VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Baigiamasis magistro darbas

Varijuojančių parametrų įtaka neuronų tinklų apmokymo procese

(Variating parameter effect on Neural Networks learning process)

Atliko: 2 magistro kurso studentas
Karolis Pečiulis
(parašas)

Darbo vadovas:
lekt. Aistis Raudys
(parašas)

Recenzentas:
Arūnas Janeliūnas
(parašas)

Vilnius
2016

Santrauka

Neuroniniai tinklai dažniausiai yra naudojami kaip klasifikatoriai tokiuose uždaviniuose kaip struktūrų, garso, vaizdo atpažinimas. Bėgant laikui, juos pradėjo naudoti ir medicinoje, finansuose ir inžinerijoje.

Šiame darbe koncentruojamės ties neuroninio tinklo optimizavimą atliekant pasirinktų faktorių modifikacijas mokymo procese. Dabar galima rasti daugybę įvairių hibridinių neuroninių tinklų modelių, kurių tikslas optimizuoti rezultatą įvairiose srityse. Mus domina, kokie optimizavimo metodai buvo naudojami siekiant gauti geresnius tinklų rezultatus, taip pat žiūrėsime.

Darbo tikslai yra ištirti ir įvertinti, kaip keičiasi neuroninių tinklų generuojami rezultatai į mokymo procesą integruojant parametrų variacijas. Tam naudosime savo pasirašytą programą, kuri apmoko aibę tinklų ir kiekvienam jų sugeneruoja grafikus atspindinčius mokymo ir testavimo duomenų aprėpimą.

Iš rezultatų galime pastebėti, kad kai kurios variacijos iš tiesų pageriną tinklo galutinį rezultatą ir kartu sutrumpina patį mokymo procesą.

Summary

Neural networks are commonly used in classification problems such as structural, sound and video recognition. Afterwards, they were integrated in medicine, finance and engineering.

In this thesis we concentrate on the optimization of the neural network by performing selected factor modifications within the learning process. Currently, there is a variety of hybrid artificial neural network models whose task is to optimize the end result. We are focusing on methods of optimization that were already applied in practice.

The goal of this thesis is to research and evaluate how the output of neural networks change when applying parameter variations within the learning process. For this we will be using our own writing software design to train multiple networks and produce results, including graphs, for each of them.

Based on our findings, we can state that some variations in fact improve the end result of the network while additionally lowering the time consumption of the training process.

Turinys

Įvadas

Neuroniniai tinklai dažniausiai yra naudojami kaip klasifikatoriai tokiuose uždaviniuose kaip struktūrų, garso, vaizdo atpažinimas. Bėgant laikui, juos pradėjo naudoti ir medicinoje. A. Fallahi ir S. Jafari aprašė sistemą pagrįstą neuroniniais tinklais skirtą atpažinti krūties vėžį [FJ11]. Naudodami Bajeso tinklą jie pasiekė 98% sėkmę atpažįstant ligą.

Nors ir yra nemažai publikacijų apie kainų prognozes neuroniniais tinklais, pačių tinklų optimizavimus, tačiau dar nėra atliktas tyrimas naudojant varijuojančius parametrus. Juos apibrėžiame kaip nepastovius neuroninių tinklų parametrus, kurių reikšmės gali priklausyti nuo kitų parametrų, mokymo iteracijos ar klaidos. Savo modelyje koncentruosimės į mokymo imties dydį, mokymosi žingsnį ir neuronų skaičių. Kitaip tariant, žiūrėsime, kaip šių parametrų kaita paveikia mūsų neuroninio tinklo mokymosi procesą ir galutinį rezultatą. Visi šie parametrai yra randami naudojant NIAB taisyklę, tai yra

$$\varphi_{i+1}(c, \alpha, \beta, \delta) = c\varphi_i + \delta(i+1)$$

, kur $\delta_1 = \alpha, \delta_N = \beta, c \geq 0, i = \overline{1..N-1}$ ir δ teigiama funkcija.

Šiame darbe koncentruojamės ties neuroninio tinklo optimizavimą atliekant pasirinktų faktorių modifikacijas mokymo procese. Dabar galima rasti daugybę įvairių hibridinių neuroninių tinklų modelių, kurių tikslas optimizuoti rezultatą įvairiose srityse. Mus domina, kokie metodai optimizavimo buvo naudojami siekiant gauti geresnius tinklų rezultatus, taip pat žiūrėsime į DNT naudotas struktūras.

L. Anastasakis ir N.Mort atliktas darbas apie JAV dolerio ir Didžiosios Britanijos svaro keitimo kurso prognozes neuroniniais tinklais [AM00] rodo tai, kad norint gauti geresnius rezultatus, neužtenka naudoti SISO (Single Input Single Output) struktūros neuroninius tinklus.

Atsižvelgdami į tai, savo modelį kursime naudodami MISO (Multi-Input Single Output) struktūrą. Tai reiškia, kad pasirinkti duomenų rinkiniai turės daugiau negu vieną įvesties sluoksnio parametrą ir jie visi rodys į vieną išvesties rezultatą. Tinklo apmokymui bus naudojamas atgalinio sklidimo algoritmas (Backpropagation), kuris yra aprašytas 3 skyriuje.

1 Tikslai ir uždaviniai

Darbo tikslas yra ištirti, kokią įtaką galutiniam rezultatui daro įvairių neuronų tinklų parametrų modifikavimas apmokymo metu. Tam reikės atlikti keletą užduočių:

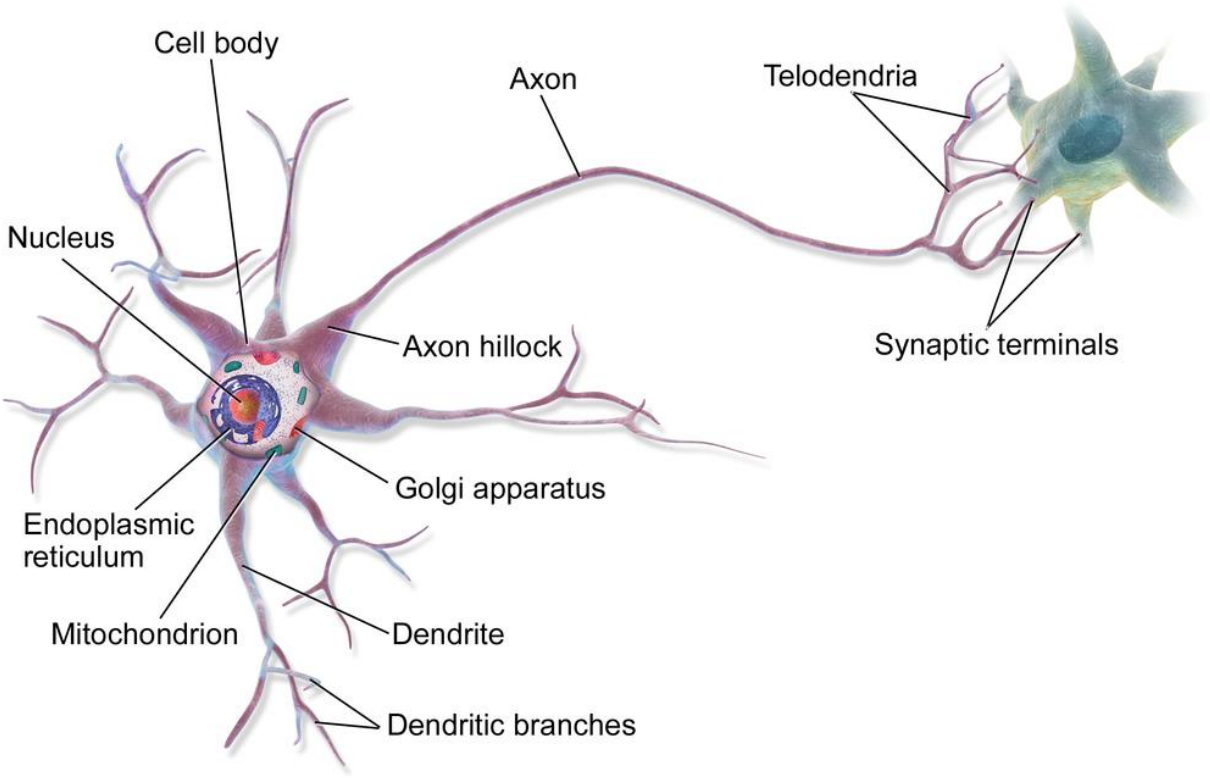
- Parinkti parametrus, kurie bus modifikuojami.
- Pasirinkti pradines ir galines tinklo struktūras turimiems atvejams.
- Parašyti konfigūruojamą programą, kuria visus bandymus galima atlikti vienu vykdymu.
- Optimizuoti kodą naudojant paralelizmą.
- Optimizuoti atminties sunaudojimą.
- Išvengti tinklų persimokymo.
- Prasmingai atlikti kiekvieno parametro modifikacijas neprarandant turimos informacijos.
- Apskaičiuoti metrikas kiekvienam tinklui.
- Nubraižyti grafikus mokymo ir testavimo duomenims ir rezultatams.

Atlikus visas užduotis tikimasi turėti:

- Daugkartinio naudojimo programą.
- Pasirinktų parametrų variacijų rezultatus ir grafikus.
- Verdiktą tokiam DNT mokymo proceso modifikavimui.

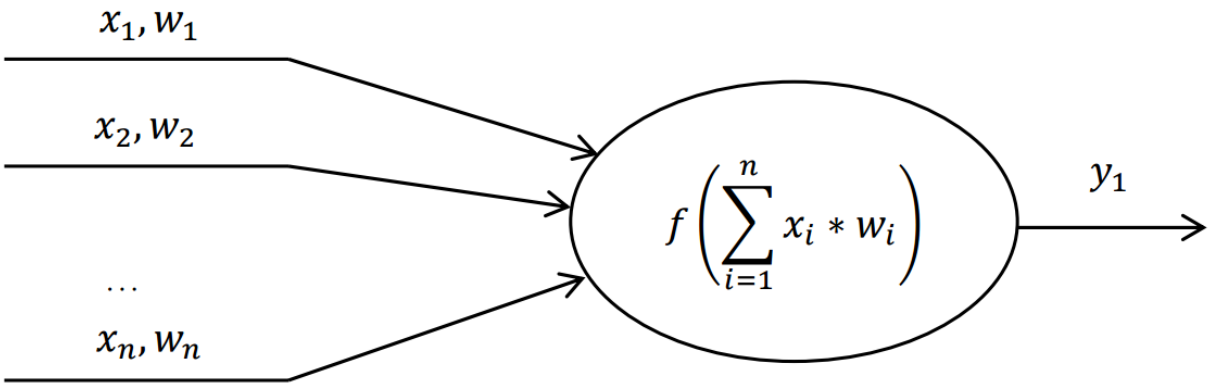
2 Dirbtinis neuronų tinklas

Neuronas – elektrai jautri ląstelė, kuri nešioja informaciją per elektrinius ir cheminius signalus (Pav. 1). Jis yra sudarytas iš ląstelės branduolio, aksono ir daugybės dendritų. Dendritai atlieka įvesties modulio funkciją, kuris surenką informaciją ir ją perduoda maršrutizavimo moduliui – aksonui. Aksonas turi daugybę išvesties sinapsių, kurių pagrindinė funkcija – apdoroti gautą informaciją ir, priklausomai nuo įvesties, siųsti arba ne į kitų prisijungusių neuronų dendritus [Mak06].



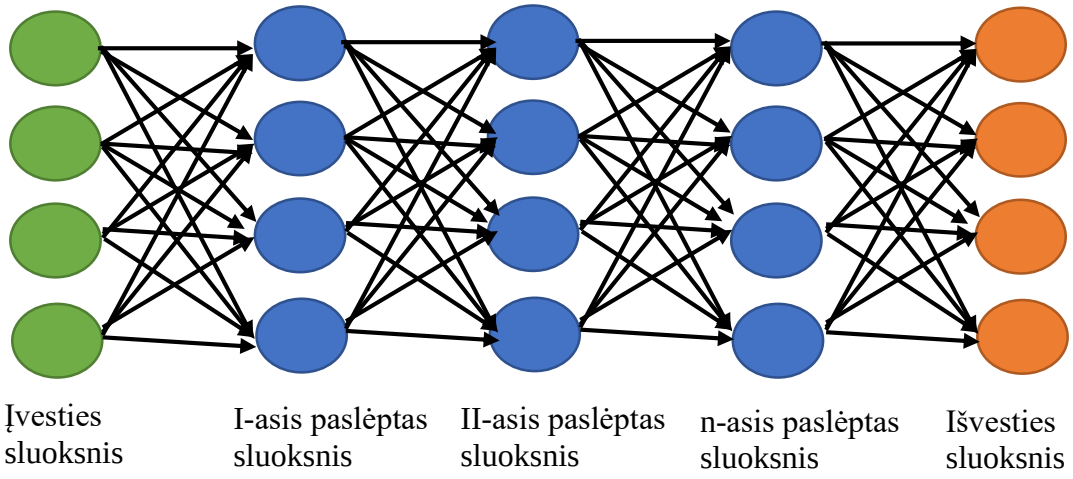
Pav. 1. Neurono biologinis modelis (Wikipedia.org)

Tokiu pat principu buvo sukurtas ir dirbtinis neuronas (Pav 2) – jis turi įvesties modulį, kuris priima informaciją ir perduoda ją aktyvavimo funkcijai – sinapsei – kuri atlikusi veiksmus perduoda signalą toliau. Su neuronu yra susieta perdavimo funkcija, kuri nurodo, kaip to neurono reikšmė yra perduodama kitam neuronui. Įprastai perdavimo funkcija [Sta13] padaugina kiekvieną įvesties reikšmę iš atitinkamų įėjimo verčių ir šias sandaugas susumavus su bazinę įėjimo verte gaunama suminė įėjimo vertė. Tada gautai suminei įėjimo vertei yra pritaikoma aktyvavimo funkcija.



Pav. 2. Dirbtinio neurono modelis

Neuronų tinklas – tai struktūra, sudaryta iš neuronų. Pagrindinė tinklo funkcija – pagal duotą įvestį pateikti atitinkamą rezultatą. Tad jie galėtų atlikti skaičiavimus yra reikalingas apmokymas, tokiu būdu yra nustatomi neuronų įvesties svoriai. [PS96] DNT gali būti vienasluoksniai ir daugiasluoksniai – šiame darbe naudosime tik daugiasluoksnius tinklus.



Pav. 3. Dirbtinių neuronų tinklo modelis

Tipinį daugiasluoksnį tinklą sudaro 3 pagrindiniai sluoksniai:

- Įvesties sluoksnis
- Paslėptasis sluoksnis
- Įvesties sluoksnis

Įvesties sluoksnio neuronai paprastai atlieka duomenų perdavimo funkciją ir patys gaunamos įvesties nekeičia, nebent reikia atlikti formatavimo procedūras duomenų srautui suvienodinti.

Paslėptojo sluoksnio neuronai atlieka visus skaičiavimus ir siunčia į tolimesnį sluoksnį. Šių sluoksnių neveikia jokie išoriniai faktoriai – t.y. jie jungiasi tiktai su kitais tinklo sluoksniais.

Išvesties sluoksnis iš vidinio tinklo gauna jau apdorotą informaciją ir siunčia ją iš tinklo. Kai kuriais atvejais šis sluoksnis dar pats atlieka paskutinius skaičiavimus prieš perduodamas rezultatą.

3 Mokymas

Yra išskirti 3 pagrindinės dirbtinių neuroninių tinklų mokymo metodų rūšys:

- Mokymas su mokytoju

  Šie metodai naudojami, kai yra žinoma, kokių rezultatų tikimasi. Jais bandoma susieti turimus duomenis su trokštamoms reikšmėms, kur vykdymo metu yra ieškomi svoriai, su kuriais skirtumas tarp norimo ir gauto rezultatų būtų mažiausias.

- Mokymas be mokytojo

  Šie metodai naudojami, kai nėra žinoma, kokių rezultatų reikia. Jų pagrindinis tikslas yra kategorizuoti turimus mokymo duomenis arba rasti juose kokius nors reguliarumus ar ypatumus.

- Hibridinis mokymas

  Tai yra mišinys mokymo su mokytoju ir be – dalis tinklo svorių yra nustatomi pagal norimus gauti rezultatas, kita dalis – gaunama naudojant tik duomenis [DKT08].

3.1 Atgalinio sklidimo algoritmas

Atgalinio sklidimo algoritmo esmė yra perduoti klaidą visiems neuronams pradedant nuo išvesties sluoksnio ir baigiant įvesties sluoksniu. Algoritmas yra skirstomas į 2 etapus:

- Tiesioginis sklidimas į priekį (nuo įvesties sluoksnio į išvesties).
- Klaidos sklidimas atgal (nuo išvesties sluoksnio iki įvesties sluoksnio).

Algoritmas patenka į „mokymo su mokytoju“ kategoriją, todėl turime įvesties vektorius ir trokštamas reikšmių vektorius.

Tiesioginio sklidimo metu visi tinklo neuronai skaičiuoja svorines įėjimo elementų sumas pagal formulę:

$$y = \sum_{i=1}^{n} w_i x_i$$

, kur w_i i-tojo neurono įėjimo svoris,

x_i i-toji reikšmė.

Gauta suma yra transformuojama su nustatyta, dažniausiai netiesine, aktyvavimo funkcija f. Vienos plačiausiai naudojamos yra:

Sigmoidas

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

Hiperbolinis tangentas

$$f(x) = \tanh(x) = \frac{\sinh(2x)}{\cosh(2x) + 1}$$

,kur

$$\sinh(y) = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!}$$

$$\cosh(y) = \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!}$$

Slenkstinė

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

Tiesinė

$$f(x) = x$$

Duomenims perėjus per tinklą, yra skaičiuojama suminė kvadratinė klaida:

$$E(W) = \sum_{i=1}^{m} E_i(W)$$

$$E_i(W) = \frac{1}{2} \sum_{j=1}^{d} (y_{ij} - t_{ij})$$

w_0 – slenksčio reikšmė,

d – neuronų skaičius,

$Y_i = \{y_{i1}, y_{i2}, \dots, y_{id}\}$ – tinklo išvesties vektorius,

$T_i = \{t_{i1}, t_{i2}, \dots, t_{id}\}$ – tinklo norimų vektorius.

Pradedant apmokymą, tinklo svoriai būna inicializuoti atsitiktinėmis reikšmėmis. Vykstant atgaliniam sklidimui, svoriai yra koreguojami pagal formulę:

$$w_{jk}\,(t+1) = w_{jk}(t) + \Delta w_{jk}(t)$$

,kur w_{jk} k-tojo vektoriaus elemento svoris į j-tąjį išėjimą. Pokytis atitinka

$$\Delta w_{jk}(t) = -\eta \frac{\partial E(t)}{\partial w_{jk}} = -\eta \sum_{i=1}^{m} \frac{\partial E_i(t)}{\partial w_{jk}} = \sum_{i=1}^{m} \Delta w_{jk}^{i}(t),$$

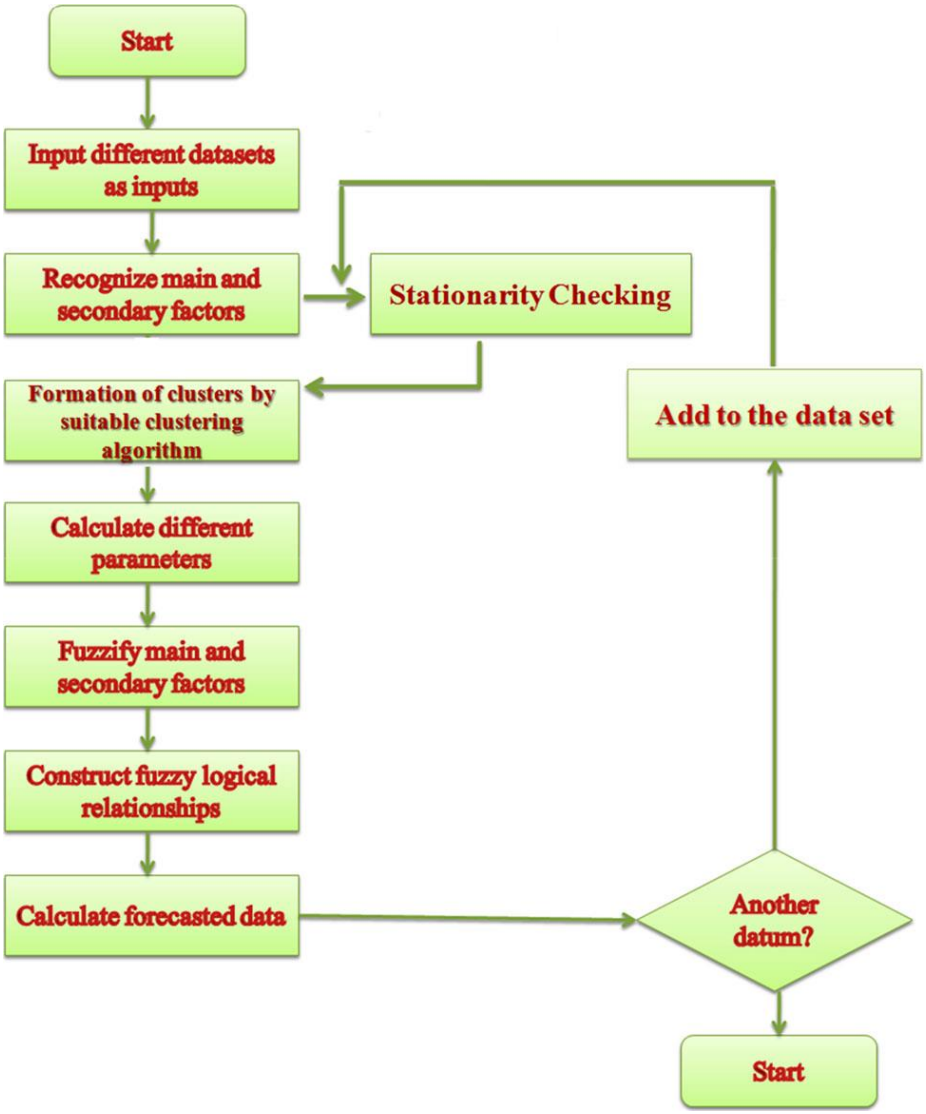
$$\Delta w_{jk}^{i}(t) = -\eta \frac{\partial E_i(t)}{\partial w_{jk}}$$

,kur $-\eta$ žymi mokymosi tempą (dažniausiai skaičius intervale (0;1)), t – iteracijos numeris [DKT08].

4 Neuroninių tinklų optimizavimo rūsys

4.1 Fuzzy

Šiuo metu yra populiaru prognozavimo uždaviniams naudoti FTS (fuzzy time series) paremtus metodus. Dauguma jų turi esminę problemą – jie naudoja savo grupavimo technikas, kurios gali būti naudojamos tik su specifiniais duomenimis, taip pat dauguma naudoja fiksuotus

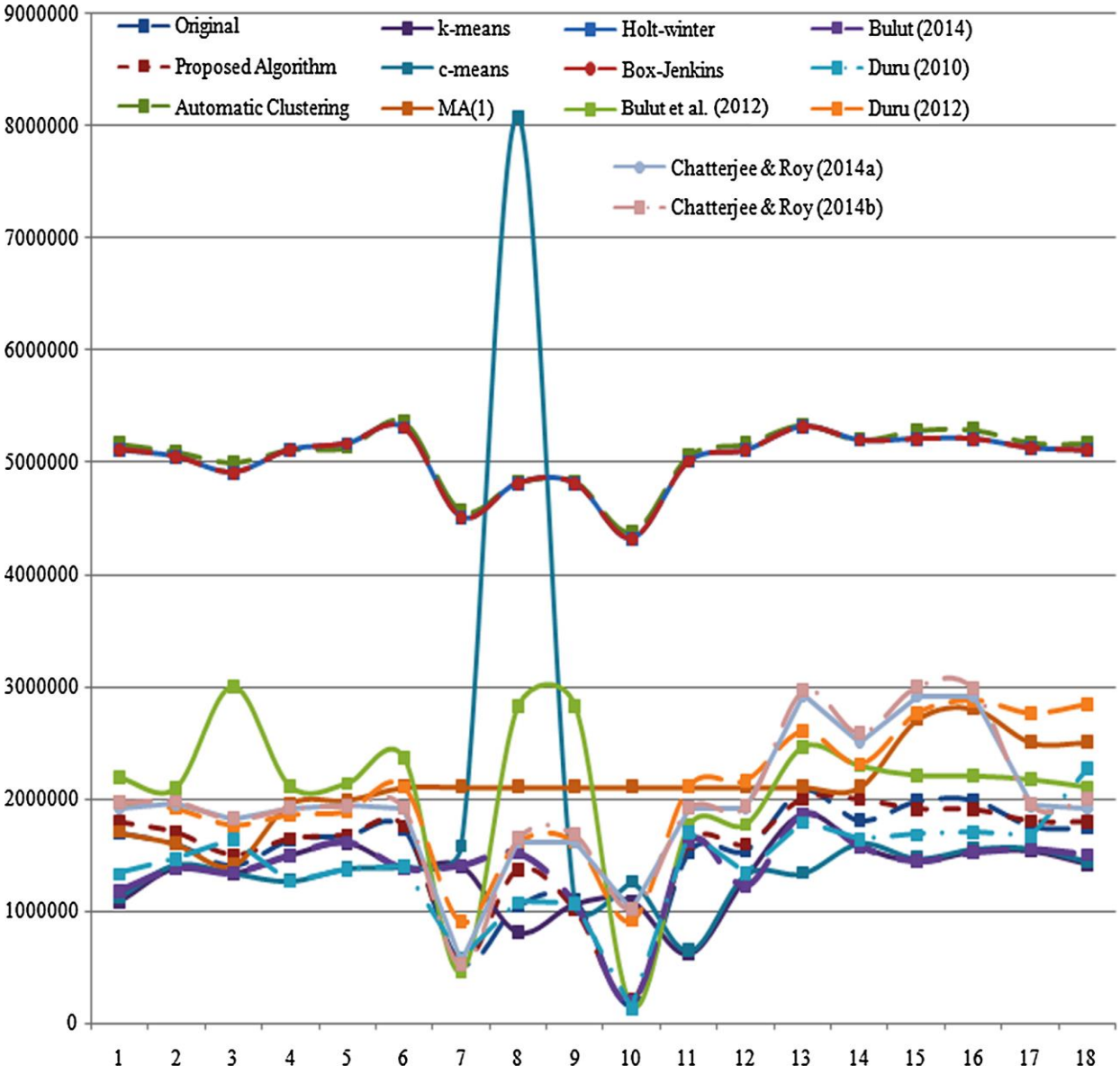


Pav. 4. [Roy15]Siūlomo algoritmo struktūrinė schema

parametrus (pvz. intervalo dydis). Dėl to, A.Roy siūlo naują daugiamatį *fuzzy* prognozavimo algoritmą, kuris gali panaikinti esamus trūkumus ir turėti geresnį rezultatą [Roy15].

Dauguma egzistuojančių *fuzzy* prognozavimo technikų susideda iš 4 žingsnių:

- Kalbos skaidymas į intervalus
- Istorinių duomenų pavertimas į *fuzzy*
- *Fuzzy* loginių ryšių ir ryšių grupių kūrimas
- Išvesties skaičiavimas



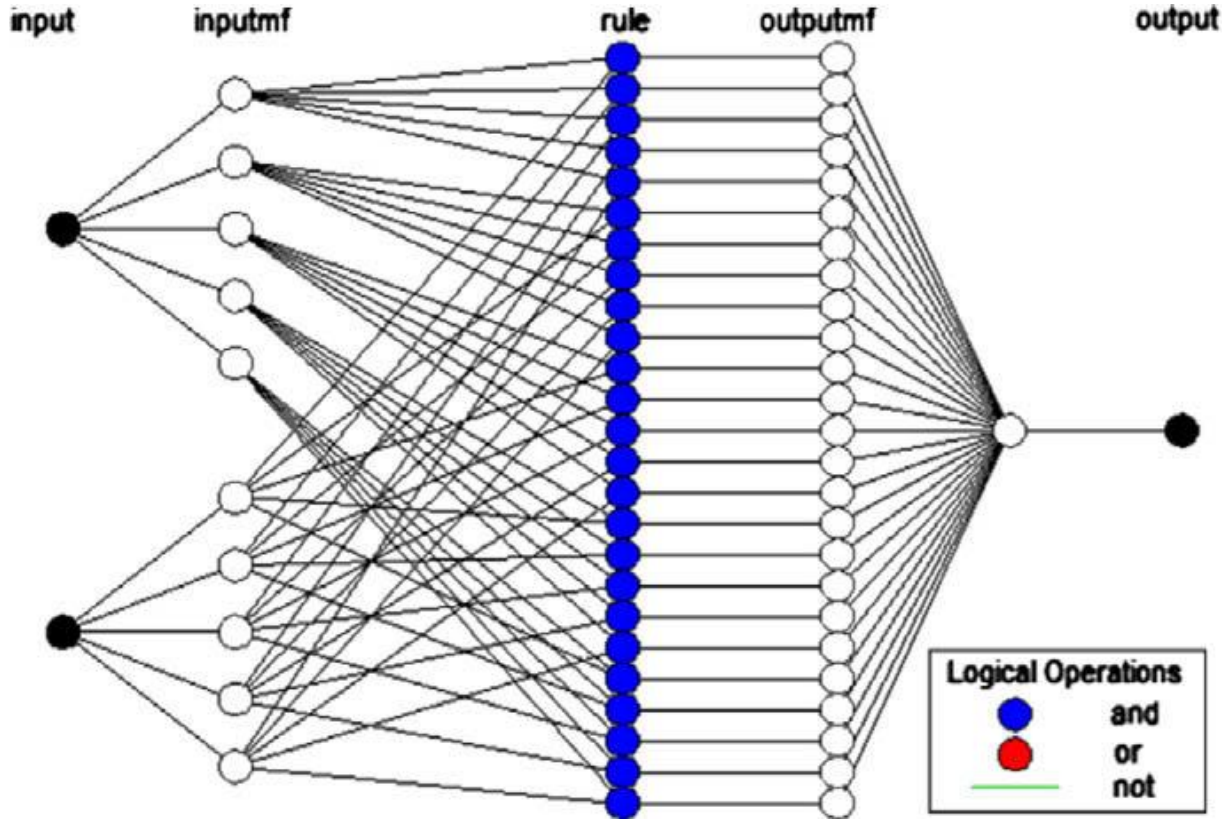
Pav. 5. [Roy15]Fuzzy daugiamačio ir kitų algoritmų rezultatai finansinių duomenų prognozėse

Autorius savo algoritmą išskaido į 5 žingsnius: stacionarumo tikrinimas, grupavimas, parametrų skaičiavimas, naryščių skirstymas, daugiamačio *fuzzy* prognozavimo algoritmo konstravimas.

Tyrimas buvo atliktas su keliais duomenų rinkiniais iš skirtingų sferų, mes žiūrėsime tik į vieną iš jų – finansinius duomenis.

Rezultatuose matoma, kad pasiūlytas algoritmas turi pranašumą prieš kitus FTS tipo metodus – vienais atvejais didesnį, kitais mažesnį. Kol kas sunku pasakyti kaip šis algoritmas veiktų naudojant daug didesnę imtį ir stipriau besikeičiančius duomenis.

Dar vienas trumpalaikės prognozės sprendimas buvo pasiūlytas G. Atsalakis ir K. Valavanis – jie siūlo naudoti *neuro-fuzzy* metodologiją [AV09]. Savo darbe jie kuria savo modelį naudodami ANFIS (Adaptive Neuro Fuzzy Inference Systems) techniką.



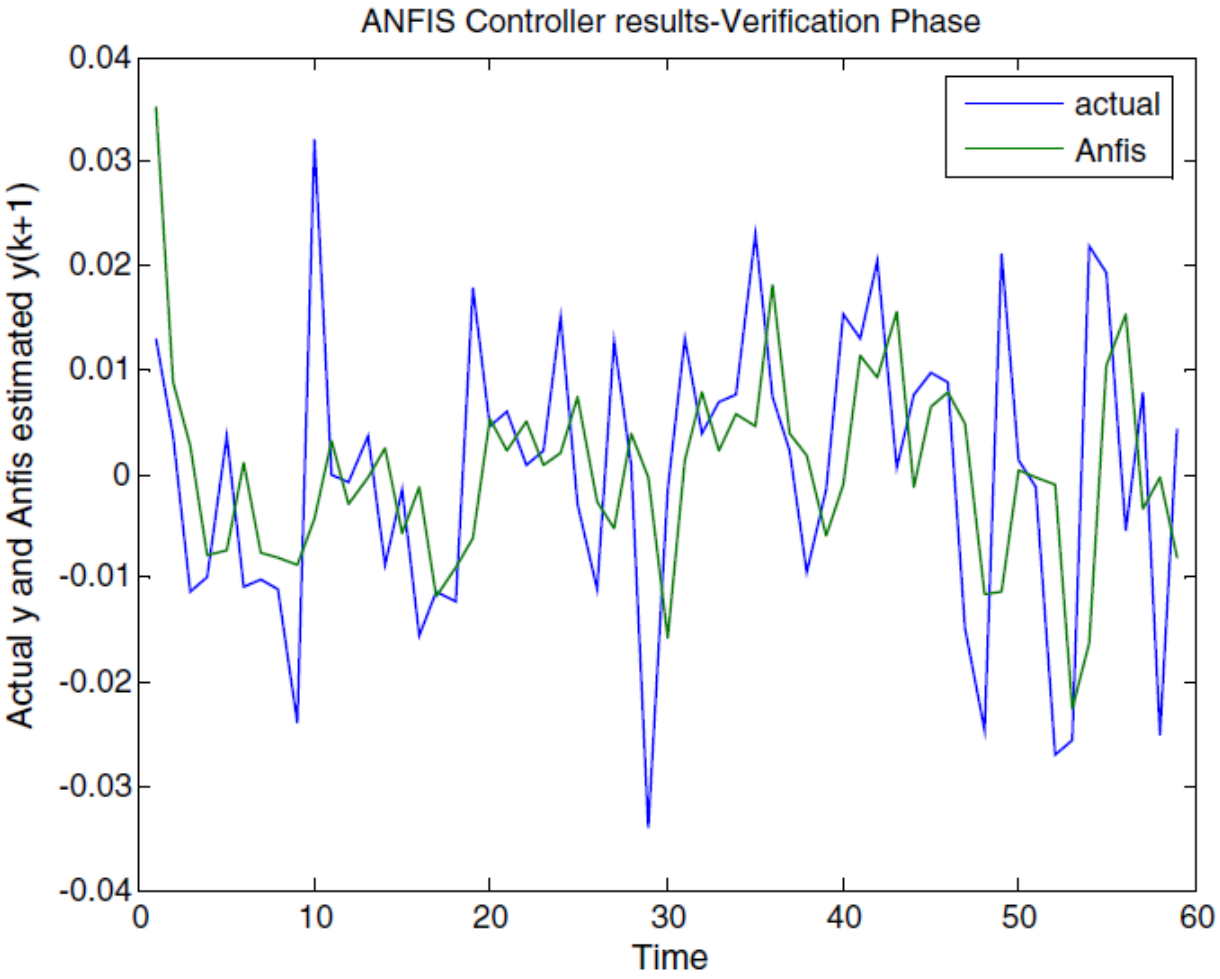
Pav. 6. [AV09]CON-ANFIS struktūra

Šis modelis naudoja esamą reikšmę, norint nustatyti būsimą, kaip ir keli prieš tai nagrinėti modeliai. Savo darbe jie teigia, kad naudojant istorinius duomenis, galima nustatyti akcijų biržos tendencijas. Mūsų kuriamas modelis remiasi šio teiginio teisingumu ir pagrįstumu.

Jeigu pažiūrėtumėme į jų gautus rezultatus, matytumėme, kad pačios tendencijos yra tiksliai randamos, bet reikšmės neatitinka realybės. Šiuo atveju mus toks rezultatas tenkina,

kadangi jis daugmaž pasako, kada ir kiek akcijos vertė pasikeis – kas yra mums naudinga informacija.

Remdamiesi tuo, kad istoriniai duomenys gali gana tiksliai pasakyti tendencijas, mes pagal tai modeliuosime savo įvestį (skirtingai negu kiti metodai, kurie išsisaugo senas reišmes) – naudosime istorinius duomenis laikydami juos dabartiniais.



Pav. 7. [AV09]ANFIS rezultatai lyginant su tikrais duomenimis

Dar vienas hibridinis *fuzzy* modelis buvo pasiūlytas naudojant c-vidurkius (c-means) ir neuroninius tinklus – jo esmė yra c-vidurkių grupavimo metodo ir DNT taikymas kuriant *fuzzy* santykius ir verčiant duomenis į *fuzzy* būseną (angl. *fuzzification*)[AEY11].

c-vidurkių grupavimo metode, duomenys į sugrupuoti taip, kad būtų minimizuota grupių suminė kvadratinė klaida, tai yra minimizuoti:

$$J_\beta(X, V, U) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{\beta} d^2(x_j, v_i)$$

kur u_{ij} yra santykio vertė, v_i – grupės centras, n – kintamųjų skaičius, β – svėrimo eksponentė ($\beta > 1$) ir $d(x_j, v_i)$ – atstumas tarp grupės centro ir stebimo objekto.

Metodas	MSE
Fuzzy1 (Song, Q., & Chissom, B. S. (1993a))	412499
Fuzzy2 (Song, Q., & Chissom, B. S. (1993b))	775687
Fuzzy3 (Sullivan, J., & Woodall, W. H. (1994))	386055
Fuzzy4 (Chen, S. M. (1996))	407507
Fuzzy5 (Cheng, Cheng, and Wang (2008))	228918
Fuzzy6 (Aladag et al. (2009))	78073
Fuzzy7 (Egrioglu et al. (2010))	60714
Fuzzy8 (Egrioglu, Aladag, Basaran et al. (2011))	66661
Fuzzy9 (Egrioglu, Aladag, Yolcu et al. (2011))	60140
Siūlomas metodas	32849

Lent. 1. [AEY11]Siūlomo hibridinio modelio MSE palyginimas su kitais fuzzy metodais

Pažiūrėję į rezultatus matome, kad pasiūlytas metodas rodo geresnius rezultatus negu prieš tai egzistavę modeliai. Deja autoriai nepateikė procentinių palyginimų, taigi sunku spręsti ar šis metodas gana tiksliai prognozuoja.

4.2 Genetiniai algoritmai

Atsiribojant nuo *fuzzy* ir DNT, dar turime genetinį EDDIE algoritmą skirtą finansinėms prognozėms – jis naudoja vienmačius duomenis, taigi lyginti su mūsų daugiamatės įvesties modeliu būtų neadekvatu, bet mes vis tiek pažiūrėsime, ką jis galės parodyti mūsų duomenims.

Pačio algoritmo tikslas yra pasakyti „Ar akcija X pakils r% per ateinančias n dienų?“ – taigi, tai galima pritaikyti ilgalaikėje prognozėje. M. Kampouridis ir F. Otero nusprendė, kad šį algoritmą galima dar labiau optimizuoti įvedus euristinę paiešką [KO15].. Savo optimizacijoje jie naudoja genetinį programavimą ir dinaminį diskretizavimą. Šie metodai labiau tinkami klasifikavimo uždaviniams spręsti – nors mūsų modelį pakoregavus, taip kad jis

	Our Approach		Standard GA with proposed neural network	
n_h	Training error	Forecasting error	Training error	Forecasting error
4	12.1116	13.9734	13.7473	14.6301
5	11.6850	13.8354	13.9495	15.7997
6	11.5730	14.0933	13.1060	14.8927
7	12.1791	14.7434	12.7207	13.9798
8	12.6076	14.3516	15.9594	19.5594

(a)

	Standard GA with traditional neural network		Back-Propagation with Momentum and Adaptive Learning Rate	
n_h	Training error	Forecasting error	Training error	Forecasting error
4	13.7666	15.6682	16.4123	20.1037
5	14.9151	15.7705	21.3844	26.2723
6	12.7433	16.9341	23.1991	28.5170
7	12.7207	14.1280	24.2294	29.6302
8	13.5383	14.2857	24.2323	29.8156

(b)

Lent. 2. [LLL03]Keleto tinklų mokymo ir prognozavimo vidutinės absoliutinės klaidos

spręstų ar reikšmė didės, ar mažės, galima būtų lyginti su šiuo metodu, bet tai jau paliekama

ateičiai. Deja naujausia EDDIE-9 versija dar nėra pasiekiama, dėl to savo lyginimus galėsime atlikinėti tik su EDDIE-8.

Dar vienas genetinis algoritmas buvo naudotas modifikuojant DNT struktūrą ir parametrus [LLL03]. H.K. Lam su kolegomis siūlo naudoti patobulintą genetinį algoritmą, apmokant neuroninį tinklą. Tikslas to yra sukurti dinamišką neuroninį tinklą – t.y. tinklas mokymosi metu gali didėti arba mažėti, pagal poreikį. Tai labai primena mūsų kuriamą modelį, tik mes naudojame fiksuotą augimą.

Iš prognozavimo rezultatų matome, kad šis modelis turi daug mažesnę klaidą negu standartinis tinklas. Taip pat matosi, kad naudojant genetinį algoritmą vietoj atgalinio sklidimo, rezultatai jau tada žymiai pranašesni. Tai svarbu, kadangi mums reikės pasirinkti, kaip mokysime savo tinklą. Kol kas genetiniai algoritmai atrodo daugiau žadantys.

Y. Kwon ir B. Moon siūlo kitokį sprendimą – periodinį neuroninio tinklo modelį, optimizuojant svorius naudojant genetinį algoritmą [MK07]. Jis yra panašus į standartinį neuroninį tinklą – turi įvesties, paslėptus ir išvesties sluoksnius. Esminis skirtumas, kad kiekvienas paslėpto sluoksnio neuronas yra sujungtas su savimi ir visais kitais sluoksnio neuronais. Mokymas vyksta įprastu atgalinio sklidimo algoritmu. Šis modelis bando nuspėti, koks bus pokytis tarp esamos ir kitos dienos. Pagal tai sprendžiama, ką daryti toliau – jeigu rezultatas yra aukščiau teigiamo slenksčio, algoritmas perka akciją, jeigu žemiau neigiamo slenksčio – algoritmas parduoda akciją, kitu atveju nieko nedaroma. Genetinio algoritmo tikslas yra dinamiškai atsinaujinti – tai yra prisitaikyti prie naujų duomenų. Tyrimas buvo atliktas manipuliuojant 36 įmonių akcijas.

Symbols	BUY	SELL	Symbols	BUY	SELL	Symbols	BUY	SELL
AA	2.96	14.92	HD	1.97	-0.02	MRK	1.26	13.64
AXP	0.99	5.50	HON	1.21	4.64	MSFT	1.29	9.06
AYP	1.30	9.01	HWP	2.70	22.62	NMSB	28.82	38.42
BA	1.24	4.49	IBM	0.57	12.26	ORCL	0.30	2.53
C	0.49	3.00	INTC	0.47	6.06	PG	1.83	12.37
CAT	1.89	6.95	IP	1.11	7.40	RYFL	43.47	42.18
DD	1.43	18.27	JNJ	1.94	5.14	SBC	1.58	10.72
DELL	0.60	14.67	JPM	2.87	10.67	SUNW	0.34	8.13
DIS	1.04	9.26	KO	1.14	8.51	T	1.30	12.57
EK	1.18	10.24	MCD	1.77	5.36	UTX	-0.17	5.58
GE	0.76	17.35	MMM	2.65	8.98	WMT	6.84	8.92
GM	6.31	11.62	MO	1.64	7.86	XOM	1.02	18.62

Lent. 3. [MK07] Vidutinis prognozės pagerėjimas naudojant genetinį algoritmą procentais

Šis prognozavimo modelis šiek tiek panašus į šio darbo – Y. Know ir B. Moon atnaujindami modelį atsisako senų duomenų ir prideda naujus, mes duomenims priskirsime svorius, taip, kad kuo naujesnė informacija, tuo didesnę reikšmę jinai turi, o seni duomenys priešingai – jų svoriai mažėja. Vietoj to, kad atsisakyti duomenų, mes mažinsime jų vertę.

Iš lentelės matome, kad genetinis algoritmas beveik visada atneša geresnius rezultatus. Iš to galime tikėtis, kad ir mūsų manipuliacijos turės teigiamą įtaką.

4.3 DNT optimizavimas

J. Ticknor siūlo naudoti Bajesu paremtą neuroninio tinklo modelį – pagal jį, tai sumažina permokymo ir triukšmo interpretavimo riziką [Tic13]. Standartiniame modelyje, tinklo mokymo žingsnyje siekiama mažinti išvesties vidutinę kvadratinę klaidą. Čia prisideda papildomas terminas:

$$F = \beta E_d + \alpha E_w$$

kur F yra tikslo funkcija, E_d – kvadratinių klaidų suma, E_w – kvadratinių tinklo svorių suma, α, β – tikslo funkcijos parametrai. Kadangi bajeso tinkle svoriai yra atsitiktiniai dydžiai, jų tankio funkciją galima užrašyti naudojant bajeso teoremą:

$$P(w|D, \alpha, \beta, M) = \frac{P(D|w, \beta, M)P(w|\alpha, M)}{P(D|\alpha, \beta, M)}$$

kur w yra svorių vektorius, D – įvesties duomenų vektorius, M – naudojamas neuroninio tinklo modelis. Norint optimizuoti tikslo funkcijos parametrus, reikia išspręsti Heseno matricą $F(w)$ minimo taške w^{MP}.

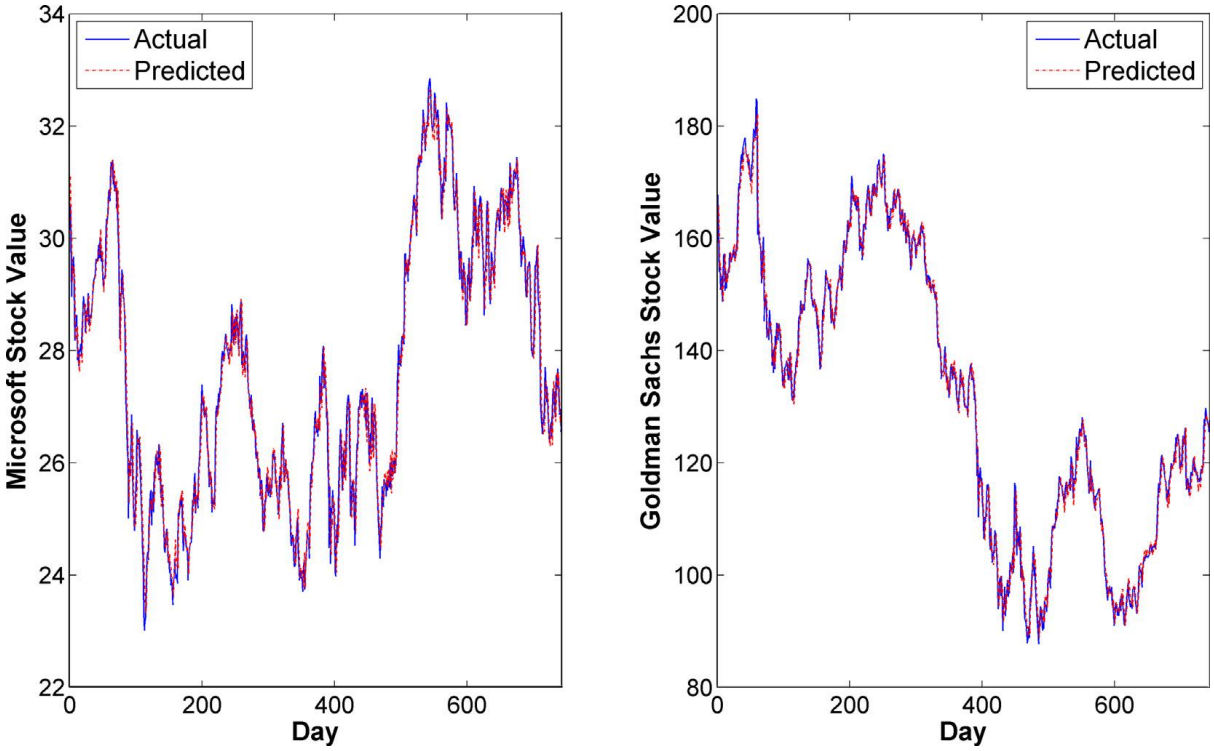
Savo tyrime J. Ticknor naudojo Goldman Sachs Group, Inc. ir Microsoft Corp. Duomenis. Tinklo apmokymui buvo naudojami pirmi 80% duomenų, o testavimui likę 20%.

Rezultatai vaizduoja vienos dienos į priekį akcijų vertę ir išprognozuotą akcijų vertę. Bajeso tinklas duoda vidutiniškai >98% atitikimą abiem atvejais.

Akcija	Mokymo VAPP	Testavimo VAPP	Bendra VAPP
Microsoft	1.0494	1.0561	1.0507
Goldman Sachs	1.5235	1.3291	1.4860

Lent. 4. [Tic13]Tinklo vidutinės absoliutinės procentinės paklaidos

Statistikoje yra gerai žinomas terminas – kryžminė patikra – ji skirta iš duomenų aibės poaibių rinkinio nustatyti, kokios prognozavimo sėkmės galima tikėtis. Panašiai, įkrovimas



Pav. 8. [Tic13]Vienos dienos akcijų vertės – modelio prognozė ir reali

(*bootstrapping*) siekia padidinti numatytos sėkmės pagrįstumą. D. Barrow ir S. Crone siūlo apjungti kryžminę patikrą ir prognozavimą [BC13].

Tyrime buvo naudojami 6 modeliai:

- BESTMLP – geriausias daugiasluoksnis perceptronas iš visų apmokytų.
- 10FOLDCV – 10 lenkimų kryžminė patikrą, visa aibė dalinama į 10 lygių dalių, apmokome su 9, testuojame su 1.
- 2FOLDCV – 2 lenkimų kryžminė patikra, išmaišyta aibė dalinama į 2 dalis, su viena mokome, su kita testuojame.
- MONTECV – Monte Carlo variacija (atsitiktinis dalinimas)
- HOLDOUT – atsitiktinai išrenkamas poaibis, kuris bus naudojamas testavimui, visi likę duomenys naudojami apmokymui.
- BAG – įkrovimas

Iš rezultatų matome, kad kryžminė patikra duoda geresnius rezultatus negu paprastas daugiasluoksnis perceptronas. Tačiau visais atvejais klaida yra ganėtinai didelė, mes norime ją dar labiau minimizuoti.

G. Sermpinis, K. Theofilatos, A. Karathanasopoulos, E. Georgopoulos ir C.Dunis siūlo prognozes neuroniniais tinklais optimizuoti radialinėmis bazinėmis funkcijomis ir dalelių spiečiumi (ARBF-PSO) [DGK12]. Jie tyrė kaip jų hibridinis modelis sugebės nuspėti valiutų kursus lyginant su trimis skirtingoms DNT architektūroms, arčiausio kaimyno (k-NN)

algoritmu, ARMA (šis modelis šiek tiek panašus į mūsų modelį – nauja reikšmė priklauso nuo prieš tai buvusių), MACD ir naivia strategija.

Laiko eilutės ilgis	Metodas	Prognozės horizontas			
		1-3	4-12	13-18	1-18
Ilga	BESTMLP	10.79	16.59	20.02	16.77
	HOLDOUT	9.34	14.96	16.20	14.43
	BAG	9.74	15.46	16.38	14.81
	MONTECV	10.86	15.16	15.43	14.54
	10FOLDCV	10.39	**14.04**	**14.82**	**13.69**
	2FOLDCV	**9.03**	14.64	15.69	14.06
Trumpa	BESTMLP	16.83	17.03	20.66	18.20
	HOLDOUT	17.59	17.04	20.12	18.16
	BAG	17.20	17.27	20.96	18.49
	MONTECV	**15.47**	14.71	19.05	16.28
	10FOLDCV	16.00	15.91	20.25	17.37
	2FOLDCV	15.86	**14.51**	**18.95**	**16.21**

Lent. 5. [GMN10]Simetrinė vidutinė absoliutinė procentinė paklaida skirtingais prognozės atvejais

Radialinės bazinės funkcijos neuroninis tinklas yra beveik tokios pačios architektūros kaip ir įprastų neuroninių tinklų, esminis skirtumas, kad paslėptame sluoksnyje vietoj aktyvavimo funkcijos yra naudojama radialinė bazinė funkcija.

Šiuo atveju tai bus Gauso funkcija:

$$\phi^{[i]}(x_t) = e^{-\frac{||x_t - C_i||^2}{2\sigma_i^2}}$$

kur C_i yra vektorius nusakantis Gauso funkcijos centrą, o σ_i – jo plotį. Dalelių spiečiaus optimizacija yra naudojama rasti C_i.

Iš rezultatų matome, kad pasiūlytas modelis ryškiai pirmauja valiutų prognozėse. Taip pat matosi, kad visi neuroniniais tinklais pagrįsti modeliai duoda geresnius rezultatus negu kiti algoritmai. Taip pat šie skaičiai atvirkščiai proporcingi vidutinėms testavimo klaidoms.

		NAIVE	**ARMA**	**MACD**	**k-NN**	**MLP**	**RNN**	**PSI**	**ARBF-PSO**
EUR/USD	Annualised Return	-0.16%	-13.31%	2.44%	7.96%	12.36%	16.40%	20.20%	25.10%
	Information Ratio	-0.02	-1.26	0.23	0.75	1.17	1.56	1.93	2.41
EUR/GBP	Annualised Return	3.22%	7.89%	-0.11%	7.20%	10.85%	16.70%	17.33%	27.06%
	Information Ratio	0.34	0.84	-0.01	0.77	1.16	1.79	1.86	2.92
EUR/JPY	Annualised Return	-11.80%	-0.07%	-1.53%	4.93%	9.36%	18.24%	16.49%	22.87%
	Information Ratio	-1.10	-0.01	-0.12	0.37	0.70	1.37	1.24	1.72

Lent. 6. [DGK12]Valiutų mainų rezultatai

Šį modelį bus sunku lyginti su mūsų kuriamu, kadangi čia figūruoja optimizacijos, o pas mus galutinis produktas yra neoptimizuotas tinklas.

Dar vienas hibridinis modelis buvo pasiūlytas T.Hsieh, H. Hsiao ir W. Yeh – jie pristatė sistemą, kur yra sujungiamos bangelių transformacijos ir dirbtine bičių kolonija (ABC) pagrįstas periodinis neuroninis tinklas (RNN) [HHY11]. Pati sistema turi 3 stadijas – pirma, bangelių transformacija naudojant Haar bangelę yra taikoma duomenų išskaidymui, taip turėtų būti panaikintas triukšmas. Antra, yra sukuriamas RNN naudojant žingsninę regresiją. Trečia, ABC yra naudojamas optimizuoti tinklo svorius. Kaip tyrimo objektai buvo pasirinkti duomenys iš keleto akcijų biržų – DJIA, FTSE, Nikkei, TAIEX. Pats ABC algoritmas atrodo taip:

- Pradinė fazė
- REPEAT
    - Darbininkių bičių fazė (ieško naujų maisto šaltinių)
    - Žiūrovių bičių fazė (pagal darbininkių informaciją tikimybiškai pasirenka maisto šaltinius)
    - Žvalgių bičių fazė (maisto šaltinius renkasi atsitiktinai)
    - Įsiminti kol kas geriausią situaciją
- UNTIL (nepasiektas maksimalus ciklų kiekis arba nesibaigė darbo laikas)

Šis algoritmas atspindi bičių elgesį ir yra visuotinis optimizavimo algoritmas.

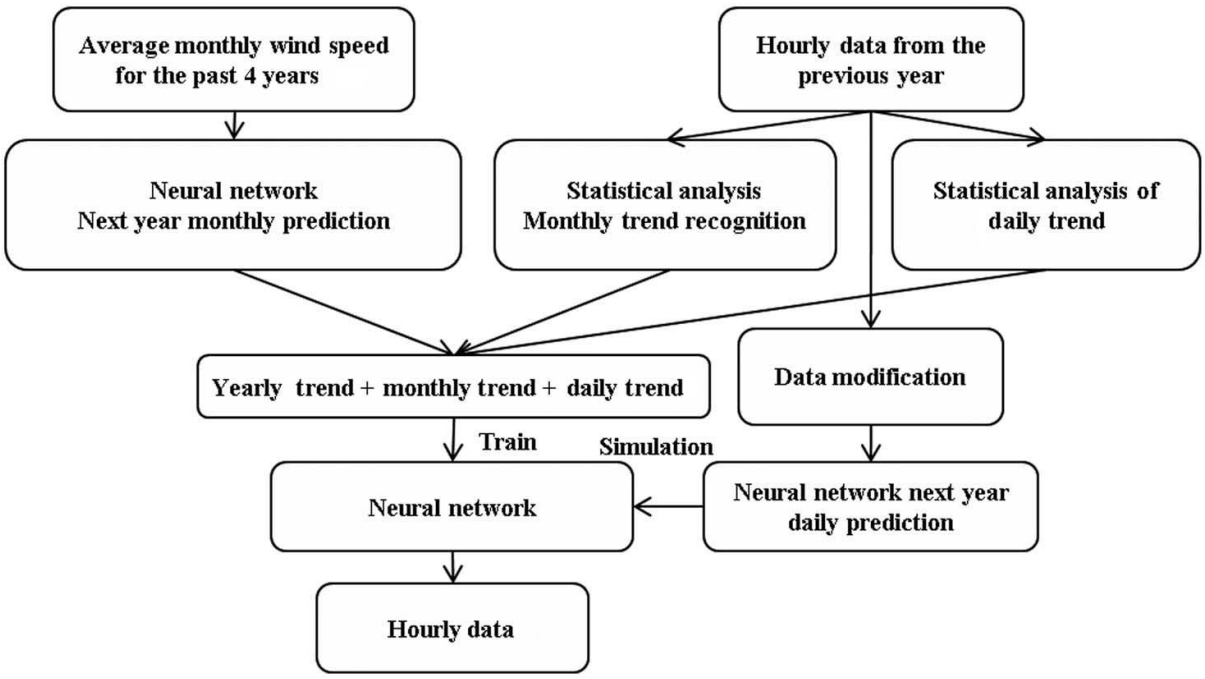
Metai	α	Metodai					
		BP-ANN	BNN	Chen	Yu	Cheng	ABC-RNN
1997	0.02	3	208	-318	-384	791	**1217**
1998	0.03	-922	695	520	-1369	577	**983**
1999	0.03	-131	92	-967	-992	58	**132**
2000	0.01	-93	182	-304	-367	-497	**197**
2001	0.02	190	113	-492	352	61	**1592**
2002	0.005	-127	-49	-46	-299	**387**	15
2003	0.005	-253	183	-533	-501	**341**	163
Bendras		-1333	1424	-2140	-3560	1718	**4299**

Lent. 7. [HHY11]Skirtingų modelių pelno palyginimas (TAIEX)

Rezultatuose modelio rezultatas yra sulygintas su kitais modeliais – BP-ANN (atgalinio sklidimo DNT), BNN (ABC optimizuotas DNT), Chen (*fuzzy*), Yu (*fuzzy*), Cheng (ANFIS). Matome, kad ne visi modeliai neša pelną – įdomu tai, kad abu ABC algoritmu optimizuoti modeliai atnešė pelną. Iš keleto tyrimų galime svarstyti apie tai, kad siekiant gerų rezultatų optimizavimas yra būtinas, tačiau mūsų modelio tikslas yra tas, kad po apmokymo tinklas jau turi generuoti gerus rezultatus.

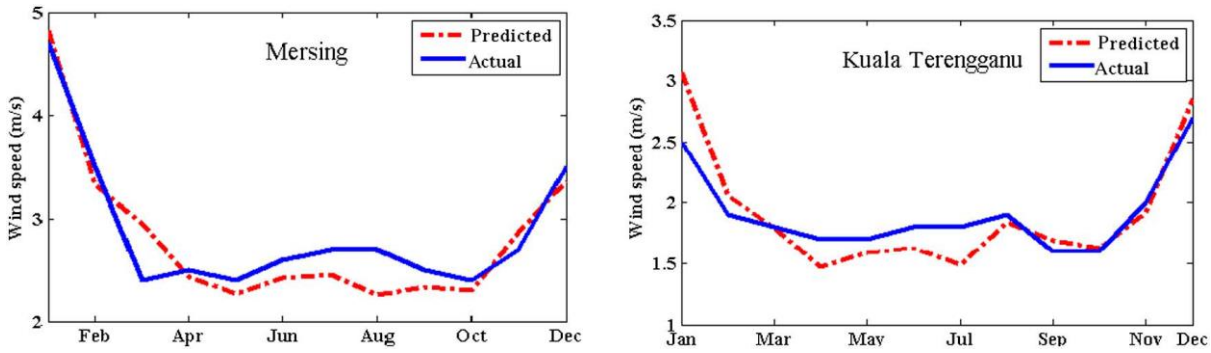
4.4 Kombinuoti tinklai

H. Azad su savo kolegomis siūlo neuroninius tinklus kaip sprendimą ilgalaikėse vėjo greičio prognozėse - tai yra orientuota į elektros rinką ir vėjo jėgaines[AGM14]. Kadangi neuroniniai tinklai pasižymi struktūrų atpažinimu, jie teigia, kad tai yra tinkamas sprendimas prognozuojant valandinius vėjo greičius ateinantiems metams.



Pav. 9. [AGM14]Modelis ilgalaikei vėjo greičio prognozei

Jų siūlomas modelis susideda iš kelių DNT, kurie turi labiau specifines užduotis – šis sprendimas yra įdomus, kadangi kiekvienas tinklas bus mokomas atpažinti stambesnius šablonus, taip galima tikėtis didesnės sėkmės apjungus į vieną. Šis modelis primena DNT tiesinę kombinaciją, kur keleto tinklų surinkta informacija yra apjungiama.



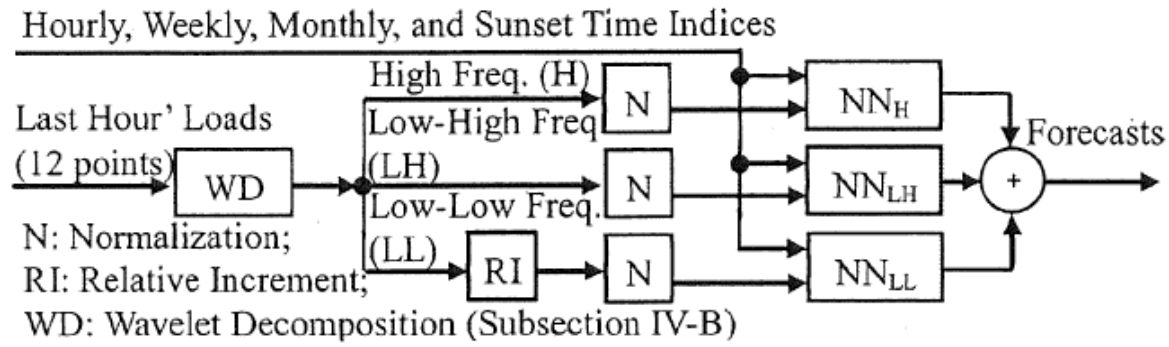
Pav. 10. [AGM14] Vėjo greičio prognozės rezultatai

Iš grafikų (Pav. 10) matome, kad rezultatai neatrodo puikūs, tačiau lyginant su kitais esamais modeliais – jis duoda daug geresnius rezultatus.

Mūsų modelis pasižymės tuo, kad jis galės augti – tai yra didinti neuronų skaičių, bet įdomu, kas būtų jeigu jis augdamas didintų ne tik neuronų skaičių, bet ir tinklų skaičių – tai yra didėdamas jis prisijunginėtų naujus tinklus. Tai yra nesunku įsivaizduoti – tinklas mokymosi eigoje pradėdamas nuo seniausių duomenų, kuriuos vis dar laiko reikšmingais ir kaskart pridėdamas naują informaciją nusprendžia prisijungti naują tinklą – šiam seni duomenys jau nebebus svarbūs ir jis labiau akcentuosis į naujus. Taip galima kurti variaciją su neuroninio tinklo kartomis – t.y. pradinis tinklas yra I kartos, jis išmoksta seną informaciją, ir poto mokosi naują, II kartos jau tik žino kad yra sena informacija, bet mokytis pradeda su naujesne, III karta jau rūpinasi tik naujausia informacija. Visus šiuos tinklus apjungus į visumą galima tikėtis įdomesnių rezultatų.

C.Guan su savo kolegomis tyrė trumpalaikes prognozes, kur siekiama gauti valandos duomenis 5 minučių intervalais į ateitį [FGL13]. Jų pasirinktas modelis naudoja duomenų filtravimą, kur bangelių dekompozicija yra naudojama duomenų išskaidymui į keletą skirtingų dažnių komponentų, kur vėliau keletas DNT iš jų rinks duomenis ir visa surinkta informacija

bus apjungiama į vieną prognozę - kas irgi skamba kaip tiesinė DNT kombinacija. Šis tyrimas labiau orientuotas į specifinį atvejį, kai prognozuojama elektros apkrova.



Pav. 11. [FGL13]Bangelių DNT modelis elektros apkrovai prognozuoti

Iš rezultatų matome, kad rezultatai yra geri tik pačioje pradžioje – toliau klaida vis didėja. Tokie rezultatai mums nėra tinkami, kadangi mums labiausiai rūpi ilgalaikė prognozė.

Min.	MAPE	MAE	SD	MASE
5	0.09	12.52	14.61	0.23
10	0.13	18.45	19.87	0.35
15	0.16	23.72	24.67	0.45
20	0.20	29.36	30.15	0.56
25	0.24	34.49	35.48	0.65
30	0.27	39.89	41.15	0.76
35	0.31	45.36	46.48	0.86
40	0.33	51.06	52.13	0.97
45	0.35	56.32	57.52	1.07
50	0.38	61.80	63.23	1.17
55	0.42	66.06	67.77	1.25
60	0.45	71.02	72.93	1.35

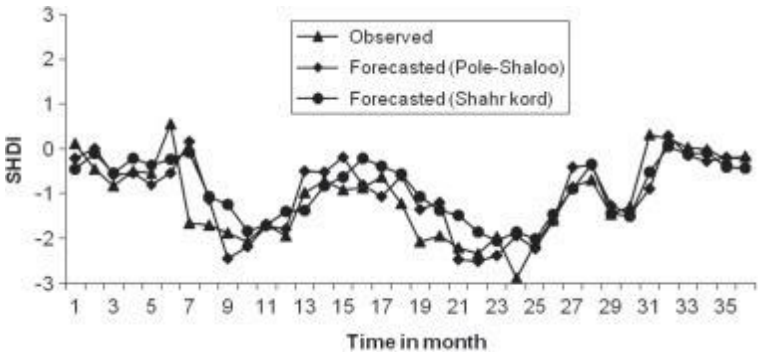
Lent. 8. [FGL13]Elektros apkrovos prognozės modelio rezultatai

4.5 Klasika ir kiti algoritmai

J. Mantri, P. Gahan ir B. Nayak tirdami akcijų biržos nepastovumą pastebėjo, kad nėra didelio skirtomo tarp DNT (dirbtinių neuroninių tinklų) ir GARCH, EGARCH, GJR GARCH ir IGARCH modelių [GMN10]. Kadangi šiuos modelius galima naudoti prognozuojant kainų indeksus, manome, kad juos taip pat galėsime panaudoti lygindami su savo sukonstruoto DNT

rezultatais. Deja, iš pačios publikacijos nieko iš anksto negalima pasakyti, kadangi buvo nagrinėjamas skirtingas objektas.

M. Dehghani su kolegomis analizavo, kaip sekasi prognozuoti sausrą naudojant DNT [DFN13]. Savo tyrime jie naudojo standartinį DNT, kad nustatytų, kokius parametrus naudoti kaip įvestį – jei naudoji principinių komponenčių analizę (PCA). Pagal gautus rezultatus, galima spręsti, kad toks duomenų mažinimas duoda geresnes prognozes – išvengiama nesvarbių parametrų.



Pav. 12. [DFN13]Sausros indekso prognozės rezultatai

Rezultatuose matome, kad prognozės daugmaž vykusios ir principinės komponentės padėjo išsirinkti pagrindinius parametrus. Savo tyrime mes irgi atlikinėsime duomenų tyrimą, kad nuspręstumėme, ar pasirinkti parametrai yra tinkami prognozuoti akcijas – naudosime keletą pagrindinių statistinių metodų kaip PCA, MLP, J48.

Tarp visų prognozavimo algoritmų taip pat egzistuoja atraminių vektorių mašina (SVM). W. Huang, Y. Nakamori ir S. Wang atliko tyrimą, kur šį metodą lygino su atsitiktinio ėjimo modeliu (RW) (šis modelis naudoja esamą reikšmę, kad nustatyti būsimą), tiesine

diskriminantine analize (LDA), kvadratine diskriminantine analize (QDA), Elmano atgalinio sklidimo neuroniniu tinklu (EBNN) bei tiesine visų šių metodų kombinacija (LC) [HNW05].

Šiame tyrime jie labiau taikėsi į klasifikavimą negu į tikslios reikšmės prognozavimą – rezultatas galėjo būti arba kaina kris, arba kils. Tokį pat tyrimą galima atlikti ir su mūsų modeliu – rezultatai turėtų būti didesni, kadangi mes naudosi daugiau įvesties parametrų.

Klasifikavimo metodas	Pataikymo procentas
RW	50
LDA	55
QDA	69
EBNN	69
SVM	73
LC	75

Lent. 9. . [HNW05]Prognozavimo rezultatai

5 Programa

5.1 Technologijos

5.1.1 .NET C#

Programa realizuota Microsoft .NET C# programavimo kalba – tai objektiškai orientuota aukšto lygio programavimo kalba sukurta, kad butų paprasta ir moderni. Naudojant .NET Framework įrankius galima nesunkiai manipuliuoti duomenų struktūromis bei duomenų bazėmis naudojant LINQ (funkcijų rinkinys leidžiantis naudoti užklausas išvardijamosiose struktūrose).

Funkcija skaičiuojanti vidutinę absoliutinę klaidą naudojant LINQ:

```
private double CalcMAE(double[] one, double[] two)
{
    return one.Select((t, i) => Math.Sqrt(Math.Pow(t - two[i],
        2))).Sum()/(double)one.Length;
}
```

5.1.2 WPF

Windows Presentation Foundation (WPF) – grafinė sub-sistema skirta vartotojo sąsajos vaizdavimui. Ši technologija kartu naudoja XAML – XML pagrįstą kalbą, kuria galima kurti elementus, juos susieti su duomenimis. Tai nesudėtinga ir labai patogi priemonė greitai sukurti moderniai atrodančią vartotojo sąsają neįdedant per daug pastangų stengiantis valdyti visus valdyklių pasikeitimus

XAML ištrauka:

```
<telerik:Label x:Name="m_statusLabel" HorizontalAlignment="Left"
    Margin="265,10,0,0" VerticalAlignment="Top" Height="700" Width="900"
    HorizontalContentAlignment="Center">
    <telerik:Label.Visibility>
        <MultiBinding Converter="{StaticResource VisiblityConverter}"
            ConverterParameter="any">
            <Binding Path="IsSelected" ElementName="m_btnServer"
                Converter="{StaticResource BoolToVisInvert}"
                ConverterParameter="Normal"/>
            <Binding Path="IsSelected" ElementName="m_btnClient"
                Converter="{StaticResource BoolToVisInvert}"
                ConverterParameter="Normal"/>
        </MultiBinding>
    </telerik:Label.Visibility>
</telerik:Label>
```

5.2 Duomenys

Buvo naudojama dešimt daugiamačių laiko eilutės trendo duomenų paketų, kur kiekvienas jų susidėjo iš apmokymo ir testavimo įrašų. Visi paketai tarpusavyje nesusiję, dauguma skiriasi savo dydžiu ir atributų skaičiumi.

Mokymo ir testavimo duomenys yra laikomo skirtinguose bylose – programos vykdymo metu jie yra apjungiami į vieną bendrą struktūra. Visi duomenys normalizuoti intervale $[0.3; 0.7]$.

Duomenys	*Mokymo*	*Testavimo*	*Atributai*
DS1	8462	9229	65
DS2	9020	8887	5
DS3	4274	5687	15
DS4	8100	8100	5
DS5	570	750	9
DS6	255	450	9
DS7	255	450	9
DS8	630	1125	9
DS9	960	1500	9
DS10	2400	85536	5

Lent. 10. Duomenys

5.3 Programinė duomenų struktūra

Visi duomenys buvo laikomi *ConcurrentDictonary* tipo objekte (žodyne), taip duodant saugų priėjimą naudojant daugiau negu vieną giją. Kaip raktas naudojamas subendrintas duomenų paketo vardas, o reikšmė yra klasė apjungianti mokymo ir testavimo duomenis, kurie yra išreikšti, kaip daugiamačiai masyvai.

5.4 Neuroninis tinklas

Realizuoti pačius DNT buvo naudota NeuronDotNet biblioteka, kurioje teko atlikti programinės sąsajos (API) pakeitimus, norint realizuoti parametrų ir struktūros modifikacijas.

Siekiant optimizuoti procesą ir atminties sunaudojimą, tinklai yra apmokomi lygiagrečiai, apribojant darbą iki šešių aktyvių gijų. Duomenys, kuriems nereikalingos modifikacijos, buvo išsisaugoti, ir panaudoti keliuose tinkluose.

5.5 Procesas

Programos veikimą procesą apibrėžiama kaip seką veiksmų:

- **Duomenų įkėlimas**

  Vartotojas pasirenka visas apmokymo ir testavimo bylas, kurias programa turės naudoti.

- **Tinklų sukūrimas**

  Kiekvienai konfigūracijai yra sugeneruojamas atskiras tinklas, visi jie yra inicializuojami identiškais svoriais.

- **Mokymo duomenų generavimas**

  Kiekvienai konfigūracijai yra sugeneruojamas duomenų rinkinys, kuris bus perduodamas tinklui. Keli tinklai gali naudoti tą patį rinkinį, jeigu pats rinkinys nėra varijuojamas.

- **Bazinis tinklo apmokymas**

  Visi tinklai yra apmokomi su atitinkamu duomenų rinkiniu. Iteracijų skaičius visais atvejai yra tas pats.

- **Papildomas tinklo apmokymas**

  Kiekvienam tinklui apmokymas yra pratęsiamas su tais pačiais duomenimis konfigūracijoje nurodytu papildomo apmokymo kiekiu.

- **Metrikos skaičiavimas**

  Kiekvienam tinklui yra apskaičiuojamos metrikos apmokymo ir testavimo duomenims.

- **Grafikų braižymas**

  Visiems tinklams yra nubraižomi du grafikai – apmokymo ir testavimo. Kiekvienam iš jų matomas tinklo generuojamas rezultatas ir rezultatas pagal duomenis.

5.6 Metrika

Kadangi duomenų paketai nėra vienodi, visuose tyrimuose buvo naudota viena bendra metrika – vidutinė absoliuti klaida (MAE).

$$MAE = \sum_{i=1}^{n} \frac{\sqrt{(o_n - o_r)^2}}{n}$$

, kur o_n yra tinklo reikšmė, o_r duomenų reikšmė, ir n – duomenų kiekis.

5.7 Konfigūracija

Programą galima konfigūruoti keičiant *NetworkConfigs.json* dokumentą. Konfigūracija atspindi *NetworkConfig* masyvą ir yra atvaizduojama JSON formatu.

```
{
  "Variation": "None",
  "TotalLayers": 3,
  "HiddenLayerNeurons": 10,
  "LearningRate": 0.2,
  "TrainIterations": 2000.0,
  "TrainSplit": 1,
  "AdditionalLearning": 0.25,
  "UseNiab": false,
  "StartingNeuronCount": 4
}
```

Konfigūracijos laukų paaiškinimas:

- Variation – kokia variacija bus naudojama
  - None – variacijos neatliekamos
  - Layer – sluoksniai
  - NeuronFast – greitas neuronų augimas
  - NeuronLastLayer – paskutinio sluoksnio neuronų augimas
  - NeuronSlow – lėtas neuronų augimas
  - RateInc – didėjantis mokymosi tempas
  - RateDec – mažėjantis mokymosi tempas
  - InputOld – senėjantys duomenys
  - InputNew – naujėjantys duomenys
- TotalLayers – paslėptų sluoksnių kiekis
- LearningRate - mokymosi tempas
- TrainIterations – mokymosi iteracijų skaičius

- TrainSplit – mokymosi išskaidymas į dalis (naudojama sluoksnių ir neuronų variacijose).
- AdditionalLearning – mokymo pratesimas kartais
- UseNiab (nustatomas pagal variaciją) – nurodymas, kad naudoti duomenų vektoriaus vidinį mokymosi tempą
- StartingNeuronCount – neuronų variacijose nurodoma, koks bus pradinis neuronų kiekis sluoksnyje

6 Baziniai atvejai

Iš viso tyrimą galima skirstyti į tris dalis – A.25, A.5, A1, kur kiekvienoje jų, apmokymo procesas yra pratęstas skirtingai – skaičius nurodo, kokia dalis visų epochų, buvo dar papildomai panaudota. T.y., jeigu mokėme 1000 epochų, tai atitinkamai kiekvienu atveju apmokymas bus pratęstas dar 250, 500, 1000.

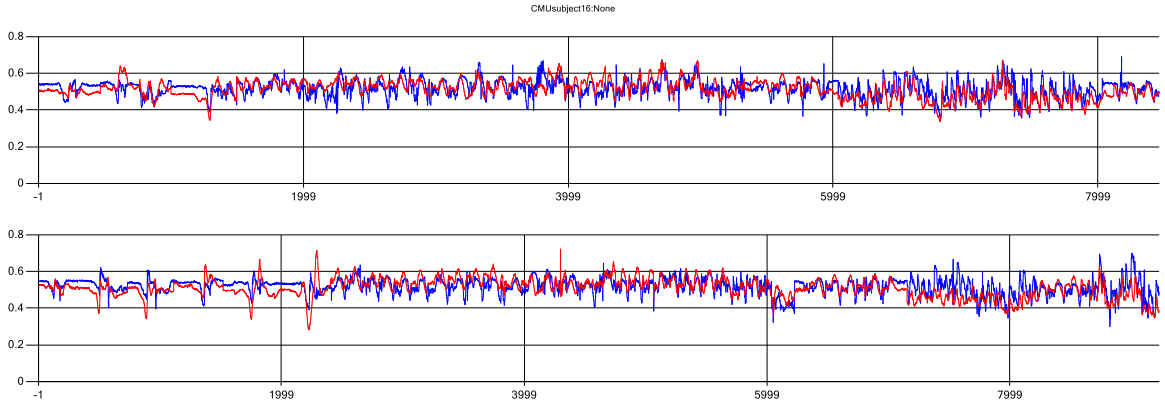
Kiekvienas atvejis turėjo savo atskirą tinklą, kuris susidarė iš vieno tiesinio įvesties sluoksnio, trijų sigmoidinių paslėptų sluoksnių ir vieno tiesinio išvesties sluoksnio. Įvesties sluoksnis susideda iš tiek neuronų, kad sutaptų su duomenų vektoriaus ilgiu, kiekvienas paslėptas sluoksnis turi dešimt neuronų, o išvesties sluoksnis – vieną neuroną, tinklų apmokymo tempas – 0.2, o bazinis iteracijų skaičius – 2000.

Atlikus bandymą su baziniais nustatymais, buvo gauti tokie rezultatai:

Duomenys	*A.25*	*A.5*	*A1*
DS1	0.0362	0.0360	0.0357
	0.0388	0.0387	0.0385
DS2	0.0201	0.0200	0.0198
	0.0207	0.0205	0.0203
DS3	0.0446	0.0447	0.0438
	0.0432	0.0437	0.0434
DS4	0.0536	0.0533	0.0531
	0.0546	0.0545	0.0545
DS5	0.0189	0.0116	0.0114
	0.0249	0.0181	0.0170
DS6	0.0140	0.0135	0.0127
	0.0117	0.0113	0.0107
DS7	0.0144	0.0141	0.0130
	0.0160	0.0158	0.0152
DS8	0.0079	0.0075	0.0071
	0.0053	0.0054	0.0057
DS9	0.0050	0.0049	0.0049
	0.0068	0.0068	0.0068
DS10	0.0839	0.0831	0.0825
	0.0842	0.0834	0.0829

Lent. 11. Tinklų MAE baziniu atveju

Pirmoje kiekvieno duomenų rinkinio lentelės eilutėje matome apmokymo rezultatus, o antrojoje – testavimo.



Pav. 13. DS1 grafikai baziniu atveju (0.25 pratęsimas)

Viršutinis grafikas atspindi mokymo duomenis, o apatinis – testavimo. Raudonai yra žymimi duomenys, mėlynai – tinklo rezultatai. Y ašis rodo normalizuotą reikšmę, X ašis – vektoriaus eilės numerį duomenų rinkinyje.

Iš abiejų grafikų matome, kad tinklas nėra iki galo apmokytas, ryškūs trūkumai. Esant didesniam apmokymo pratęsimui, tikimasi matyti mažesnes klaidas.

Toliau gautus rezultatus lyginsime su baziniais ir bandysime išsiaiškinti, ar pasirinkta variacija daro teigiamą įtaką tinklo apmokymo procesui.

7 Parametrų varijavimas

Viena didžiausių neuroninių tinklų problema yra topologijos parinkimas. Norėdamas gauti geriausius rezultatus, tam turi sugebėti surasti tam tinkamą tinklo struktūrą – kas nėra triviali užduotis. Jeigu tinklas, pats galėtų išsiaiškinti, kokia topologija yra optimaliausia, tai stipriai pagreitintų visą darbą naudojant DNT. Tokie sprendimai jau yra taikomi tinkluose, kurie nenaudoja „mokymo su mokytoju“, kaip saviorganizuojantis neuroninis tinklas (SOM)[ABC11] ir robotikoje, kur naudojamas palaipsniui augantis DNT [CMM00]. Manome, kad tai galima pritaikyti ir mūsų nagrinėjamiems tinklams.

7.1 Mokymosi tempas

7.1.1 Didėjantis tempas

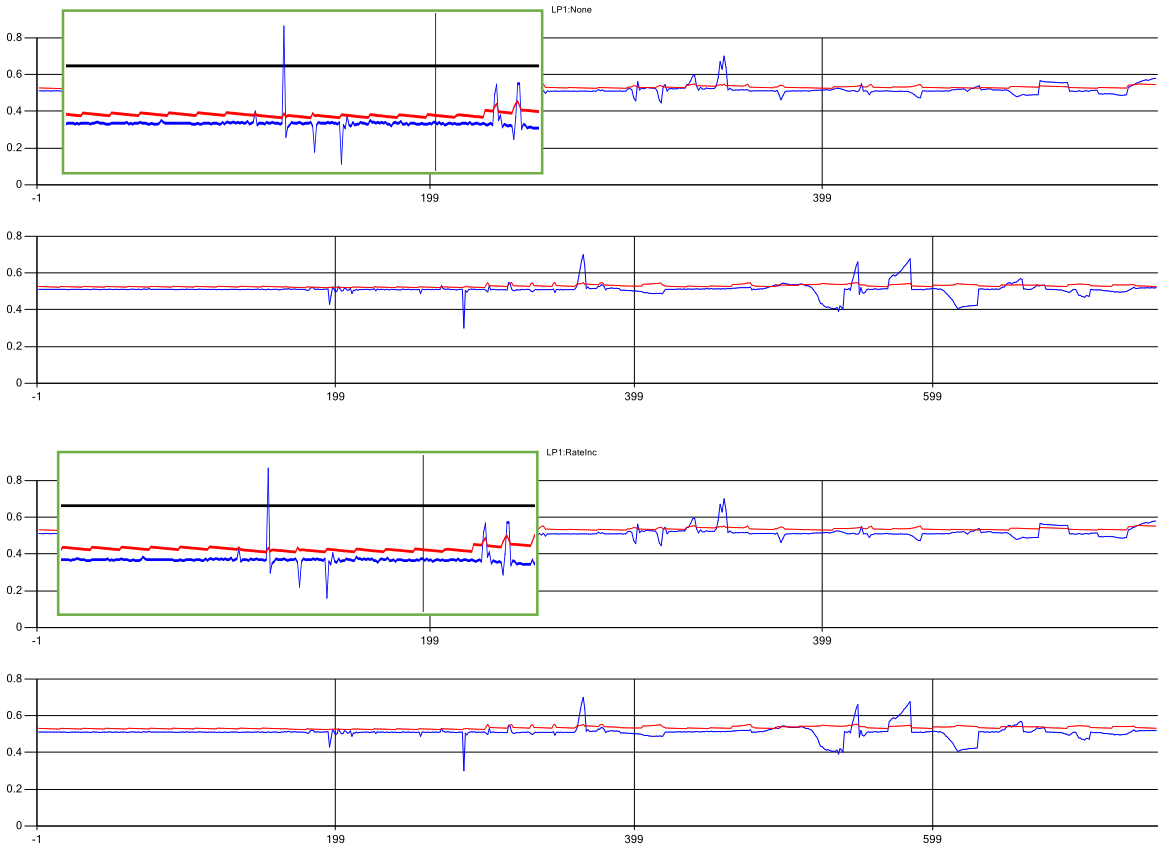
Šio varijavimo metu, viso tinklo mokymosi tempas yra didinamas palaipsniui – panaudojus duomenų vektorių, jo vidinis mokymosi tempas yra didinamas. Kadangi stengtasi išlaikyti kuo panašesnes sąlygas, tempas buvo tiesiškai išskaidytas intervale [0.1; 0.3], taip gaunant vidutinę vertę tokią pat, kaip ir baziniu atveju - 0.2.

Duomenys	*A.25*	*A.5*	*A1*
DS1	0.0383 (5.8%)	0.0376 (4.4%)	0.0362 (1.3%)
	0.0412 (5.9%)	0.0407 (4.9%)	0.0391 (1.6%)
DS2	0.0203 (0.4%)	0.02 (0%)	0.0198 (-0.4%)
	0.0208 (0.3%)	0.0205 (-0.2%)	**0.0203 (-0.6%)**
DS3	**0.0496 (11.1%)**	**0.0482 (7.7%)**	0.0444 (1.4%)
	0.0476 (10%)	**0.0467 (6.9%)**	0.0438 (0.9%)
DS4	0.0551 (2.7%)	0.0549 (3%)	0.0567 (6.7%)
	0.0556 (1.7%)	0.056 (2.8%)	**0.0591 (8.5%)**
DS5	**0.0143 (-24.5%)**	**0.0116 (-0.5%)**	**0.0128 (11.6%)**
	0.0213 (-14.4%)	**0.0175 (-3.7%)**	0.0176 (3.3%)
DS6	0.0136 (-2.8%)	0.0129 (-5%)	**0.01143939 (-10.1%)**
	0.0115 (-2.5%)	0.011 (-3.1%)	0.0111 (2.7%)
DS7	0.0146 (0.6%)	0.0141 (-0.3%)	0.0126 (-3.7%)
	0.0165 (2.4%)	0.0163 (2.7%)	0.0154 (0.9%)
DS8	0.008 (0.3%)	0.0076 (0.8%)	0.0073 (1.1%)
	0.0055 (2.8%)	0.0058 (5.5%)	0.006 (5.1%)
DS9	0.005 (-0.6%)	0.005 (-0.5%)	0.0049 (-1.1%)
	0.0068 (0.1%)	0.0069 (0.5%)	0.0069 (0.6%)
DS10	0.0837 (-0.3%)	0.0829 (-0.2%)	0.0827 (0.2%)
	0.084 (-0.3%)	0.0832 (-0.3%)	0.0831 (0.2%)

Lent. 12. Didėjančio mokymosi tempo variacijos MAE rezultatai

Iš rezultatų matome, kad labai nedidelis kiekis tinklų turėjo geresnius rezultatus. A.25 atveju galima pastebėti, kad DS5 tinklas vienintelis turėjo didesnį klaidos sumažėjimą – beveik visais kitais atvejais, klaida svyruoja mažu procentu.

Didinant pratęsimą, klaida vis mažiau svyruoja, mažiau tinklų turi teigiamą pokytį. Pasižiūrėjus į sugeneruotus grafikus, skirtumai yra minimalūs – kai kur sunkiai pastebimi.



Pav. 14. DS5 A.25 tinklo bazinis (viršuje) ir rezultatas didinant mokymo tempą (apačioje)

Peržvelgę visus skaičius, nematome jokių ryškių pokyčių – grafikuose taip pat vyrauja minimalūs nukrypimai, kurie nerodo jokio teigiamo ar neigiamo poveikio. Galime spręsti, jog ši variaciją, pasirinktoje struktūroje ir turimiems duomenims, nedaro nei teigiamos, nei neigiamos įtakos.

7.1.2 Mažėjantis tempas

Čia situacija panaši kaip ir didinant mokymosi tempą. Esminis skirtumas – mokymo tempas mažėja, kas atitinka mūsų NIAB taisyklę – svarbiausiai viskas, kas yra nauja.

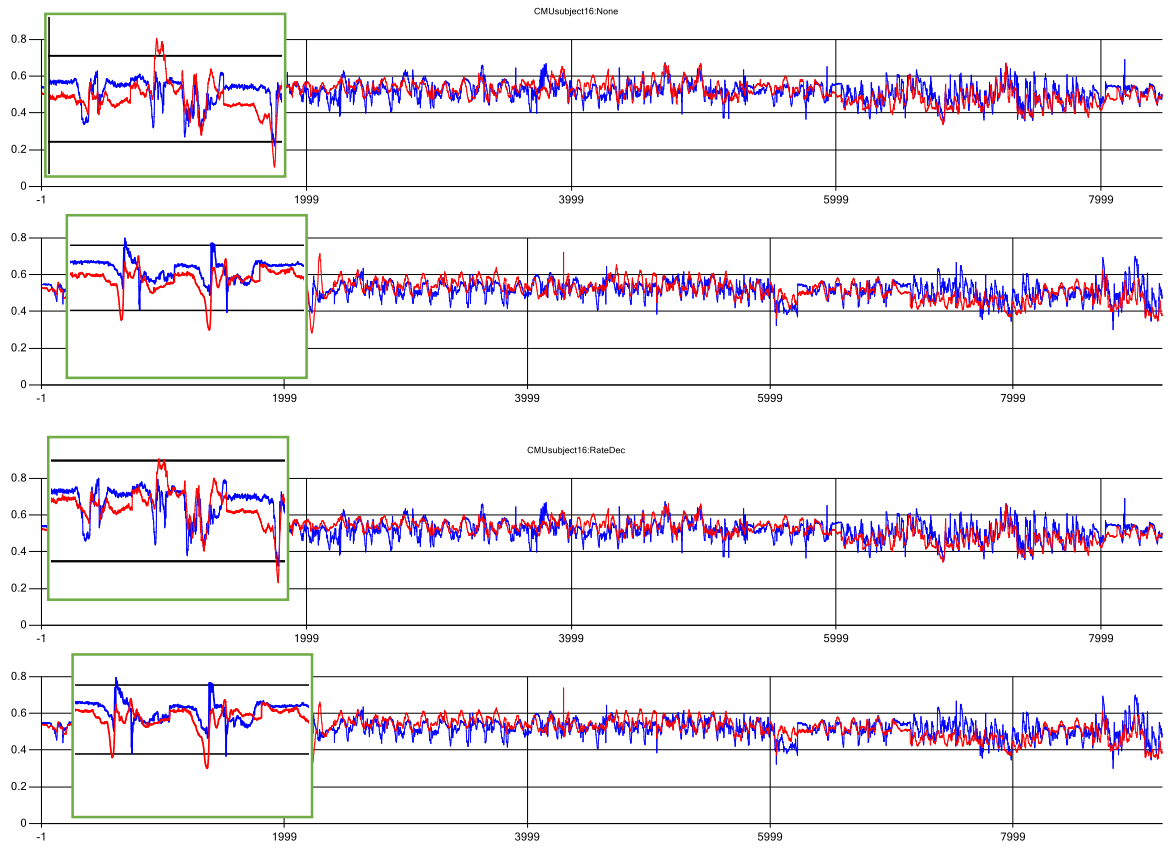
Didžiausias iššūkis yra teisingai parinkti mokymosi tempo mažėjimo funkciją, kadangi jeigu mažinimas vyksta per lėtai – bus veltui naudojamas tinklo darbo laikas šokinėjant dideliais pokyčiais, jam vykstant per greit – tinklas pradės vis mažiau keistis ir gali nebespėti apsimokyti [JKL16].

Ši variacija jau yra naudojama kai kuriuose DNT bibliotekose.

Duomenys	*A.25*	*A.5*	*A1*
DS1	0.0333 (-7.9%)	0.0336 (-6.9%)	0.0341 (-4.7%)
	0.0361 (-7.2%)	0.0361 (-6.9%)	0.0365 (-5.3%)
DS2	0.0202 (-0.2%)	0.02 (-0.2%)	0.0199 (0.1%)
	0.0208 (0.2%)	0.0206 (0.2%)	0.0205 (0.5%)
DS3	**0.0396 (-11.4%)**	**0.0398 (-10.9%)**	**0.0403 (-8%)**
	0.0391 (-9.6%)	**0.0396 (-9.3%)**	**0.0405 (-6.7%)**
DS4	0.0535 (-0.3%)	0.0535 (0.3%)	0.0534 (0.4%)
	0.0549 (0.5%)	0.0553 (1.4%)	0.0552 (1.2%)
DS5	**0.0193 (1.5%)**	**0.0162 (39.5%)**	0.0109 (-5%)
	0.0242 (-2.8%)	**0.0216 (18.6%)**	0.0168 (-1.3%)
DS6	0.0139 (-0.7%)	0.0137 (1.1%)	0.0341 (-4.7%)
	0.0117 (-1%)	0.0115 (0.8%)	**0.0133 (4.8%)**
DS7	0.0137 (-5.7%)	0.0134 (-4.7%)	0.0111 (2.8%)
	0.0149 (-7.2%)	0.0147 (-7.3%)	0.0129 (-1.2%)
DS8	0.008 (1.1%)	0.0079 (4.8%)	**0.0143 (-6.1%)**
	0.0052 (-3.6%)	0.0053 (-3.5%)	0.0077 (6.7%)
DS9	0.005 (0.7%)	0.005 (0.9%)	0.0054 (-5.5%)
	0.0068 (0.2%)	0.0068 (0%)	0.005 (1.1%)
DS10	0.084 (0%)	0.0835 (0.4%)	0.0068 (-0.7%)
	0.0846 (0.4%)	0.0841 (0.8%)	0.0827 (0.2%)

Lent. 13. Mažėjančio mokymosi tempo variacijos MAE rezultatai

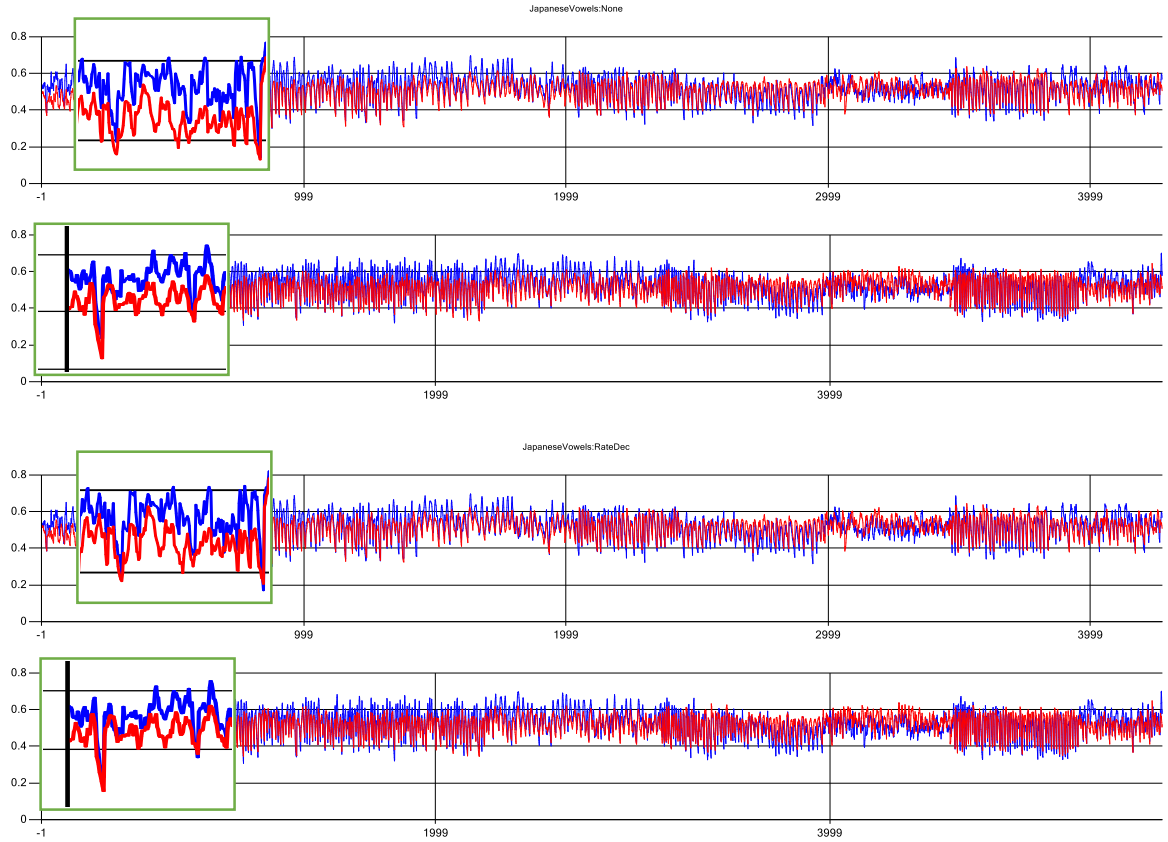
Iš rezultatų matome, kad daugiau tinklų turi šiokį tokį pagerėjimą – skaičiai vis vien išlieka daugmaž artimi baziniam atvejui. Pažvelgus į grafikus galima matyti ryškesnius pokyčius kai kuriems duomenims.



Pav. 15. DS1 A.25 tinklo bazinis (viršuje) ir rezultatas mažinant mokymo tempą (apačioje)

DS1 tinklo grafike (Pav. 6) aiškiai pastebimas skirtumas pirmame ketvirtyje – tiek mokymo, tiek testavimo reikšmės yra geriau atpažintos. Tinklui praėjus didžiąją dalį mokymo, nustatytas mokymosi tempas buvo mažesnis negu baziniu atveju, kas galėjo lemti tai, kad duomenys buvo mažiau pasistūmėję – turint žemesnį tempą, tinklas lėčiau mokosi, bet kartu ir turi mažesnius nukrypimus.

Labai panaši situacija gavosi ir DS3 tinkle (Pav. 7) – pirmasis ketvirtis yra pasislinkęs žemyn baziniu atveju, didesnius skirtumus pamatyti sunku. Pagal MAE šie tinklai ir turėjo rodyti tikslesnius grafikus, likusiuose sunku matyti skirtumus ir ar buvo aiškus pagerėjimas ar pablogėjimas.



Pav. 16. DS3 A.25 tinklo bazinis (viršuje) ir rezultatas mažinant mokymo tempą (apačioje)

Atsižvelgę į visus turimus rezultatus matome, kad kai kuriais atvejais variacija duoda geresnius grafikus ir mažesnes klaidas, kitais skirtumai nereikšmingi. Kadangi tokia variacija neatnešė matomų pablogėjimų, laikome ją vykusia – taip pat galima ją tirti naudojant ne tiesinę mažėjimo funkciją.

7.2 Duomenys

Čia varijuosime duomenų vektorių vidinius mokymosi tempus – tai reikš, kad vieni duomenys turės didesnę įtaką rezultatui negu kiti. Išskyrėme 2 atvejus – kai vidinis tempas mažėja artėjant link duomenų rinkinio pabaigos, kitas – kai didėja. Nustatinėdami tempus naudosime intervalą $[0.1; 0.4]$, iš kurio tiesiškai priskirsime vertes duomenims. Ši variacija pagal teoriją atrodo teisingiausia naudoti, kai yra žinoma, kad dalis duomenų mažai svyruoja, o kita stipriai šokinėja – taip tinklas labiau orientuosis į svyruojančią dalį.

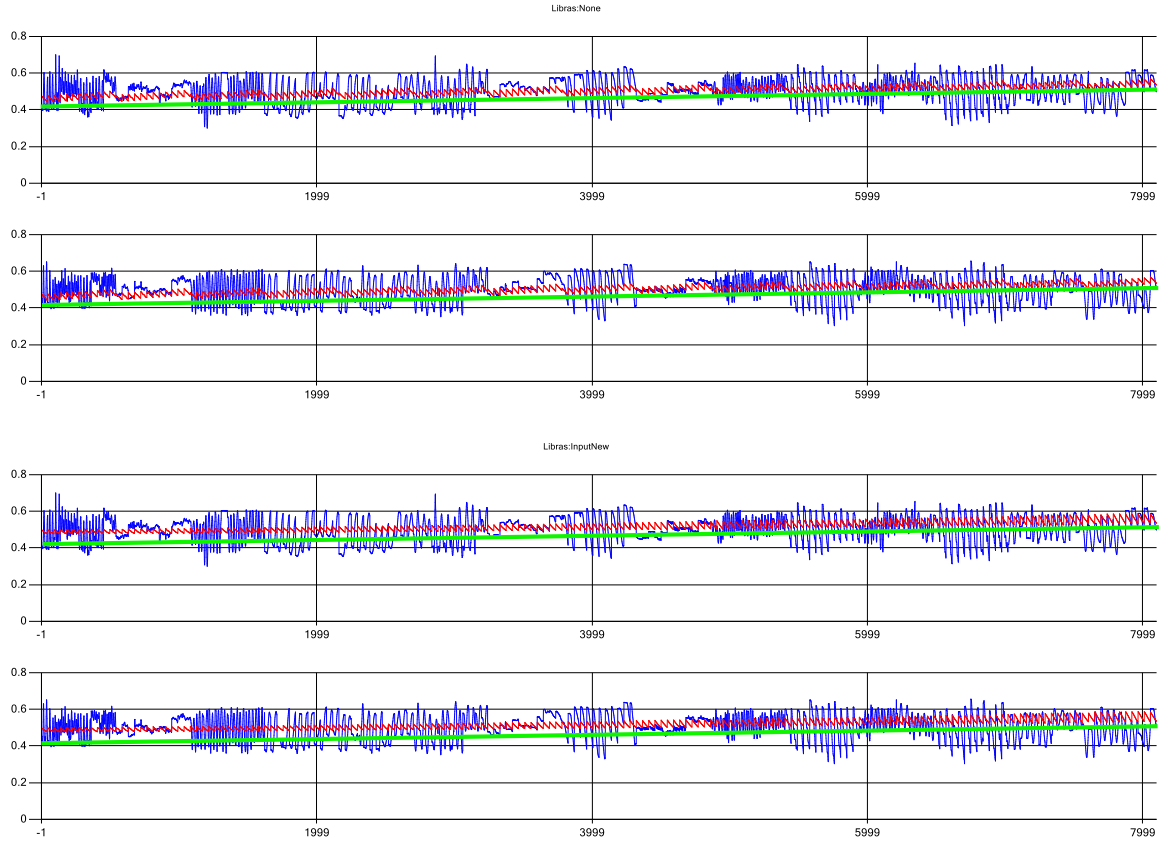
7.2.1 Naujėjantys duomenys

Šiuo atveju vidinis tempas didės – tai reiškia, kad pirmas duomenų rinkinio elementas turės mažiausią įtaką, jo paskutinis didžiausią.

Duomenys	***A.25***	***A.5***	***A1***
DS1	0.0382 (5.5%)	0.0377 (4.4%)	0.0361 (1%)
	0.0412 (6%)	0.0408 (5.1%)	0.0392 (1.7%)
DS2	0.0203 (0.4%)	0.02 (-0.1%)	0.0198 (-0.4%)
	0.0208 (0.3%)	0.0205 (-0.3%)	0.0203 (-0.6%)
DS3	0.0495 (10.8%)	**0.0481 (7.7%)**	0.0445 (1.6%)
	0.0477 (10.2%)	**0.0469 (7.2%)**	0.044 (1.3%)
DS4	0.0545 (1.6%)	0.0535 (0.4%)	0.053 (-0.2%)
	0.0551 (0.9%)	0.0542 (-0.5%)	**0.054 (-0.9%)**
DS5	**0.023 (21.2%)**	0.0115 (-1.1%)	**0.0127 (10.4%)**
	0.0292 (17.1%)	0.0181 (-0.2%)	0.0177 (3.8%)
DS6	0.0144 (2.9%)	0.0137 (1.1%)	**0.0123 (-3.6%)**
	0.0122 (3.6%)	0.0116 (1.6%)	0.0108 (0.3%)
DS7	0.0148 (2.1%)	0.0144 (2.3%)	0.013 (-0.6%)
	0.0165 (2.9%)	0.0165 (4%)	0.0157 (2.9%)
DS8	0.0082 (2.9%)	0.0077 (1.5%)	0.0072 (0.8%)
	0.0055 (1.8%)	0.0057 (3.7%)	**0.006 (4.4%)**
DS9	**0.005 (-0.5%)**	**0.005 (-0.4%)**	0.0049 (-0.9%)
	0.0068 (0.1%)	0.0069 (0.5%)	0.0069 (0.6%)
DS10	0.0838 (-0.2%)	0.083 (-0.2%)	0.0827 (0.2%)
	0.0841 (-0.2%)	**0.0832 (-0.3%)**	0.0831 (0.2%)

Lent. 14. Naujėjančių duomenų variacijos MAE rezultatai

Iš rezultatų ganėtinai aiškus vaizdas – minimalūs pokyčiai į teigiamą pusę, didesni į neigiamą. Visais atvejais matomi tokie pat rezultatai.



Pav. 17. DS4 A.25 tinklo bazinis (viršuje) ir naujėjančių duomenų rezultatas(apačioje)

Iš grafikų galima pastebėti, kad baziniu atveju jis labiau vingiuotas, variacijos – labiau „tiesinis“ ir kylantis. Dalis problemas gali būti dėl vidinio tempo intervalo – jeigu išrinktas per daug platus spektras, tinklas gali per daug koncentruotis į vieną dalį, atvirkščiai, gali gautis rezultatas beveik identiškas baziniui. Spėjame, kad ši variaciją turėtų būti naudojama tik specifiniams duomenims, kur yra aišku, kad vienoj puse didesni šuoliai.

7.2.2 Senėjantys duomenys

Šiuo atveju tempas mažėja - tikimasi matyti panašius rezultatus kaip ir didinant tempą, kadangi sąlygos yra identiškos.

Duomenys	A.25	A.5	A1
DS1	0.0336 (-7.3%)	0.0339 (-5.9%)	0.0344 (-3.9%)
	0.0363 (-6.6%)	0.0365 (-5.8%)	0.0368 (-4.5%)
DS2	0.0201 (-0.5%)	0.02 (-0.2%)	0.0199 (0.3%)
	0.0207 (-0.2%)	0.0206 (0.1%)	**0.0205 (0.7%)**
DS3	0.0398 (-10.9%)	**0.04 (-10.4%)**	0.0405 (-7.6%)
	0.0394 (-9%)	0.0398 (-9%)	**0.0406 (-6.6%)**
DS4	0.0536 (-0.2%)	0.0535 (0.4%)	0.0534 (0.5%)
	0.0551 (0.9%)	**0.0553 (1.5%)**	0.0552 (1.2%)
DS5	**0.0133 (-30%)**	0.011 (-5.1%)	0.0107 (-7.1%)
	0.0191 (-23.4%)	0.017 (-6.6%)	0.0164 (-4.1%)
DS6	0.0128 (-8.6%)	0.0127 (-6.5%)	0.0124 (-2.3%)
	0.0106 (-10.4%)	0.0105 (-7.8%)	0.0104 (-4%)
DS7	0.0131 (-9.5%)	0.0129 (-8.8%)	0.0124 (-5.5%)
	0.0144 (-10.4%)	**0.0142 (-10.5%)**	**0.0138 (-9.4%)**
DS8	0.0079 (-0.3%)	**0.0079 (3.6%)**	**0.0073 (2.1%)**
	0.0053 (-1.2%)	0.0054 (-1.5%)	0.0054 (-5.6%)
DS9	**0.005 (0.5%)**	0.005 (0.7%)	0.005 (0.9%)
	0.0068 (0.3%)	0.0068 (0%)	0.0068 (-0.5%)
DS10	0.0839 (-0.1%)	0.0834 (0.3%)	0.0826 (0.1%)
	0.0844 (0.2%)	0.084 (0.6%)	0.0832 (0.3%)

Lent. 15. Senėjančių duomenų variacijos MAE rezultatai

Čia rezultatai kur kas kitokie – daug daugiau tinklų rodo mažesnes klaidas, kai kur grafikai rodo labai panašius vaizdus kaip ir mažinant mokymosi tempą – dalis grafiko mažiau nukrypęs. Bendram vaizde, matom mažesnes klaidas, tačiau dauguma grafikų nerodo pagerėjimų.

Ši variacija, kaip ir naujėjančių duomenų, yra tinkama naudoti tiktais specifiniams duomenims.

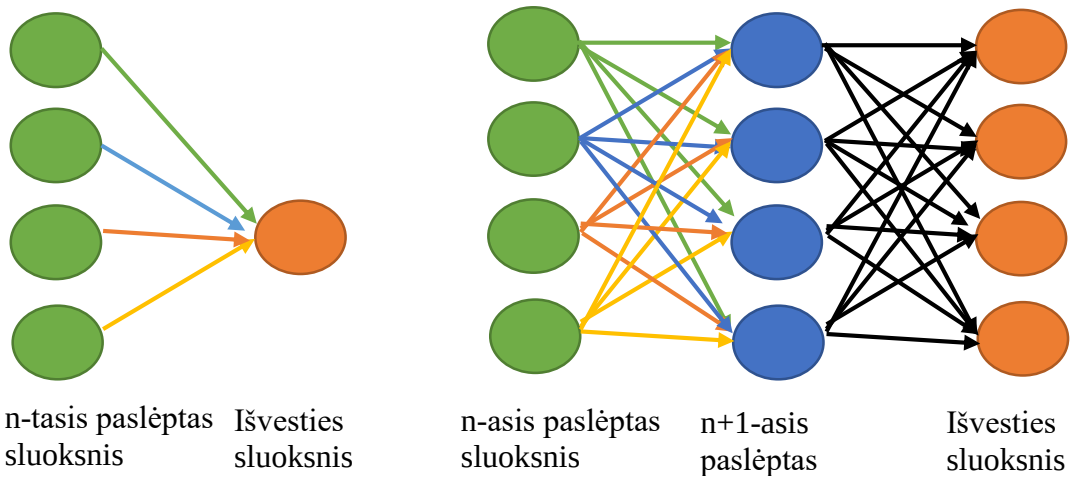
7.3 Struktūra

Teisingas DNT struktūras parinkimas visada buvo iššūkis. Buvo ne vienas bandymas, kaip dinamiškai sugeneruoti gerą tinklą [Soo13]. Ką tai padaro, tai tik duoda galutinį variantą, o mums įdomu, kas būna žinant galutinę topologiją – kaip tinklo mokymo procesas reaguoja į struktūrinius pakitimus.

7.3.1 Sluoksniai

Šios variacijos metu, kiekvienas tinklas savo mokymąsi pradės turėdamas tik vieną paslėptą sluoksnį. Proceso eigoje, jam bus pridėti du nauji sluoksniai – taip galutiniame variante turėsi tokios pat struktūros tinklą, kaip ir baziniu atveju.

Sluoksniai yra pridedami du kartus, praėjus $\frac{1}{3}$ ir $\frac{2}{3}$ bazinio mokymo. Įterpimo metu visos sinapsės, jungiančios paskutiniojo paslėptojo sluoksnio neuronus su išvesties sluoksnio neuronais, yra pašalinamos. Toliau tarp minėtųjų sluoksnių įterpiamas naujas ir yra sukuriamos naujos sinapsės į abu sluoksnius. Svarbu paminėti, kad naujojo sluoksnio sinapsės, jungiančios su prieš tai esančiu sluoksniu, yra inicializuotos tokiais pat svoriais kaip ir prieš tai buvusios, t.y. ėjusios į išvesties sluoksnį – taip bandoma išvengti informacijos praradimo. Senos sinapsės yra „klonuojamos“ ir priskiriamoms naujo sluoksnio neuronams. Naujas sluoksnis yra jungiamas su išvesties sluoksniu naujomis sinapsėmis – svoriai yra nustatomi funkcija $w(i) = (-1)^i$, kur i – sinapsės numeris.



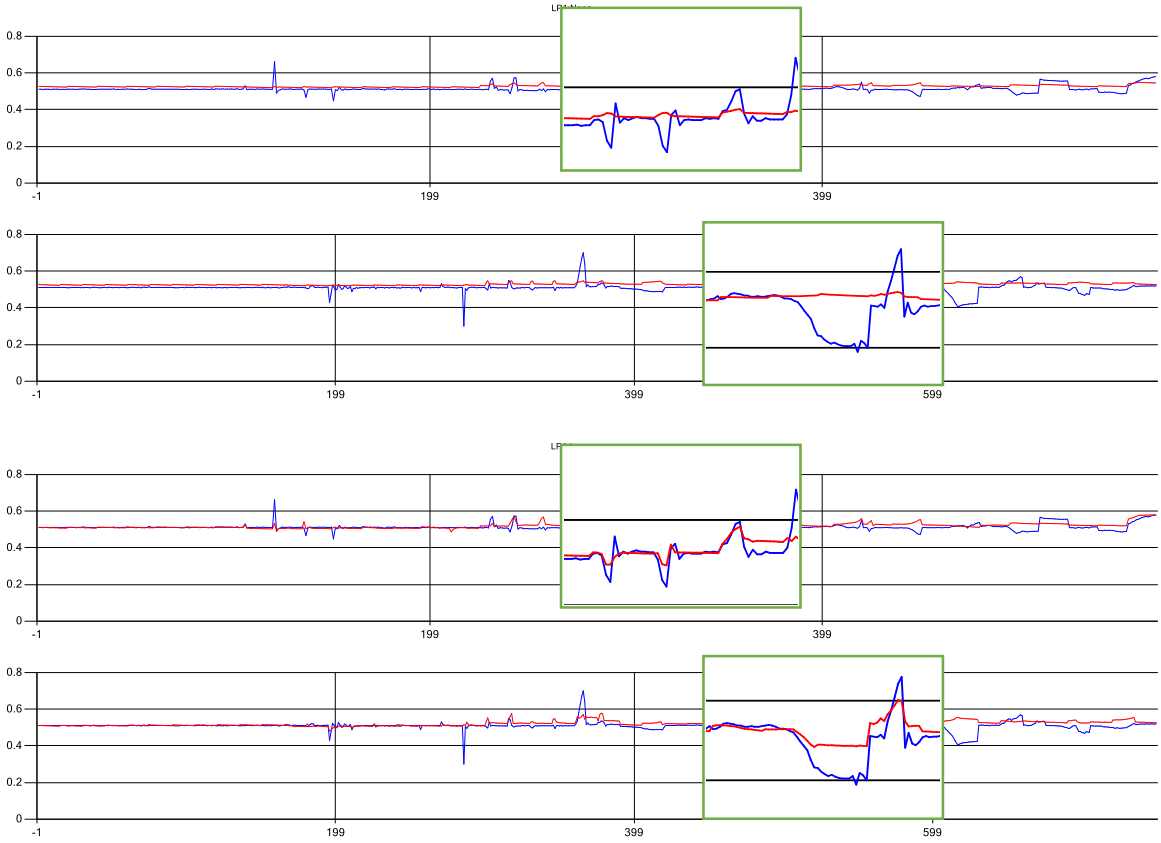
Pav. 18. Naujo sluoksnio pridėjimas

Duomenys	A.25	A.5	A1
DS1	0.0421 (16.3%)	0.0379 (5.1%)	0.0342 (-4.2%)
	0.0399 (2.6%)	0.0387 (-0.3%)	0.036 (-6.6%)
DS2	0.0206 (2.1%)	0.0196 (-2.4%)	0.0191 (-4%)
	0.0209 (0.7%)	0.0199 (-3.2%)	0.0195 (-4.5%)
DS3	0.0451 (1%)	0.0426 (-4.6%)	0.0434 (-0.9%)
	0.0429 (-0.8%)	0.0413 (-5.5%)	0.0425 (-2.2%)
DS4	0.0564 (5.2%)	0.0558 (4.6%)	0.0552 (3.9%)
	0.0591 (8.2%)	0.058 (6.3%)	0.057 (4.5%)
DS5	0.0129 (-31.8%)	0.0128 (9.8%)	0.0118 (2.9%)
	0.0187 (-24.9%)	0.0173 (-4.9%)	0.0163 (-4.5%)
DS6	0.0129 (-8.2%)	0.0125 (-7.5%)	0.0112 (-11.9%)
	0.0097 (-18.1%)	0.0098 (-13.8%)	0.0102 (-5.2%)
DS7	0.0115 (-20.6%)	0.0109 (-22.8%)	0.0105 (-19.5%)
	0.0175 (8.7%)	0.0178 (12.2%)	0.0173 (13.9%)
DS8	0.0091 (14.7%)	0.0097 (27.5%)	0.0092 (27.5%)
	0.0057 (6%)	0.0084 (52.9%)	0.0085 (48%)
DS9	0.0051 (2.6%)	0.005 (-0.4%)	0.0068 (36.6%)
	0.0068 (-0.1%)	0.007 (2.1%)	0.0089 (29.6%)
DS10	0.0848 (1%)	0.0841 (1.2%)	0.0843 (2.1%)
	0.0845 (0.3%)	0.0843 (1%)	0.085 (2.5%)

Lent. 16. Sluoksnių variacijos MAE rezultatai

Tinklui prijungus naują sluoksnį, gauname situaciją, kad prieš tai paskutinio tinklo rezultatas, kurio generuojama vertė yra artima išvesties rezultatui, yra siunčiamas į naująjį, kur vertės turėtų būti nežymiai pakoreguojamos. Taip pat, kadangi pradžioje tinklas turi mažesnį kiekį neuronų, jis turėtų greičiau apsimokyti ir atėjus laikui prijungti sluoksnį, gali būti tokia situacija, kad tinklas jau generuoja ganėtinai tikslius rezultatus. Kartu išlieka rizika, kad sluoksnio pridėjimas kaip tik sugadins rezultatą.

Iš rezultatų (Lent. 7) matome, kad A.25 tinklai rodo gerus rezultatus, tačiau toliau pradeda prastėti. Kas yra gerai, tai kad grafikuose matosi ženklūs pagerėjimai. Pav.10 aiškiai pastebima, kad tinklas yra stipriau apmokintas ir geriau atspindi duomenis. Taip pat DS8 (A1) klaida yra stipriai padidėjusi, tačiau iš grafiko matosi, kad jis yra daug geriau apmokytas – baziniu atveju klaida yra mažesnė dėl to, kad rezultatas yra tiesė, ir tai galioja dar keliems tinklams.



Pav. 19. DS5 A.25d tinklo bazinis (viršuje) ir sluoksnių variacijos rezultatas(apačioje)

Atsižvelgiant į klaidų lentelę ir grafikus, kur aiškiai matome tendenciją gerinti galutinį rezultatą – žymių pabloginimų nematyti, darome išvadą, kad ši variaciją yra gera ir galima ją taikyti beveik visiems duomenims. Taip pat pats apmokymas trunka lėčiau, kas taupo mūsų resursus.

7.3.2 Neuronai

Čia atliksime įvairias neuronų kiekio variacijas. Išskirsime tris atvejus – kaip juos varijuosime:

- paskutiniojo sluoksnio augimas; esminis skirtumas bus, kad paskutinis sluoksnis pradžioje turės mažiau neuronų ir apmokymo eigoje, jo skaičius augs iki kol susivienodins su kitais.
- Lėtas sluoksnių augimas; visi sluoksniai pradės su mažiau neuronų. Apmokymo proceso eigoje jie augs pridedant naujus neuronus. Pridėjimas vyks tik viename sluoksnyje, kol jis nepasieks nustatytos ribos. Toliau neuronai bus pridedami po jo einančiam sluoksniui.
- Greitas sluoksnių augimas; visi sluoksniai pradės su mažiau neuronu. Čia pridėjimas vyks ne viename, o visuose sluoksniuose. Taip jis greičiau pasieks bazinę būseną ir darys didesnę įtaką rezultatui.

Taip pat kaip optimizavimo metodas gali būti naudojamas neuronų atkirpimas – mokymas pradedamas su milžinišku tinklu ir proceso metu jo neuronai yra šalinami [ABB15].

7.3.2.1 Paskutinio sluoksnio augimas

Šio tyrimo metu, modifikuojamas sluoksnis, esantis arčiausiai išvesties sluoksnio. Manoma, kad taip pridedami nauji neuronai darys rezultato korekcijas ir duos tikslesnius pastūmimus prognozėse. Mūsų tinklas pradės su 4 neuronais ir bus didinamas kiekvieną šeštadalį bazinio mokymo iki kol turės tokį pat neuronų skaičių, kaip ir kiti paslėpti sluoksniai.

Lyginant su sluoksnių variaciją, čia jau nebėra tos rizikos, kad bus ištrinta svarbi informacija, kadangi senų sinapsių naikinti nebereikia – užtenka pridėti naujas.

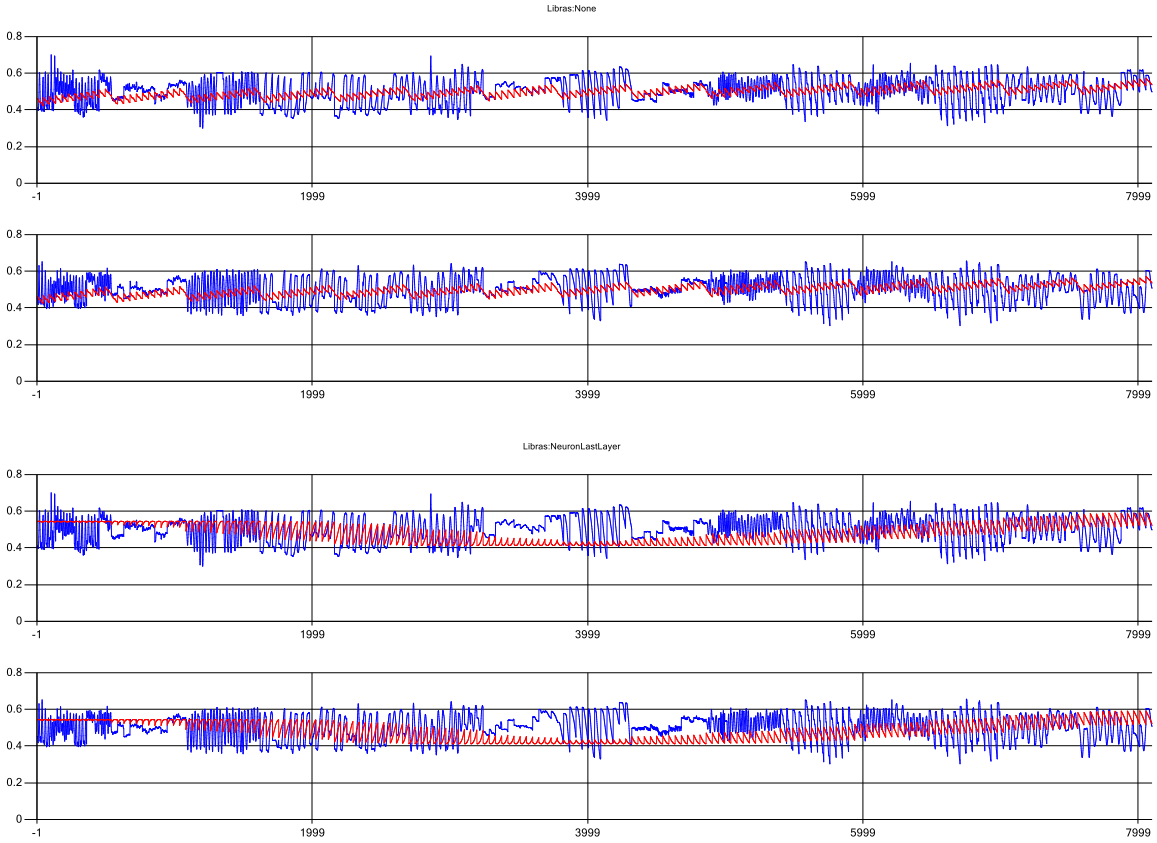
Duomenys	A.25	A.5	A1
DS1	**0.035 (-3.3%)**	0.035 (-3.1%)	0.0349 (-2.3%)
	0.037 (-4.8%)	**0.0369 (-4.9%)**	**0.0367 (-4.6%)**
DS2	0.0201 (-0.6%)	0.0199 (-0.5%)	0.0198 (-0.4%)
	0.0206 (-0.5%)	0.0205 (-0.4%)	0.0203 (-0.3%)
DS3	0.0465 (4.1%)	0.0469 (4.8%)	0.0448 (2.2%)
	0.045 (4.1%)	0.0458 (4.7%)	0.0441 (1.4%)
DS4	0.0662 (23.3%)	**0.0655 (22.9%)**	**0.0669 (26%)**
	0.0696 (27.4%)	**0.0687 (26.1%)**	**0.07 (28.4%)**
DS5	**0.0269 (41.6%)**	0.014 (20.4%)	0.0136 (18.5%)
	0.0348 (39.4%)	0.0214 (17.8%)	0.0185 (8.3%)
DS6	**0.0119 (-14.9%)**	**0.0112 (-17.1%)**	**0.0114 (-10.1%)**
	0.0118 (0%)	0.0112 (-1.2%)	0.0108 (0.5%)
DS7	0.0146 (0.9%)	0.0148 (4.8%)	0.0142 (8.7%)
	0.0157 (-2.1%)	0.0161 (1.6%)	0.016 (4.8%)
DS8	0.0085 (7.2%)	0.0078 (3.5%)	0.0073 (1.4%)
	0.0054 (1%)	0.0055 (0.2%)	0.0059 (3.8%)
DS9	0.0056 (11.4%)	0.0056 (11.8%)	0.0054 (9.2%)
	0.007 (3.6%)	0.007 (3.2%)	0.007 (2%)
DS10	0.0824 (-1.9%)	0.0824 (-0.9%)	0.0823 (-0.2%)
	0.0824 (-2.2%)	0.0825 (-1.2%)	0.0827 (-0.3%)

Lent. 17. Paskutinio sluoksnio neuronų variacijos MAE rezultatai

Pagal turimus rezultatus matome, kad ši variacija neduoda gerų rezultatų. Viena priežastis gali būti tai, kad esant mažiau neuronų paskutiniame sluoksnyje, jis nesugeba apimti viso duomenų spektro. Jam pridėdinėjant neuronus, sluoksnio apimtis plečiasi, tačiau jau nebespėjama apmokyti tinklo iki galo.

Vienas spendimas būtų paskutinį sluoksnį auginti pradedant su daugiau neuronų, kas tikriausiai ir yra didžiausia variacijos problema. Jeigu pradėtumėm su daugiau negu puse neuronų, tada pradiniame etape tinklas jau apimti daugiau taškų ir naujų neuronų įtraukimas turėtų nešti naudą.

Jeigu pažiūrėtume į tinklo DS4 grafikus (Pav. 11), įsitikintumėme, kad problemos yra dėl mažo kiekio neuronų apmokymo pradžioje. Nors ir baziniu atveju tinklui trūko laiko ar net neuronų, kad tinkamai apsimokyti, kadangi tai vienas didesnių duomenų paketų (8100 įrašų). Galima tikėtis, kad turint daug didesnį kiekį duomenų bazinė tinklo struktūra nespės teisingai pabaigti mokymo, o ši variacija pagal teoriją turėtų padėti išspręsti tai.



Pav. 20. DS4 A1 tinklo bazinis (viršuje) ir paskutinio sluoksnio neuronų variacijos rezultatas(apačioje)

7.3.2.2 Lėtas sluoksnių augimas

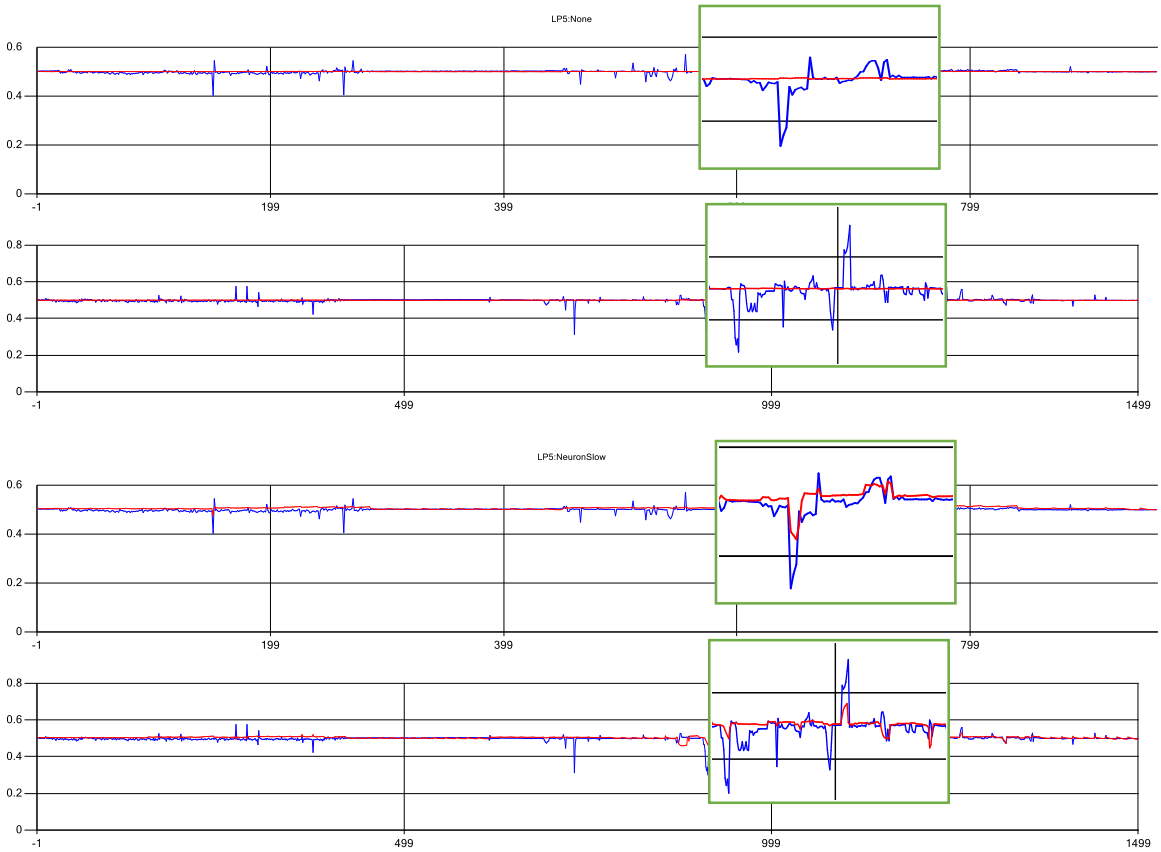
Šį variacija veiks panašiai kaip ir paskutiniojo – skirtumas, kad visi pradės su 4 neuronais ir augs. Sluoksniui pasiekus bazinio tinklo paslėptojo sluoksnio neuronų skaičių, pradės augti kitas, po jo esantis, sluoksnis. Pagal prieš tai buvusius rezultatus, matome, kad gali iškilti daug problemų turint didesnį kiekį duomenų, tačiau šį kartą bendras tinklo neuronų skaičius bus ganėtinai mažas – pats svorių generavimas vyksta lėčiausiai pirmame paslėptame sluoksnyje, kur prieš tai informacijos ateidavo mažai.

Duomenys	A.25	A.5	A1
DS1	0.0297 (-18%)	0.0316 (-12.3%)	0.0311 (-12.9%)
	0.036 (-7.5%)	0.0377 (-2.7%)	0.0369 (-4.3%)
DS2	0.0183 (-9.5%)	0.0186 (-7.2%)	0.019 (-4.4%)
	0.0188 (-9.1%)	0.0191 (-6.9%)	0.0197 (-3.6%)
DS3	**0.0325 (-27.1%)**	**0.0319 (-28.7%)**	**0.0231 (-47.3%)**
	0.0327 (-24.5%)	**0.0322 (-26.4%)**	**0.0257 (-40.8%)**
DS4	0.0555 (3.3%)	0.0517 (-3.1%)	0.0501 (-5.7%)
	0.0554 (1.4%)	0.0521 (-4.4%)	0.0505 (-7.4%)
DS5	0.0147 (-22.4%)	0.012 (3.2%)	0.0093 (-19%)
	0.0207 (-16.9%)	0.017 (-6.5%)	0.0145 (-15.2%)
DS6	0.0121 (-13.9%)	0.012 (-11.7%)	0.011 (-13.4%)
	0.0115 (-2.2%)	0.0112 (-1.5%)	0.0114 (5.7%)
DS7	0.0128 (-11.3%)	0.012 (-15.1%)	0.0108 (-17.2%)
	0.0141 (-12.1%)	0.0144 (-9%)	0.0157 (3%)
DS8	0.0083 (4.7%)	0.0078 (2.6%)	**0.0074 (2.6%)**
	0.007 (30.6%)	0.0065 (18.5%)	0.0062 (7.9%)
DS9	**0.0081 (60.7%)**	**0.0061 (21.8%)**	0.0043 (-13.3%)
	0.0096 (41.1%)	**0.0082 (19.9%)**	0.0074 (7%)
DS10	0.0845 (0.6%)	0.0817 (-1.7%)	0.0679 (-17.7%)
	0.0902 (7.1%)	0.087 (4.2%)	**0.0957 (15.4%)**

Lent. 18. Lėto neuronų augimo variacijos MAE rezultatai

Rezultatai geresni – matome mažesnes klaidas, negu auginant tik paskutinį sluoksnį. Pagal lentelę tinklas DS3 stabiliausiai rodo didžiausią klaidos kritimą, o DS9 – prasčiausius per pirmus 2 tyrimus. Jeigu pažiūrėtumėme į pastarojo grafiką (Pav. 12), pamatytumėme, kad vis dėl to variacija nesuprastino rezultato – tinklas daug geriau apmokytas. Tokia pati situacija ir DS8 tinkle.

Pagal gautus rezultatus galime daryti išvadą, jog ši variacija yra gera ir ją galima taikyti, apmokant neuroninius tinklus. Ji, kaip ir sluoksnių variacija, trumpina apmokymo procesą.



Pav. 21. DS9 A.25 tinklo bazinis (viršuje) ir lėto neuronų augimo variacijos rezultatas(apačioje)

7.3.2.3 Greitas sluoksnių augimas

Ši variacija, kitaip negu su lėtu augimu, visiems sluoksniams prijungs naujus neuronus. Reiškiasi, kad nebus ilgo užsisėdėjimo su mažu nenorų skaičiumi. Vienas galimas minusas tai, kad kai neuronas pridedamas į visus sluoksnius, atsiranda didesnė dalis neapmokyto tinklo ir yra tikimybė iškraipyti rezultatus.

Pagal lentelės duomenis (Lent. 10), kad rezultatai rodo klaidos sumažėjimus, nedidelis kiekis tinklų rodė didesnes klaidas. Jeigu pažiūrėtumėme ir į grafikus, matytųsi, kad grafikai geriau sugeneruoti. Taip pat klaidos didėjimą parodęs tinklas DS9, turi tikslesnį grafiką, tik jis yra silpniau apmokytas negu auginant lėtai.

Duomenys	A.25	A.5	A1
DS1	0.0334 (-7.9%)	0.0305 (-15.5%)	0.031 (-13.3%)
	0.0411 (5.7%)	0.0393 (1.3%)	0.041 (6.4%)
DS2	0.0196 (-2.8%)	0.0194 (-3.3%)	0.0164 (-17.2%)
	0.02 (-3.4%)	0.0197 (-4.4%)	0.0169 (-17.1%)
DS3	0.0326 (-27%)	**0.0328 (-26.7%)**	**0.031 (-13.3%)**
	0.0331 (-23.4%)	**0.0331 (-24.2%)**	**0.041 (6.4%)**
DS4	0.048 (-10.5%)	0.0453 (-15.1%)	0.0473 (-10.9%)
	0.0499 (-8.7%)	0.0478 (-12.2%)	0.0502 (-7.9%)
DS5	**0.0099 (-48.1%)**	0.0096 (-17.5%)	0.0088 (-23.6%)
	0.0153 (-38.6%)	0.0145 (-20.4%)	0.0134 (-21.4%)
DS6	0.0132 (-5.7%)	0.011 (-18.7%)	0.0103 (-19.1%)
	0.0113 (-4.1%)	0.0117 (2.8%)	0.0128 (18.5%)
DS7	0.0148 (2.1%)	**0.0175 (24.2%)**	0.0116 (-11.2%)
	0.0161 (0.1%)	**0.021 (32.2%)**	**0.0194 (27.4%)**
DS8	0.0086 (8.6%)	0.0085 (12.6%)	**0.0068 (-6%)**
	0.0058 (8.9%)	0.0059 (7.4%)	0.0057 (-0.4%)
DS9	**0.0056 (12.2%)**	0.0052 (3.3%)	0.0044 (-11.6%)
	0.0072 (5.1%)	0.0077 (12.6%)	0.0074 (8%)
DS10	0.0784 (-6.7%)	0.0696 (-16.2%)	0.0623 (-24.5%)
	0.0872 (3.5%)	0.0806 (-3.4%)	0.0751 (-9.4%)

Lent. 19. Greito sluoksnių augimo variacijos MAE rezultatai

Remiantis turimais duomenimis, galima sakyti, kad ši variacija taip pat yra gera ir neša teigiamą naudą neuroninių tinklų apmokymo procese – tinklas greičiau apmokomas, mažėja apmokymo laikas, mažesnės klaidos.

8 Išvados

Apžvelgę visų bandymų rezultatus, galime daryti tokias išvadas:

- Didėjantis mokymosi tempas ir duomenų vidinių svorių variavimas nėra efektyvus neuroninio tinklo apmokymo proceso optimizatorius – nėra ryškių pagerėjimų, per didelė priklausomybė nuo duomenų.
- Neuronų tinklų struktūros variacijos ir mokymosi tempo mažinimas yra efektyvios priemonės pagerinti DNT apmokymo procesą
- Vien klaidos nepakanka norint teisingai įvertinti metodo efektyvumą.
- Naudojant struktūrines variacijas galima sutrumpinti patį tinklų apmokymo procesą.
- Programa yra tinkama naudoti įvairiems duomenų rinkiniams norint ištirti variacijos įtaką

	Suminė MAE	***Pokytis (%)***	***WinCount***	***WinCount (%)***
Bazinis	0.9009	-	-	-
Sluoksniai	0.9091	0.7	14	46.7
Neuronai (Greitas)	0.8368	-11.3	16	53.3
Neuronai (Pask. sluoksnis)	0.9571	5.9	12	40
Neuronai (Lėtas)	0.865	-7.5	18	60
Didėjantis tempas	0.9179	1.7	8	26.7
Mažėjantis tempas	0.8842	-1.8	17	56.7
Senėjantys duomenys	0.8712	-3.4	20	66.7
Naujėjantys duomenys	0.9211	2.3	7	23.3

Lent. 20. Bendri testavimo duomenų rezultatai. Suminė MAE – visų tinklų MAE suma, Pokytis(%) – kiek % suminė MAE skiriasi nuo bazinio, WinCount – kiek tinklų rodė klaidos sumažėjimą (iš 30), WinCount(%) – kiek % tinklų rodė klaidos sumažėjimą

Literatūra

[ABB15] A. Bondarenko, A. Borisov, L. Aleksejeva „Neurons vs Weights Pruning in Artificial Neural Networks“ Riga Technical University, Faculty of Computer Science and Information Technology, Decision Support Systems Group, 2015

[ABC11] V. Chaudhary, A.K. Ahlawat, R.S. Bhatia „Growing neural networks using Soft Competitive Learning“, International Journal of Computer Applications (0975-8887) Volume 21 – No.3, 2011

[AEY11] E. Egrioglua, C. Aladagb, U. Yolcu „Fuzzy time series forecasting with a novel hybrid approach combining fuzzy c-means and neural networks“, 2nd International Fuzzy Systems Symposium 17-18, Ankara, Turkey, 2011.

[AGM14] H. Azad, S. Mekhilef, V. Ganapathy „Long-Term Wind Speed Forecasting and General Pattern Recognition Using Neural Networks“, IEEE Transactions on sustainable energy, vol. 5, no. 2, 2014.

[AM00] L.Anastasakis, N. Mort “Neural Network-Based Prediction of the USD/GBP Exchange Rate: The Utilisation of Data Compression Techniques for Input Dimension Reduction”, Technical Report, University of Sheffield, 2000.

[AV09] G. Atsalakis, K.Valavanis “ Forecasting stock market short-term trends using a neuro-fuzzy based methodology ”, Expert systems with applications, 2009.

[BC13] D. Barrow, S. Crone „Crogging (Cross-Validation Aggregation) for Forecasting – a novel algorithm of Neural Network Ensembles on Time Series Subsamples“, International Joint Conference on Neural Networks, 2013.

[CMM00] A. Carlevarino, R. Martinotti, G. Metta and G. Sandini „An Incremental Growing Neural Network and its Application to Robot Control“ Lira Lab – DIST – University of Genova Via Opera Pia, 13 – 16145 Genova, Italy

[DKT08] G. Dzemyda, O. Kurasova, J. Tilinskas. “Daugiamačių duomenų vizualizavimo metodai”. Vilnius, 2008

[DFN13] M. Dehghani, B. Saghafian, F. Saleh, A. Farokhniac,R. Noorie“ Uncertainty analysis of streamflow drought forecast using artificial neural networks and Monte-Carlo simulation”, International Journal of Climatology, 2013.

[DGK12] G. Sermpinis, K. Theofilatos, A. Karathanasopoulos, E. Georgopoulos, C. Dunis „Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and particle swarm optimization“, European Journal of Operational Research, 2012.

[FGL13] C. Guan, P. Luh, L. Michel, Y. Wang, P. Friedland „Very Short-Term Load Forecasting: Wavelet Neural Networks With Data Pre-Filtering“, IEEE Transactions on power systems, vol. 28, no. 1, Feb 2013.

[FJ11] A. Fallahi, S. Jafari „An Expert System for Detection of Breast Cancer Using Data Preprocessing and Bayesian Network“, International Journal of Advanced Science and Technology Vol. 34, September, 2011

[GMN10] J. Mantri, P. Gahan, B. Nayak “Artificial neural networks – an application to stock market volatility ”, International Journal of Engineering Science and Technology Vol. 2(5), 1451-1460, 2010.

[HHY11] T. Hsieh, H. Hsiao, W.Yeh „Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm“, Department of Industrial Engineering and Engineering Management, National Tsing Hua University, P.O. Box 24-60, Hsinchu 30013, Taiwan, ROC Applied Soft Computing, 2510-2525, 2011.

[HNW05] W. Huang, Y. Nakamori, S. Wang „Forecasting stock market movement direction with support vector machine“, Computers & Operations Research vol. 32, no. 10, 2513–2522, 2005.

[JKL16] J.Johnson, A.Karpathy, F.Li „Convolutional Neural Networks for Visual Recognition“, Stanford, CS213N, 2016, http://cs231n.github.io/neural-networks-3/

[KO15] M. Kampouridis, F. Otero “ Heuristic Procedures for Improving the Predictability of a Genetic ProgrammingFinancialForecastingAlgorithm”, Soft Computing, ISSN 1432-7643, 2015.

[LLL03] H. Lam, S. Ling, F. Leung, P. Tam „Tuning of the Structure and Parameters of Neural Networks using an Improved Genetic Algorithm“, IEEE Transactions on neural networks, vol. 14, no. 1, 2003.

[Mak06] R. Makūnaitė. Neuroniniai tinklai. Kompiuterija – PC World, 2006

[MK07] Y. Kwon, B. Moon „A Hybrid Neurogenetic Approach for Stock Forecasting“, IEEE Transactions on neural networks, vol. 18, no. 3, May 2007.

[PS96] B.Kröse, P.Smagt. An introduction to Neural Networks. University of Amsterdam, 1996, p.15-19.

[Roy15] A. Roy „A novel multivariate fuzzy time series based forecasting algorithm incorporating the effect of clustering on prediction“, Intelligent Security Systems Research Lab, Department of Computer Science, University of Memphis, Memphis, TN 38111, USA, 2015.

[Soo13] A. Sood „Artificial Neural Networks - Growth & Learn: A Survey" International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-3, July 2013

[Sta13] Š. Stankis. Neuroniniai tinklai. Referatas, Programų inžinerijos katedra, Informatikos fakultetas, KTU, Kaunas, 2013

[Tic13] J. Ticknor “A Bayesian regularized artificial neural network for stock market”, Duke University, Pratt School of Engineering, Durham, NC, USA, 2013.

Priedas Nr. 1

Programa „DynamicNN“ skirta tinklų apmokymui. Paleidimui naudoti DynamicNN.exe bylą.

Vieta: CD\DynamicNN

Priedas Nr. 2

Programos duomenys skirti tinklų apmokymui. Įkelti visus naudojantis programa.

Vieta: CD\Data

Priedas Nr. 3

Programos sugeneruoti grafikai visiems tinklams.

Vieta: CD\Graph

Priedas Nr. 4

Programos rezultatai visiems tinklams Excel byloje.

Vieta: CD\Results

Priedas Nr. 5

Duomenų pavadinimų žodynas skirtas susieti darbe vartojamas duomenų pavadinimų koduotes su pavadinimais prieduose.

Vieta: CD\README.txt