows* transponiramo matricu u oblik prihvatljiv za treniranje neuronske mreže (Slika 14).



Slika 12. Prikaz uvodnog prozora Neural Network Fitting Tool-a.

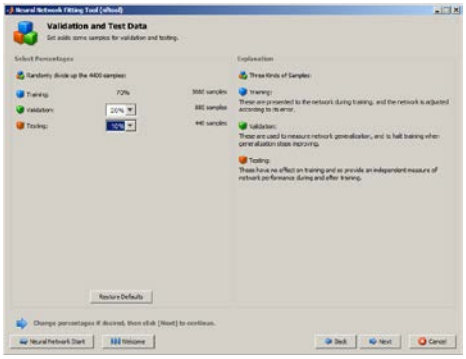


Slika 13. Prikaz prozora za unos podataka za treniranje.

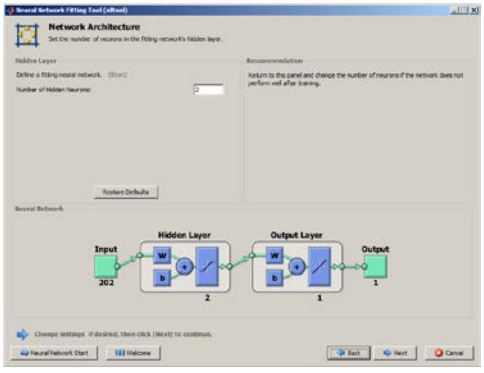


Slika 14. Prikaz prozora za unos podataka s unešenim podacima.

Nakon unešenih podataka pritiskom na aktivno polje *Next* prelazimo na prozor za odabir postotka podataka korištenih za treniranje, validaciju i testiranje. Odaberemo za validaciju 20% te za testiranje 10% (Slika 15). Valja napomenuti kako je testiranje spomenuto u okviru treniranja mreže unutarnje testiranje, dok ćemo nakon treniranja testirati s vanjskim podacima. Zatim pritiskom na aktivno polje *Next* prelazimo na prozor za odabir skrivenih slojeva mreže (Slika 16).

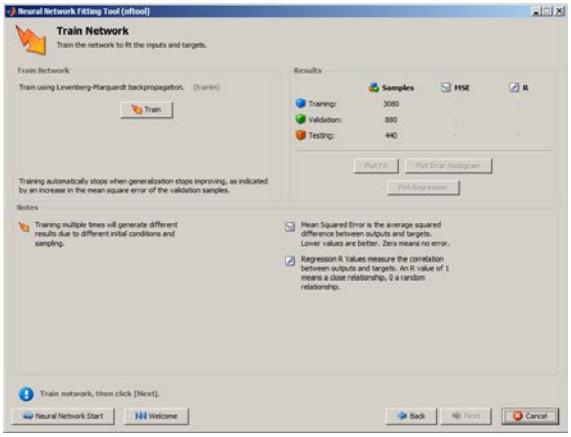


Slika 15. Prikaz prozora za odabir podataka za validaciju i testiranje.



Slika 16. Prikaz prozora za odabir skrivenih slojeva mreže.

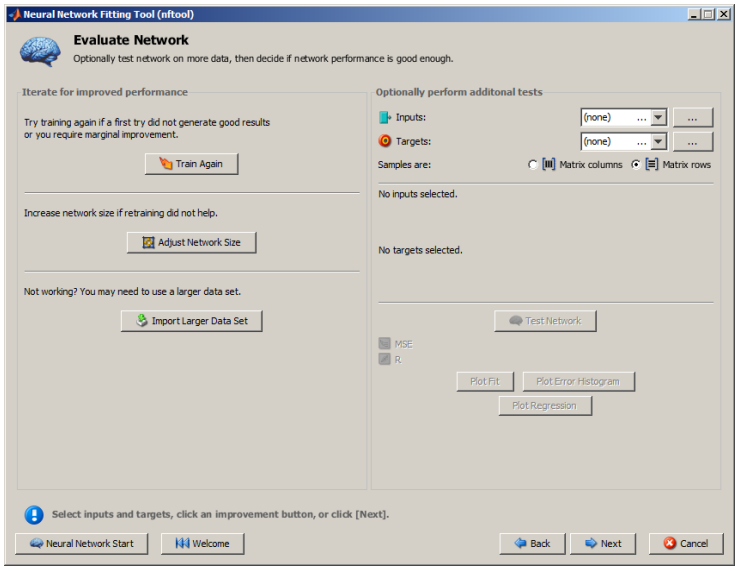
Pritiskom na aktivno polje *Next* otvara se prozor za početak treniranja mreže (Slika 17) i pritiskom na aktivno polje *Train* počinje treniranje i otvara se prozor za treniranje (Slika 18).



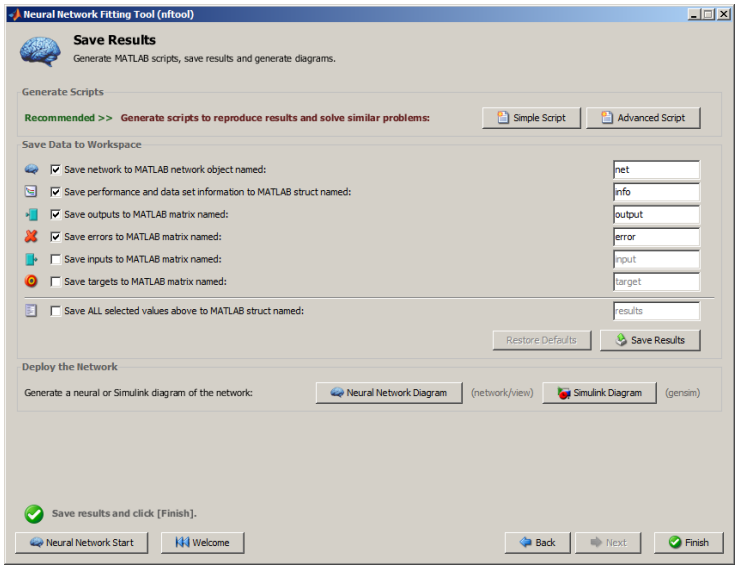
Slika 17. Prikaz prozora za početak treniranja mreže.

Slika 18. Prikaz prozora za treniranje mreže.

Nakon završetka treniranja mreže pritiskom na aktivno polje *Next* otvara se prozor za evaluaciju istrenirane mreže (Slika 19) i još jednim pritiskom na aktivno polje *Next* otvara se prozor za spremanje mreže (Slika 20) nakon čega je proces treniranja završen.



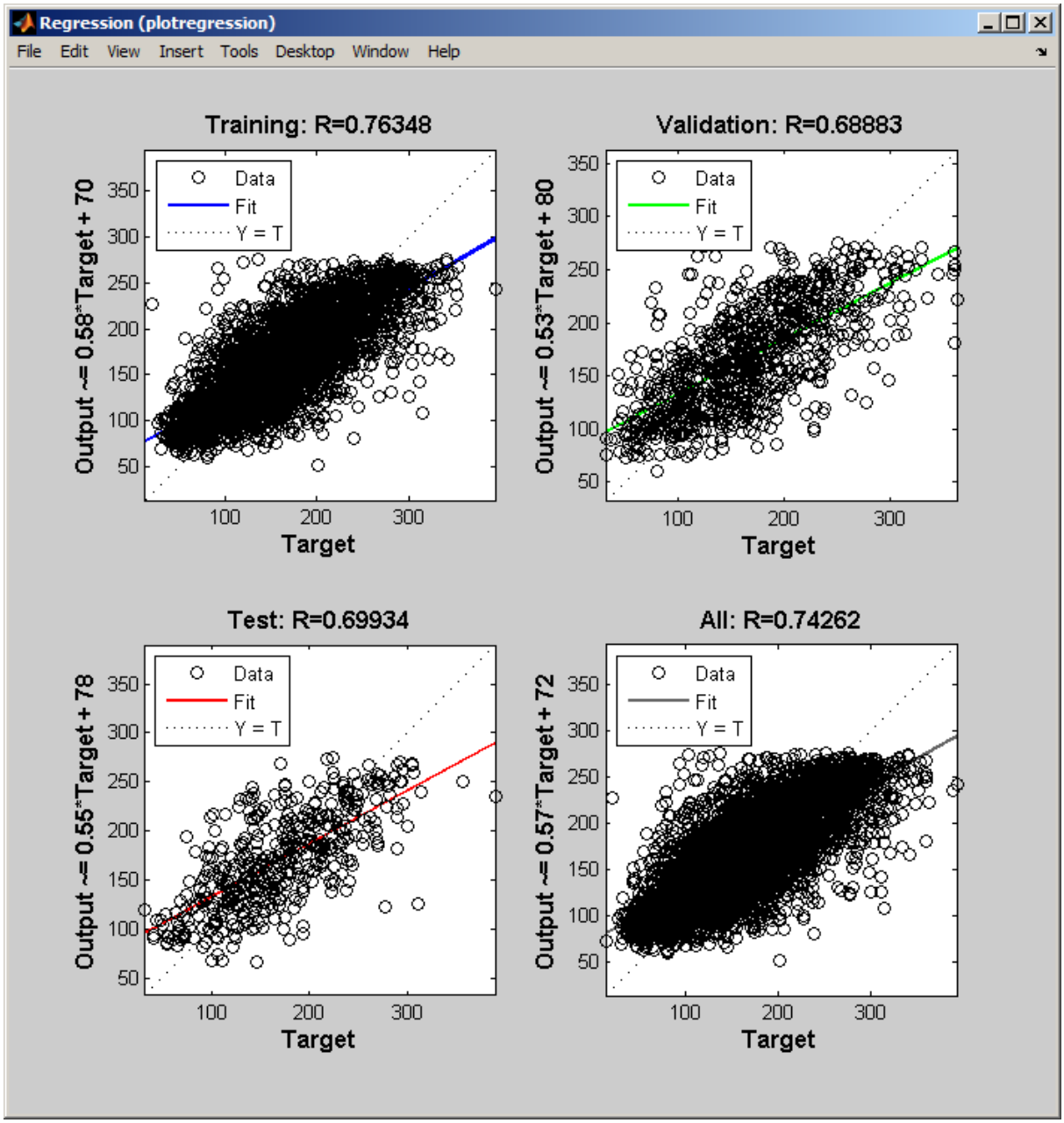
Slika 19. Prikaz prozora za evaluaciju mreže.



Slika 20. Prikaz prozora za spremanje mreže.

5. Rezultati

Kao rezultat treniranja (Slika 21) dobivene su prihvatljive vrijednosti koeficijenta korelacije R i to za treniranje R = 0,76348; za validaciju R = 0,68883; te za testiranje R = 0,69934.



Slika 21. Grafički prikaz regresije treniranja, validacije i testiranja mreže.

Tijekom treniranja mreže mijenjaju se težine između ulaznih i izlaznih podataka. U svakoj iteraciji računaju se nove težine kako bi mreža dala najmanju pogrešku. Naučena mreža u sljedećem koraku se validira s ciljem pronalaska boljeg rezultata uz pohranu podataka.

Postupak se ponavlja sve dok se rezultat poboljša. Dobivena mreža testira se na uzorku za testiranje, a dobiven rezultat se uzima kao konačno mjerilo uspješnosti mreže.

Nakon treniranja mreže i *plotanja* grafičkog prikaza regresije treniranja, validacije i unutarnjeg testiranja slijedi vanjsko testiranje s deskriptorima za preostalih 50 spojeva. Dobivene su vrijednosti Tablica 2 koje se u potpunosti ne podudaraju s originalnim vrijednostima temperature vrelišta za tih 50 spojeva, te je zaključak da je za provedbu *QSAR*-a ipak učinkovitija metoda poput metode parcijalnih najmanjih kvadrata.

Tablica 2. Usporedba eksperimentalnih temperatura vrelišta i temperatura vrelišta dobivenih uz pomoć umjetnih neuronskih mreža.

Tv (ANN)	Tv (eksp)	Tv (ANN)	Tv (eksp)
133,631	210	**176,358**	180
242,4828	252	**169,7372**	169
178,7029	164	**142,2768**	119
133,7803	156	**166,9445**	153
176,8881	118,5	**109,9049**	95
100,7528	80	**130,9041**	223
146,7127	179	**154,5679**	170
180,5087	231	**129,6752**	94
158,0855	160,5	**205,6429**	163
185,0233	186	**117,0997**	45
233,1188	269	**79,13346**	89
75,25432	68	**155,6121**	233
145,7496	116	**157,4732**	152
93,51875	75	**227,1264**	203
199,0473	139	**80,76426**	113,5
192,279	295	**149,8917**	156
162,1013	100	**183,0153**	189
186,218	281	**111,0812**	209
91,16086	198	**155,8788**	120
125,9409	142	**147,3234**	178
212,9976	122	**123,6295**	131
186,2269	146,5	**203,6799**	234
178,1284	148	**208,2676**	106
73,6352	128	**147,7615**	157
185,8164	257	**210,7403**	242
		118,9759	90

6. Zaključak

Neuronske mreže kao računalni sustav predstavljaju noviji i relativno dobar alat za obradu i predviđanje podataka ali nisu u potpunosti točne. Korisne su u različitim znanostima poput psihologije, medicine, fizike, ekonomije, neurologije i slično.

Samu neuronsku mrežu definiraju slojevi, aktivacijske funkcije, težinski koeficijent... Upotreba neuronskih mreža zahtijeva 2 odnosno 3 koraka. Prvi je učenje (treniranje) mreže gdje je potrebno definirati ulazne i izlazne podatke, odrediti aktivacijsku funkciju, težinske koeficijente i slično. Drugi je testiranje mreže na nekim novim podacima kako bismo dobili odgovor od mreže (izlazni podatak), a potom treći dio: ocjenjivanje mreže izračunavanjem greške i usporedbom izlaza mreže stvarnim izlazima. Kod treniranja treba voditi računa ukoliko ne želimo da dođe do pretreniranja.

Kako su dobivene vrijednosti koje se u potpunosti ne podudaraju s originalnim vrijednostima temperature vrelišta za tih 50 spojeva, može se zaključiti da je za provedbu *QSAR*-a ipak učinkovitija neka druga metoda poput metode parcijalnih najmanjih kvadrata.

7. Literatura

1. Handbook of Chemometrics and Qualimetrics Volume 2, Elsevier
2. Patani GA, LaVoie EJ, "*Bioisosterism: A Rational Approach in Drug Design*". Chemical Reviews **96** (1996) 3147–3176
3. http://www.epa.gov/nrmrl/std/cppb/qsar/index.html (*14. listopada, 2011.*)
4. Tomasz Baczek, Roman Kaliszan, "*Combination of linear solvent strength model and quantitative structure–retention relationships as a comprehensive procedure of approximate prediction of retention in gradient liquid chromatography*". Journal of Chromatography A, **962** (2002) 41–55
5. K. Mačkić, Primjena neuronskih mreža u fizici i informatici (diplomski rad), Prirodoslovno-matematički fakultet, 2009.
6. http://hr.wikipedia.org/wiki/Umjetna_inteligencija
7. http://oliver.efos.hr/informatika/strucni/files/p10_52i3.pdf
8. http://binaurals.110mb.com/str2.html
9. http://autopoiesis.foi.hr/wiki.php?name=KM%20-%20Tim%2023&parent=NULL&page=5.%20Neuronske%20mreze
10. http://eris.foi.hr/11neuronske/nn-predavanje3.html
11. D. Klemen, Raspoznavanje uzoraka primjenom umjetne kolonije pčela, Fakultet elektrotehnike i računarstva, 2011.
12. http://eris.foi.hr/11neuronske/nnetwork_v6_a3.html
13. http://eris.foi.hr/11neuronske/nn-predavanje4.html
14. B. Dalbelo Bašić, M. Čupić, J. Šnajder, Umjetne neuronske mreže, Fakultet elektrotehnike i računarstva, 2008.
15. I. Petrović, M. Baotić, N. Perić, Inteligentni sustavi upravljanja: Neuronske mreže, evolucijski i genetički algoritmi (skripta), Fakultet elektrotehnike i računarstva, (2011./2012.)
16. http://www.fer.unizg.hr/_download/repository/Automati_2011_12_5.pdf
17. S. Lončarić, Neuronske mreže: proces učenja (predavanja), Fakultet elektrotehnike i računarstva
18. D. Vukadinović, Procjena brzine vrtnje vektorski upravljanog asinkronog motora primjenom neuronske mreže (doktorska disertacija), Fakultet elektrotehnike, strojarstva i brodogradnje,2005.

19. V. Vidulin, Neuronske mreže: Algoritmi i primjene u obrazovanju (diplomski rad), Filozofski fakultet u Rijeci, 2005.
20. http://www.cheminformatics.org/datasets/karthikeyan/4173_185_92_DataSetMTPSMIDescr.zip

8. Prilozi

8.1. Prilog 1

```
>> Tv=Podaci(1:4400,1);

>> Des=Podaci(1:4400,2:203);

>> Tv_test=Podaci(4400:4450,1);

>> Des_test=Podaci(4400:4450,2:203);

>> Tv_test=transpose(Tv_test);

>> Des_test=transpose(Des_test);
```

8.2. Prilog 2

```
>> Tv_ANN=net(Des_test);
```