Sveučilište u Zagrebu
Fakultet prometnih znanosti
Zavod za inteligentne transportne sustave
Katedra za primijenjeno računarstvo

Vježba:	#7
Kolegij:	Umjetna inteligencija
Tema:	Neuronske mreže - primjena
Vježbu pripremili:	doc. dr. sc. Edouard Ivanjko Martin Gregurić, mag. ing. traff. Mario Buntić, mag. ing. traff.

Upute za izradu vježbi/zadataka

Prije dolaska na vježbu potrebno je proučiti pripremu za vježbu. Vježbe je potrebno izraditi pomoću alata koji se koriste. Vrijeme za izradu zadanih zadataka iznosi 90 minuta.

Cilj vježbe

Pregled osnovnih MATLAB metoda i funkcija potrebnih za proces učenja umjetne neuronske mreže u okviru skupa alata „Neural Network Toolbox“. Upoznavanje s osnovnim opcijama alata „Neural Network Training“. Izrada ciklične umjetne neuronske mreže sa provedbom postupka učenja u cilju pronalaženja uzoraka u prometnim podatcima dobivenih s brojila prometa cestovnih vozila.

Opis vježbe

MATLAB Neural Network Toolbox podržava više sustavnih metoda učenja, odnosno prilagođavanja težina višeslojnih umjetnih neuronskih mreža kako bi se postigao željeni stupanj dobrote izlaza. Svim metodama zajedničko je da promjenom vrijednosti težina teže smanjiti veličinu pogreške između izračunatog izlaznog i zadanog ciljnog vektora. Kroz vježbu će se pronaći određeni uzorci u unaprijed pripremljenim podatcima sa brojila prometa cestovnih vozila prikupljanih kroz tjedan dana, posredstvom umjetnih neuronskih mreža i odgovarajuće metode učenja.

Standardni parametri i metode učenja umjetne neuronske mreže u MATLAB Neural Network Toolbox skupu alata

Većina korištenih umjetnih neuronskih mreža podučava se metodom povratnog rasprostiranja (eng. „*backpropagation*“) pogreške. MATLAB Neural Network Toolbox ime više vrsta metoda podučavanja metodom povratnog rasprostiranja. Najkorištenija je metoda povratnog rasprostiranja sa silaznim gradijentom (eng. „*Gradient descent backpropagation*“) u MATLAB okruženju se poziva funkcijom „traingd“. Ova metoda vrši učenje u kojem se koristi metoda povratnog rasprostiranja za računanje derivacija performansi učenje „perf“ prema težinama i biasima u varijabli „X“. Svaka se varijabla prilagođava prema silaznim gradijentu koji se računa prema sljedećoj jednadžbi:

$$dX = lr * dperf/dX$$

gdje je:

lr – stupanj dobrote učenja (eng. „*Learning rate*“);
dperf – derivacija performansi prethodnog učenja;
dX – derivacija parametra koji se prilagođava.

Parametri koji se mogu mijenjati u funkciji „traingd“ su prikazani u tablici 1. sa svojim početnim vrijednostima. Napomenimo kako su date MATLAB naredbe pisane za objekt umjetne neuronske mreže naziva „net“.

Tab 1. Tablica parametra funkcije „traingd“ sa objašnjenjem i početnim vrijednostima

MATALB naredba	**Početne vrijednosti**	**Obajšnjenje**
net.trainParam.epochs	10	Maksimalni broj iteracija/epoha učenja
net.trainParam.goal	0	Ciljna vrijednost greške učenja završetaka učenja
net.trainParam.showCommandLine	0 (ili 1)	Generiranje izlaza u obliku ispisa u komandnom prozoru vrijednosti izlaznih grešaka
net.trainParam.showWindow	1 (ili 0)	Prikaži prozor alata „Neural Network Training“
net.trainParam.lr	0.01	Stupanj dobrote učenja
net.trainParam.show	25	Razlika između prikaza

		vrijednosti greške učenja iteracije/epohe
net.trainParam.time	inf	Vrijednost vremena u sekundama do kraja učenja
net.trainParam.min_grad	10^{-10}	Vrijednost minimalnog gradijenta kod kojeg prestaje učenje

Analizirajmo primjer prepoznavanja brojeva većih od dva u kojem varijabla „p“ predstavlja jednostavan skup podataka za učenje brojeva od vrijednosti „0“ do „5“, a varijabla „t“ će predstavlja ciljni vektor. Jedinicom će biti prepoznati brojevi veći od vrijednosti „2“, a nulom vrijednosti manje od vrijednosti „2“.

```
p = [0 1 2 3 4 5];
t = [0 0 0 1 1 1];
```

Za potrebe primjera stvorimo cikličnu neuronsku mrežu (engl. *„feed-forward neural network*“) pomoću funkcije „newff“ i nazovimo objekt u kojem spremamo umjetnu neuronsku mrežu imenom „net“. Prvi sloj neka ima tri neurona, a drugi jedan, prijenosna funkcija prvog sloja neka bude Tan-sigmoidnu funkciju, a drugog linearna funkcija, te za metodu učenja izaberimo metodu povratnog rasprostiranja sa silaznim gradijentom.

```
net=newff(minmax(p),[3,1],{'tansig','purelin'},'traingd');
```

Napomenimo kako se performanse umjetne neuronske mreže u Neural Network Toolbox-u mjere u jedinici srednjeg kvadrata pogreške - MSE (eng. „*Mean Squared Error*“).

Uzmimo:

x_n – podatci za učenje;

$\hat{x}_n$ - izlazni rezultati umjetne neuronske mreže;

n – broj elementa podataka za učenje i izlaznog rezultata;

N – ukupni broj elemenata za učenje.

MSE računamo prema sljedećim jednadžbama:

$$1 \leq n \leq N$$

$$MSE\left(\hat{x},x\right) = SE\left(\hat{x},x\right)/N$$

$$SE\left(\hat{x},x\right) = \sum_{n=1}^{N} SE\left(\hat{x}_n,x_n\right)$$

$$SE\left(\hat{x}_n,x_n\right) = (\hat{x}_n - x_n)^2$$

Postavimo simulaciju umjetne neuronske mreže na prikazivanje vrijednosti greške učenja svakih „50“ epoha/iteracija. Maksimalni broj iteracija/epoha neka bude „300“, stupanj dobrote učenja neka bude „0.01“, ciljana vrijednost učenja „10^{-3}“, te neka se prikazuju parametri učenje tokom trajanja procesa učenja u komandom prozoru MATLAB-a:

```
net.trainParam.show = 50;
net.trainParam.epochs = 300;
net.trainParam.goal = 1e-3;
net.trainParam.lr = 0.01;
net.trainParam.showCommandLine = 1;
```

Funkcija koja će primijeniti metodu učenja povratnog rasprostiranja sa silaznim gradientom ili bilo koju drugu metodu učenja koja je određena funkcijom „newff“ je „train“ koja ima sljedeću sintaksu:

- *[net,tr,Y,E,Pf,Af] = train(net,P,T,Pi,Ai)* – funkcija za pokretanje procesa učenja umjetne neuronske mreže je „train“. Vraća varijablu „net“ - ime objekta u kojem je spremljena naučena umjetna neuronska mreža, „tr“ – broj iteracije/epohe i pripadajuću grešku, „Y“ – izlazne vrijednosti umjetne neuronske mreže, „E“ – greške učenja, „Pf“ – konačna ulazna kašnjenja, „Af“ – konačno kašnjenje po slojevima. A uzima varijablu „net“ - ime objekta u kojem je spremljena nenaučena umjetna neuronska mreža, „P“ ulaze u mrežu, „T“ – ciljani vektor, „Pi“ – početna ulazna kašnjenja, „Pi“ – početna kašnjenja po slojevima.

Stoga u našem slučaju naredba za pokretanje procesa učenja je:

```
[net,tr]=train(net,p,t);
```

Za pokretanje simulacije rada umjetne neuronske mreže koristimo funkciju „sim“ koja ima sljedeću sintaksu:

- *[Y,Pf,Af,E,perf] = sim(net,P,Pi,Ai,T)* – funkcija za pokretanje procesa simulacije rada umjetne neuronske mreže je „sim“. Vraća varijablu, „Y“ – izlaznih vrijednosti umjetne neuronske mreže, „Pf“ – konačna ulazna kašnjenja, „Af“ – konačno kašnjenje po slojevima, „E“ – greške učenja, „perf“ – perfomanse rada mreže. A uzima varijablu „net“ - ime objekta u kojem je spremljena nenaučena umjetna neuronska mreža, „P“ ulaze u mrežu, „Pi“ – početna kašnjenja po slojevima, „Ai“ – početna kašnjenja sloja, „T“ – ciljani vektor.

Za pokretanje simulacije primjera koristimo iduću naredbu i potvrdimo tipkom Enter:

```
a = sim(net,p)
```

Na kraju trajanja učenja umjetne neuronske mreže u komandnom prostoru MATLAB-a pojavit će se idući parametri:

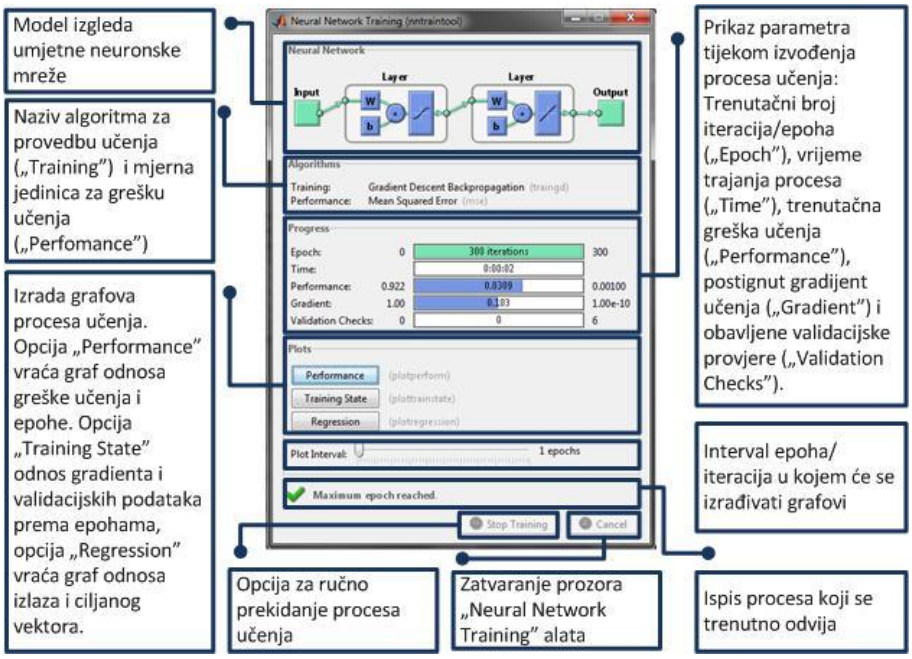
```
Training with TRAINGD.
Epoch 0/300, Time 1.888, Performance 1.0804/0.001, Gradient 3.0884/1e-010,
Validation Checks 0/6
```

```
Epoch 50/300, Time 2.699, Performance 0.12586/0.001, Gradient 0.38095/1e-
010, Validation Checks 0/6
Epoch 100/300, Time 3.167, Performance 0.089406/0.001, Gradient 0.21084/1e-
010, Validation Checks 0/6
Epoch 150/300, Time 3.588, Performance 0.071433/0.001, Gradient 0.17099/1e-
010, Validation Checks 0/6
Epoch 200/300, Time 4.025, Performance 0.059071/0.001, Gradient 0.14414/1e-
010, Validation Checks 0/6
Epoch 250/300, Time 4.446, Performance 0.050138/0.001, Gradient 0.12357/1e-
010, Validation Checks 0/6
Epoch 300/300, Time 4.867, Performance 0.043485/0.001, Gradient 0.10742/1e-
010, Validation Checks 0/6
Training with TRAINGD completed: Maximum epoch reached.
```

Uviđamo kako je za svakih „50“ epoha/iteracija ispisano vrijeme trajanja promjena težina i proračun greške („*Time*“), performanse greške učenja u odnosu na stupanj dobrote učenja („*Performance*“), trenutačnu u odnosu na ciljanu vrijednost gradijenta („*Gradient*“), te validacijsku provjeru koju trenutno nemamo, jer nemamo validacijski skup podataka. Zadnja linija nam govori zašto je učenje prekinuto u ovom slučaju zbog dostignuća maksimalnog broja epoha/iteracija. Kao rezultat rada umjetne neuronske mreže u varijabli „a“ dobit će se sljedeće vrijednosti:

a = -0.3367 0.1490 0.2934 0.8262 0.9427 1.0757

Uviđamo kako se s obzirom na ciljani vektor pohranjen u varijabli „t“ i podatcima za učenje u varijabli „p“ (brojeva od „0“ do „5“), mijenja izlaz umjetne neuronske mreže. Brojevi veći od dva daju vrijednosti bliže broju jedan, a vrijednosti manje od dva daju vrijednosti bliske vrijednosti „0“. Ranije je napomenuto u tablici 5.1. kako je pokretanje „*Neural Network Training*“ alata podešeno u početnim postavkama funkcije „*showWindow*“ naredbe “net.trainParam.showWindow“ na vrijednost „1“, odnosno obavezno pokretanje. Na slici 1. možemo vidjeti prikaz „*Neural Network Training*“ alata sa objašnjenje nekih osnovnih funkcija.



Slika 1. Osnovne funkcije alata „Neural Network Training“

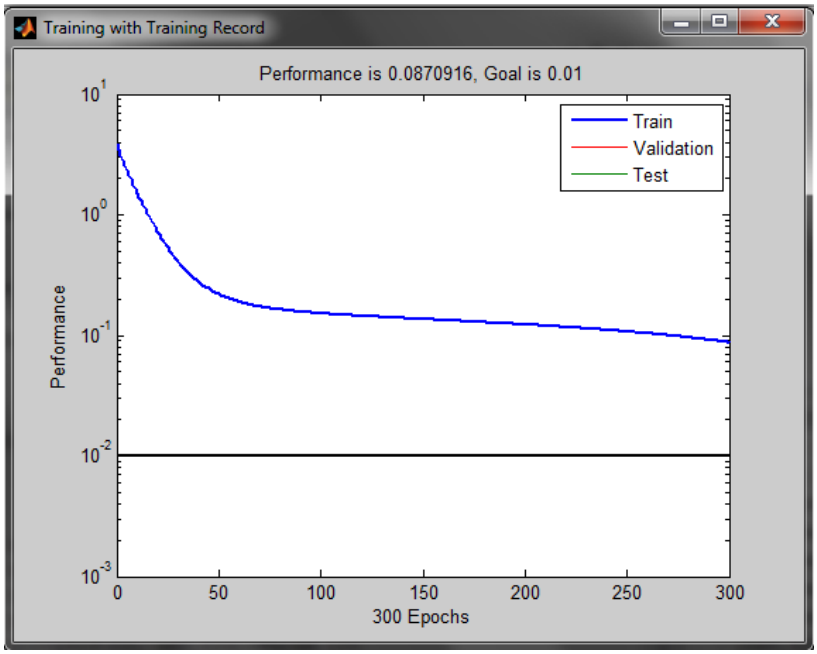
Ukoliko želimo izraditi graf performansi učenja umjetne neuronske mreže, bez korištenja alata „*Neural Network Training*“ koristimo funkciju „plotperf“ sa sljedećom sintaksom:

- *plotperf(tr, a)* – ulazni argument „tr“ predstavlja skup podataka o performansama, odnosno greškama učenja umjetne neuronske mreže dobivenih nakon završetka procesa učenja. Argument „a“ označava bilo koji ciljni parametar (eng. „Goal“) vezan za učenje s kojim želimo usporedit performanse, odnosne greške učenja umjetne neuronske mreže.

Ukoliko želimo u primjeru izraditi graf odnosa greška učenja i faktora dobrote učenja koristimo iduću liniju koda:

```
plotperf(tr, net.trainParam.lr)
```

Nakon potvrde unosa koda u komandni prostor MATLAB-a tipkom Enter dobivamo idući prozor sa slikom grafa kako je prikazano na slici 2.



Slika 2. Prikaz odnosa performansi umjetne neuronske mreže i stupnja dobrote učenja (engl. „*Goal*“)

Brze metode učenja umjetne neuronske mreže

Funkcija učenja umjetnih neuronskih mreža koja bitno ubrzava proces učenja je metoda povratnog rasprostiranja silaznog gradijenta sa adaptivnim stupnjem dobrote učenja (eng. *“Gradient descent with adaptive learning rate backpropagation“*) funkcija ima naredbu „traingda“ . Razlikuje se od prethodne metode učenja po tome, ukoliko se pri pojedinoj iteraciji/epohi greška učenja kreće prema ciljanoj konačnoj greški, koeficijent stupnja dobrote učenja će se povećat za faktor „lr_inc“. Ako se greška učenja poveća više od faktora „max_perf_inc“. Stupanj dobrote učenja se prilagodi faktoru smanjenja

stupnja dobrote učenja „lr_dec“, te se promjene koje utječu na povećanje greške učenja ne dogode. Za funkciju „traingda“ vrijede sve naredbe iz tablice 1. uz naredbe iz tablice 2.

Tab 2. Tablica dodatnih parametra funkcije „trainga“ sa objašnjenjem i početnim vrijednostima

MATALB naredba	Početne vrijednosti	Obajšnjenje
net.trainParam.lr_inc	1.05	Fakor povećanja stupanja dobrote učenja
net.trainParam.lr_dec	0.7	Fakor smanjenja stupanja dobrote učenja
net.trainParam.max_perf_inc	1.04	Najveće povećanje greške učenja nakon kojeg se primjenjuje „lr_dec“

Funkcija otpornog povratnog rasprostiranja (eng.“Resilient backpropagation“) sa naredbom „trainrp“ istu način rada ima kao prethodne dvije metode učenja osim što je brža, te se varijable težina, biasa i prijenosnih funkcija prilagođavaju po sljedećoj jednadžbi:

$$dX = deltaX.*sign(gX)$$

Gdje su elementi od „deltaX“ inicijalizirani na početku vrijednošću „delta0“, te je „gX“ gradijent. Kod svake iteracije elementi „deltaX“ se modificiraju. Funkcija „sign“ vraća vektor istih dimenzija kao i gradijent „gX“. Odgovarajuću poziciju puni vrijednošću „1“ ako je na toj poziciji gradijenta „gX“ pozitivni broj, „-1“ ako je negativni, i „0“ ako je vrijednost „0“. Ako element od gradijenta „gX“ promjeni predznak iz jedne iteracije u drugu, onda će se odgovarajući element vektora „deltaX“ smanjiti za vrijednost „delta_dec“. Ako je element gradijenta „gX“ zadrži isti predznak iz jedne u drugu iteraciju, onda se odgovarajući element vektora „deltaX“ uveća za vrijednost „delta_inc“. Na tablici 3. su prikazane parametri funkcije „trainrp“ sa objašnjenjem i početnim vrijednostima.

Tab 3. Tablica dodatnih parametra funkcije „trainrp“ sa objašnjenjem i početnim vrijednostima

MATALB naredba	Početne vrijednosti	Obajšnjenje
net.trainParam.delt_inc	1.2	Uvećanje za promjenu težina
net.trainParam.delt_dec	0.5	Smanjenje za promjenu težina
net.trainParam.delta0	0.07	Inicijalna vrijednost za promjenu težina
net.trainParam.deltamax	50.0	Maksimalna promjena težina

Priprema za vježbu

- Proučiti predavanja vezana za umjetne neuronske mreže.
- Proučiti vježbe iz osnova MATLAB-a i Simulink-a, te vježbe iz uvoda u umjetne neuronske mreže.

Rad na vježbi

Rad na vježbi sastoji se od stvaranja MATLAB skripte za pripremu podataka za učenje i kreiranja umjetne ciklične neuronsku mreže pomoću „newff“ funkcije sa argumentima. Riješit će se problem predviđanja dolaska vozila na temelju generiranih podataka i podataka iz datoteke „vjezbe.mat“ provesti će se učenje, evaluacija i simulacija rada umjetne neuronske mreže, te komentirati izlazne rezultate.

Priprema podataka iz „vjezbe.mat“ za prezentaciju cikličnoj umjetnoj neuronskoj mreži

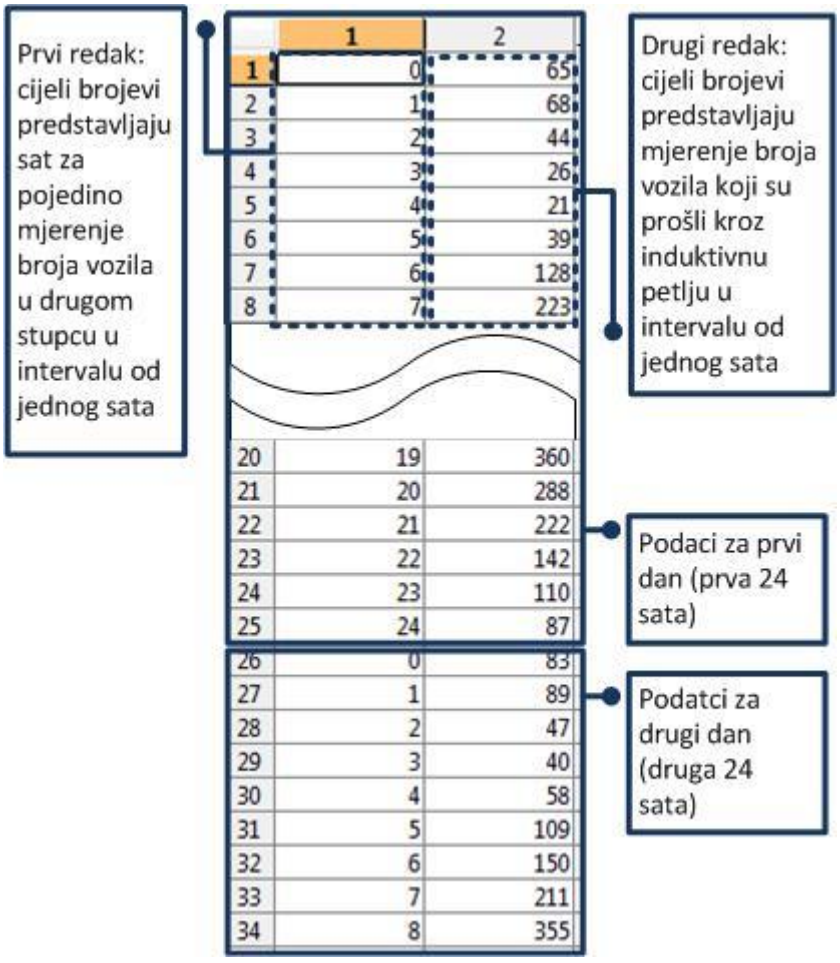
Datoteka „vjezbe.mat“ sadrži podatke sa brojila prometa na lokaciji grada Vira, prikupljenih induktivnom petljom 2009. godine (dane na raspolaganje od strane mr. sc. Marka Ševrovića, dipl.ing.). Kako je napomenuto u prethodnoj vježbi. Vrijednosti stupca koji označava vremenski trenutak kada je mjerenje prolaska vozila zabilježeno kao broj sati unutar dana su prebrojane u intervalima u rasponu „0.5“ između vrijednosti „0“ i „24“, tj. jednog dana. Interval u rasponu „0.5“ predstavlja broj vozila koji su prošli preko induktivne petlje u 30 minuta. Mjerenje broja prolaska vozila je provedeno tokom sedam dana, odnosno cijelog tjedna.

Obrada skupa podataka dobivenog brojanjem prolaska vozila u jednom satu

Kako bi se provelo potrebno učenje umjetne neuronske mreže sa skupom podataka dobivenog brojanjem prolaska vozila u jednom satu potrebno je stvoriti MATLAB skriptu pod nazivom „ucenje.m“, koja će generirati varijablu – „rez“.

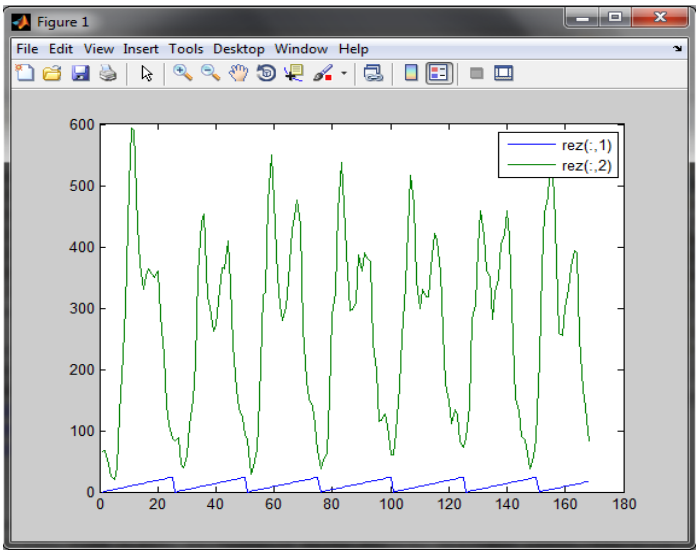
Prvi stupac varijable „rez“ će sadržavati cijele brojeve u rasponu od „0“ do „24“ sa intervalom „1“ za svaki od sedam dana, on će predstavljati broj sati u jednom danu. Drugi stupac sadržavati će broj vozila u jednom satu, tj. zbroj vrijednosti vozila u dva susjedna intervala od pola sata. Broj vozila po satu u drugom stupcu neka odgovara odgovarajućem satu kad je izmjeren u prvom stupcu. Varijabla „rez“ će imati izgled koji je prikazan na tablici 4.

Tab 4. Izgled i objašnjenje elemenata varijable „rez“



	1	2
1	0	65
2	1	68
3	2	44
4	3	26
5	4	21
6	5	39
7	6	128
8	7	223
20	19	360
21	20	288
22	21	222
23	22	142
24	23	110
25	24	87
26	0	83
27	1	89
28	2	47
29	3	40
30	4	58
31	5	109
32	6	150
33	7	211
34	8	355

Pripazite pri generiranju varijable „rez“, da podatci o broju prolaska vozila ne postoje od 17:00h sedmog dana (imaju vrijednost „0“). Kako bi rezultati učenje bili što kvalitetniji izbrišite retke koji u drugom stupcu (broj prolaska vozila u satu) imaju vrijednost „nula“. Izradite graf varijable „rez“ kao što je to prikazano na slici 4. Analizirajte odnos prvog retka „rez(:,1)“ i drugog retka „rez(:,1)“ uočavanjem uzorka ponašanja broja vozila u odnosu na određeno vrijeme dana kroz tjedan.



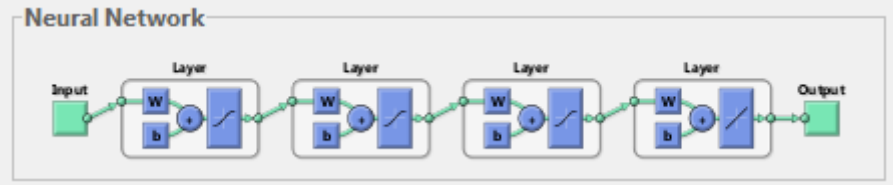
Slika 4. Grafički prikaz varijable „rez“

Varijablu „rez“ podijelite u dvije varijable „ucenje“ i „validacija“. U varijablu „validacija“ spremite vrijednosti varijable „rez“ za mjerenja u prva 24 sata (broj vozila u satu sa pripadajućim vrijednostima sata mjerenja). Varijabla „ucenje“ neka sadrži mjerenja preostalih šest dana, zajedno sa vrijednostima pripadajućeg sata mjerenja.

Izrada i priprema učenja ciklične umjetne neuronske mreže

Stvorite cikličnu neuronsku mrežu (*engl. „feed-forward neural network“*) pomoću „newff“ funkcije bez argumenta pod imenom „mreza“. Strukturu ciklične neuronske mreže kreirajte na način da ima jedan ulaz, te četiri sloja .

Prvi sloj postavite da ima „223“ neurona. Drugi sloj postavite na „184“ neurona, treći na „83“ neurona, dok četvrti postavite na jedan neuron, pošto će umjetnoj neuronskoj mreži biti predstavljen samo jedan ciljani (engl. „*Targets*“) vektor. Pri definiranju strukture umjetne neuronske mreže prijenosnu funkciju (engl. *„Transfer Function“*) u prva tri sloja postavite na Tan-sigmoidnu funkciju, a za treći sloj linearnu funkciju. Metodu učenja definirajte u samoj konstrukciji umjetne neuronske mreže i postavite ju na vrijednost funkcije otpornog povratnog rasprostiranja. Grafički model umjetne neuronske mreže je prikazan na slici 5.



Slika 5. Grafički model umjetne neuronske mreže

Ulazni skup za učenje neka sadržava vrijednost sati unutar dana od „0“ do „24“ (prvi stupac u varijabli „ucenje“), a ciljani vektor neka sadrži odgovarajući broj vozila po satu iz skupa podataka za učenje (drugi stupac u varijabli „ucenje“). Obratite pažnju kako stvorena ciklična umjetna neuronska mreža ima samo jedan ulaz. Jedan ulaz u umjetnu neuronsku mrežu u Neural Network okruženju predstavlja vektor u obliku matrice 1 x n.

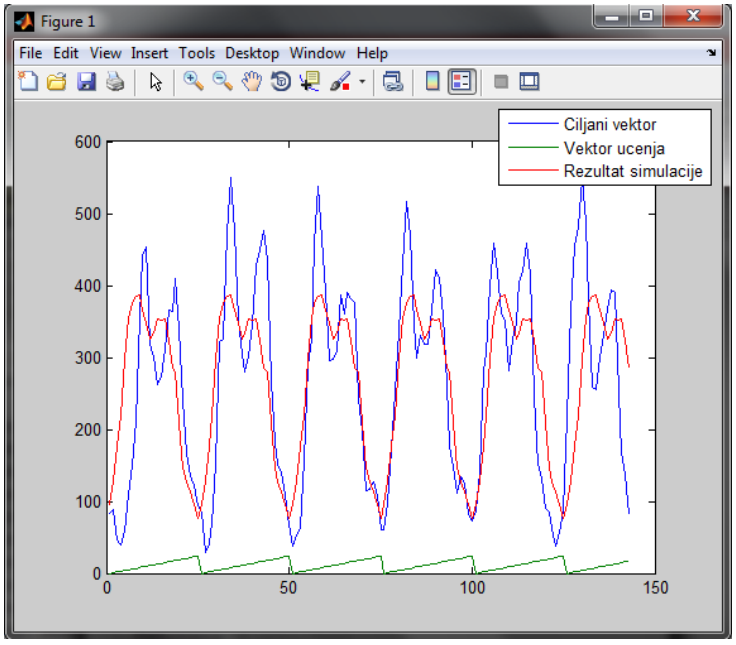
Maksimalni broj iteracija/epoha učenja postavite na vrijednost „300“, a vrijednost razlike između prikaza vrijednosti greške učenja iteracije/epohe postavite na „50“. Omogućite generiranje izlaza u obliku ispisa u komandnom prozoru vrijednosti izlaznih grešaka, te prikaz „Neural Network Training“ alata.

Nakon umjeravanja učenja umjetne neuronske mreže, pokrenite proces učenja uključujući podatke iz skupa za učenje i ciljani vektor.

Po završetku učenja, simulirajte rad umjetne neuronske mreže pomoću funkcije „sim“ sa dva tipa ulaznih podataka za simulaciju:

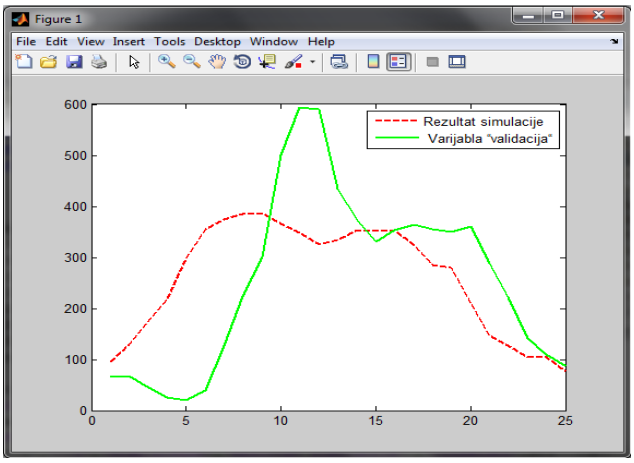
1. Najprije kao skup podatka za simulaciju rada umjetne neuronske mreže postavite već korišteni skup podataka za učenje, tj. vektor učenja;
2. Zatim iskoristite za simulacijski ulaz prvi stupac varijable „validacija“ (sadržava vrijednost sati unutar prvog dana od „0“ do „24“).

Izradite graf rezultata učenja, za prvi tip podatka u odnosu na skup podataka za učenje i ciljani vektor, kao što je prikazano na slici 6.



Slika 6. Grafički prikaz odnosa ciljanog vektora, vektora učenja i rezultata simulacije rada umjetne neuronske mreže.

Izradite graf prikaza rezultata simulacije rada umjetne neuronske mreže sa drugim tipom ulaznih simulacijskih podataka (varijabla „validacija“) u odnosu na početne vrijednosti prolaska broja vozila u varijabli „validacija“, kao što je to prikazano na slici 7.

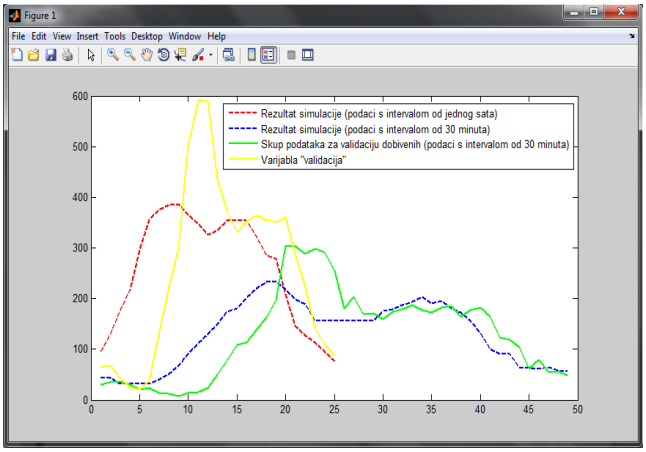


Slika 7. Prikaz rezultata simulacije rada umjetne neuronske mreže sa drugim tipom ulaznih podataka (varijabla „validacija“) u odnosu varijablu „validacija“

Obrada skupa podataka dobivenog brojanjem prolaska vozila u 30 minuta

Ponovite isti postupak sa istim parametrima ciklične umjetne neuronske mreže, ali koristite čitav skup podataka iz datoteke „vjezba.mat“, bez postupka raučunanja prolaska broja vozila u jednom satu. U novom slučaju imamo intervale od 30 minuta. Prema tome nova varijabla, ekvivalentna varijabli „rez“ će sadržavat brojeve u rasponu „0“ do „24“, ali s intervalom od „0.5“ (30 minuta). Ostatak naputka za pripremu podataka za prezentaciju umjetnoj neuronskoj mreži vrijede od prijašnjeg zadatka.

Uradite sve korake obrade rezultata učenja iz prijašnjeg primjera po istim naputcima. Za kraj dodajte grafičku usporedbu rezultat simulacije umjetne neuronske mreže sa brojanjem prolaska vozila u jednom satu (prvi zadatak) i u 30 minuta (drugi zadatak) izradom grafa prema podatcima za validaciju dobivenih obradom skupa podataka dobivenog brojanjem prolaska vozila u 30 minuta i varijablom „validacija“. Izgled grafa je prikazan na slici 8.



Slika 8. Graf prikaza analiza stupnja dobrote rezultata simulacije s podatcima s intervalom od jednog sata i 30 minuta

Zadatak

Usporedite rezultate simulacije s podatcima s intervalom od jednog sata i 30 minuta. Odredite u kojem slučaju umjetna neuronska mreža daje bolje rezultate nakon simulacije rada mreže. Odredite uzrok boljim rezultatima. Izraditi graf performansi učenja umjetne neuronske mreže u oba načina obrade podataka korištenjem alata „Neural Network Training“ ili funkciju „plotperf“, te odredite kod koje epohe/iteracija su dostignute najbolje performanse. Promatrajte kvalitetu rezultata simulacije povećanjem ili smanjenjem broja epoha/iteracija u oba načina prikupljanja podataka.