VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Gilaus mokymo algoritmų taikymas efektyviam sandorių vykdymui

Deep Learning Algorithms For Efficient Trade Execution

Magistro baigiamasis darbas

Atliko:	Saulius Tautvaišas	(parašas)
Darbo vadovas:	dr. Aistis Raudys	(parašas)
Darbo recenzentas:	dr. Valdas Dičiūnas	(parašas)

Vilnius 2017

Santrauka

Pastaruoju metu labai išpopuliarėję gilaus mokymo algoritmai leido išspręsti labai sudėtingas problemas vaizdo, balso ir kalbos atpažinime. Šiame darbe buvo siekiama ištirti ir nustatyti geriausią gilaus neuroninio tinklo architektūrą tinkančią finansinių duomenų prognozavimui. Iš pradžių buvo atlikta literatūros analizė, kurioje pateikiama gilaus neuroninių tinklų pritaikymo apžvalga bei trumpai pristatomi modeliai iki šiol naudoti efektyvių sandorių vykdymo problematikai spręsti. Vėliau buvo atliekami tyrimai pasirinkus skirtingų architektūrų neuroninius tinklus su skirtingomis aktyvacijos funkcijomis bei reguliarizacijos metodais. Ištyrus neuroninių tinklų architektūras, standartiniai neuroninių tinklų modeliai buvo palyginti su maišyto tankio neuroniniais tinklais. Galiausiai buvo siekiama įvertinti neuroninių tinklų prognozavimo patikimumą bei pritaikyti juos efektyvių sandorių vykdymui.

Summary

Deep learning algorithms have dramatically improved the state-of-the-art in speech recognition, visual object recognition, natural language processing and many other domains. This Master's thesis applies deep neural networks to financial domain problem and tries to find the best neural network architecture for financial data forecasting. In the introduction of this thesis we gave an overview about deep neural networks and their applications. We briefly covered few algorithms used for solving efficient order execution problem. In the following sections, we presented results for different neural network architectures trained with different activation functions and regularization methods. Also, we have compared standard neural network and mixture density network forecasting results. Finally, we created models to get neural network uncertainty about its forecasts and applied these models to efficient order execution problem.

Turinys

1 Įvadas

Pastaraisiais metais algoritminės prekybos apimtys finansiniais instrumentais sparčiai augo. 2010 metais algoritminės prekybos apyvarta akcijomis JAV ir Europoje atitinkamai sudarė 56 % ir 38 % nuo visos prekybos akcijomis apyvartos [TV12], [Ald13]. Algoritminė prekyba turi didelę įtaką ne tik akcijų rinkose, tačiau ir kitose rinkose, ypač ateities sandorių. Pagal 2011 metų paskelbtus duomenis algoritminės prekybos apyvarta naftos ateities sandorių kontraktais JAV sudarė 95 % nuo visos prekybos apyvartos [Mey11].

Algoritminė prekyba gali būti apibrėžiama kaip investavimo strategija, kurioje sprendimai pirkti ar parduoti finansinius instrumentus yra paremti kompiuteriniais algoritmais, labai trumpam investavimo periodui, dažniausiai neilgiau kaip keletas sekundžių ar milisekundžių [TV12], [Ald13]. Algoritminę prekybą dažniausiai naudoja alternatyvaus investavimo fondai (ang. hedge funds) spekuliatyviniams tikslams bei instituciniai investuotojai norėdami efektyviai įsigyti ar parduoti didelį tam tikro finansinio instrumento kiekį su kuo mažesniais kaštas ir kuo mažiau įtakojant finansinio instrumento kainą.

Algoritminėje prekyboje sandorių vykdymo kaštai gali sudaryti reikšmingą dalį nuo visos gautos investicijų grąžos. Atlikti moksliniai tyrimai parodė, jog sandorių vykdymo kaštai instituciniams investuotojams vidutiniškai sudaro apie 1 % nuo sandorio vertės ir gali pasiekti net iki 3-4 % nelikvidžiose rinkose [Lor08]. Alternatyvaus investavimo fonduose per dieną sandorių sudaroma be galo daug, todėl sandorių kaštai gali sudaryti nuo 20 iki 50 % gautos investicijų grąžos [Nar09]. Todėl kiekvienam rinkos dalyviui yra labai svarbu turėti efektyvią sandorių vykdymo strategiją, kuri leistų minimizuoti kaštus.

Sandorio kaštai gali būti skirstomi į fiksuotus ir kintamus kaštus [Nar09]. Fiksuoti kaštai yra tokie kaštai, kurie yra žinomi iš anksto. Jie yra sudaryti iš komisinių brokeriams, mokesčių vertybinių popierių biržoms bei pelno mokesčio. Kintami kaštai nėra žinomi iš anksto ir jie gali keistis priklausomai nuo rinkos sąlygų. Kintami kaštai yra sudaryti iš alternatyvių kaštų (angl. *opportunity cost*), paklausos-pasiūlos skirtumo (angl. *bid-ask spread*), rinkos įtakos (angl. *market impact*) bei praslydimo (angl. *slippage*) kaštų. Mokslinėje literatūroje sandorių vykdymo kaštų problematika sprendžiama optimizuojant kintamus kaštus.

Efektyvios sandorių vykdymo strategijos yra paremtos naudojant du pagrindinius sandorių tipus: rinkos (angl. *market order*) ir limituotų pavedimų (angl. *limit order*) sandorius [CK12],

[SW12]. Rinkos sandoris yra instrukcija vykdyti sandorį su tuo metu rinkoje esančia geriausia kaina, o limituoto pavedimo sandoris yra instrukcija įvykdyti sandorį su iš anksto nustatyta kaina [CMZ12], [NKP+05]. Pagrindinis skirtumas tarp šių sandorių tipų yra laikas, per kurį bus įvykdyti šie sandoriai. Rinkos sandorio atveju, sandoris bus įvykdytas iš karto, tačiau patiriant kaštus dėl paklausos-pasiūlos kainos skirtumo. Limituoto pavedimo atveju šių kaštų išvengiama, tačiau dėl įvesto kainos apribojimo padidėja tikimybė, jog sandoris gali būti įvykdytas tik dalinai arba išvis neįvykdytas [Yin12].

Akcijos ir kiti finansiniai instrumentai yra prekiaujami pavedimų sandoriais paremtoje sistemoje (angl. order-driven system), kur pirkimo sandoris atitinka pardavimo sandorį. Pavedimai gali būti atliekami tik su konkrečia diskrečia kaina, dažniausiai mažiausias kainos žingsnis (angl. tick size) yra 0,01 cento. Visi limituoti pavedimai yra registruojami į limituotų pavedimų knygą (angl. limit order book), kurioje išlieka iki tol kol jie yra įvykdomi arba atšaukiami. Limituotų sandorių knyga yra sudaryta iš visų limituotų pavedimų pirkti ir parduoti su skirtingomis kainomis ir agreguotais sandorių dydžiais. Geriausia pirkimo kaina (angl. best ask price) yra žemiausias limituoto pavedimo parduoti kainą pavedimų knygoje. Geriausia pardavimo kaina (angl. best bid price) yra aukščiausia limituoto pavedimo pirkti kainą pavedimų knygoje. Dažnai tarp geriausios pirkimo ir pardavimo kainos limituotų pavedimų knygoje gali atsirasti kainų skirtumas vadinamas paklausos-pasiūlos skirtumas (angl. *bid-ask spread*). Rinkos pavedimas pirkti yra įvykdomas geriausia limituoto pavedimo parduoti kainą, o rinkos pavedimas parduoti yra įvykdomas geriausia limituoto pavedimo pirkti kaina esančią limituotų pavedimų knygoje. Rinkos pavedimas tiesiogiai sumažina geriausios pirkimo ar pardavimo limituoto pavedimo kiekį.

Šiame darbe efektyvaus sandorių įvykdymui bus naudojama bitkoinų (angl. bitcoin) limituotų pavedimų knyga. Bitkoinų limituotų pavedimų knygos pavyzdys pavaizduotas 1.1 pav. Limituotų pavedimų knyga parodo kokį kiekį galima nusipirkti ar parduoti finansinį instrumentą tam tikra kaina. Limituotų pavedimų pirkti ir parduoti dydžiai pavedimų knygoje parodo šio finansinio instrumento likvidumą tame kainos lygyje.

Pardavimo kiekis	Pardavimo kaina	Pirkimo kaina	Pirkimo kiekis
		1363.48	0.0186
		1363.14	2.3402
		1363.11	0.0276
		1362.97	0.1
		1362	19.7412
		1361.5	13.3890
4.00	1361		
0.88063	1360		
0.33799	1359.66		
6.39805	1359.5		
0.970	1358.66		
0.080	1358.08		

1.1 pav. Limituotų pavedimų knygos pavyzdys.

Limituotų pavedimų knyga parodo paklausą ir pasiūlą prie skirtingų kainos lygių. Bėgant laikui, limituotų pavedimų knygos geriausia pirkimo ir pardavimo kaina gali kisti dėl atsiradusių naujų limituotų pavedimų geresne ar blogesne kaina, atšauktų esamų limituotų pavedimų ar dėl rinkos pavedimų. Geriausia limituoto pavedimo pardavimo kaina ir geriausias pirkimo kaina laiko momentu t yra kaina, kuria rinkos dalyviai gali iš karto nusipirkti ar parduoti finansinį instrumentą.

Šiame darbe remsimės [NKP+05] pasiūlyta sandorių vykdymo metodologija, kurioje daroma prielaida, jog investuotojas nori nusipirkti arba parduoti tam tikrą finansinio instrumento kiekį K per tam tikrą fiksuotą laiko periodą T su kuo mažesniais sandorio kaštais. Investuotojas turi tik tris galimus pasirinkimus: pirkti akcijų kiekį K laiko periodu t = 0 esančia rinkos kaina, pirkti tą patį akcijų kiekį K rinkos kaina, laiko periodu t=T, arba sudaryti limituotą pavedimo sandorį laiko periodu t=0 ir jei sandoris nebuvo pilnai įvykdytas, nusipirki likusį akcijų kiekį rinkos kaina, laiko periodu t=T. Visais šiais scenarijais investuotojas nusipirks (parduos) tą patį akcijų skaičių, tačiau su skirtingais sandorio kaštais. Pagal šią metodologiją, sandorio vykdymo kaštų problemai spręsti, galima panaudoti neuroninius tinklus ir prognozuoti kainos pokytį per T periodą.

1.1 Temos aktualumas bei naujumas

Standartiniai mašininio mokymo (angl. machine learning) algoritmai dažniausiai negali apdoroti duomenų jų originalia forma. Dėl šios priežasties, mokslininkai norėdami naudoti šiuos algoritmus turėdavo skirti didelę laiko dalį iš originalios duomenų imties rasti ir išgauti požymių vektorius, kuriuos būtų galima panaudoti algoritmų mokymui [BLH15]. Dažnai norit surasi ir parinkti gerus požymius reikėdavo labai gerai išmanyti sprendžiamos problematikos specifiką. Dėl šios priežasties nuo pat mašininių algoritmų atsiradimo pradžios buvo siekiama automatizuoti požymių atrinkimą, kad patys algoritmai sugebėtų rasti reprezentatyvius požymius.

Reprezentatyvus mokymas (angl. representation learning) yra metodų aibė, kurie leidžią algoritmui pateikus originalią duomenų imtį automatiškai rasti reprezentatyvius požymių vektorius. Gilaus mokymo metodai (angl. deep learning) priklauso šių metodų šeimai, kurie turi keletą reprezentatyvių sluoksnių sudarytų iš netiesinių modulių. Kiekvienas sluoksnis transformuoja prieš jį buvusio sluoksnio rastus reprezentatyvius požymius į vis aukštesnio ir abstraktesnio lygio požymius. Tai leidžia algoritmui išmokti labai kompleksines funkcijas naudojant originalią duomenų aibę. Pagrindinis šių algoritmų tikslas yra automatiškai rasti reprezentatyvius požymius be žmogaus įsikišimo, kuriais remiantis algoritmas gali išmokti atliktį norimą užduotį [BEN09], [BLH15].

Vienas paprasčiausių gilaus mokymo algoritmų yra tiesioginio skleidimo neuroninis tinklas (angl. feedforward neural network) sudarytas iš kelių paslėptų sluoksnių netiesinių modulių [GV07]. Iki 2006 metų buvo manoma, kad neuroninį tinklą turinti daugiau nei du paslėptus sluoksnius yra sunku apmokyti, nes jie neduodavo geresnių rezultatų negu neuroninis tinklas turintis mažiau paslėptų neuroninių sluoksnių. Tačiau nuo 2006 metų atradus naujus neuroninių tinklų mokymo metodus leido sėkmingai apmokyti neuroninius tinklus turinčius keturis ar daugiau paslėptus sluoksnius [HS06]. Gilaus mokymo algoritmai leido išspręsti problemas, kurios iki tol nebuvo įmanoma atlikti bei pagerino ankstesnius rezultatus daugelyje problemų [BEN09], [BCV13].

Mažai tyrimų yra atlikta efektyvių sandorių vykdymo problematikai spręsti panaudojant mašininio mokymosi algoritmus. [NFK06] panaudojo mokymosi su paskatinimu algoritmą (angl. reinforcement learning) optimalios sandorių vykdymo strategijos radimui akcijų rinkose. Tačiau pagrindinis šio darbo trūkumas yra toks, kad šis standartinis mokymosi su paskatinimu algoritmas

negali būti pritaikomas tiesiogiai aukštos dimensijos problemoms modeliuoti. Dėl šios priežasties, rankiniu būdu bandoma rasti reprezentatyvius požymius iš duomenų imties. [NFK06] savo darbe iš laiko eilučių duomenų atrinko pagrindinius požymius ir juos panaudojo optimalios strategijos radimui. [KZ15] panaudojo atraminių vektorių mašiną vidutinės akcijų kainos prognozavimui limituotų pavedimų knygoje. Šiame darbe tyrėjai, taip pat išskyrė pagrindinius požymius, kurie buvo panaudoti modelio mokymui.

Gilūs neuroniniai tinklai turi daug privalumų lyginant su atraminėmis vektorių mašinomis ar kitais masinio mokymosi algoritmais. Vienas iš pagrindinių privalumų yra tai, jog gilūs neuroniniai tinklai gali tiesiogiai modeliuoti aukštos dimensijos duomenis ir dėl šios priežasties nebereikia rankiniu būdu išskirti požymius iš mokymo duomenų.

1.2 Darbo tikslas

Ištirti neuroninio tinklo architektūrą geriausiai tinkančią limituotų pavedimų knygos kainų pokyčio prognozavimui.

1.3 Uždaviniai

Tikslui pasiekti bus sprendžiami šie uždaviniai:

- Ištirti gilaus neuroninio tinklo architektūrą geriausiai tinkančią kainų pokyčių prognozavimui;
- Palyginti gilių neuroninių tinklų architektūras su seklesnės architektūros neuroniniais tinklais;
- Palyginti neuroninio tinklo mokymo ir prognozavimo rezultatus naudojant skirtingas aktyvacijos funkcijas;
- Ištirti gilaus neuroninio tinklo reguliarizavimo metodus;
- Palyginti standartinį neuroninį tinklą su maišyto tankio neuroniniu tinklu;
- Įvertinti gilaus neuroninio tinklo mokymo prognozavimo rezultatų patikimumą;
- Neuroninių tinklų pritaikymas efektyvių sandorių vykdymui.

Uždaviniams išspręsti bus naudojami šie metodai: mokslinės literatūros analizė ir apibendrinimas, modelio sudarymas ir jo taikymas.

1.4 Laukiami rezultatai

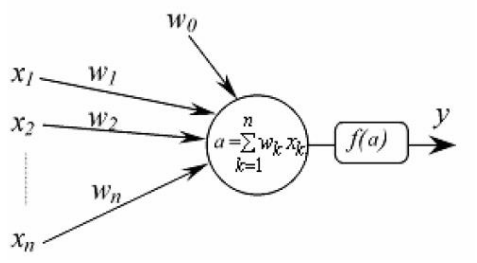
Toliau pateikiami uždavinių rezultatai:

- Sudaryti ir palyginti skirtingų architektūrų neuroniniai tinklai;
- Ištirtos aktyvacijos funkcijos geriausiai tinkančios gilaus neuroninio tinklo mokymui;
- Rastas geriausias gilaus neuroninio tinklo reguliarizavimo metodas;
- Palyginti standartiniai ir maišyto tankio neuroniniai tinklai;
- Įvertintas neuroninių tinklų prognozavimo patikimumas;
- Neuroninių tinklų modeliai pritaikyti efektyvių sandorių vykdymui.

2 Dirbtiniai neuroniniai tinklai

Dirbtiniai neuroniniai tinklai (DNT) (angl. artificial neural networks) – tai informacijos apdorojimo struktūros, netiksliai imituojančios, kai kuriuos gyvųjų organizmų smegenyse vykstančius informacijos apdorojimo procesus. DNT sudaromi iš daugelio tarpusavyje sujungtų labai paprastų skaičiavimo elementų. Šie elementai jungiami vieni su kitais įvairaus stiprumo jungtimis, yra apytikris biologinių neuronų modelis. Dirbtiniai neuroniniai tinklai pradėti nagrinėti XX a. šeštajame dešimtmetyje, tačiau iki devintojo dešimtmečio vidurio jie nebuvo plačiai naudojami. Tik išradus greitus ir galingus mokymo mechanizmus DNT galėjo spręsti realus uždavinius [GV07], [MED07], [BIS07], [BLH15].

Labiausiai paplitusi dirbtinio neurono schema pavaizduota 2.1 pav. Čia $x_1,..,x_n$ žymi neurono įėjimo signalus. Atitinkamai svoriai pažymėti $w_1,...,w_n$. Žymėjimas w_0 reiškia slenksčio vertę, f() – perdavimo funkcija arba dar kitaip vadinama aktyvacijos funkcija ir y – neurono išėjimą.



2.1 pav. Dirbtinis neuronas

Neurono išėjimo reikšmę y galime užrašyti ir formule:

$$y = f(\sum_{k=1}^{n} w_k x_k). \tag{2.1}$$

Dažniausiai naudojamos netiesinės aktyvacijos funkcijos [PAS14] yra sigmoidinė

$$f(a) = \frac{1}{1+e^{-a}} \tag{2.2}$$

ar hiperbolinis tangentas

$$f(a) = \tanh(a) = \frac{e^{a} - e^{-a}}{e^{a} + e^{-a}}. \tag{2.3}$$

Pastarasiais metais labai išpopuliarėjo lyginanti (angl. rectifier) aktyvacijos funkcija [NH10], [GBB11].

$$rect(x) = \max\{o, x\} = x \cdot II_{X>0}(x) = \begin{cases} x, iff\ \text{x} > 0 \\ 0, \text{kitu atveju} \end{cases}, \tag{2.4}$$

kur $x \in R$ ir II yra funkcijos indikatorius :

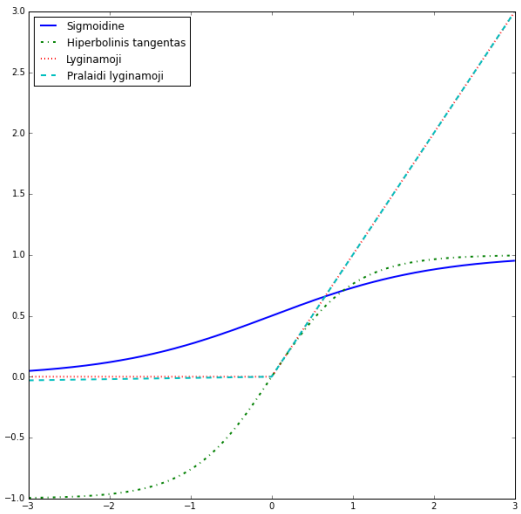
$$II_{x>b}(x) = \begin{cases} 1, iff\ \text{x} > \text{b} \\ 0, \text{kitu atveju} \end{cases}. \tag{2.5}$$

Atlikti tyrimai parodė, jog naudojant šią aktyvacijos funkciją neuroninio tinklo mokymo paklaida konverguoja daug greičiau nei naudojant sigmoidinę ar hiperbolinio tangento funkcijas [KSH12].

Kita populiari šiuo metu naudojama aktyvavijos funkcija yra pralaidi lyginamoji (angl. leaky rectifier) [MHN13].

$$leaky\_rect(x) = \max\{0, x\} = \begin{cases} x, iff\ \text{x} > 0 \\ 0{,}01 * \text{x}, \ \text{kitu atveju} \end{cases}, \tag{2.6}$$

Šios pagrindinės netiesinės aktyvacijos funkcijos buvo pavaizduotos 2.2 pav.



2.2 pav. Netiesinės aktyvacijos funkcijos naudojamos neuroninio tinklo paslėptuosiuose sluoksniuose

Neuroninio tinklo pradinių svorių iniciavimui dažniausiai yra naudojamas normalusis ar tolygusis skirstinys. [GB10] savo darbe pasiūlė šių skirstinių modifikacijas: Glorot normalųjį ir

Glorot tolygųjį skirstinį. Pradiniai svoriai naudojant Glorot normalųjį skirstinį inicijuojami panaudojant tokią formulę:

$$W \sim N(0, \sigma),\ kur\ \sigma = \sqrt{\frac{2}{n_j + n_{j+1}}} \tag{2.7}$$

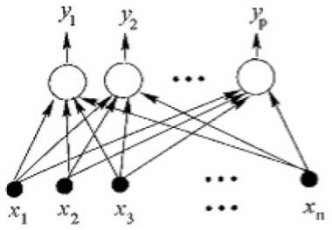
Šioje formulėje n_j ir n_{j+1} yra neuronų skaičius dabartiniame ir sekančiame neuroniniame sluoksnyje. Tuo tarpu, svoriai pagal Glorot tolygųjį skirstinį inicijuojami:

$$W \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}\right] \tag{2.8}$$

Buvo įrodyta, kad šie skirstiniai leidžia greičiau ir geriau apmokyti neuroninius tinklus [GB10], [KSH12] , [HZR+15] .

2.1 Neuroninių tinklų architektūra

Atskiri neuronai jungiami į neuroninį tinklą, kuriame neuronai yra sugrupuoti į atskirus sluoksnius. Paprasčiausios architektūros yra taip vadinami tiesioginio sklidimo neuroniniai tinklai, kuriuose galimos tik vienkryptės jungtys į priekį. Paprasčiausias tokio tipo neuroninis tinklas – perceptronas, susidedantis iš vieno sluoksnio p neuronų, sujungtų su n įėjimais (3.3 pav.) [ROS58]. Dažnai neuronai kurie sudaro neuroninį tinklą paprasčiausiai vadinami netiesiniais moduliais [BL07].

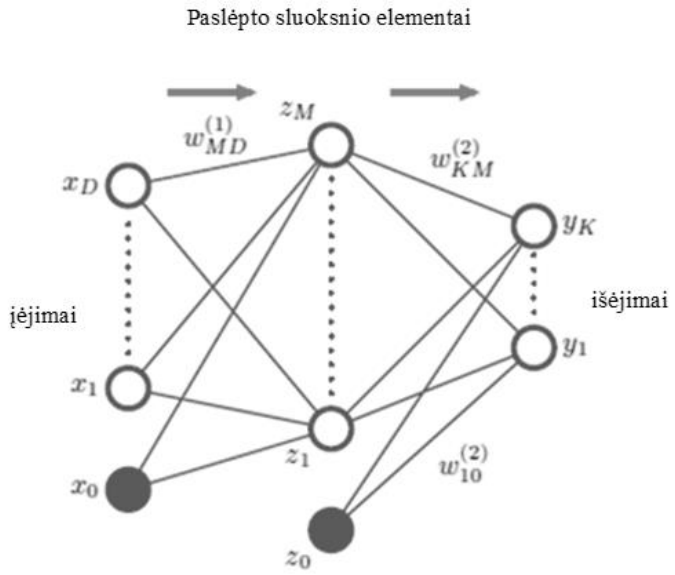


2.2 pav. Perceptronas

Vienasluoksniai dirbtiniai neuroniniai tinklai nėra tinkami labai sudėtingiems uždaviniams spręsti. Tam naudojami tinklai turintys keletą netiesinių modulių sluoksnių. Nustatyta, jog perceptronai turintys tarp įėjimo ir išėjimo sluoksnio viena ar daugiau netiesinių modulių sluoksnius yra tinkami spręsti klasifikavimo problemas, kur klasės nėra tiesiškai atskiriamos

[TP89]. Yra žinoma, kad neuroniniai tinklai turintys vieną ar daugiau netiesinių modulių sluoksnių pasižymi universalaus apibendrinimo savybe [Cyb89], [Hor91].

Dažniausia daugiasluoksnio tinklo struktūra – vienas po kito einantys neuronų sluoksniai, kurių kiekvienas neuronas sujungtas su visais kito sluoksnio neuronais ir neturintis jokių kitų jungčių. Tiesioginio sklidimo neuroninis tinklas turintis daugiau nei vieną neuronų sluoksnį vadinamas daugiasluoksniais tiesioginio sklidimo neuroniniais tinklais (angl. multilayer feedforward neural networks) [GV07].



2.3 pav. Daugiasluoksnis neuroninis tinklas

Neuroninio tinklo sudaryto iš dviejų neuronų sluoksnių pavaizduoto 2.3 pav., tarpinių sluoksnių neuronai yra vadinami paslėptaisiais neuronais. Parodytame tinkle yra D įėjimo, M paslėptųjų ir K išėjimo neuronų. Analitinė neuroninio tinklo realizuojama funkcija užrašoma taip: j-tojo paslėptojo neurono svorinė suma gaunama formuojant svorinę visų d įėjimų kintamųjų kombinaciją ir pridedant laisvąjį narį:

$$a_j = \sum_{i=1}^{d} w_{ji}^{(1)} x_i + w_{j0}^{(1)}; \tag{2.9}$$

Čia $w_{ji}^{(1)}$ reiškia pirmojo sluoksnio svorį iš i-tojo paslėptojo neurono į j-tąjį kito sluoksnio neuroną, $w_{j0}^{(1)}$ žymi j-tojo paslėptojo neurono laisvojo nario svorį. Paslėpto neurono išėjimo vertė gaunama transformuojant tiesinę sumą (formulė 2.9) aktyvacijos funkcija g(·). Neurono išėjimo vertė

$$z_j = g(a_j) \tag{2.10}$$

Kiekvienam neuroninio tinklo išėjimui skaičiuojamos tiesinės paslėptojo sluoksnio neuronų išėjimų kombinacijos:

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} + w_{k0}^{(2)}. \tag{2.11}$$

Tada neuroninio tinklo išėjimas:

$$y_k(x) = \tilde{g}(a_k) \tag{2.12}$$

Išėjimų ir paslėptųjų sluoksnių neuronų perdavimo funkcijos $g(\cdot)$ ir $\tilde{g}(\cdot)$ nebūtinai turi būti tos pačios. Išėjimo sluoksnio aktyvacijos funkcijos forma dažniausiai pasirenkama pagal tai kokia problema yra sprendžiama. Pavyzdžiui, jei sprendžiama binarinė klasifikacija, tuomet gali būti naudojama ir sigmoidinė aktyvacijos funkcija, kuri garantuoja, kad išėjimo reikšmės bus iš intervalo [0,1]. Jei reikia atlikti klasifikavimą į daugiau klasių, tuomet dažnai naudojama minkštojo maksimumo (angl. softmax) aktyvacijos funkcija. Ši funkcija remiasi visomis tinklo išėjimo reikšmėmis ir garantuoja, kad visų išėjimo reikšmių suma bus lygi 1.

$$\text{softmax}(x)_i = \frac{e^{x_i}}{\sum_{j=0}^{0} e^{x_j}} \tag{2.13}$$

Sprendžiant regresijos problemą naudojama vienetinė (angl. identity) funkcija, kur

$$id(x) = x \tag{2.14}$$

Sujungę formules 2.9-2.12 formule bei laisvuosius svorius į bendrą svorių vektorių gautume:

$$y_k(x) = \tilde{g}\left(\sum_{j=0}^{M} w_{kj}^{(2)} g\left\{\sum_{i=0}^{d} W_{ji}^{(1)} X_i\right\}\right) \tag{2.15}$$

Neuroninių tinklų mokymas dažnai grindžiamas klaidos minimizavimo metodais. Klaida, kurią minimizuojant mokomas tinklas, gali būti parenkama pagal sprendžiamą uždavinį ir neuroninio tinklo tipą. Viena iš klaidos funkcijų, plačiai naudojama neuroniniams tinklams apmokyti yra suminė kvadratinė klaida. Ji gaunama sumuojant klaidas visuose tinklo išėjimuose visai duomenų imčiai:

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{c}\left\{y_k(x^n, w) - t_k^n\right\}^2 \tag{2.16}$$

čia $y_k(x^n, w)$ yra k-tasis tinklo išėjimas, apskaičiuojamas kaip n-tojo įėjimo vektoriaus x^n ir svorių vektoriaus w funkcija, N yra įėjimo duomenų skaičius, o c žymi tinklo išėjimų skaičių.

Sprendžiant klasifikavimo problemas, kur duomenys yra skirstomi į daugiau nei dvi klases, dažniausiai naudojama kryžminės entropijos (angl. cross-entropy) klaidos funkcija. Jeigu pasirenkamas 1-iš-c klasių kodavimo būdas neuroninio tinklo išėjime, kur užduoties vektorius nusakomas vienetu norimame išėjime k elementui ir nuliais likusiuose, tuomet klaidos funkcija apskaičiuojama:

$$E(w) = \sum_{n=1}^{N}\sum_{k=1}^{c} t_{nk} \ln y_{nk} \tag{2.17}$$

Neuroninio tinklo klaida gali būti minimizuojama įvairiais metodais. Jei neuroninio tinklo neuronų aktyvacijos funkcijos diferencijuojamos, tai išėjimo sluoksnio neuronų aktyvacijos funkcijos diferencijuojamos įėjimo kintamųjų ir svorių atžvilgiu. Jei klaidos funkciją parinksime diferencijuojamą neuroninio tinklo išėjimo atžvilgiu, tai ši funkcija bus diferencijuojama ir svorių atžvilgiu. Galima apskaičiuoti klaidos išvestines svorių atžvilgiu ir naudoti jas svoriams, kuriems esant klaidos funkcija minimali, rasti gradientinio priartėjimo ar kitais optimizavimo metodais. Klaidos išvestinės pagal svorius apskaičiavimo algoritmas yra žinomas kaip klaidos atgalinio skleidimo (angl. error backpropogation) algoritmas, kadangi jis atitinka klaidos skleidimą neuroniniu tinklu nuo išėjimo link įėjimo neuronų [GV07], [BIS07].

Daugumoje mokymo algoritmų naudojama pasikartojanti svorių modifikavimo procedūra. Mokymo metu svoriai žingsnis po žingsnio keičiami taip, kad mažėtų klaida. Naudojant gradientinius mokymo metodus, galima išskirti du kiekvieno tokio žingsnio etapus. Pirmajame apskaičiuojamos klaidos funkcijos išvestinės svorių atžvilgiu naudojant klaidos atgalinio

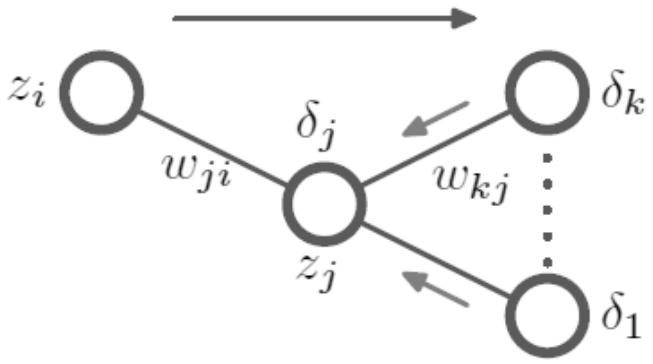
skleidimo algoritmu. Kitame etape išvestinės naudojamos reikiamam svorių pokyčių dydžiui nustatyti.

2.2 Klaidos atgalinio skleidimo algoritmas

Klaidos atgalinio skleidimo algoritmas daugiasluoksniam neuroniniams tinklams buvo pasiūlytas [RHW88]. Tiesioginio sklidimo neuroninio tinklo atveju kiekvienas elementas skaičiuoja svorinę įėjimo verčių sumą:

$$a_j = \sum_i w_{ji} z_i \qquad (2.18)$$

Kur z_i yra neuroninio tinklo įėjimas arba žemesniojo sluoksnio i-tojo elemento išėjimas, einantis į j-tąjį elementą, o w_{ji} yra jungties svoris (žr. 2.4 pav.).



2.4 pav. Mazgų ir svorių žymėjimas.

Sumuojami visų neuronų, sujungtų su j-tuoju elementu, išėjimai. Suma (2.18) transformuojama naudojant netiesinę aktyvavimo funkciją g() ir neurono išėjimas yra šios funkcijos vertė:

$$z_j = g(a_j) \qquad (2.19)$$

Tinkami neuroninio tinklo svoriai parenkami minimizuojant tam tikrą klaidos funkciją. Klaidos funkcija gali būti apskaičiuojama kaip suma klaidų, gaunamų atskirai kiekvienam sluoksnio duomenų vektoriui:

$$E = \sum_n E^n \tag{2.20}$$

čia n žymi duomenų skaičius imtyje.

Klaidos E^n išraiška turi būti diferencijuojama pagal neuroninio tinklo išėjimo kintamuosius.

Tarkim, ieškoma klaidos E^n išvestinė svorio w_{ji} atžvilgiu. Neuronų išėjimų vertės priklauso nuo n - tojo įėjimo duomenų vektoriaus, esančio įėjime. Nenorint užgriozdinti žymėjimo sistemos, įėjimo duomenų indeksas n, žymint neuronų įėjimo, svorinių sumų ir išėjimo kintamuosius, bus paleidžiamas.

Klaida E^n priklauso nuo svorio w_{ij} tik per svorinę įėjimų į j - tąjį elemento sumą a_j . Dalinė klaidos išvestinė pagal svorį w_{ij} užrašoma taip:

$$\frac{\partial E^n}{\partial w_{ji}} = \frac{\partial E^n}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}} \tag{2.21}$$

Pažymima:

$$\delta_j \equiv \frac{\partial E^n}{\partial a_j} \tag{2.22}$$

Naudojant (2.18), galima užrašyti:

$$\frac{\partial a_j}{\partial w_{ji}} = z_i \tag{2.23}$$

Įrašę (2.22) ir (2.23) į (2.21), gauname:

$$\frac{\partial E^n}{\partial w_{ji}} = \delta_j z_i \tag{2.24}$$

Taigi reikalinga išvestinė gaunama padauginant vieno neurono δ_j iš kito neurono z_i (neuronai sujungti jungtimi w_{ji}).

δ galima traktuoti kaip tam tikros rūšies neurono išėjimo klaidą. Kaip matyti iš (2.24), tinklo klaidos išvestinės svorio atžvilgiu, t. y. svorio pokyčio, įtaka bendrai klaidai yra didesnė, kuo didesnę klaidą daro neuronas. Taip pat išvestinė tuo didesnė, kuo stipresnis signalas, perduodamas svorio jungtimi.

Taigi norint rasti išvestinę, tereikia pagal (2.24) išraišką apskaičiuoti paslėptųjų ir išėjimo neuronų δ_j vertes.

Išėjimo mazgų δ_k apskaičiuojama tiesiogiai. Iš (2.22) gauname

$$\delta_k \equiv \frac{\partial E^n}{\partial a_k} = \frac{\partial y_k}{\partial a_k}\frac{\partial E^n}{\partial y_k} = g'(a_k)\frac{\partial E^n}{\partial y_k} \tag{2.25}$$

kur y_k apskaičiuojama pagal (2.19) formulę, z_k pažymėjus y_k.

Apskaičiuojant paslėptųjų mazgų δ, vėl keičiami diferencijavimo kintamieji:

$$\delta_j \equiv \frac{\partial E^n}{\partial a_j} = \sum_k \frac{\partial E^n}{\partial a_k}\frac{\partial a_k}{\partial a_j} \tag{2.26}$$

čia sumuojami visi k neuronai, į kuriuos mazgas j turi jungtis.

Svorių ir neuronų ryšiai parodyti 2.4 pav. Neuronas, pažymėtas indeksu k, gali būti aukštesnio nei nagrinėjamojo neurono j paslėptojo sluoksnio arba išėjimo neuronas.

(2.22) išraišką įrašius į (2.26) ir panaudojus (2.18) bei (2.19), gaunama atgalinio skleidimo formulė:

$$\delta_j \equiv \frac{\partial E^n}{\partial a_j} = \sum_k \frac{\partial E^n}{\partial a_k}\frac{\partial a_k}{\partial a_j} = g'(a_j)\sum_k w_{kj}\delta_k \tag{2.27}$$

nes

$$a_k = \sum_j g'(a_j)w_{kj} \tag{2.28}$$

ir

$$\frac{\partial a_k}{\partial a_j} = g'(a_j)w_{kj} \tag{2.29}$$

Išraiška (2.27) rodo, jog δ vertė tam tikram neuronui gali būti gauta atgaliniu δ verčių skleidimu iš tolimesnių neuronų (mazgų), kaip parodyta 2.4 pav. Kadangi išėjimo neuronų δ vertės yra žinomos iš (2.25), rekursyviai naudojant (2.27) išraišką, galima apskaičiuoti visų

paslėptųjų neuronų δ nepriklausomai nuo neuroninio tinklo topologijos, su sąlyga, kad tai tiesioginio sklidimo neuronais tinklas.

Atgalinio skleidimo procedūrą klaidos E^n išvestinei svorių atžvilgiu apskaičiuoti galima apibendrinti keturiais žingsniais:

1. Neuroninio tinklo įėjime esant įėjimo vektoriui x^n, pagal(2.18) ir (2.19) formules apskaičiuoti visų paslėptųjų ir išėjimo neuronų išėjimų vertes.
2. Naudojant (2.25) apskaičiuoti visų išėjimo mazgų δ_k.
3. Skleisti δ_k atgal, naudojant (2.27), ir apskaičiuoti visų neuroninio tinklo paslėptųjų neuronų δ_j.
4. Naudoti (2.24) išraišką reikiamoms išvestinėms skaičiuoti.

Bendra klaida E visai įėjimo duomenų imčiai gali būti gaunama kartojant šiuos keturis žingsnius visiems imties duomenims ir sumuojant atskirų duomenų taškų klaidas:

$$\frac{\partial E}{\partial w_{ji}} = \sum_n \frac{\partial E^n}{\partial w_{ji}} \tag{2.30}$$

2.3 Neuroninio tinklo mokymas

Klaidos išvestinės svorių atžvilgiu gali būti efektyviai skaičiuojamos daugiasluoksniams neuroniniams tinklams, naudojant atgalinio skleidimo algoritmą. Informaciją apie klaidos gradientą svorių atžvilgiu svarbi kuriant pakankamai greitus mokymo algoritmus, kurie gali būti taikomi praktikoje [GV07].

Mokant neuroninį tinklą, reikia rasti svorių vektorių, kuriam esant klaidos funkcija E(**w**) yra mažiausia. Dažnu atveju klaidos funkcija yra netiesinė, todėl bendruoju atveju neįmanoma rasti matematinės išraiškos klaidos minimumui apskaičiuoti, todėl yra naudojami iteracijų algoritmai, susidedantys iš žingsnių sekos:

Dėl to, kad klaidos funkcija netiesinė, bendruoju atveju neįmanoma rasti matematinės išraiškos klaidos minimumams apskaičiuoti. Vietoj to naudojami iteracijų algoritmai, susidedantys iš žingsnių sekos

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)} \tag{2.31}$$

kur τ žymi iteracijos žingsnį. Skirtinguose algoritmuose skirtingai parenkamas svorių žingsnis $\Delta w^{(\tau)}$. Šis žingsnis gali būti skaidomas į dvi komponentes: žingsnio dydį ir kryptį.

Neuroninio tinklo svorių su kuriais klaidos funkcija yra mažiausia radimui yra naudojamas gradientinio nusileidimo algoritmas. Šis algoritmas yra vienas paprasčiausių neuroninių tinklų mokymo algoritmų. Svorių pokytis gali būti apskaičiuojamas iš visos duomenų imties:

$$\Delta w^{(\tau)} = -\eta \nabla E^n \Big|_{w^{(\tau)}} \tag{2.32}$$

Kur Ɛ yra mokymosi greičio žingsnis, kuris reguliarizuojamas kaip svorių keitimo dydį [Bis07]. Jei mokymo žingsnis per mažas, mokymas vyksta lėtai.

Svorių keitimo taisyklėje dažnai pridedamas judėjimo svorių erdvėje „inercijos“ narys [GV08]. Inercijos (angl. momentum) metodai naudoja gradientinę informaciją svorių koregavimui kaupiant gradiento greitį ta linkme, kuri nuosekliai mažina klaidos funkciją. Svorių keitimo taisyklė su šiuo inercijos nariu atrodo taip:

$$\Delta w^{(\tau)} = -\eta \nabla E \Big|_{w^{(\tau)}} + \mu \Delta w^{(\tau-1)} \tag{2.33}$$

Gradientinis nusileidimas svoriu modifikavimui naudoją visą duomenų imtį, todėl praktikoje šis algoritmas dažnai gali būti neefektyvus kuomet duomenų imtis yra pakankamai didelė. Dėl šios priežasties dažnai naudojamas stochastinis gradientinio nusileidimo (angl. stochastic gradient descent) algoritmas. Šis algoritmas svorių koregavimui naudoja tik atsitiktinai pasiriktą duomenų imties poaibį. Tai leidžia žymiai greičiau pasiekti svorių konvergenciją kuomet duomenų imtis yra didelė [LBB+98], [BL04], [BB08].

Kiti gerai žinomi ir efektyvus algoritmai optimalių svorių radimui yra RPROP ir Quickprop, kurie taip pat yra paremti klaidos atgalinio skleidimo algoritmu [RB93], [FAH88].

2.4 Neuroninio tinklo reguliarizavimas

Apmokius sudėtingą neuroninį tinklą, jo funkcija dažnai turi didelių svyravimų (dėl per didelio prisitaikymo prie mokymo duomenų)[GV07]. Norint gauti glotnesnę ir tolygesnę tinklo perdavimo funkciją, turinčią geresnes apibendrinimo savybes, prie klaidos funkcijos pridedamas sudėtingumą kontroliuojantis narys **Ω**:

(2.34)

$$\tilde{E} = E + \nu\Omega \ ()$$

Čia E – standartinė paklaidos funkcija, o parametras v kontroliuoja papildomo nario įtaką bendrai klaidos funkcijai. Minimizuojant naująją klaidos funkciją gradientiniais metodais, reikalaujama, kad narys **Ω** būtų diferencijuojamas svorių atžvilgiu. Parinkus funkciją *y(x)*, gerai atitinkančią duomenis, gaunama maža E vertė. Narys **Ω** parenkamas toks, kad jo vertė būtų maža, kai funkcija *y(x)* glodi. Galutinė perdavimo funkcija turėtų būti kompromisas tarp gero duomenų atitikimo ir minimalaus nario **Ω**.

Vienas iš paprasčiausių reguliarizavimo narių yra neuroninio tinklo svorių kvadratų suma, kurioje sumuojami visi svoriai ir laisvieji nariai:

$$\Omega = \frac{1}{2}\sum_i w_i^2 \tag{2.35}$$

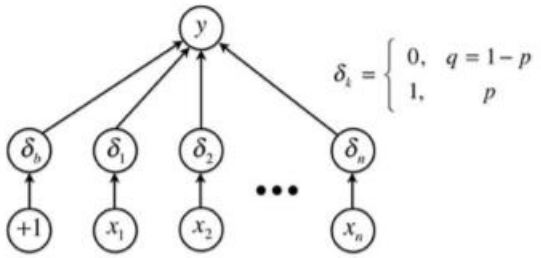
Dažnai ši reguliarizavimo forma yra vadinama L2 reguliarizavimu [Ng04], [GV07]. Atliekant šį reguliarizavimą siekiama, kad svorių kvadratinė suma butų, kuo mažesnė. Tai leidžia žymiai pagerinti neuroninio tinklo apibendrinimo savybes. Neuronų tinklui per daug prisitaikius prie mokymo duomenų, tinklo funkcija turi didelių svyravimų ir staigių linkių. Norint gauti labai kreivus paviršius, reikalingi dideli svoriai, nes esant mažiems svoriams daugiasluoksnio neuroninio tinklo perdavimo funkcija tampa tiesinė. Todėl šis narys riboją svorių didėjimą, tuo ribodamas ir funkcijos sudėtingumą.

Panašiai kaip ir L2 reguliarizavimas yra naudojamas L1 reguliarizavimui [Ng04]. Naudojant šį reguliarizavimą prie klaidos funkcijos pridedama absoliuti neuroninio tinklo svorių suma:

$$\Omega = \frac{1}{2}\sum_i \left\| w_i \right\| \tag{2.36}$$

Šiuo reguliarizavimo metodu siekiama, kad absoliuti svorių suma butų kuo mažesnė. Atliktuose tyrimuose nustatyta, kad naudojant dauguma modelio svorių vertės priartėja prie 0 [Ng04]. Dėl šios priežasties, modelis naudoja tik svarbiausius ir pačius reprezentatyviausius požymius.

Pastaruoju metu labai išpopuliarėjo naujas neuroninių tinklų reguliarizavimo metodas vadinamas išjungimo metodu (angl. dropout) [SHK+14]. Šio metodo pagrindinė idėja yra atsitiktiniu būdu išjungti neuroninio tinklo vienetus (neuronus) mokymo metu. Šie neuronai parenkami atsitiktiniu būdu su tikimybe q = 1-p. Kai neuronas išjungiamas, įeinančios ir išeinančios jungtys su kitais neuronais yra taip pat panaikinamos. Tai pavaizduota 2.5 pav.



2.5 pav. Neuroninio tinklo mokymo pavyzdys su išjungimo reguliarizavimo metodu

Visa tai leidžia kiekvienam neuronui išmokti naudingos informacija ir per daug nepasikliauti šalia esančiais kitais neuronais. Tokiu būdu mokymo metu yra gaunama daug skirtingų neuroninių tinklų architektūrų, kurie kiekvienas gali išmokti atpažinti geriau vieną ar kita požymį nei kiti neuroniniai tinklai. Taip pat gali ir persimokyti skirtingais būdais, tačiau kuomet iš visų skirtingų neuroniniai tinklų rezultatų gaunamas vidurkis, bendras viso tinklo persimokymas yra išvengiamas.

Naudojant neuroninį tinklą apmokytą su išjungimo reguliarizacijos metodu prognozavimui su testavimo duomenimis reikia išjungimo reguliarizacijos metodą išjungti.

2.5 Maišyto tankio neuroninis tinklas

Neuroninio tinklo apmokyto naudojant mažiausių kvadratų klaidos metodą išėjimo vertės aproksimuoja sąlyginius užduoties vidurkius. Daugeliui klasifikacijos problemų spręsti vidurkis yra optimalus sprendinys. Tačiau sprendžiant regresijos problemas sąlyginis vidurkis yra labai limituotas parametras. Dėl šios priežasties dažnai yra siekiama rasti parametrus, kurie geriau apibūdintų sprendžiamos užduoties duomenų pasiskirstymą.

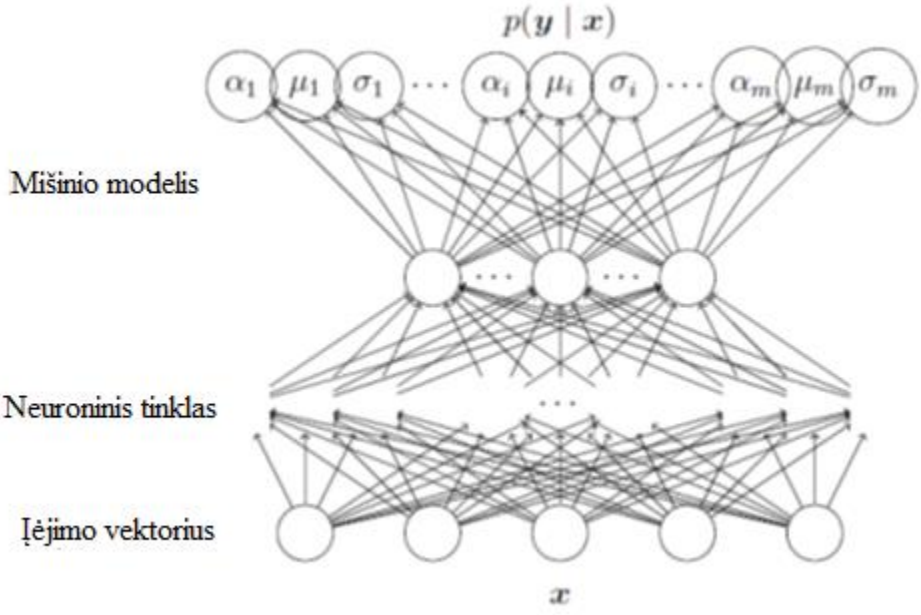
Mažiausių kvadratų klaidos išraiška gali būti išvesta remiantis maksimalaus tikėtumo principu, laikantis prielaidos, kad sprendžiamos užduoties duomenys pasiskirstę pagal normalųjį skirstinį. Dėl šios priežasties galima normalųjį skirstinį pakeisti mišinio modeliu (angl. mixture model), kuris leidžia modeliuoti bendresnę duomenų pasiskirstymo funkciją. Tuomet užduoties duomenų sąlyginės tikimybės tankio funkciją galima modeliuoti kaip branduolio funkcijų tiesinę kombinaciją:

$$p(y|x) = \sum_{i=1}^{m} \alpha_i(x)\, \phi_i(y|x) \tag{2.37}$$

Kur m yra branduolių funkcijų skaičius mišinio modelyje ir $\alpha(x)_i$ yra vadinamas maišymo koeficientu (angl. mixing coefficient). Autorius [Bis94] savo darbe naudojo branduolio funkcija turinčią normaliojo skirstinio formą:

$$\phi_i(y|x) = \frac{1}{(2\pi)^{c/2}\, \sigma_i(x)^c}\, exp\left\{-\frac{\|y-\mu_i(x)\|^2}{2\sigma_i(x)^2}\right\} \tag{2.38}$$

Kur $\mu(x)_i$ yra branduolio centro vertė, $\sigma(x)_i$ branduolio reikšmių standartinis nuokrypis. Toks mišinio modelis leidžia gauti bendrine salyginio tankio *p(y|x)* funkcijos formą. Mišinio modelio parametrus galime modeliuoti kaip įėjimo vektoriaus funkcijas ir todėl šias reikšmes galima modeliuoti kaip neuroninio tinklo išėjimo reikšmes [Bis94]. Tokio tipo modelis yra vadinamas maišyto tankio neuroniniu tinklu (angl. Mixture density network). Šis tinklas grafiškai pavaizduotas 2.6 pav.



2.6 pav. Maišyto tankio neuroninis tinklo modelis.

Šio tinklo mokymui yra naudojama tokia paklaidos funkcija:

$$\log \mathcal{L}(y|x) = -\log(p(y|x)) = -log\left(\sum_{i=0}^{m} \alpha_i(x)\, \phi_i(y|x)\right) \tag{2.38}$$

2.6 Gilaus mokymo algoritmai

Neuroninis tinklas turintis tris ar daugiau paslėptus nelinijinių modulių sluoksnius vadinamas giliu neuroniniu tinklu [BL07]. Pastaruoju metu tokio tipo algoritmus imta vadinti gilaus mokymo algoritmais (angl. deep learning).

Kad neuroninį tinklą galėtumėme pavadinti gilaus mokymo algoritmu, jis turi atitikti dvi pagrindines sąlygas:

1. Tinklas turi turėti galimybe būti praplėstas pridedant papildomų modulių sluoksnius
2. Kiekvieno tinklo parametrus turi turėti galimybę būti išmokstamas

Iki 2006 metų buvo manoma, kad neuroninį tinklą turinti daugiau nei du paslėptus sluoksnius yra sunku apmokyti, nes jie neduodavo geresnių rezultatų negu seklios architektūros neuroniniai tinklai, turintys tik viena ar du paslėptus sluoksnius [BL07], [LBH15].

Viena iš svarbiausių hipotezių bandanti paaiškinti kodėl neuroninis tinklas turintis daugiau paslėptų sluoksnių neduoda geresnių rezultatų, buvo pasiūlyta [BLP+07]. Šiame darbe buvo nustatyta, jog neuroniniame tinkle žemesni paslėpti sluoksniai esantys prie įėjimo sluoksnio nėra gerai išnaudojami, kai tuo tarpu kituose trijuose sluoksniuose, susidedančiuose iš dviejų paslėptų ir vieno išėjimo sluoksnio, įvyksta persimokymas (angl. overfitting). Dėl šios priežasties tinklas negali gerai apibendrinti nematytų duomenų. Dėl šios priežasties neuroniniai tinklai turintys daugiau nei du paslėptus sluoksnius praktikoje buvai retai naudojami.

Tačiau [HS06], [HIN07], [BLP+07] pasinaudojus godžiu individualių sluoksnių apmokymo algoritmu (angl. greedy layerwise pre-training) pavyko sėkmingai apmokyti neuroninį tinklą turintį daug paslėptų sluoksnių ir jį pritaikyti labai sudėtingoms užduotims spręsti. Šis algoritmas naudojamas pradiniam tinklo svorių parinkimui, kuriame tinklo sluoksniai yra apmokomi vienas po kito ir vėliau naudojant atrastus svorius viso tinklo apmokymui pritaikomi standartiniai optimizavimo algoritmai kaip stochastinis gradientinis nusileidimas. Po šios sėkmės šie neuroniniai tinklai sulaukė vis daugiau mokslininkų dėmesio [BLH15].

[BLP+07] pasiūlyta hipotezė nebuvo vienintelis paaiškinimas kodėl nepavyko apmokyti gilių neuroninių tinklų. [MAR10] pasiūlė kita paaiškinimą, jog to pagrindinė priežastis gali būti nenormalus klaidos funkcijos išlenktumas. Autorius teigia, kad klaidos funkcija lyginant su neuroninio tinklo sluoksniais esančiais arčiau įėjimo sluoksnio parametrais turi mažesnį

išlenktumą nei sluoksniai esantys arčiau išėjimo sluoksnio. Dėl šios priežasties pasinaudojus tik pirmos eilės klaidos funkcijos išvestinės informacija, gradientinio nusileidimo algoritmas negali sparčiai pajudėti ta kryptimi, kur kreivės išlenktumas yra labai mažas ir todėl tinklo svoriai tuose sluoksniuose dažniausiai išlieka nepakoreguoti.

[MAR10] parodė, kad panaudojus laisvuoju hesiano optimizavimo (angl. Hessian-free optimization (HF) algoritmu ir nenaudojant [HS06] pasiūlyto algoritmo pradinio svorių parinkimui, pavyko taip pat sėkmingai apmokyti gilius neuroninius tinklus ir gauti mažesnę tinklo paklaidą toms pačios užduotims, kurios buvo atliktos [HS06].

Neseniai pasirodę tyrimai parodė, kad gilius neuroninius tinklus galima sėkmingai apmokyti pasinaudojus agresyviu inercijos žingsnio planavimu bei geru pradinių tinklo svorių pasirinkimu. [GB10] ir [MDH12] pavyko nesunkiai apmokyti neuroninius tinklus turinčius atitinkamai 5 ir 8 sluoksnius pasinaudojus atsitiktiniais pradiniais svoriais. [CE11] apmokė 11 sluoksnių auto-koduojanti neuroninį tinklą (angl. autoencoder) naudojant tik [GB10] pasiūlytą pirminių svorių parinkimo algoritmą ir stochastinį gradientinį nusileidimą. Jų gauti rezultatai buvo geresni nei [HS06].

Dar vienas iš pagrindinių atradimų, kurie palengvino neuroninių tinklų mokymą buvo naujų aktyvacijos funkcijų atradimas, tokių kaip individualus linijiniai vienetai (ang. Piecewise liner unit). Ši aktyvacijos funkcija susideda iš dviejų linijinių komponentų. [JKR+09] pastebėjo, kad šios funkcijos panaudojimas buvo vienas iš svarbiausių faktorių pagerinant vaizdo atpažinimo sistemą lyginant su kitais faktoriais. Bandymai parodė, kad su maža duomenų imtimi, aktyvacijos funkcija buvo daug svarbesnis faktorius nei pradinių svorių parinkimas paslėptame sluoksnyje. [GBB11] parodė, kad gilų neuroninį tinką yra lengviau apmokyti panaudojant lyginančią aktyvacijos funkciją negu naudojant standartines aktyvacijos funkcijas turinčias kreivės išlenkimus kaip sigmoidinė funkcija, dėl kurių atsiranda funkcijos išvestinės perpildymas (angl. saturation). Ši nauja funkcija gerai tiek skleidžia informacija tinklu į priekį, tiek perduoda gerai informacija apie klaidos išvestines visiems parametrams atgalino skleidimo etapu.

Sigmoidinės aktyvacijos funkcijos vis dar yra sėkmingai panaudojamas apmokyti seklios architektūros neuroninius tinklus, tačiau pastaruoju metu populiariausia yra lyginančioji aktyvacijos funkcija [NH10], [GBB11].

Norint išvengti tinklo persimokymo yra naudojamas modelio reguliarizavimas. Naujas reguliarizavimo metodas buvo pasiūlytas [SHK+14], kuris buvo pavadintas išjungimo (angl.

dropout) metodu. Šis metodas apjungia iš skirtingų neuroninių tinklų gaunamus rezultatus, kurie gaunami iš neuroninio tinklo atsitiktiniu būdu išjungiant neuronus. Tai leidžia iš vieno neuroninio tinklo gauti daug skirtingų neuroninių tinklų, kurie gali geriau apibendrinti duomenis ir išvengti persimokymo. Šis metodas leido pagerinti iki tol buvusius rezultatus vaizdo ir kalbos atpažinimo problemose [KSH12]. Be to, su šiuo reguliarizavimo metodu tinklo apmokymas buvo nuo dviejų iki trijų kartų greitesnis nei su prieš tai buvusiais metodais.

Kitas svarbus atrastas faktorius, kuris leido apmokyti gilius neuroninius tinklus buvo grafinio procesoriaus (GPU) panaudojimas tinklo mokymui [RMN09]. [CMG+12] parodė, kad nenaudojant jokių prieš tai išvardytų metodų, o tik stochastinį gradientinį nusileidimą, pavyko apmokyti gilų neuroninį tinklą ir pagerinti iki tol buvusius rezultatus skaičių atpažinimo problemoje panaudojant grafinį procesorių.

3 Tyrimo metodikos

Šioje dalyje bus aprašoma tyrimui naudojami mokymo duomenys, aprašomos pasirinktos neuroninių tinklų architektūros ir atliktų tyrimų rezultatai.

3.1 Mokymo duomenys

Tyrimui buvo naudojami trijų mėnesių bitkoinų limituotų pavedimų knygos duomenys nuo 2016-01-01 iki 2016-04-01 iš okCoin.cn biržos. Naudojamas duomenų kiekis sudarė daugiau nei 10 GB. Tyrimui buvo naudojama 5 geriausių limituotų pavedimų knygos pirkimo ir pardavimo kainos ir kiekio duomenys, iš viso 20 kintamųjų kiekvienu laiko momentu. Norint perteikti kainų pokyčių dinamiką neuroniniams tinklams buvo imami paskutiniai 20 pavedimų knygos pokyčių.

Limituotų pavedimų duomenų normalizavimui buvo naudojamas kainų ir kiekių kodavimas pagal 1-iš-c schemą, pagal praeities vidutinius kainų ir kiekių pokyčių dydžius. Dydžiai buvo koduojami į vektorius tokiu principu:

- [1,0,0] reiškė didelis neigiamas pokytis;
- [0,1,0] mažas pokytis;
- [0,0,1] didelis teigiamas pokytis.

Tokiu būdu sudaryta duomenų imtis, kurioje dauguma atributų įgyja reikšmes lygias nuliui yra vadinama reta duomenų aibe (angl. sparse data). Neuroninio tinklo įėjimo duomenų vektoriaus dydis yra 20x20x3. Prognozuojamas buvo geriausias pirkimo kainos pokytis po 5 limituotų knygos pokyčių į ateitį.

Duomenys, kuriuose nebuvo dideliu pokyčių buvo neįtraukiami į mokymo imtį. Iš viso buvo sudaryta duomenų imtis, kurią sudarė 50000 duomenų. Mokymo ir testavimo duomenims buvo skirta atitinkamai 80 % ir 20 % duomenų nuo visos duomenų aibės.

Neuroninių tinklų prognozavimo rezultatų palyginimui naudojama vidutinė kvadratinė paklaida (angl. Mean Squared Error(MSE)).

$$MSE = \frac{1}{n}\sum_{i}^{n}(f_i - y_i)^2$$

Neuroniniai tinklai buvo programuojami naudojant populiarias Python programavimo kalbos bibliotekas: Theano, Lasagne, TensorFlow.

3.2 Neuroninio tinklo architektūros

Norint ištirti tinkamiausią neuroninio tinklo architektūrą kainų prognozavimui buvo sudaromi 3 skirtingų architektūrų neuroniniai tinklai:

1. Neuroninis tinklas sudarytas tik iš vieno paslėptojo neuroninio sluoksnio su 100 neuronų. Šis tinklas turi 160403 parametrų;
2. Gilus neuroninis tinklas turintis 3 paslėptus neuroninius sluoksnius. Kiekvienas paslėptasis sluoksnis turi po 100 neuronų. Šis neuroninis tinklas iš viso turi 180603 parametrų;
3. Gilus neuroninis tinklas turintis 5 paslėptus neuroninius sluoksnius ir kiekviename paslėptajam sluoksnyje turintis po 100 neuronų. Šis tinklas iš viso turi 210903 parametrų.

Atliekant tyrimus neuroninių tinklų mokymui buvo naudojamas paketinis rėžimas su 5000 mokymo vektoriaus duomenų dydžiu ir tinklai mokomi 500 epochų. Mokymui buvo naudojamas stochastinis gradientinis nusileidimo metodas su 0.005 mokymo žingsniu ir 0.9 inercijos dydžiu. Tinklo kaštų funkcijai buvo naudojama vidutinė kvadratinė paklaida.

3.3 Aktyvacijos funkcijų tyrimas

Tyrimams buvo naudojamos tokios neuroninio tinklo aktyvacijos funkcijos:

1. Sigmoidinė;
2. Hiperbolinis tangentas;
3. Lyginamoji funkcija;

Trijų skirtingų neuroninių tinklų architektūros buvo apmokomos su skirtingomis aktyvacijos funkcijomis. Tokiu būdu buvo sudaryti 9 skirtingi neuroniniai tinklai:

- NT1_1, NT1_2, NT1_3 — neuroninis tinklas turintis 1 paslėptąjį neuroninį sluoksnį, su skirtingomis aktyvacijos funkcijomis;
- NT2_1, NT2_2, NT2_3 — neuroninis tinklas turintis 3 paslėptuosius sluoksnius;
- NT3_1, NT3_2, NT3_3 — neuroninis tinklas turintis 5 paslėptuosius sluoksnius.

Kiekvienas neuroninis tinklas buvo mokomas 10 kartų ir mokymo duomenys buvo sumaišomi kiekvieno mokymo metu. Norint geriau palyginti tinklų prognozavimo vidutinę kvadratinę paklaidą buvo naudojami tie patys testavimo duomenys. Šių neuroninių tinklų prognozavimo rezultatai pateikiami 3.1 lentelėje ir stačiakampėje diagramoje 3.1 pav.

Kaip matyti iš pateiktos lentelės mažiausia vidutinė prognozavimo klaida buvo seklaus neuroninio tinklo NT1_1 turinčio 1 paslėptąjį sluoksnį ir naudojant sigmoidinę aktyvacijos funkciją. Nepaisant to, jog šio neuroninis tinklo prognozavimo rezultatų standartinis nuokrypis buvo didžiausias, tačiau prognozuojant su šiuo tinklu buvo gauta pati mažiausia vidutinė kvadratinė paklaida 0,007652 ir maksimali reikšmė buvo taip pat maža lyginant su kitais neuroniniais tinklais.

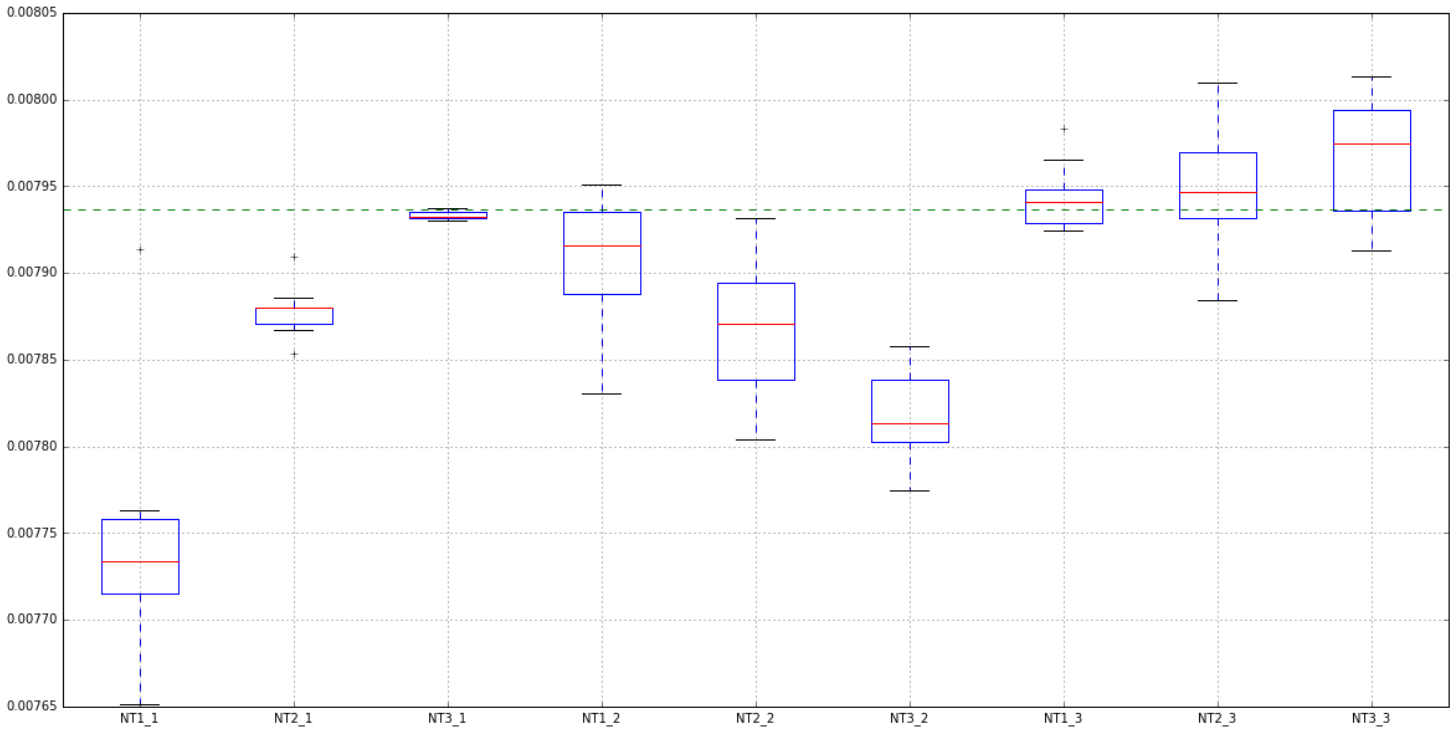
	NT1_1	NT2_1	NT3_1	NT1_2	NT2_2	NT3_2	NT1_3	NT2_3	NT3_3
vidurkis	0.007743	0.007878	0.007933	0.007907	0.007867	0.007816	0.007944	0.007949	0.007967
std	0.000068	0.000014	0.000002	0.000036	0.000042	0.000028	0.000019	0.00004	0.000035
min	0.007652	0.007854	0.007931	0.007831	0.007804	0.007775	0.007924	0.007884	0.007913
max	0.007913	0.007909	0.007937	0.007951	0.007932	0.007858	0.007983	0.00801	0.008013

3.1 lentelė. Aktyvacijos funkcijų įtaka skirtingų architektūrų neuroninių tinklų prognozavimo rezultatams

Prasčiausi prognozavimo rezultatai buvo gauti su neuroniniais tinklais NT1_3, NT2_3 ir NT3_3 kurie naudojo lyginančiąją aktyvacijos funkciją. Geresni rezultatai buvo gauti naudojant hiperbolinio tangento funkciją. Naudojant šią aktyvacijos funkciją buvo gauta mažiausią vidutinė kvadratinė paklaida su giliu neuroniniu tinklu turinčiu 5 paslėptuosius sluoksnius lyginant su kitais tos pačios architektūros neuroniniais tinklais.

Norėdami įvertini ar neuroninių tinklų prognozavimo rezultatai nėra atsitiktiniai, palyginome juos su baziniu scenarijumi, kuomet neuroninis tinklas visada prognozuotų reikšmę 0. Bazinio scenarijaus vidutinė kvadratinė paklaida lygi 0,00793696. Kaip matyti iš 3.1 pav. stačiakampės diagramos, kurioje bazinio scenarijaus vidutinė paklaida pavaizduota taškuota horizontalia linija, neuroninių tinklų su sigmoidine ir hiperbolinio tangento aktyvacijos funkcijomis prognozavimo rezultatai buvo geresni už bazinį variantą. Tačiau naudojant

lyginančiąją aktyvacijos funkciją, visų architektūrų neuroninių tinklų prognozavimo rezultatai buvo blogesni nei bazinio varianto.



3.1 pav. Skirtingų architektūrų neuroninių tinklų su skirtingomis aktyvacijos funkcijomis prognozavimo rezultatų stačiakampė diagrama.

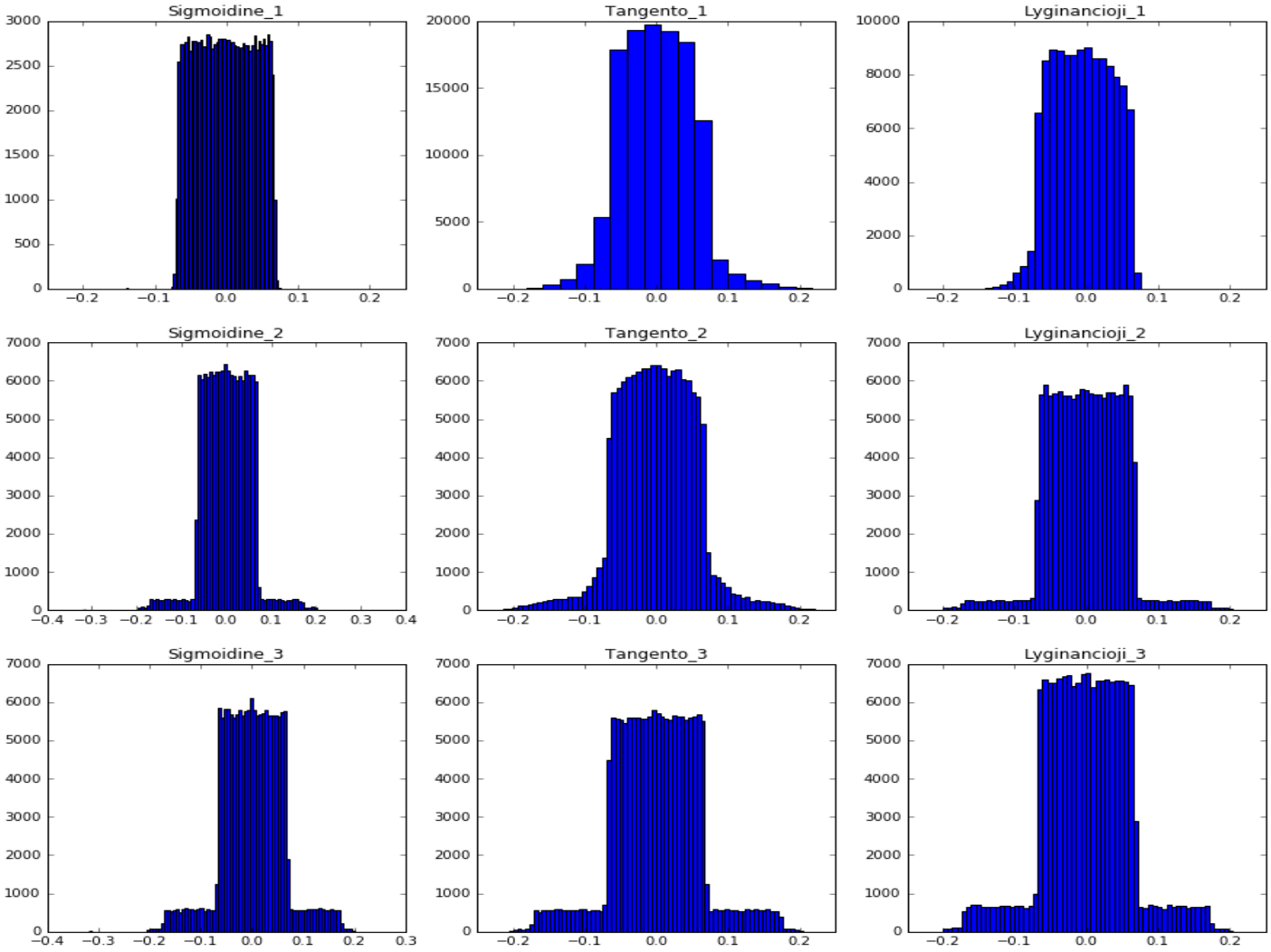
Gauti rezultatai prieštarauja, daugumos kitų tyrėjų [GB10], [GBB11] gautiems rezultatams. Kuriuose teigiama, kad sigmoidinė aktyvacijos funkcija nėra tinkami gilaus neuroninio tinklo mokymui, nes aktyvacijos paslėptuose neuroniniuose sluoksniuose esančiuose arčiau prie išėjimo sluoksnio yra prisotinamos, todėl aktyvacijos vertė tampa labai artima nuliui ir neuroninio tinklo mokymas sulėtėja arba išvis sustoja. Dėl šios priežasties šie autoriai rekomenduoja naudoti lyginančiąja aktyvacijos funkciją, kuri neturi neuronų prisotinimo problemos. Tačiau kaip matyti iš šio tyrimo, naudojant sigmoidinę aktyvacijos pavyko apmokyti gilų neuroninį tinklą ir rezultatai buvo geresni nei naudojant lyginančiąją aktyvacijos funkciją. Viena iš pagrindinių priežasčių kodėl taip atsitiko, galėjo būtu tai, kad mokymui buvo naudojami reta duomenų imtis. Yra žinoma, kad vienas iš pagrindinių lyginančiosios aktyvacijos funkcijos trūkumų yra tas, kad neuronai neuroniniame tinkle gali būti neaktyvuoti, t. y. jų reikšmės bus lygios

nuliui ir todėl mokymas tinklų nevyks.

Norėdami nustatyti ar reta duomenų imtis turėjo įtakos, kad neuroninių tinklų su lyginančiąją aktyvacijos funkcija prognozavimo rezultatai buvo prastesni lyginant su kitais tinklais buvo sudarytos apmokytų neuroninių tinklų svorių stulpelinės histogramos. Iš viso buvo sudarytos 9 histogramos ir reikšmes suskirstomos į 50 intervalų. Tos pačios architektūros apmokytų tačiau su skirtingomis aktyvacijos funkcijomis apmokytų neuroninių tinklų svoriai pavaizduoti 3.2 pav, kur:

- Sigmoidinė_1, Tangento_1, Lyginančioji_1 — neuroninio tinklo turinčio 1 paslėptąjį neuroninį sluoksnį svorių histogramos su skirtingomis aktyvacijos funkcijomis;
- Sigmoidinė_2, Tangento_2, Lyginančioji_2 — neuroninio tinklo turinčio 3 paslėptuosius sluoksnius svorių histogramos;
- Sigmoidinė_3, Tangento_3, Lyginančioji_3 — neuroninio tinklo turinčio 5 paslėptuosius sluoksnius svorių histogramos;

Kaip matyti iš 3.2 pav., apmokyto seklaus neuroninio tinklo turinčio 1 paslėptąjį sluoksnį svorių stulpelinės histogramos su skirtingomis aktyvacijos funkcijomis buvo labai skirtingos. Naudojant sigmoidinę aktyvacijos funkcija svorių reikšmės buvo tolygiai ir tankiai pasiskirstę visame intervale (-0,1; 0,1) bei sudarytų intervalų skaičius labai didelis. Tuo tarpu neuroninio tinklo apmokyto su hiperbolinio tangento aktyvacijos funkcija svorių reikšmės buvo pasiskirstę platesniame intervale bei jų reikšmės labai panašios ir tankiai labai dideli. Dėl šios priežasties, intervalų skaičius mažesnis ir histogramos stulpelių intervalai platesni. Panaši situacija ir su lyginančiąja aktyvacijos funkcija, svorių reikšmių tankis buvo daug didesnis nei lyginant su sigmodine aktyvacijos funkcija ir intervalai platesni. Panašus rezultatai gauti ir su kitomis neuroninių tinklų architektūromis. Iš to galime daryti išvadas, kad neuroniniai tinklai apmokyti su sigmoidine aktyvacijos funkciją turėjo mažiau svorių reikšmių, kurių vertės buvo lygios nuliui. Vadinasi, lyginančioji aktyvacijos funkcija nėra tinkama neuroninių tinklų mokymui su reta duomenų aibe.



3.2 pav. Skirtingų architektūrų apmokytų neuroninių tinklų svorių stulpelinės histogramos.

3.4 Reguliarizamo metodų tyrimas

Norint palyginti skirtingų reguliarizavimo metodu efektyvumą neuroniniams tinklams buvo sudaryti tokie modeliai:

- NT1_1, NT1_2, NT1_3 — vienasluoksnis neuroninis tinklas nenaudojant reguliarizacijos, su L2 ir išjungimo reguliarizavimo metodais;
- NT3_1, NT3_2, NT3_3 — gilus neuroninis tinklas su 3 paslėptais neuroniniais sluoksniais nenaudojant reguliarizacijos, su L2 ir išjungimo reguliarizavimo metodu.

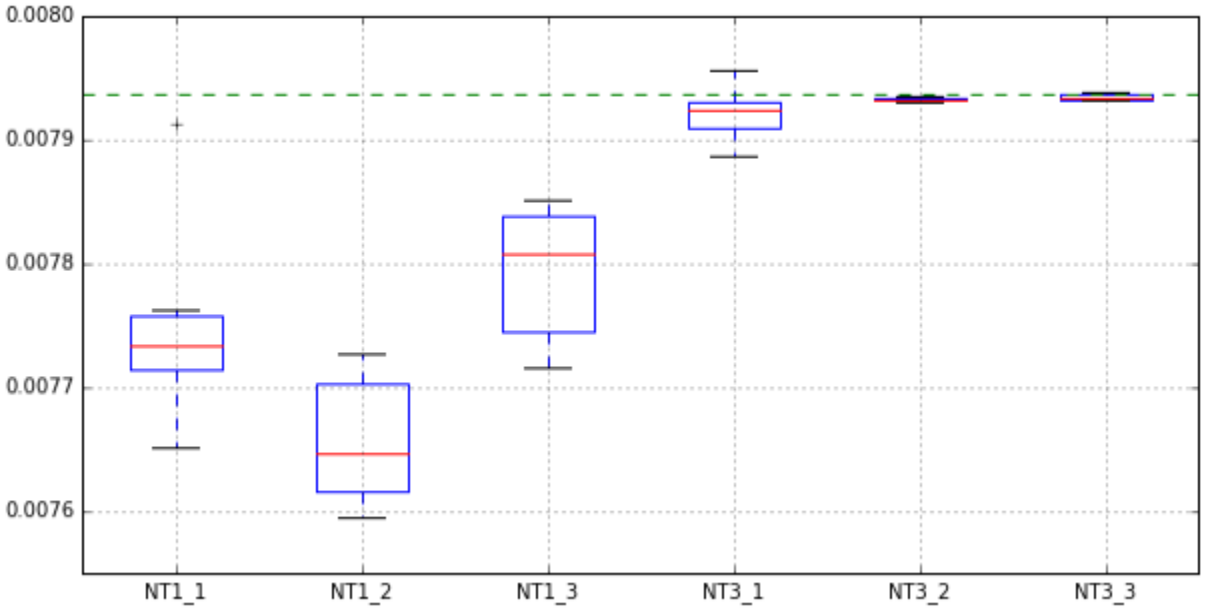
Buvo naudojama sigmoidinė aktyvacijos funkcija ir pradinių svorių iniciavimui buvo naudota Glorot normalusis skirstinys. Visi neuroniniai tinklai buvo mokomi 500 epochų ir mokymui naudojant stochastinį gradientinį nusileidimą su paketiniu rėžimu. Kiekvieno neuroninio tinklo mokymas buvo kartojamas 10 kartų. Mokymo duomenys buvo sumaišomi kiekvieną kartą permokinant visus neuroninius tinklus, tačiau testavimo duomenys buvo nekeičiami. Gautų rezultatų vidutinė kvadratinė paklaida buvo palyginta su baziniu scenarijumi, kuomet prognozavimo rezultatas visada yra lygus nuliui.

Kaip matyti iš 3.2 lentelėje pateiktų statistinių duomenų, tiksliausias prognozavimo rezultatas buvo gautas naudojant vienasluoksnį neuroninį tinklą NT1_2 su L2 reguliarizacijos metodu. Šio neuroninio tinklo vidutinės kvadratinės paklaidos vidurkis buvo pats mažiausias 0.007656, bet to, šis tinklas turėjo mažiausią minimalią reikšmę 0.007595 bei mažiausią maksimalią reikšmę 0.007728. Gilaus neuroninio tinklo rezultatai buvo prastesni nei vienasluoksnio neuroninio tinklo. Geriausi rezultatai gauti nenaudojonat reguliarizacijos metodo, tačiau šio tinklo rezultatų standartinis nuokrypis buvo pats didžiausias 0,000019 lyginant su kitais giliais neuroniniais tinklais.. Todėl šios priežasties, neuroninio tinklo prognozavimo rezultatai nėra patikimi. Stabiliausi rezultatai gauti naudojant neuroninį tinklą su L2 reguliarizacijos metodu, kurios prognozavimo vidutinės kvadratinės paklaidos vidurkis 0,007933 buvo didesnis už kitus reguliarizacijos metodus išskyrus be reguliarizacijos. Su šiuo metodu buvo gauti pati mažiausia maksimali reikšmė 0,007935 ir mažiausias prognozavimo reikšmių standartinis nuokrypis 0,000001.

	NT1_1	NT1_2	NT1_3	NT3_1	NT3_2	NT3_3
Vidurkis	0.007743	0.007656	0.007792	0.007921	0.007933	0.007934
Standartinis nuokrypis	0.000068	0.000050	0.000052	0.000019	0.000001	0.000003
Minimali reikšmė	0.007652	0.007595	0.007716	0.007888	0.007931	0.007931
Maksimali reikšmė	0.007913	0.007728	0.007852	0.007957	0.007935	0.007939

3.2 lentelė. Gilaus neuronio tinklo prognozavimo rezultatų vidutinė kvadratinė paklaida naudojant skirtingus reguliarizavimo metodus.

Kaip matyti iš 3.3 pav. stačiakampės diagramos, kurioje pavaizduoti neuroninių tinklų prognozavimo rezultatų vidutinių kvadratinių klaidų pasiskirstymo, visų naudotų neuroninių tinklų prognozavimo paklaidos vidurkis buvo mažesnis už bazinio scenarijaus paklaidą. Patys geriausi rezultatai gauti naudojant seklų neuroninį tinklą su vienu paslėptuohu sluoksniu ir L2 reguliarizacijos metodu. Šio tinklo prognozacimo rezultatų vidurkis, minimali ir maksimali reikšmė buvo mažesnė už bazinio scenarijaus.



3.3 pav. Reguliarizacijos metodu palyginimas naudojant sigmoidinę aktyvacijos funkcija.

3.4 Standartinio ir maišyto tankio neuroninių tinklų palyginimas

Norėdami palygini maišyto tankio neuroninio tinklo prognozavimo rezultatus su standartinio neuroninio tinklo buvo sudaryti tokie modeliai:

- Standartinis neuroninis tinklas su 1 paslėptuoju neuroniniu sluoksniu;
- Maišyto tankio neuroninis tinklas su 1 paslėptuoju neuroniniu sluoksniu ir 3 branduoliais;
- Maišyto tankio neuroninis tinklas su 3 paslėptais neuroniniais sluoksniais ir 3 branduoliais.

Kaip matyti iš 3.3 lentelėje pateiktų duomenų tiksliausi prognozavimo rezultatai gauti naudojant maišyto tankio neuroninį tinklą su 3 paslėptais neuronais sluoksniais. Abiejų sudarytų maišyto tankio neuroninių tinklų rezultatai buvo geresni nei standartinio neuroninio tinklo. Bet to, visų neuroninių tinklų vidutinė kvadratinė paklaida buvo mažesnė už bazinio scenarijaus.

Modelis	Prognozės vidutinė kvadratinė paklaida	Mokymo laikas, sekundėmis
Bazinis scenarijus	0,007936	-
Neuroninis tinklas su 1 paslėptuoju neuroniniu sluoksniu	0,007743	194.36
Maišyto tankio neuroninis tinklas su 1 paslėptuoju neuroniniu sluoksniu	0,007664	235.51
Maišyto tankio neuroninis tinklas su 3 paslėptais neuroniniais sluoksniais	0,007582	551.25

3.3 lentelė. Sudarytų modelių prognozavimo vidutinė kvadratinė klaida

Lyginant tinklų mokymo laikus galime matyti, kad mažiausiai laiko užtrukto apmokyti standartinį neuroninį tinklą, 194.36 sekundes. Ilgiausiai užtruko apmokyti maišyto tankio neuroninį tinklą su 3 paslėptais neuroniniais sluoksniais

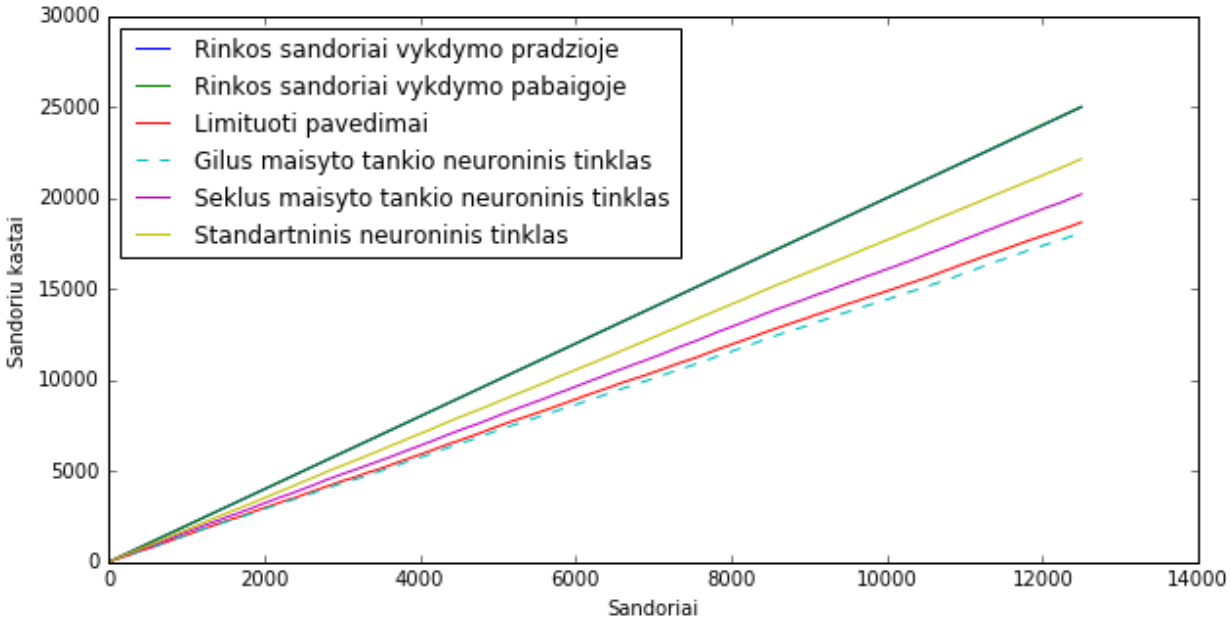
3.4.1 Neuroninių tinklų pritaikymas efektyviam sandorių vykdymui

Praeitame skyrelyje sudarytus neuroninius tinklus pritaikysime efektyviu sandoriu vykdymui. Sandoriai buvo vykdomi tik tuomet jei modelio prognozuojama reikšmė yra didesnė už nulį. Laikysimės prielaidų, jog rinkos sandorio kaina yra 0,2 % ir kiekvienas sandoris bus vykdomas už 1000 EUR, tuomet 12500 rinkos sandorių kaštai bus 25000 EUR. Sudarytu modeliu rezultatai bus palyginti su bazinėmis strategijoms, kuomet sandoriai visada vykdomi rinkos sandoriais periodo pradžioje ir pabaigoje, bet limituotais pavedimais.

Kaip matyti iš 3.4 lentelės ir 3.4 pav. mažiausi sandorių vykdymo kaštai buvo gauti naudojant maišyto tankio neuroninį tinklą su 3 paslėptais neuroniniais sluoksniais. Šis tinklas gerokai aplenkė kitus modelius. Nepaisant to, šis modelis tik labai nedaug aplenkė bazinę strategiją kuomet sandoriai vykdomi limituotais pavedimais ir jei sandoris neįvykdomas, tuomet sandoris vykdomas rinkos sandoriu periodo pabaigoje.

Strategijos	Sandorių vykdymo kaštai
Rinkos sandoriai periodo pradžioje	25000.00
Rinkos sandoriai periodo pabaigoje	25027.74
Limituoti pavedimai	18676.41
Neuroninis tinklas su 1 paslėptuoju neuroniniu sluoksniu	22154.42
Maišyto tankio neuroninis tinklas su 1 paslėptuoju neuroniniu sluoksniu	20216.25
Maišyto tankio neuroninis tinklas su 3 paslėptais neuroniniais sluoksniais	18105.35

3.4 lentelė. Sandorių vykdymo kaštai naudojant skirtingas strategijas



3.4 pav. Sandorių vykdymo kaštai naudojant skirtingas strategijas

3.5 Neuroninio tinklo prognozavimo rezultatų patikimumas

Naudojant neuroninius tinklus prognozavimui dažniausiai norima žinoti kiek neuroninis tinklas yra tikras dėl savo prognozės. Tai yra ypatingai svarbu norint pritaikyti neuroninius tinklus efektyviam sandorių įvykdymui, kadangi sandorius norime vykdyti tik tuomet kai prognozė yra didelės tikimybės. Mažas prognozuojamos reikšmes standartinis nuokrypis parodys, kad neuroninio tinklo prognozavimo duomenys yra patikimi Norėdami nustatyti neuroninių tinklų prognozavimo rezultatų patikimumą ir pritaikyti juos efektyvių sandorių vykdymui buvo sudaryti tokie modeliai:

1. 10 gilios ir 10 seklios architektūros neuroninių tinklų turinčių atitinkamai 1 ir 3 paslėptuosius neuroninius sluoksnius junginys. Mokinant kiekvieną neuroninį tinklą duomenys buvo sumaišomi atsitiktiniu būdu. Tokiu būdu kiekvienam testavimo duomenų elementui gaunama po 20 prognozių, iš kurių galime apskaičiuoti vidurkį ir standartinį nuokrypį.
2. Gilus neuroninis tinklas sudarytas iš 3 paslėptųjų neuroninių sluoksniu su išjungimo reguliarizacijos metodu. Prognozavimo metu išjungimo metodas yra paliekamas įjungtas ir dėl šios priežasties, kiekviena kartą prognozuojant yra išjungiami vis kiti

skirtingi neuronai. Dėl šios priežasties yra gaunamos skirtingos prognozės, iš kurių galime apskaičiuoti prognozavimo rezultatų standartinį nuokrypi kuris parodo kiek neuroninis tinklas yra tikras dėl prognozuojamos reikšmės [GG15]. Prognozavimas tokiu būdu buvo atliekamas 100 kartų.

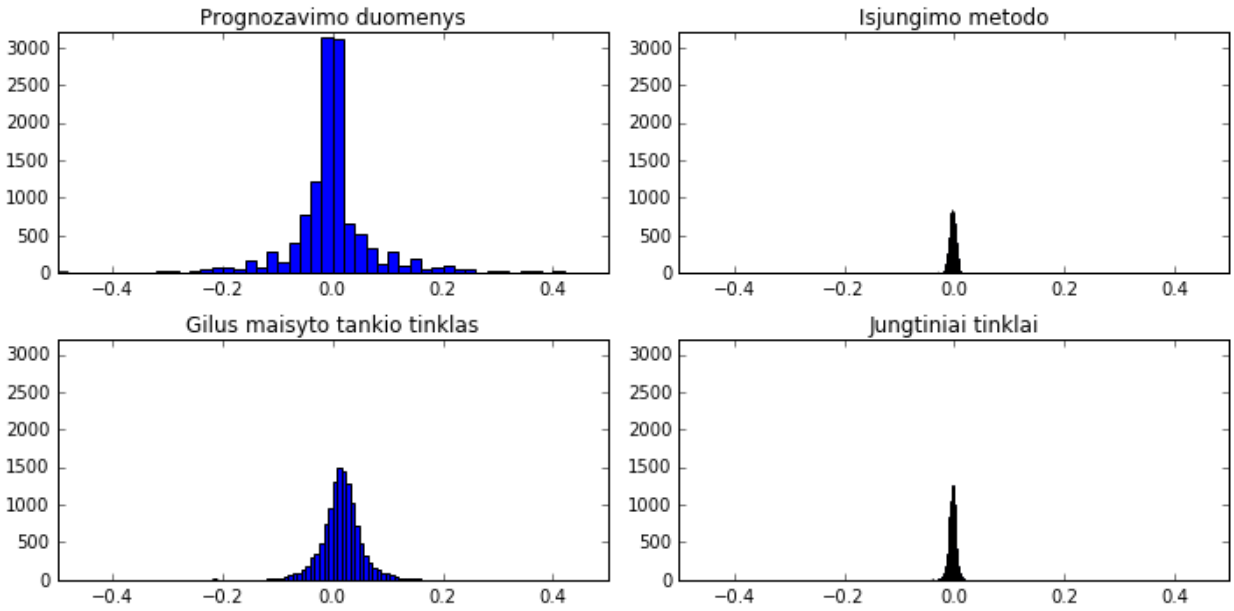
3. Gilus maišyto tankio neuroninis tinklas sudarytas iš 3 paslėptųjų neuroninių sluoksnių ir turintis 3 branduolius.

Šių modelių prognozavimo vidutinė kvadratinė paklaida pateikta 3.5 lentelėje. Kaip matyti iš lentelės, mažiausia paklaida buvo gilaus maišyto tankio neuroninio tinklo, o didžiausia neuroninio tinklo su reguliarizacijos metodu.

Modelis	Prognozavimo rezultatų vidutinė kvadratinė paklaida
Gilus maišyto tankio neuroninio tinklo	0,007582
Tinklas su išjungimo metodu	0.0079737
Neuroninių tinklų junginys	0.0078336

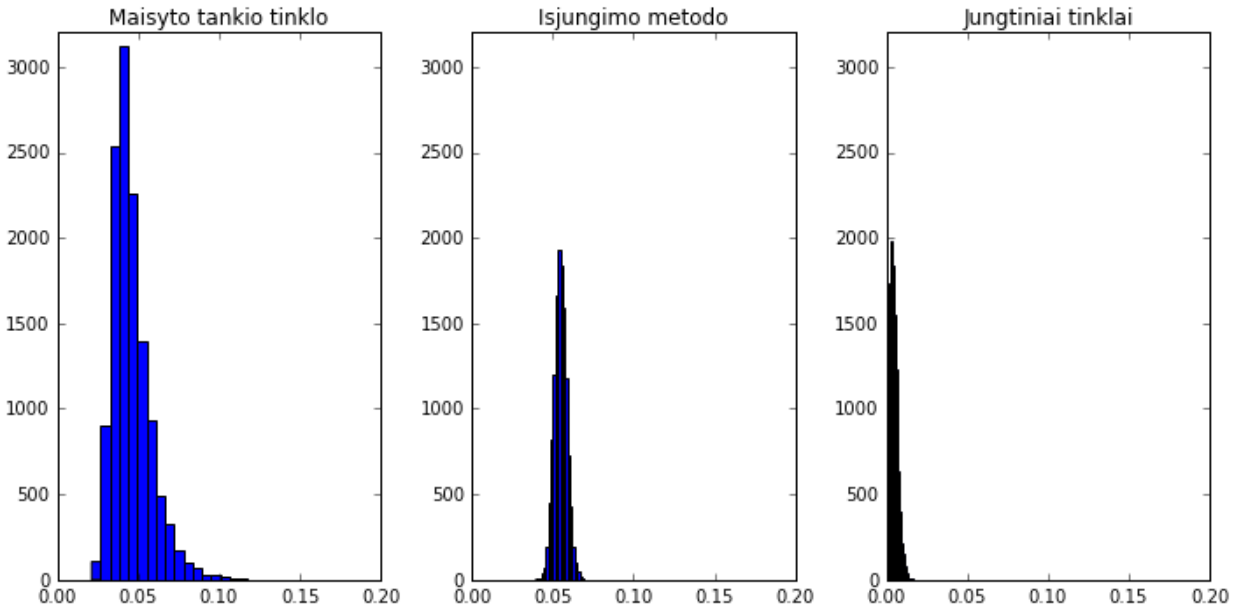
3.5 lentelė. Sudarytų modelių prognozavimo vidutinė kvadratinė klaida

Tačiau vidutinė paklaida neparodo kaip gerai prognozuojami duomenys atitinka tikrus duomenys. Norėdami palyginti kaip gerai sudarytų modelių prognozavimo rezultatai atitinka tikrus duomenimis buvo sudarytos 4 histogramos. Kaip matyti iš 3.5 pav. geriausiai tikrų duomenų pasiskirstymą atitinka gilaus maišyto tankio neuroninio tinklo prognozuojamos reikšmės. Kitų modelių prognozuojamos reikšmės labai susikoncentravusios arti nulio.



3.5 pav. Sudarytų modelių prognozuojamų reikšmių skirstiniai

Sudarytų modelių standartinis nuokrypis gaunamas skirtingais būdais. Maišyto tankio neuroninio tinklo standartinis nuokrypis yra apskaičiuojamas kaip kiekvieno branduolio svertinę sumą. Branduolio standartinis nuokrypis yra prognozuojamas taip pat neuroninio tinklo. Kitų modelių standartinis nuokrypis gaunamas apskaičiuojant standartinį nuokrypį tarp prognozavimo rezultatų. Kaip matyti iš 3.6 pav., neuroninio tinklo su išjungimo reguliarizacijos metodo ir jungtinių tinklų standartinis nuokrypis nėra informatyvus, nes jų reikšmės yra labai tankiai pasiskirsčiusios apie viena reikšmę.



3.6 pav. Sudarytų neuroninių tinklų prognozuojamų reikšmių standartinis nuokrypis

Tuo tarpu, maišyto tankio neuroninio tinklo standartinis nuokrypis yra labiau informatyvus, nes jo reikšmės yra labiau pasiskirsčiusios platesniame intervale.

3.5.1 Neuroninių tinklų patikimumo pritaikymas efektyviam sandorių vykdymui

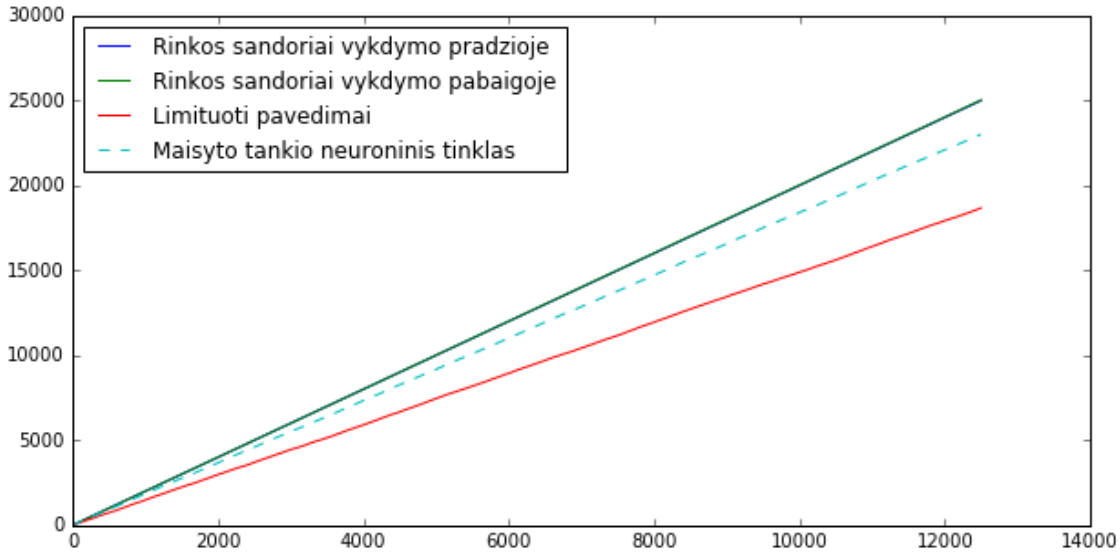
Norėdami pritaikyti neuroninius tinklus efektyviam sandorių vykdymui yra labai svarbu žinoti kiek neuroninis tinklas yra tikras dėl savo prognozių. Naudodami neuroninius tinklus iš praeito skyrelio sudarėme sandorių vykdymo strategiją. Sandoris bus vykdomas tik tokiu atveju jeigu neuroninio tinklo prognozuojamos reikšmės vidurkio atėmus šių reikšmių standartinį nuokrypį reikšmės bus teigiamos. Kitaip tariant, norima, kad su 68 % tikimybe norime, kad reikšmės aplink vidurkį būtų teigiamos. Kaip matyti iš 3.6 lentelėje pateiktų rezultatų sandorius su pasirinkta strategija pavyko vykdyti tik naudojant gilaus maišyto tankio neuroninio tinklą. Naudojant šį tinklą iš viso buvo vykyti 1125 sandoriai iš galimų 12500, iš kurių pelningai įvykdyti 607 sandoriai.

Modelis	Iš viso vykdytų sandorių	Sėkmingai įvykdyti sandoriai	Sėkmingai įvykdytų sandorių santykis, %
Gilus maišyto tankio neuroninio tinklo	1125	607	53,96
Tinklas su išjungimo metodu	0	0	0
Neuroninių tinklų junginys	0	0	0

3.6 lentelė. Efektyvių sandorių strategijos vykdymo rezultatai naudojant skirtingus neuroninių tinklų modelius.

Sėkmingai įvykdytų sandorių santykis buvo 53,96 %. Žinant, kad finansiniuose duomenys labai daug triukšmo ir sėkmingas prognozavimo rezultatų santykis dažnai nesiekia 50 %.

Gautus rezultatus palyginome su bazinėmis strategijomis. Kaip matyti iš 3.7 pav. naudojant maišyto tankio neuroninis tinklas leidžia sutaupyti sandorių vykdymo kaštus lyginant su rinkos sandoriais vykdomais periodo pradžioje ar pabaigoje, tačiau nusileidžia bazinei strategijai kuomet sandoriai vykdomi limituotais pavedimais.



3.7. pav Sandorio vykdymo kaštai naudojant rinkos sandorius ir strategija su maišyto tankio neuroniniu tinklu (MTNT)

4 Išvados

Palyginus 3 skirtingų neuroninių tinklų architektūrų prognozavimo rezultatus su skirtingomis aktyvacijos funkcijomis buvo nustatyta, kad seklesnės architektūros neuroninių tinklų prognozavimo rezultatų vidutinė kvadratinė paklaida buvo mažesnė nei gilių neuroninių tinklų.

Apmokius seklius ir gilius neuroninius tinklus su skirtingomis aktyvacijos funkcijomis ir reguliarizacijos metodais buvo gauta, kad prognozavimo rezultatų mažiausia vidutinė kvadratinė paklaida buvo gauta naudojant seklų neuroninį tinklą su sigmoidine aktyvacijos funkcija ir L2 reguliarizacijos metodu.

Palyginus standartinį neuroninį tinklą su mišraus tankio neuroniniu tinklu nustatyta, kad gilaus mišraus tankio neuroninio tinklo su 3 paslėptais neuroniniais sluoksniais prognozavimo rezultatų vidutinė kvadratinė paklaida buvo pati mažiausia. Naudojant šį modelį buvo gauti mažiausi sandorų vykdymo kaštai lyginant su kitais modeliais ir baziniais scenarijais.

Norint įvertinti gilaus neuroninio tinklo prognozavimo rezultatų patikimumą buvo sudaryti trys skirtingi neuroninių tinklų modeliai. Rezultatų patikimumui įvertinti buvo naudojamas prognozuojamų rezultatų standartinis nuokrypis. Šių modelių mažiausias vidutinės kvadratinės paklaidos vidurkis buvo gautas naudojant gilų mišraus tankio neuroninį tinklą. Gauti rezultatai parodė, kad gilaus maišyto tankio neuroninio tinklo prognozuojamas reikšmės labiau atitinka tikrus duomenys ir šio modelio prognozuojamas rezultatų standartinis nuokrypis buvo daug informatyvesnis negu kitų modelių. Nepaisant to, naudojant šį modelį nepavyko aplenkti bazinės strategijos modelio kuomet sandoriai vykdomas limituotu pavedimu strategija.

Literatūros sąrašas

[Ald13] I. Aldridge. High-frequency trading: a practical guide to algorithmic strategies and trading systems. John Wiley & Sons, 2013.

[BB08] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems 20, pages 161–168. MIT Press, Cambridge, MA, 2008.

[BCV13] Y. Bengio, A. Courville, P. Vincent. Representation learning: A review and new perspectives., IEEE Transactions on, 35.8 Pattern Analysis and Machine Intelligence. 1798-1828. 2013.

[Ben09] Y. Bengio. Learning deep architectures for AI. Foundations and trends® in Machine Learning, 2.1: 1-127. 2009.

[BIS07] Bishop C. Pattern Recognition and Machine Learning. Mokomoji knyga. 2007

[Bis94] C. Bishop. Mixture density networks. Neural Computing Research Group Report. 1994

[BL04] L. Bottou and Y. LeCun. Large scale online learning. In S. Thrun, L. Saul, and B. Scholkopf, editors, Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA, 2004.

[BL07] Y. Bengio, Y. LeCun. "Scaling learning algorithms towards AI."Large-scale kernel machines 34.5. 2007.

[BL07] Y. Bengio, Y. LeCun. Scaling learning algorithms towards AI. Large-Scale Kernel Machines, 2007, 34: 1-41.

[BLP+07] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In Advances in Neural Information Processing Systems (NIPS), pp. 153-160. 2007.

[CE11] O. Chapelle, D. Erha. Improved Preconditioner for Hessian Free Optimization. In NIPS Workshop on Deep Learning and Unsupervised Feature Learning. 2011.

[Cho14] K. Cho. Foundations and Advances in Deep Learning. Daktaro disertacija. (2014).

[Cyb89] G. Cybenko. Approximation by superposition of sigmoidal functions.Mathematics of Control, Signals and Systems, 1989, 2.4: 303-314.

[CK12] R. Cont, A. Kukanov. Optimal order placement in limit order markets. arXiv preprint arXiv:1210.1625v1. 2012

[CMG+12] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep big multilayer perceptrons for digit recognition. In G. Montavon, G. Orr, and K.-R. Muller, editors, Neural Networks: Tricks of the Trade, volume 7700 of Lecture Notes in Computer Science, pages 581–598. Springer Berlin Heidelberg, 2012b.

[CMZ12] R. Cesari, M. Marzo, P. Zagaglia. Effective Trade Execution. arXiv preprint arXiv:1206.5324, 2012.

[EGW05] D. ERNST, P. GEURTS, L. WEHENKEL " Tree-based batch mode reinforcement learning". In: Journal of Machine Learning Research. 2005. p. 503-556.

[FAH88] Fahlman, S. E. 1988. Faster-learning variations on back-propagation: An emperical study. in Proceedings, Connectionist Models Summer School, Morgan-Kaufmann, Los Altos CA. 1988.

[GB10] X. Glorot, Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of AISTATS 2010, volume 9, pages 249–256. 2010.

[GBB11] X. Glorot, A. Bordes, .Bengio. "Deep Sparse Rectifier Neural Networks." *Aistats*. Vol. 15. No. 106. 2011.

[GBB11] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In International Conference on Machine Learning (ICML), 2011

[GBB11] X. Glorot, A. Bordes, Y. Bengio. Deep sparse rectifier neural networks.In International Conference on Artificial Intelligence and Statistics (AISTATS). 2011

[GG15] Y. Gal, Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. arXiv preprint arXiv:1506.02142. 2015 Jun 6;2.

[GV07] A. Gelžinis, A. Verikas. Neuroniniai tinklai ir neuroniniai skaičiavimai. Mokomoji knyga; Kauno technologijos universitetas, 2007

[GWM+13] I. J. Goodfellow, D. Warde-Farley, M. Mirza,A. Courville, Y. Bengio. Maxout networks. arXiv preprint arXiv:1302.4389. 2013

[Hin07] G. E. Hinton. Learning multiple layers of representation. Trends In Cognitive Sciences, 2007, 11.10: 428-434.

[Hor91] K. Hornik. Approximation capabilities of multilayer feedforward networks.Neural networks, 1991, 4.2: 251-257.

[HS06] G. Hinton, R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 2006, 313.5786: 504-507.

[HZR+15] K. He, X. Zhang, S. Ren, J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1026-1034), 2015.

[Yin12] C. Yingsaeree. Algorithmic Trading: Model of Execution Probability and Order Placement Strategy. 2012

[JKR+09] K. Jarrett, K. Kavukcuoglu, M. Ranzato, Y. LeCun. What is the best multi-stage architecture for object recognition? In Computer Vision, 2009 IEEE 12th International Conference on, pages 2146–2153. IEEE.

[KSH12] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[KZ15] A. Kercheval, Y. Zhang. Modeling high-frequency limit order book dynamics with support vector machines. Quantitative Finance 15(8), 1315–1329, 2015

[LBB+98] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. Gradient based learning applied to document recognition. In IEEE Procedings. 1998

[LBH15] Y. LeCun, Y. Bengio, G. Hinton. Deep learning. Nature, 2015. 521. 436–444 (28 May 2015).

[Lor08] J. Lorenz. Optimal trading algorithms: Portfolio transactions, multiperiod portfolio selection, and competitive online search. Diss. ETH, 2008

[LR10b] S. Lange, M. Riedmiller. Deep learning of visual control policies. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2010), Brugge, Belgium.2010

[MAR10] J. Martens. Deep learning via Hessian-free optimization. In J. Furnkranz and T. Joachims, editors,Proceedings of the 27th Internation Conference on Machine Learning (ICML 2010), pages 735–742, Haifa, Israel, June 2010.

[MDH12] A. Mohamed, G. Dahl, G. Hinton, G. Acoustic modeling using deep belief networks. Audio, Speech, and Language Processing, IEEE Transactions on, 20(1):14 –22. 2012.

[MED07] V. Medvedev. Tiesioginio Sklidimo Neuroninių Tinklų Taikymo Daugiamačiams Duomenims Vizualizuoti Tyrimai, Daktaro disertacija. 2007.

[Mey11] G. Meyer. CFTC data reveal day traders' role in volatile oil markets. July 5, 2011. [žiūrėta 2014-12-08]. Prieiga per internetą: http://www.ft.com/intl/cms/s/0/b29b2b1e-a743-11e0-b6d4-00144feabdc0.html#axzz3Lqhe8D00

[MH08] L. Van Der Maaten, G. Hinton. Visualizing data using t-SNE. Journal of Machine Learning Research, 2008, 9.2579-2605: 85.

[MHN13] A. Maas, A. Y. Hannun, A. Y. Ng. "Rectifier nonlinearities improve neural network acoustic models." *Proc. ICML*. Vol. 30. No. 1. 2013.

[Nar09] R. Narang. Inside the black box : the simple truth about quantitative trading. Hoboken, N.J., Wiley. 2009.

[NFK06] Y. Nevmyvaka, Y. Feng, M. Kearns. Reinforcement learning for optimized trade execution. In Proceedings of the 23rd international conference on Machine learning (pp. 673-680). ACM. 2006

[Ng04] A. Y. Ng. "Feature selection, L 1 vs. L 2 regularization, and rotational invariance." Proceedings of the twenty-first international conference on Machine learning. ACM, 2004.

[NH10] V. Nair, G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In International Conference on Machine Learning (ICML), 2010. pp. 807-814

[NKP+05] Y. Nevmyvaka, M. Kearns, M. Papandreou, K. Sycara Electronic trading in order-driven markets: Efficient execution. In: E-Commerce Technology, 2005. CEC 2005. Seventh IEEE International Conference on. IEEE, 2005. p. 190-197.

[PAS14] Pascanu R. On Recurrent and Deep Neural Networks, Daktaro disertacija. 2014

[RB93] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In H. Ruspini, editor, Proceedings of the IEEE International Conference on Neural Networks (ICNN), pages 586 – 591, San Francisco, 1993.

[RB93] M. Riedmiller, H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In Proc. of the IEEE Intl. Conf. On Neural Networks, pages 586–591, San Francisco, CA. 1993.

[RHW88] D. E. Rumelhart, G. Hinton, R. J. Williams. In: Neurocomputing: foundations of research. MIT Press, 1988. p. 673-695.

[RMN09] R. Raina, A. Madhavan, and A. Ng. Large-scale deep unsupervised learning using graphics processors. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009), pages 873–880, New York, NY, USA, 2009. ACM.

[ROS58] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological Review, 65(6):386–408, Nov. 1958.

[SB98] R. S. Sutton, A. G. Barto. Reinforcement learning: an introduction. MIT Press. 1998.

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958. 2014

[SW12] S. Stoikov, R. Waeber. Optimal Asset Liquidation Using Limit Order Book Information. Available at SSRN 2113827, 2012

[TP89] D. S. Touretzky and D. A. Pomerleau. What is hidden in the hidden layers? Byte, 14:227–233, 1989.

[TV12] J. Tse, D. Vincent. High Frequency trading – The good, the bad, and the Regulation. Credit Suisse AES Analysis. 2012. [žiūrėta 2013-12-08]. Prieiga per internetą:https://edge.creditsuisse.com/edge/Public/Bulletin/Servefile.aspx?FileID=23284&m=1815212669.