Kauno technologijos universitetas
Mechanikos inžinerijos ir dizaino fakultetas

Gamybos efektyvumo prognozavimo naudojant dirbtinius neuroninius tinklus tyrimas

Baigiamasis magistro projektas

Šarūnas Neimontas
Projekto autorius

Doc. Dr. Marius Rimašauskas
Vadovas

Kaunas, 2019

Kauno technologijos universitetas
Mechanikos inžinerijos ir dizaino fakultetas

Gamybos efektyvumo prognozavimo naudojant dirbtinius neuroninius tinklus tyrimas

Baigiamasis magistro projektas
Gamybos inžinerija (6211EX015)

Šarūnas Neimontas
Projekto autorius

Doc. Dr. Marius Rimašauskas
Vadovas

Prof. Dr. Egidijus Dragašius
Recenzentas

Kaunas, 2019

Kauno technologijos universitetas
Mechanikos inžinerijos ir dizaino fakultetas
Šarūnas Neimontas

Gamybos efektyvumo prognozavimo naudojant dirbtinius neuroninius tinklus tyrimas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Šarūno Neimonto, baigiamasis projektas tema „Gamybos efektyvumo prognozavimo naudojant dirbtinius neuroninius tinklus tyrimas“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka)	(parašas)

Kauno technologijos universitetas

Mechanikos inžinerijos ir dizaino fakultetas

Studijų programa Gamybos inžinerija (6211EX015)

Magistrantūros studijų baigiamojo projekto užduotis

Studentui Šarūnui Neimontui

1. Baigiamojo projekto tema:

   Gamybos efektyvumo prognozavimo naudojant dirbtinius neuroninius tinklus tyrimas
   Research of Manufacturing Efficiency Forecasting Using Artificial Neural Network
   Patvirtinta 2018 m. lapkričio 29 d. dekano potvarkiu Nr. V25-11-14

2. Projekto tikslas ir uždaviniai:

   Šio baigiamojo darbo tikslas – ištirti dirbtinių neuroninių tinklų panaudojimo galimybes gamybos efektyvumo prognozavimui.
   Darbo uždaviniai:
   1. Atlikti analitinę dirbtinių neuroninių tinklų panaudojimo galimybių gamybinėje praktikoje analizę;
   2. Išanalizuoti DNT kūrimo *Matlab* programinėje aplinkoje procesą;
   3. Naudojant realius gamybinės įmonės duomenis, sukurti DNT modelį gamybos efektyvumo prognozavimui;
   4. Atlikti sukurto modelio testavimą ir patikrinimą realiomis sąlygomis

3. Pradiniai projekto duomenys:

   Trylikos mėnesių UAB „Mars Lietuva“ gamybinių linijų efektyvumo duomenys: 3428 gaminamos produkcijos procesų rezultatai iš 6 kintamųjų.

4. Pagrindiniai reikalavimai ir sąlygos:

   Sukurti neuroniniais tinklais grįstą gamybos efektyvumo prognozavimo modelį. Darbas turi būti originalus, atliktas pagal galiojančius reikalavimus taikomus magistro baigiamiesiems darbams.

5. Projekto aprašomosios dalies struktūra:

   Įvadas; Dirbtinių neuroninių tinklų panaudojimo galimybių gamybinėje praktikoje analizė; DNT kūrimo *Matlab* programinėje aplinkoje analizė; DNT modelio pritaikymas gamybos efektyvumo prognozavimui, aprašymas, tobulinimas ir rezultatų patikrinimas; Išvados; Literatūros sąrašas.

6. Grafinės projekto dalies sudėtis: nėra.

7. Projekto konsultantai: UAB „Mars Lietuva“ gamybos vadovas Vincas Augaitis. UAB „Mars Lietuva“ Tiekimo ir logistikos skyriaus vadovė Renata Jurtautė.

Studentas Šarūnas Neimontas
(vardas, pavardė, parašas, data)

Projekto vadovas Doc. Dr. Marius Rimašauskas
(vardas, pavardė, parašas, data)

Krypties studijų programos vadovas Doc. Dr. Regita Bendikienė
(vardas, pavardė, parašas, data)

Turinys

Paveikslų sąrašas

Lentelių sąrašas

Neimontas, Šarūnas. Gamybos efektyvumo prognozavimo naudojant dirbtinius neuroninius tinklus tyrimas. Magistro baigiamasis projektas / vadovas doc. Marius Dr. Rimašauskas; Kauno technologijos universitetas, Mechanikos inžinerijos ir dizaino fakultetas.
Studijų kryptis ir sritis (studijų krypčių grupė): Gamybos inžinerija (E10), Inžinerijos mokslai (E).
Reikšminiai žodžiai: dirbtiniai neuroniniai tinklai, gamyba, prognozavimas, efektyvumas.
Kaunas, 2019. 56 p.

Santrauka

Darbo tikslas – ištirti dirbtinių neuroninių tinklų (toliau DNT) panaudojimo galimybes gamybos efektyvumo prognozavimui.

Atlikus DNT panaudojimo gamybinėje praktikoje analizę, nustatyta, kad plačiausiai dirbtiniai neuroniniai tinklai naudojami optimizavimo, modeliavimas, kokybės kontrolės, vaizdų atpažinimo ir prognozavimo problemų sprendimui. Išnagrinėjus dirbtinio neuroninio tinklo kūrimo *Matlab* programinėje aplinkoje procesą, išskirti ir išanalizuoti pagrindiniai jo etapai: duomenų surinkimas, apdorojimas ir padalinimas į funkcines grupes; tinklo struktūros ir architektūros parinkimas; optimalios DNT mokymo funkcijos pasirinkimas; neuroninio tinklo mokymas, tikrinimas, vertinimas bei tobulinimas. Naudojant realius UAB „Mars Lietuva“ gamybos proceso generuojamus duomenis, sukurtas DNT modelis gamybos efektyvumo prognozavimui pagal kiekvieno fabrike gaminamo produkto rūšį, formatą, gamybos kiekį, pamainą ir gamybos liniją. Atlikti daugiau nei šimto sukurtų įvairių architektūrų modelių testavimas ir patikrinimai, lyginant su realiais UAB „Mars Lietuva“ gamybos duomenimis. Tinklas patikrintas tiesioginio sklidimo ir kaskadinio sklidimo architektūrose su penkiais pagrindiniais Matlab programinės aplinkos algoritmais, lyginant suprognozuotas vertes su realiomis tikslo vertėmis. Įvertinta neuronų skaičiaus paslėptame sluoksnyje svarba. Remiantis rezultatais nustatyta, kad geriausia dirbtinio neuroninio tinklo struktūra gauta naudojant tiesioginio sklidimo DNT su Levenberg-Marquardt (trainlm) algoritmu, septyniais neuronais paslėptame sluoksnyje ir naudojant ankstyvo stabdymo metodiką. Sukurto daugiasluoksnio dirbtinio neuroninio tinklo modelio prognozuojamas procentinis rezultatas individualiai skiriasi nuo realios vertės vidutiniškai 3,17 procento ir yra 54 proc. tikslesnis lyginant su dabar įmonės naudojamu rezultatų imties vidurkiu (paklaida ± 6,9 proc.). Sukurtas modelis ateityje gali būti naudojamas įmonės gamybos procesų tikslesnėms prognozėms.

Neimontas, Šarūnas. Research of Manufacturing Efficiency Forecasting Using Artificial Neural Network. Master's Final Degree Project / supervisor assoc. prof. Dr. Marius Rimašauskas; Faculty of Mechanical Engineering and Design, Kaunas University of Technology.
Study field and area (study field group): Production and Manufacturing Engineering (E10), Engineering Sciences (E).
Keywords: artificial neural network, manufacturing, production, efficiency, forecasting.
Kaunas, 2019. 56 pages.

Summary

The aim of this project was to present the possibility of artificial neural networks (ANN) application for manufacturing efficiency forecasting. By analyzing the nowadays practices of applying ANN in manufacturing processes, five main areas was identified - optimization, modeling/simulation, quality control, image recognition and forecasting. Also the process of artificial neural network creation in the Matlab computing environment was analyzed and four key phases was defined: data collection, processing and division into functional groups; network structure and architecture selection; optimal training function selection; network training, validating, valuating and optimizing. Operating data from manufacturing process of UAB Mars Lietuva used to create ANN driven forecasting model. Using the product type, format, production volume, shift and production line as input data and efficiency index as an output. During the research, more than one hundred neural networks with different structure trained, validated and valuated using actual UAB Mars Lietuva production process data. Fast-forward and cascade-forward networks verified with five main Matlab multilayer neural network training functions. In addition, ANN optimized by finding the optimal number of neurons in a hidden layer and using early-stopping methodology. Based on the results, the best structure for forecasting model was obtained by using fast-forward artificial neural network with Levenberg-Marquardt (trainlm) algorithm, with seven neurons in the hidden layer and using early-stopping. The predicted percentage result of the generated multilayer ANN model differs from the actual value by an average of 3.17 percent and is 54 percent more accurate than the model (± 6,9%) currently used by the company. Company could use the created model in the future for more precise forecasting of production efficiency.

Įvadas

XXI amžiaus pradžioje prasidėjusi ketvirtoji pramonės revoliucija atveria naujas galimybes pramonės ir gamybos įmonėms efektyviau kurti vertę ir mažinti gaminamos produkcijos savikainą. Nuolatinis gamybos procesų tobulinimas ir žmogiškosios darbo jėgos automatizavimas ar pakeitimas robotizuotais įrenginiais nebėra vienintelės kryptys, nulemiančios įmonės ekonominį pranašumą rinkoje. Ketvirtoji pramonės revoliucija yra kuriama ant trečiosios skaitmeninės revoliucijos pamatų, kuri jau užpildė pasaulį kompiuteriais ir automatizavo duomenų kaupimą, todėl tikslingai apdorojus ir interpretavus teisingai sukauptą informaciją galima kurti sistemas, leidžiančias dirbtiniu intelektu valdyti kasdieninius įmonių procesus, taip ne tik taupant fizinio ir intelektinio darbo resursus, bet ir įgalinant atlikti tai efektyviau, tiksliau bei objektyviau. Viena iš pažangiausių dirbtinio intelekto atmainų – biologinę nervų sistemą imituojantys kompiuterinėje aplinkoje kuriami dirbtiniai neuroniniai tinklai (toliau DNT).

Šio baigiamojo darbo tikslas – ištirti dirbtinių neuroninių tinklų panaudojimo galimybes gamybos efektyvumo prognozavimui.

Darbo uždaviniai:

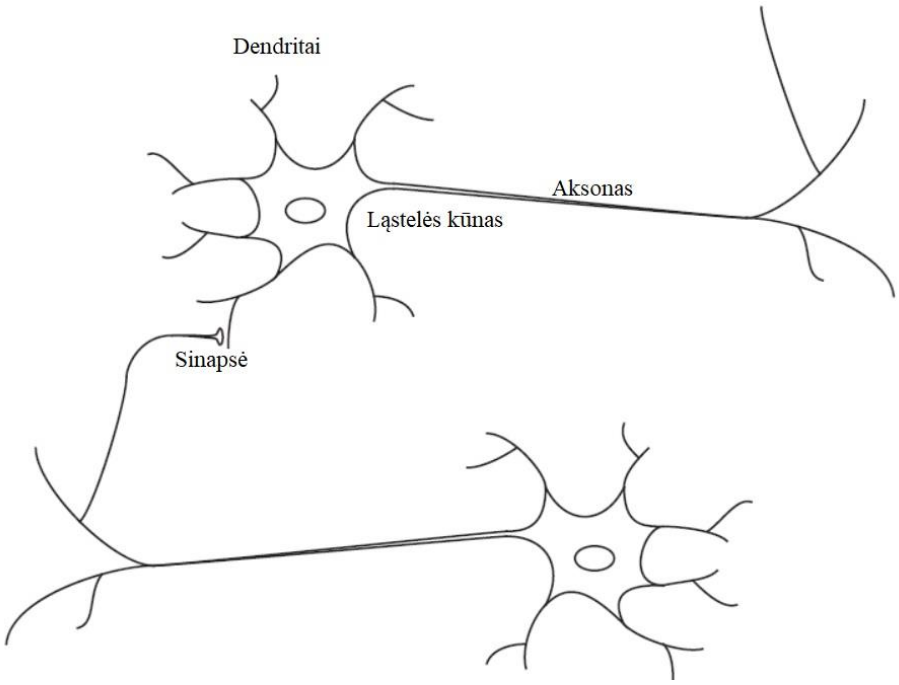
1. Atlikti dirbtinių neuroninių tinklų panaudojimo galimybių gamybinėje praktikoje analizę;
2. Išanalizuoti DNT kūrimo *Matlab* programinėje aplinkoje procesą;
3. Naudojant realius gamybinės įmonės duomenis, sukurti DNT modelį gamybos efektyvumo prognozavimui;
4. Atlikti sukurto modelio testavimą ir patikrinimą realiomis sąlygomis.

1. Neuroninių tinklų koncepcija

Dirbtinis neuroninis tinklas yra tarpusavyje susietų apdorojimo elementų, vienetų ar mazgų junginys, kurio funkcionavimas paremtas supaprastintais žmogaus neuronų veikimo principais. Tinklo potencialas generuojamas pagal jungčių tarp vienetų ryšio stiprumą ar svorius, gaunamus adaptavimo arba mokymo procesų metu pagal modelio mokymo duomenų rinkinį.

1.1. Biologiniai neuroniniai tinklai

Žmogaus smegenys sudarytos iš 10^{11} vienetų nervinių ląstelių, dar vadinamų neuronais. Neuronai tarpusavyje komunikuoja elektriniais signalais, kurie yra trumpalaikiai impulsai ląstelės sienelės arba membranos įtampoje. Tarpneuroninės jungtys yra komunikuojamos elektrocheminiais junginiais, sinapsiais, kurie randasi ląstelių šakose, dar vadinamose dendritais. Kontaktinis taškas tarp vienos ląstelės aksono ir kitos ląstelės dendrito vadinamas sinapse (angl. synapse). Kiekvienas elementas paprastai turi apie 10^4 jungčių su kitais neuronais, todėl nuolat gauna daugybę įeinančių signalų, kurie galiausiai dandritais pasiekia ląstelės kūną [1]. Čia signalai suintegruojami tarpusavyje ir jeigu gautas suminis signalas viršija numatytą slenkstį, neuronas yra inicijuojamas, tai yra sugeneruojamas naujas atsakomasis įtampos mikroimpulsas (1 pav.). Sugeneruotas impulsas, vientisų neuronų elementų, aksonų (dar vadinamų neuritais) tinklu siunčiamas toliau kitiems neuronams. Nuo neuronų išdėstymo ir individualių sinapsių pajėgumo, kuris nustatomas sudėtingo cheminio proceso metu, priklauso neuroninio tinklo funkcija [2].



1 pav. Dviejų susietų biologinių neuronų schema [2]

Nors neuromokslas tik pradeda suvokti, kaip veikia biologiniai neuroniniai tinklai, aišku, kad visos biologinės funkcijos, o taip pat ir atminties, yra atliekamos neuronais ir jungtimis tarp jų, o mokymasis suvokiamas kaip naujų jungčių diegimas tarp neuronų arba jau esamų jungčių modifikavimas.

1.2. Dirbtiniai neuroniniai tinklai

Nors dirbtiniai neuroniniai tinklai neprilygsta biologiniams savo kompleksija ir sudėtingumu, galima įvardinti du pagrindinius panašumus: abiejų lyginamų tinklų pagrindiniai elementai yra atskiri skaičiavimo vienetai (neuronai), tampriai susieti tarpusavyje į vieną visumą, o jungtys tarp neuronų nusako viso tinklo funkciją. Dažniausiai DNT naudojami didžiulės apimties duomenų statistinei analizei ir duomenų modeliavimui, kaip alternatyva standartinei netiesinei regresijai ar tradicinėms klasterių analizės technikoms, todėl įprastai yra aktualiausi prognozavimo ir klasifikavimo problemų sprendimuose [1]. Kaip teigia Smith ir Gupta [3], dirbtiniai neuroniniai tinklai yra funkcijos aproksimacijos įrankis, vertinantis sąryšį tarp nepriklausomų ir priklausomų kintamųjų. Principinis skirtumas tarp DNT ir tradicinių statistinių metodikų yra toks, kad neuroninių tinklų metodai nepateikia besąlygiškų prielaidų apie statistinį duomenų paskirstymą ar savybes, o ieško funkcinės priklausomybės netiesiniu ir neparametriniu būdu. Todėl praktinėse situacijose teisingai sumodeliuoti dirbtiniai neuroniniai tinklai yra labiau universalūs. Kitaip sakant, DNT metodikai būdingas netiesinis problemų sprendimo būdas, todėl galima tikėtis itin tikslių rezultatų, ypač operuojant kompleksiškomis ir itin didelėmis duomenų imtimis.

1.2.1. Dirbtiniai neuroniniai tinklai pagal mokymo tipą

DNT pagal mokymo algoritmą skirstomi į tris rūšis:

- mokymo su mokytoju (angl. Supervised learning);
- mokymo be mokytojo (angl. Unsupervised learning);
- Hibridinis mokymas (angl. Semi-supervisedvlearning).

Inžinerinėje praktikoje populiaresnis yra mokymosi su mokytoju būdas. Šiuo atveju duomenų aibėje aiškiai nustatomos įvesties ir išvesties reikšmės, nenurodant ir net nežinant priklausomybės tarp jų. DNT mokydamasis modeliuoja priklausomybės funkciją tarp įvesties ir išvesties kintamųjų. Tinklo mokymosi eiga ir potencialas nuolat tikrinami tarp generuojamų naujų prognozuojamų reikšmių ir validacijos bei tikrinimo duomenų, atitinkamai ieškant optimalių slenksčių (angl. bias) ir svorių (angl. weights). Kartu mokymosi procesas nuolat vertinamas pagal numatytus parametrus, kol neuroninio tinklo rezultato klaida pasiekia norimą ar artimą reikšmę. Dažniausiai mokant daugiasluoksnį tinklą

naudojamas klaidos sklidimo atgal algoritmas (angl. error back propagation). Mokymosi su mokytoju DNT itin naudingi prognozavimo, vaizdo atpažinimo ir tendencijų nustatymui.

Dirbtiniai neuroniniai tinklai, mokomi be mokytojo, naudoja tiktai įvesties duomenų rinkinius ir pats tinklas mokymo eigoje modifikuojamas be išvesties reikšmių. Itin svarbu teisingai parinkti matavimo ir tikrinimo reikšmes, kuriomis bus vertinamas tinklo potencialas ir įsitikinama, kaip tinklas teisingai atlieka mokymosi funkciją. Dažniausiai neuroniniai tinklai, mokomi be mokytojo, naudojami didelės apimties duomenų pirminiam apdorojimui, klasifikavimui, klasterizavimui ir optimizavimui.

Kombinacinis (tiek prižiūrimas su mokytoju, tiek ir neprižiūrimas) mokymasis, dar vadinamas mišriu, apjungia apibrėžtus ir neapibrėžtus tikslo duomenis, siekiant kuo tikslesnio galutinio modelio [4]. Dažniausiai toks tinklo tipas pasirenkamas, kai naudojama daugiau neįvardintų, nei įvardintų duomenų rinkinių.

1.2.2. Vienasluoksniai ir daugiasluoksniai neuroniniai tinklai

Pagal savo struktūrą neuroniniai tinklai skirstomi į vienasluoksnius (angl. single–layer) ir daugiasluoksnius tinklus. Vienasluoksniai – tai viena pirmųjų primityviausios architektūros neuroninių tinklų formų, dar vadinamų parceptronais (angl. – parceptron). Vienasluoksniai neuroniniai tinklai klasifikuojami kaip viena iš tiesioginio sklidimo (angl. – feedforward) DNT rūšių, kadangi informacija juose juda tik viena kryptimi – nuo įvesčių į išvestis. Šis tinklas sudarytas iš vieno įvesties mazgų sluoksnio, kuris siunčia jau įvertintas įvestis į sekantį priimančių mazgų sluoksnį, o kai kuriais atvejais tik į vieną priimantį mazgą. Kadangi parceptronas pagal savo loginį algoritmą arba siųs tolesnį signalą, arba ne, toks loginis modelis tinka klasifikavimo uždavinių sprendimui. Kitaip tariant, parceptronas generuos du išėjimus TAIP (+1) arba NE (-1).

Kompleksiškesnėms užduotims spręsti reikalingi daug sudėtingesnės struktūros, daugiau nei vieną sluoksnį turintys, dirbtiniai neuroniniai tinklai. Įprastai jie sudaryti iš bent vieno įvesties sluoksnio, kuris siunčia nustatyta funkcija įvertintas įvestis į sekančius paslėptus sluoksnius ir galiausiai į išvesties sluoksnį pabaigoje [5]. Kuo daugiau tinklas turi paslėptų sluoksnių ir kuo daugiau neuronų randasi juose, tuo tinklas yra sudėtingesnis, o kartu yra imlesnis kalkuliacijų skaičiui.

Tačiau didelė tinklo architektūra neužtikrina jo funkcionalumo, kadangi dažnai tinklas pasidaro per daug adaptyvus ir nepajėgus improvizuoti su naujomis įvestimis. Įrodyta, kad ir dviejų sluoksnių DNT gali aproksimuoti bet kurią tolydžią funkciją. Viena populiariausių daugiasluoksnio tinklo struktūrų – vienas po kito sekantys sluoksniai, kurių neuronai tarpusavyje yra susiję be jokių papildomų jungčių. Tokie tinklai paprasčiau analizuojami ir lengviau modeliuojami nei bendri tiesioginio sklidimo dirbtiniai neuroniniai tinklai.

1.3. Dirbtinių neuroninių tinklų praktinis panaudojimas

Dirbtiniai neuronai yra itin supaprastintos biologinių neuronų abstrakcijos, todėl ir DNT nėra įgalūs tiesiogiai imituoti žmogaus smegenų veiklos, tačiau funkcinis panaudojimas ir potencialas yra labai platus. Svarbiausios DNT panaudojimo sritys ir sprendžiamų problemų tipai:

- Funkcijos aproksimacija – skirtingai nuo daugelio statistinių metodų, atvaizduojant kelias įvestis į vieną pageidaujamą išvestį, atliekama adaptyviu modeliu laisvu parametrų interpretavimu;
- Modelio atpažinimas (angl. pattern association) – DNT yra labai efektyvus greitai atpažįstant, identifikuojant ir klasifikuojant garso, vaizdo ir video elementus. Dažnai neuroniniai tinklai yra pajėgūs šiuos veiksmus atlikti be išankstinio modelio apibrėžimo ir klasifikavimo nusakymo, kadangi DNT pats išmoksta identifikuoti visiškai naujas modelio tendencijas;
- Asociatyvi atmintis – tai tinkamiausio modelio parinkimas problemos sprendimui, turint tik dalinį duomenų rinkinį. Dažniausiai naudojami itin sudėtingos struktūros neuroniniai tinklai, sudaryti iš dinaminių tarpusavyje sąveikaujančių neuronų;
- Naujų išmaniųjų modelių karta (angl. meaningful patterns) – viena naujausių DNT mokslo krypčių, kur tiriamos itin sudėtingos neuroninės struktūros, galinčios demonstruoti pradinius kūrybiškumo elementus [6];
- Kitos panaudojimo sritys.

Populiariausios neuroninių tinklų panaudojimo sferos [2]:

- Aviacija, kosmoso ir karinė pramonė (orlaivių autopilotų, skrydžio kelio simuliacijos ir jų tobulinimas, orlaivių valdymo sistemų ir komponentų modeliavimas, gedimų detektorių projektavimas);
- Automobilių pramonė **(**automatinio valdymo ir navigacijos sistemos, kuro įpurškimo kontrolė, automatinės stabdymo sistemos, gedimų identifikavimas, virtualūs išmetamųjų dujų jutikliai, garantinio aptarnavimo analizė);
- Bankinis sektorius **(**dokumentų automatiniai skaitytuvai, kredito paraiškų vertinimas, grynųjų pinigų prognozavimas, įmonių kvalifikavimas, valiutų kursų prognozavimas, rizikos vertinimas ir t.t.);
- Gynybos sektorius **(**ginklų nukreipimas, taikinių ar objektų identifikavimas, veido atpažinimas, vaizdinio signalo apdorojimas, garso slopinimas, naujos kartos sensoriai, sonarai, radarai);
- Elektronika **(**kodų sekos prognozavimas, integruotos grandinės mikroschemų išdėstymas ir jų gedimo analizė, robotizuotas matymas, balso sintezė); finansų sektorius (nekilnojamo turto,

paskolų ir užstato vertinimas, įmonių obligacijų reitingavimas, kredito linijos analizė, fondų prekybos programos, įmonių finansinė analizė, akcijų rinkų analizė ir prognozė);

- Draudimo sektorius (draudimo produktų optimizavimas, polisų išdavimo politikos tobulinimas); medicina (vėžinių ląstelių analizė; protezų dizainas ir tobulinimas; transplantacijos laikų optimizacija; ligoninių išteklių panaudojimo optimizacija);
- Naftos ir gamtinių dujų pramonė (paieškos ir gavybos procesų bei įrankių tobulinimas, seisminių duomenų interpretavimas, telkinių priežiūros sprendimai);
- Telekomunikacijų pramonė (vaizdo ir garo realių laikų kompresija, automatizuotų informacijos suteikimo paslaugos, užsienio kalbų vertimas realiu laiku, klientų atsiskaitymo sprendimai);
- Logistika (navigacijos sistemos, optimalaus maršruto parinkimas, transporto grafiko sudarymas) ir t.t.

1.4. Dirbtinių neuroninių tinklų taikymas gamybos inžinerijos problemų sprendimams

Dėl išaugusios gaminių asortimento įvairovės, vartotojų poreikių ir sutrumpėjusio produkto gyvavimo ciklo, postindustrinėje aplinkoje gamybos technologijų ir pramonės sistemų pokyčiai vyksta nuolat ir itin mažais intervalais.

Anot Hu ir An [7], dėl imlumo laikui ir piniginiams kaštams, tradiciniai gamybos modeliavimo metodai ir simuliavimo sistemos tampa neįgalūs laiku prisitaikyti prie nuolatinių pokyčių ir vis labiau praranda savo pirminę prasmę, todėl reikalingas naujas požiūris į gamybos procesų organizavimą, kontrolę ir planavimą, tiek trumpuoju, tiek ir ilguoju periodu.

Šiame darbe didžiausias dėmesys skiriamas dirbtinių neuroninių tinklų panaudojimo galimybių tyrimui gamybos ir pramonės aplinkoje, ypač didelį dėmesį skiriant prognozavimo sistemų ir modelių taikymui praktikoje ir jų tobulinimui.

Remiantis 1.3 šio darbo skyriuje aprašytais populiariausiais DNT sprendžiamų problemų tipais, išskirtos ir aptartos šios aktualiausios gamybinės industrijos sritys, kuriose dirbtiniai neuroniniai tinklai yra sėkmingai naudojami praktikoje, parodė potencialą ar galbūt bus naudojamos ateityje:

- Optimizavimo uždaviniai;
- Modeliavimas;
- Kokybės kontrolė;
- Vaizdų atpažinimas;
- Prognozavimas.

1.4.1. Optimizavimo uždaviniai

Sprendžiant optimizavimo uždavinius, siekiama apibrėžtoje aibėje surasti elementą, kuriam kriterijaus reikšmė būtų arba minimali, arba maksimali. Inžinerinėje praktikoje įvairūs optimizacijos metodai yra itin efektyvūs įrankiai, tačiau tradiciniai optimizacijos metodai (pavyzdžiui parametriniai tyrimai, simuliacijomis paremti optimizacijos metodai) yra brangūs ir daug laiko sąnaudų reikalaujantys uždaviniai.

Siekiant nustatyti optimalų aušinimo vandeniu kiekį ir kampą puslaidininkių medžiagų litavimo procese, dėl itin didelio panašumo į klasikinį puslaidininkių litavimo lydinį (Pb-Sn), buvo ištirtas bismuto – sidabro (Bi-Ag) lydinys [8]. Pradiniai duomenys surinkti atsižvelgiant į senėjimo laiko variacijos ir skirtingas sidabro svorio procentines dalis lydmetalyje, o jo kontakto kampas su varine (Cu) plokšte pamatuotas optiniu mikroskopu. Tada pagal apdorotus pradinių duomenų rinkinius, panaudojant dirbtinį neuroninį tinklą, simuliacijoms sukurtas skaitmeninis modelis. Tyrimo rezultatai ir simuliacijos nustatė, kad eksperimento metu determinacijos koeficientas (R^2) siekė apie 0,97, kas suponuoja, jog šis DNT optimizacijos modelis buvo sukurtas tinkamai ir yra patikimas tolesniam naudojimui.

Tiriant duonmedžio krakmolo hidrolizatus, kaip anglies šaltinį bioetanolio gamyboje, siekiant maksimaliai optimizuoti fermentacijos procesą, nustatyta, kad mielės buvo pajėgios utilizuoti hidrolizatus tiek su, tiek ir be maistinių medžiagų papildų, o didžiausia įprasta bioetanolio išeiga atitinkamai – 3.96 ir 3.60 proc. tūrio dalyje po 24 val. fermentacijos [9]. Statistiškai įprastas kvadratinės regresijos modelis ($p < 0,05$) buvo sukurtas bioetanolio išeigai prognozuoti, o panaudojus paviršiaus atsako metodologiją (angl. Response Surface Methodology), optimalios sąlygų vertės bioetanolio gamyboje apskaičiuotos – duonmedžio krakmolo hidrolizatų koncentracija 134.81 g L^{-1}, 21.33 val. laikui ir 5.01 pH., prognozuojant 3,95 proc. tūrio dalyje. Ištestavus įvairias dirbtino neuroninio tinklo architektūras, daugiasluoksnis įprastas tiesioginio sklidimo inkrementinis grįžtamojo perdavimo tinklas su hiperboline tangento funkcija parodė geriausią rezultatą. DNT apskaičiuotos optimaliausios sąlygų reikšmės: duonmedžio krakmolo hidrolizatų koncentracija – 120 g L^{-1}, laikas – 24 val ir 4.5 pH, prognozuojant 4,21 proc. tūrio dalyje bioetanolio išeigą. Prognozuojama bioetanolio išeiga buvo patikrinta eksperimentiškai. Gauta 4,10 proc. tūrio dalis paviršiaus atsako metodologijai ir 4,22 proc. – DNT metodui. Nustatymo koeficientas (R^2) ir absoliutus vidutinis nuokrypis atitinkamai 1 ir 0,09 proc. DNT modeliui, ir 0,9882 bei 1,67 proc. paviršiaus atsako metodui, tad patvirtinta, kad praktikoje pastarasis ženkliai nusileido dirbtinio neuroninio tinklo metodologijai.

Kita svarbi DNT panaudojimo optimizavimo problemų sprendimams sfera yra bendriniai gamybos valdymo ir vadybos procesai: įrenginių ar darbuotojų darbo laiko grafikų optimizavimas, kompleksinių projektų ir daugiaprocesinių užduočių planavimas, užsakymo dydžio prognozavimas [3].

1.4.2. Surogatiniai modeliai

Daugelis gamybos procesų reikalauja preciziškai tikslaus parametrizavimo, siekiant optimalių veiklos kaštų, kokybės ir laiko prasme. Vertinamo proceso parametrų skaičius gali svyruoti nuo keleto iki kelių šimtų, todėl pavyzdžiui dizaino ar parametrų optimizavimo problemos inžinerijoje yra daugiadimensinės ir netiesinės.

Įprastas gamybos procesų parametrų optimizavimas reikalauja daugybės brangių bandomųjų eksperimentų ir vertinamųjų skaičiavimų. Kai kuriais atvejais naudojamas fiziškai tikslaus modelio simuliavimas (pavyzdžiui paremtas baigtinių elementų metodu ar skysčių dinamikos skaičiavimais), galintis pakeisti faktinį eksperimentą ir numatyti fizinį elgesį itin tiksliai. Tačiau toks modeliavimo metodas dažnai yra itin brangus dėl imlumo laiko resursams.

Vienas iš būdų sumažinti bandomųjų eksperimentų (simuliacijų) skaičių – modeliavimui naudoti surogatiniais modeliais paremtą optimizavimo metodą. Surogatiniai modeliai (dar vadinami metamodeliais) yra supaprastintos, todėl lengviau apskaičiuojamos, itin tikslios vertinamų struktūrų aproksimacijos. Įprasti surogatiniai modeliai sudaromi iš polinomų, splainų, stochastinių procesų, radialinių bazinių funkcijų ar dirbtinių neuroninių tinklų. Taip atrenkamas optimalus galimas sprendimas, ženkliai sumažinus skaičiavimo kaštus. Kadangi surogatinio modelio skaičiavimuose naudojamas slenksčio koeficientas, gautų sprendinių spektre gali nebūti realiai optimalaus sprendinio, todėl surogatais paremtos optimizacijos tikslas – eliminuoti surogatinio modelio slenksčio paklaidą, pakartotinai tobulinant modelį naujais eksperimentiniais (simuliaciniais) rezultatais prieš tai pasirinktiems sprendimams.

Šiuolaikinės gamybos procesuose naudojama modeliu pagrįsta savioptimizacija remiasi gamybos įrenginiais, aprūpintais galimybe einamuoju momentu automatiškai adaptuoti optimalius parametrus, pagal proceso eigoje kintančius išorinius veiksnius ir sąveikas. Surogatais paremtoje optimizacijoje negalima proceso susieti su statišku optimizavimo proceso modeliu, kai užfiksuoti galutiniai išeities parametrai, nes čia procesas yra iteratyviai tobulinamas naujais stebėjimais ir jų rezultatais.

Parametriniai modeliai turi fiksuotą parametrų kiekį, o neparametriniai modeliai, pavyzdžiui interpoliaciniai metodai, neturi fiksuoto parametrų skaičiaus. Nors dažnai dirbtiniai neuroniniai tinklai turi konkretų ir tiksliai apibrėžiamą parametrų skaičių, jie yra taip pat priskiriami prie

neparametrizuotų modelių, kadangi (ypač gilaus mokymosi neuroniniai tinklai) gali būti sudaryti iš daugiatūkstantinės neuronų imties ir, prieš atliekant tinklo mokymą, parametrai, reglamentuojantys kiekvieno neurono elgesį ir jų santykius, neturi išankstinės tiesioginės reikšmės procesui, kurį jie imituos.

Kompozitinių medžiagų tekstilės pramonėje surogatinių modelių sukūrimui buvo panaudoti gilaus mokymo dirbtiniai tinklai. Suformuluota daugiau nei 50 dimensijų tęstiniu pluoštu sustiprintos plastmasės gamybos proceso duomenų imtis [10]. Siekiant gerinti modelio tikslumą šalia norimo optimalaus sprendinio, galimas sprendimas patikrintas baigtinių elementų simuliacijomis ir pridėtas prie tinklo mokymo duomenų. Poslinkio kampo prognozavimas dirbtiniais neuroniniais tinklais daugiau nei 24 tūkstančiams kompozitinės medžiagos elementų buvo ženkliai tikslesnis nei įprasti modeliai. Užteko dvidešimties pirminių DNT surogatinio modelio patobulinimų, kol buvo apskaičiuota parametrų kombinacija, pralenkianti prieš tai buvusį geriausią tiesioginės optimizacijos sprendinį.

Norint tobulinti gamybinius procesus ir sistemas, kurios jau yra gana efektyvios, teisinga duomenų analizė suteikia konkurencinį pranašumą. Tiriant DNT surogatinių modelių panaudojimo galimybes metalo gręžimo procesuose, sukurtos surogatinių modelių specifikacijos ir algoritmas, nusakantis neuroninio tinklo manipuliacijas, prognozavimo modelių generavimui [11]. Algoritmas iliustruoja, kaip neuroninis tinklas, sukurtas surogatinio modelio pagrindu, yra naudojamas kaip įvestis su duomenimis standartinių ar neutralių formatų prognozavimo modelių generavimui. Todėl toks algoritmo naudojimas, supaprastina DNT surogatinio modelio panaudojimą kasdieninėms prognozavimo problemoms spręsti ir nereikalauja itin gilių duomenų mokslo žinių.

Atliekant lyginamuosius tyrimus tarp įprastų *software-in-the-loop* optimizavimo metodų (dalelių virpėjimo, skruzdžių kolonijos) ir tarp DNT surogatiniais modeliais pagrįstų metodų, nustatyta, kad pastatų optimizavimas naudojant metamodelius sugeneruoja optimalų pastato dizaino projektą ir suponuoja potencialius energijos taupymo sprendimus skirtingose klimato zonose ženkliai mažesniais skaičiavimo kaštais [12]. Be to, optimizavimo rezultatų analizė suteikė architektams geresnį suvokimą apie pastatų dizaino kintamųjų opcijų optimalias vertes skirtingomis klimato sąlygomis ir padėjo kurti būsimus statybos projektavimo ruošinius, skirtus didelio masto architektūros sprendimams.

1.4.3. Kokybės kontrolė

Kokybės kontrolė gamybos procese yra kertinė dalis, užtikrinanti aukštą gaminamos produkcijos kokybę. Pagal pažangios gamybos paradigmą, efektyvus proceso būklės stebėsena yra pirminė

kokybės vadybos ir kontrolės sąlyga, leidžianti laiku identifikuoti ir eliminuoti netinkamo gamybos proceso pasekmes ir įgalina proceso stabilizavimą bei grąžiną jį į numatytas sklaidos ribas.

Pagal gamybos proceso pobūdį galima išskirti daugybę kokybinių charakteristikų, kurios privalo būti nenutrūkstamai kontroliuojamos. Statistiniai procesų kontrolės (toliau SPK) metodai pripažįstami kaip vieni svarbiausių šių charakteristikų nuolatinio stebėsenos įrankių, tačiau automatizuoto SPK efektyvumas yra tiesiogiai priklausomas nuo statistinių prielaidų ir savarankiškai nėra pajėgus išmatuoti bei vizualizuoti itin platų realaus proceso būsenos duomenų spektrą. Todėl, atsižvelgiant į klaidų spektro (triukšmo) tolerancijos ribas, rekomenduojama statistinę procesų kontrolę kombinuoti su dirbtiniais neuroniniais tinklais.

Pramonėje dažniausiai naudojami neprižiūrimo mokymo saviorganizuojantys neuroniniai tinklai (angl. Self Organizing Map), kurie generuoja topologiškai atpažįstamas kartografijas iš daugybės dimensijų įvesčių erdvės į dviejų dimensijų atvaizdavimo erdvės žemėlapį [13]. Šis darinys sudarytas iš neuronų ir ryšių tinklo, tarp kurių nusakoma bendra topologija (dažniausiai iš tinklo ar grandinės). Tinklo mazgai inicializuojami atsitiktinėmis reikšmėmis daugiadimensinėje duomenų erdvėje ir yra pritaikomi prie faktinio duomenų kolektoriaus pagal mokymosi taisyklę suformuota lygtimi [14]. Tai leidžia ženkliai sustiprinti ir pagreitinti konvergenciją bei suteikia galimybę vizualizuoti mokymo procesą taip, kad būtų galima papildomai įvertinti mokymo duomenų struktūrą. Be to, taip įgalinamas nuolatinis statistinis mokymasis gamybos proceso kontrolėje, siekiant išvengti nuokrypių, o adaptyvus saviorganizuojantis neuroninio tinklo algoritmas tampa smarkiai pranašesnis prieš tradicinius neuroninius tinklus.

Nustatant optimalias įrenginių parametrų vertes, minimizuojančias kokybinius nuokrypius, tai yra gaminamų detalių paviršių nelygumus, sukurti prognozavimo ir optimizavimo modeliai, apjungiant dirbtinius neuroninius tinklus kartu su genetiniu algoritmu, kaip alternatyva tradiciniams metodams [15]. Paraleliai, sukurto modelio patikrinimui, buvo atlikti mašininiai mechaninio apdirbimo eksperimentai. Suprognozuoti kokybiniai parametrai pasirodė esantys arti eksperimentinių duomenų. Paskaičiuota vidutinė santykinė 4,11 proc. paklaida (angl. mean relative error) įrodė sukurto modelio pranašumą prieš regresijos ir neapibrėžtumo logikos (angl. fuzzy logic) modelius, taip pat atitiko iškeltus proceso reikalavimus.

1.4.4. Vaizdų atpažinimas

Gamyboje vaizdų atpažinimo neuroniniais tinklais funkcija yra tampriai susijusi ir su kokybės kontrolės procesais. Panaudojimo sritis labai plati – visos pramonės šakos, reikalaujančios kruopštaus kokybės tikrinimo procesų, turi siekių panaudoti greitesnį, efektyvesnį ir tikslesnį nuokrypių fiksavimą vadinamu mašininiu matymu, lyginant su imlia klaidoms žmogiškąja darbo jėga. Mašininis

matymas yra puikus įrankis, leidžiantis tikrinti įvairių dydžių ir struktūrų gaminius, tokius kaip tekstilės, vaisių ar daržovių, spausdintinių plokščių, elektroninių komponentų, etikečių, integrinių grandynų, įrengimų įrankių ir t.t.

Terminių saugiklių gamyboje, pritaikius išmaniąsias sistemas, inkorporuojant mašininį matymą su dirbtiniu neuroniniu tinklu, sukurtas ir aprašytas efektyvus automatizuotas inspekcijos procesas, skirtas keturių pagrindinių kokybinių defektų fiksavimui [16]. Klaidos sklidimo atgal neuroninis tinklas ir mokymosi vektoriaus kvantavimo efektyvumo rezultatai sulyginti kampo defekto (angl. bur defect) nustatyme. Daugybė defektų pavyzdžių ištirti gamybos proceso metu ir panaudoti siekiant įsitikinti siūlomos sistemos veiksmingumu. Tyrime aprašyta kokybės užtikrinimo sistema ne tik efektyviai naudojama darbe aprašomoje kompanijoje, tačiau ir pakeitė šešis kokybės patikrinimo darbuotojus. Tokiu būdu procesas ne tik užtikrino geresnius kokybės procesus terminių saugiklių gamyboje, bet kartu sumažino tiesioginius gamybos kaštus.

Atliekant tyrimą tekstilės pramonėje nustatyta, kad automatinė tekstilės siūlų kokybės kontrolė pagerino visus procese standartizuotus kokybės rodiklius, be to, sumažino gamybos kaštus [14]. Sistema buvo sudaryta iš atitinkamo atvaizdo priėmimo įrenginio, algoritmo, leidžiančio identifikuoti siūlą, įrenginio reikšmių nustatymo funkcijos ir save organizuojančio neuroninio tinklo, skirto identifikuoto objekto klasifikacijai. Sistemos rezultatai įvertinti klasifikuojant siūlų mėginius, kurie yra naudojami tekstilės gamybos industrijoje, reguliuojant siuvimo mašinų parametrus. Nustatyta, kad net ir kelios, tačiau teisingai nustatytos funkcijos, pasiekia gerus klasifikacijos rezultatus. Klasifikavimo norma sudarė per 80 proc. teisingai įvardintų klasifikavimo klasių, o tai buvo neblogesnis rezultatas nei pasiekiamas įprastos darbuotojų ekspertizės metu, kur paklaida atsiranda dėl subjektyvaus nesutarimo, vertinant skirtingus defektus. Be to, klasifikavimas užtruko apie vieną sekundę, naudojant standartinę kompiuterinę įrangą, o individualus eksperto vertinimas užtrunka apie 30 sekundžių.

Nagrinėjant lazerinės miltelių sukepimo technologijos priedų gamybą, nustatyta kritinė savalaikio proceso defektų identifikavimo svarba [17]. Daugelis šių defektų atsiranda sąveikoje tarp padavimo stūmoklio, kuris išstumia reikalingą miltelių kiekį, ir jau padengto sluoksnio [18]. Todėl nustatyta, kad gilaus mokymosi konvoliucinis neuroninis tinklas (toliau KNT) ypač efektyvus autonominiam daugelio kokybės nuokrypių fiksavimui ir klasifikavimui. Įvesties KNT sluoksnis buvo modifikuotas taip, kad įgalintų algoritmą mokytis miltelių sluoksnio anomalijų išvaizdos, o kartu ir pagrindinės kontekstinės informacijos daugybės dydžių skalėse. Be to, šios konvoliucinio neuroninio tinklo architektūros modifikacijos ženkliai pagerino tinklo lankstumą, bendrą algoritmo klasifikavimo tikslumą ir kompensavo paklaidas, atsirandančias dėl žmogiškojo faktoriaus.

1.5. Prognozavimas dirbtiniais neuroniniais tinklais

Prognozavimas – ateities reiškinių ir rodiklių dydžių numatymo būdas. Prognozuojant taikomi įvairūs metodai, modeliavimas, analogija, ekstrapoliacija ir interpoliacija, aiškinimas, interpretacija, ekspertų apklausa ir kt. [19]. Prognozuojant pateikiama unikalių problemų rinkiniai mokymo sistemai, kai viena iš išvesčių ar išvesčių rinkinių yra nežinomi [20]. Yra daugybė įvairių prognozavimo metodų ir autoriai klasifikuoja juos skirtingai, bet dažniausiai siūlomas prognozavimo jie yra skirstomi į dvi pagrindines grupes: kokybinius ir kiekybinius. Kokybiniai metodai, dar vadinami intuityviu prognozavimu, tai subjektyvūs modeliai, besiremiantys nepamatuojama informacija (apklaustųjų ekspertine nuomone, nuotaika, nuojauta ir t.t.) ir naudojami, kai nėra realios galimybė naudoti kiekybinius metodus.

Kiekybiniai prognozavimo metodai, dar vadinami moksliniais, naudojami, kai buvusios tendencijos turi potencialo tęstis ir atsikartoti ateityje, kai yra surinkta pakankamai duomenų statistiškai įrodytoms koreliacijoms, priklausomybėms ir tendencijoms įvardinti. Tai daug objektyvesni metodai lyginant su kokybiniu prognozavimu, kadangi leidžia išvengti nuokrypių ir paklaidų dėl šališkumo apraiškų. Kiekybiniai prognozavimo metodai yra skirstomi į laiko eilutės metodus ir priežastinius (regresinius) metodus. Prie pastarųjų priskiriami prognozavimo dirbtiniais neuroniniais tinklais metodas [21].

Tyrimai sunkiojoje pramonėje įrodė praktinę dirbtinių neuroninių tinklų metodo naudą ir prognozavimo galimybių potencialą, kai pavyko sukurti itin tikslius produkto kokybės prognozės modelius sudėtingiems metalo liejimo procesams [22]. Suprognozuoti kokybės modeliai leido priimti objektyviau pagrįstus sprendimus, atnaujinant ir renovuojant stambius metalurgijos pramonės objektus. Iš pradžių, remiantis statistine analize, iš procesinių duomenų buvo atrinkti pagrindiniai parametrai. Vėliau bandymais nustatyta optimali DNT struktūra: paslėptų sluoksnių ir neuronų skaičius, pasirinkta tiesioginio sklidimo architektūra ir Levenberg-Marquardt tinklo mokymo algoritmas. Atlikta suprognozuoto produkto kokybės simuliacija patvirtino sukurto prognozavimo modelio efektyvumą.

Tikslus įrankių darbo laiko resursų apskaičiavimas tiesiogiai daro įtaką gamybos procesų efektyvumą ir produkcijos kokybę, pagrinde dėl procesų stebėjimo ir kontrolės realiu laiku galimybių, kai duomenys apdorojami ir analizuojami iš karto, proceso metu. Siekiant surasti būdą tiksliau prognozuoti frezavimo įrankių efektyvų darbo laiką ir optimaliai apskaičiuoti, kada įrankis turėtų būti keičiamas, buvo atlikti tyrimas, kai frezuojant nerūdijančio plieno ruošinio paviršius, keičiamas veleno greitis ir fiksuojamas skirtingas veleno veikimo galingumas [23]. Surinkus duomenis, buvo pritaikyta DNT kreivės fiksavimo (angl. curve fitting) metodas ir *Matlab* programinės platformos

skirtingos mokymo funkcijos galios aritmetinio vidurkio kvadrato šaknies (angl. root mean square) P_{rms} koeficientui apskaičiuoti. Be to, pavyzdinės P_{rms} augimo kreivės buvo sugeneruotos, siekiant atsižvelgti į proceso dalių neapibrėžtumą. Tyrimo metu nustatyta, kad P_{rms} reikšmė laiko perspektyvoje jautri frezavimo įrankių nusidėvėjimui ir rezultatais buvo pagrįsta koreliacija tarp DNT prognozuojamo ir realaus likutinio įrankių darbo ciklo. Autoriai taip pat pažymi, kad prognozavimas *Matlab* programinėje platformoje su įprastu *Intel i7* procesoriumi užtrūko apie 0.5s, metodas yra gana pigus ir gali būti pritaikomas daugybėje pramonės šakų.

Panašus tyrimas buvo atliktas nikelio lydinių apdirbimo pramonėje, kur nuolatinė ir savalaikė procesų kontrolė bei stebėsena yra kritinė, kadangi teisingai nuspėti įrankių darbo laiko resursai leidžia ne tik užtikrinti įrangos netikėtų gedimų prevenciją, gaminamo produkcijos kokybę, bet ir ženkliai sumažina darbų saugos incidentų riziką. Tai labai svarbu ypač nikelio ir titano superlydinių gamyboje (Ni-Ti), kadangi katastrofiški gedimai yra neprognozuojami ir ankstyvas įrankių susidėvėjimas ištinka beveik atsitiktine tvarka, dėl apdirbamų metalų fizinių savybių ir agresyvaus suvirinimo, pjaustymo, šlifavimo bei kitų operacijų pobūdžio [24]. Kaip ir prieš tai aptartame nerūdijančio plieno frezavimo tyrime, veleno darbo galingumas, greičio ir padavimo duomenys buvo surinkti iš modernių nikelio lydinių gręžimo įrenginių, siekiant įvertinti veleno galingumo realiu laiku įtaką įrankio susidėvėjimui / lūžiui prognozuoti, apdirbant nikelio ir chromo pagrindo *Inconel 625* superlydinį. Veleno darbo eigos galingumo statistiniai duomenys galios matuokliais perduoti funkciniam apdorojimui tiesiogiai į sumodeliuotą dirbtinį neuroninį tinklą. Modelio patikimumui patikrinti surinkti ir palyginti veleno jėgos duomenys. Rezultatai rodo, kad šių dviejų skirtingų tipų duomenų tendencijos per įrenginio darbo laiką yra panašios bet kuriuose spaudimo jėgos ir greičio deriniuose. Nustatyta įrankio nusidėvėjimo DNT prognozės modelio paklaida su galingumo duomenimis – tarp 0,8 ir 18,4 proc., palyginus su 0,7 – 17,9 proc. paklaida, remiantis jėgos duomenimis. Tyrimu įrodyta, kad veleno galingumo duomenys, integruoti į DNT sistemą, gali būti naudingi įrankių susidėvėjimų / lūžimo stebėjimui realiu laiku ir efektyvesnei proceso kontrolei, o kartu ir stambiosios pramonės skaitmenizavimui.

Naudodama Lenkijos vandentiekio duomenis, Kutyłowska [25] sukūrė prognozavimo modelį, kuris leidžia prognozuoti gedimų dažnumą vieno iš Lenkijos miesto vandens tiekimo sistemoje. Buvo nustatyta, kad dirbtinis neuroninis tinklas (daugiasluoksnis perceptronas), mokytas naudojant *quasi-Newton* algoritmą, pasiekė inžinerinėje praktikoje reikalaujamą konvergenciją. Tinklas buvo mokytas naudojant ir vertinant 173 namų vandentiekio jungtis ir 147 vandens padavimo vamzdynus 2001-2006 metų laikotarpiu. 50 proc. visų duomenų naudoti mokymui, 25 proc. validacijai ir 25 proc. testavimui. Prognozavimo fazėje geriausias sukurtas modelis naudojo 100 proc. iš 133 ir 144 testavimo verčių. Geriausias namo jungčių gedimų prognozavimo modelis turėjo 8 įvesčių neuronus,

18 neuronų paslėptame ir vieną išvesties sluoksnį. Koreliacija tarp eksperimentinių ir prognozuojamų duomenų buvo charakterizuota mokymosi fazės indikatoriumi ir pasiekė gan aukštus rezultatus: R^2=0,9510 ir R^2=0,9260.

Remiantis aukščiau aprašytais moksliniais tyrimais galima daryti išvadą, kad kompleksiškuose, daugybės faktorių veikiamuose ir iš itin didelės pradinių duomenų rinkinių aprašomuose gamybos procesuose, dirbtinių neuroninių tinklų panaudojimo galimybės yra itin plačios ir panaudojimo potencialas yra didžiulis. Prognozavimas dirbtiniais neuroniniais tinklais gali būti itin naudingas pramonėje, siekiant nuspėti ateities įvykius – būsimus kokybės nuokrypius, potencialius techninės įrangos gedimus, būsimą veiklos rezultatą ir kt.

1.6. Efektyvumo samprata

Kadangi sąvokos panaudojimo spektras yra itin platus (nuo statistikos, matematikos, ekonomikos iki biologijos, fizikos), egzistuoja daugybė efektyvumo apibrėžimų. Kaip apibūdina Vainienė [19], efektyvumas, tai išteklių panaudojimo veiksmingumas, kai norimas rezultatas pasiekiamas mažiausiomis įmanomomis sąnaudomis arba naudojant turimus išteklius pasiekiamas maksimalus įmanomas rezultatas.

Pramonėje, aptariant gamintojo ekonominės veiklos rezultatus, dažniausiai vertinamos dvi prizmės: produktyvumas ir efektyvumas. Gaminimo proceso produktyvumas – tai santykis tarp išvesties ir įvesties kiekių. Šis santykis yra gana lengvai apskaičiuojamas, kai gamybos procesas turi vieną įvestį ir pagaminamą vieną išvestį. Tačiau dažniausiai gamybos procesas naudoja kelias, keliolika ar kelis šimtus įvesčių, gaminant daugybę išvesčių, todėl procesas turi būti apibendrinamas ir apskaitomas pagal nustatytas valdymo taisykles, siekiant užtikrinti ekonomiškai pagrįstą objekto veiklą. Gamybos proceso efektyvumas – tai palyginimas tarp stebimo proceso realaus ir optimalaus įvesčių ir išvesčių santykio [26].

Galimybė tiksliai prognozuoti gamybos efektyvumą, tai yra nuspėti, kiek produkcijos bus pagaminta per apsibrėžtą laikotarpį, suteikia neabejotiną konkurencinį pranašumą ir leidžia:

- Optimaliai paskirstyti gamybos užduotis per disponuojamus įrenginius ar žmogiškuosius išteklius;
- Teisingai suplanuoti pinigų srautus;
- Maksimaliai įvykdyti užsakymus;
- Minimizuoti pagamintos produkcijos logistikos kaštus ir pristatymo laiką;
- Optimaliai prognozuoti žaliavų tiekimo planavimą;
- Racionaliai pagrįsti priimtus sprendimus.

Šiame darbe aptariama standartizuota MARS, Incorporated (toliau Mars, Inc.) tarptautinės įmonės gamybinių procesų efektyvumo skaičiavimo metodika, plačiau aprašyta šio darbo 3.1.2 skyriuje „Įmonės procesų efektyvumo vertinimo metodika“, naudojama globaliu mastu, daugiau nei 130 gamybinių padalinių visame pasaulyje.

2. Dirbtinio neuroninio tinklo modelio kūrimo eiga

Matlab programinė platforma pateikia dirbtinių neuroninių tinklų kūrimo įrankius, kurių pagalba kuriami DNT modeliai užsibrėžtų uždavinių sprendimui. Šiame skyriuje aptariama ir analizuojama dirbtinio neuroninio tinklo modelio, skirto prognozavimo uždavinių sprendimui ir pritaikyto konkrečiai užduočiai – gamybos efektyvumo prognozavimui, sukūrimo metodika. Sukurtas modelis bus patikrintas ir validuotas, naudojant realius gamybinės įmonės duomenis, lyginant juos su realiais tikslo rezultatais.

Praktikoje ir *Matlab* metodologijoje rekomenduojama septynerių pagrindinių žingsnių dirbtinio neuroninio tinklo kūrimo eiga [27]:

1. Duomenų surinkimas;
2. Tinklo sukūrimas;
3. Tinklo konfigūravimas;
4. Slenksčių (angl. bias) ir svorių (angl. weights) inicijavimas;
5. Tinklo mokymas;
6. Tinklo tikrinimas;
7. Tinklo naudojimas.

2.1. Pradinių duomenų surinkimas, apdorojimas ir padalinimas į funkcines grupes

Duomenų surinkimas ir sistematizavimas atliekamas pagal neuroninio tinklo tikslą, pobūdį, užduotį bei programinės įrangos rūšį ir programinius reikalavimus. Tai fundamentali proceso dalis, tiesiogiai veikianti visų vėlesnių proceso etapų efektyvumą.

Svarbu, kad duomenys apimtų įvesčių diapazoną, kuriam tinklas bus naudojamas. Daugiasluoksniai neuroniniai tinklai gali būti mokomi gerai generalizuotis įvesčių diapazone, kuriame jie buvo mokomi, tačiau jie negali tiksliai ekstrapoliuoti už šio diapazono ribų, todėl svarbu, kad tinklo mokymo duomenys apimtų visą įvesčių erdvės diapazoną. Prieš panaudojant surinktus duomenis dirbtinio neuroninio tinklo mokymo metu, atliekami du pagrindiniai duomenų apdorojamo etapai: sistematizavimas ir klasterizavimas į funkcinius pogrupius ar matricas.

Iš anksto žinant duomenų specifiką, rūšį ir struktūrą, galima tiksliau ir greičiau parinkti tinkamą neuroninio tinklo modelio kūrimo metodiką, architektūrą ir techniką. Kai kurie su duomenimis susiję faktoriai gali daryti ženklią įtaką tinklo kūrimo eigai, pavyzdžiui duomenų triukšmo, imčių

praleidimo, pasikartojimo ar nenuoseklumo, šališkumo, nereprezentatyvumo ir t.t. problemoms. Todėl gerinant pradinės imties duomenų kokybę atitinkamai gerėja analizės kokybė.

Yra daugybė skirtingų duomenų gerinimo technikų, leidžiančių pritaikyti ir modifikuoti pradines imtis pagal panaudojamą metodiką ir lengvesnį apdorojimą. Kourou ir kt. [4] išskiria šiuos pagrindinius metodus: dimensionalumo mažinimas, ypatybių atranka ir išskyrimas.

Pagrindinė praktika mokant daugiasluoksnius DNT yra suskirstyti duomenis į tris pagrindinius rinkinius – funkcines grupes. Pirmoji yra mokymo dalis, skirta gradiento, tinklo svorių ir slenksčių apskaičiavimui. Antra – tikrinimo, validacijos dalis, kurios išskaičiuota klaida atliekama mokymo kokybės stebėsenos tinklo mokymo metu. Įprastai, kaip ir nustatyta tinklo mokymo klaida, validacijos klaida mažėja kartu su tinklo mokymu, tačiau, kai tinklas tampa itin adaptyvus (angl. overfit) mokymo duomenims, tinklo validacijos duomenų klaidos reikšmė dažniausiai pradeda didėti. Trečioji – testavimo duomenų dalis yra nenaudojama tinko mokymo metu ir yra skirta skirtingų tinklo modeliams palyginti ir išbandyti. Be to, ji naudinga apibrėžiant testavimo rinkinio klaidą tinklo mokymo metu. Jei klaida testavimo duomenų dalyje pasiekia minimumą po visiškai kito iteracijų skaičiaus lyginant su minimalia klaida validacijos duomenų rinkinyje, tai tikriausiai indikuoja netikslų rinkinių grupės padalinimą arba pasirinkimą.

Dažniausiai programinėje DNT modeliavimo aplinkoje duomenų padalinimas į mokymo, validacijos ir testavimo rinkinius atliekamas specialiomis funkcijomis. Pavyzdžiui padalinant duomenis:

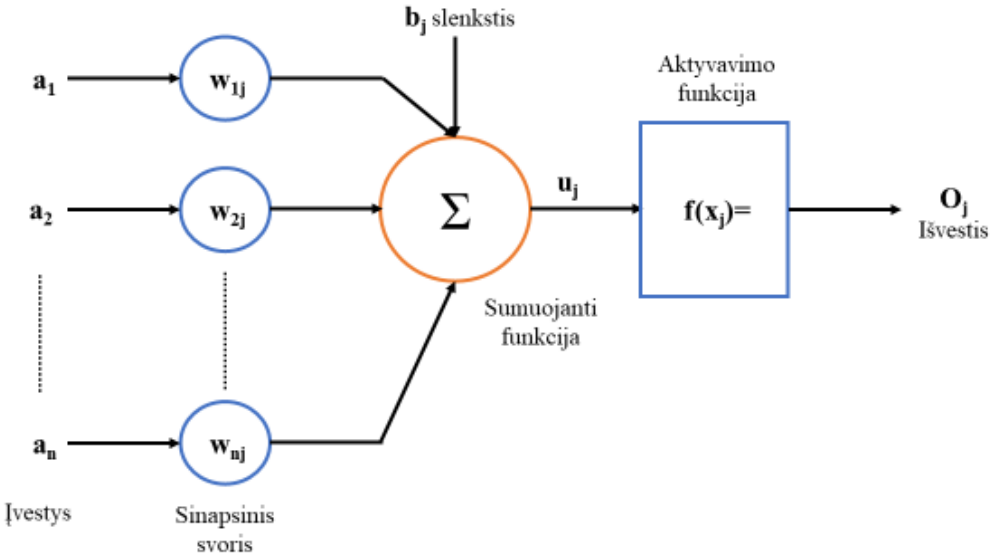
- Atsitiktine tvarka (*Matlab* komanda – *dividerand*);
- Į gretimus blokus (*divideblock*);
- Pasirenkant tarpinius rinkinius (*divideint*);
- Indeksuojant (*divideind*).

Siekiant itin tikslios dirbtinio neuroninio tinklo modelio architektūros, galimas rankinis duomenų pasirinkimas, tokiu atveju optimalus modelis gali būti pasiekiamas tik atliekant nemažai testų.

2.2. Dirbtinio neuroninio tinklo struktūra

Pagrindinis DNT komponentas yra neuronas, literatūroje dažnai vadinamas mazgu (angl. node). Vieno mazgo dirbtinis neuroninio tinklo modelis yra pateiktas 2 paveiksle. Čia įvestys (angl. inputs) pavaizduotos kaip a_1, a_2 ir a_n, o tinklo išvestis (angl. output) – O_j. Gali būti daugybė įvesties signalų į vieną mazgą. Vertės w_{1j}, w_{2j} ir w_{nj} yra svorių (angl. weights) faktoriai asocijuojami su įvestimis į mazgą.

Svoriai yra adaptyvūs koeficientai tinklo ribose, kurie nusako įeinančio signalo intensyvumą. Kiekviena įvestis (a_1, a_2, . . . , a_n,) yra dauginama iš atitinkančio svorių koeficiento (w_{1j}, w_{2j} ir w_{nj}), o mazgai naudoja svorių įvesčių sumavimą (w_{1j} *a_1, w_{2j}*a_2, . . ., w_{nj}*a_n), kad įvertintų išvesties signalą, naudodami perdavimo (dar vadinama aktyvavimo) funkciją.



2 pav. Dirbtinio neurono anatomija *[28]*

Kita įvestis į mazgą, tai yra tinklo neuroną, yra b_j – mazgo vidinis slenkstis (angl. bias), dar vadinamas poslinkiu. Tai atsitiktine tvarka pasirenkama vertė, kuri reguliuoja mazgo grynąją įvestį per 1 formulę.

$$u_j = \sum_{i=j}^{n} (w_{ij} * a_i) + b_j \qquad (1)$$

Mazgo išvestis nusakoma naudojant matematinius veiksmus visoms mazgų (neuronų) išvestims. Šie veiksmai vadinami perdavimo (angl. transfer) funkcija, kuri gali transformuoti visą mazgų įvestį tiesine arba netiesine tvarka [28]. Dažniausiai naudojamos trys pagrindinės perdavimo funkcijų rūšys: sigmoidinė, hiperbolinė ir tiesinė.

Sigmoidinė (angl. sigmoid) perdavimo funkcija:

$$f(x) = \frac{1}{1+e^{-x}}, \quad 0 \leq f(x) \leq 1 \qquad (2)$$

Sigmoidinė funkcija generuojami rezultatai apima nuo 0 iki 1 intervalą. Nors sigmoidinė funkcija yra dažniausiai naudojama praktikoje, tačiau ji turi ir trūkumų dėl įgaunamų tik teigiamų reikšmių, kadangi tiriant kompleksiškai sudėtingesnes problemas reikalingos ir neigiamos reikšmės. Hiperbolinio tangento perdavimo funkcija:

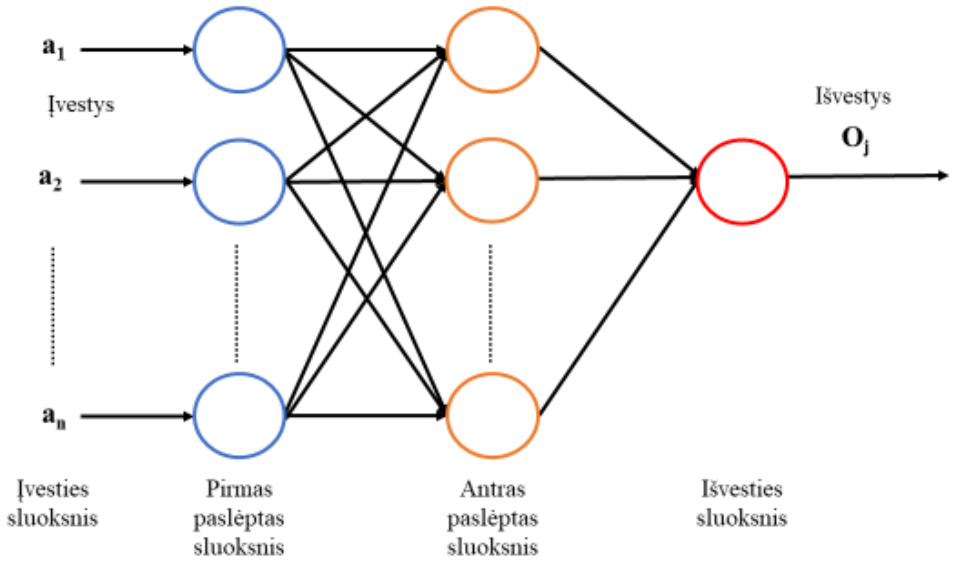
$$f(x) = \tan(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}, \quad -1 \leq f(x) \leq 1 \tag{3}$$

Kadangi potencialiai ši funkcija yra įgali įgauti tiek teigiamas, tiek ir neigiamas išvesties reikšmes, neuroniniam tinklui suteikia papildomas savybes ir galimybes. Tiesinė perdavimo funkcija generuoja neapibrėžto intervalo rezultatus:

$$f(x) = x\ , \ -\infty \leq f(x) \leq +\infty \tag{4}$$

Dirbtinio neuronų tinklo rezultatas O_j nustatomas remiantis viena iš šių funkcijų neuronų bendrai įvesčiai u_j.

Ši aprašyta vieno neurono struktūra yra vieno dirbtinio neuroninio tinklo sluoksnio individuali dalis. Jeigu tinklas yra daugiasluoksnis, pradinis įvesties sluoksnis sudarytas iš įvesčių duomenų, o funkciniai paslėpti neuronų sluoksniai toliau mokomi vis kitokių savybių apie pateiktą duomenų imtį.



3 pav. Daugiasluoksnio tiesioginio sklidimo DNT modelio sluoksnių išsidėstymas

Sluoksniai vadinami paslėptais, kadangi skaičiuojamos reikšmės nestebimos ir nėra kontroliuojamos mokymo metu. Dažniausiai išvesties sluoksnis pasižymi pačia paprasčiausia struktūra ir sudaromas iš aiškiai nustatyto išvesčių kiekio. Tačiau, kaip ir anksčiau aprašyta, jeigu sluoksnis sudaromas nors ir iš vieno neurono, vis tiek yra laikomas sluoksniu.

2.3. Tinklo architektūros parinkimas

Kaip aprašoma šio darbo 1.2 skyriuje, dirbtiniai neuroniniai tinklai yra skirstomi pagal mokymo rūšį – su mokytoju, be mokytojo ir mišrūs. Kadangi šiame darbe kuriamas gamybos efektyvumo prognozavimo modelis, remiantis surinktais aiškiai apibrėžtais įvesties ir tikslo išvesties duomenimis,

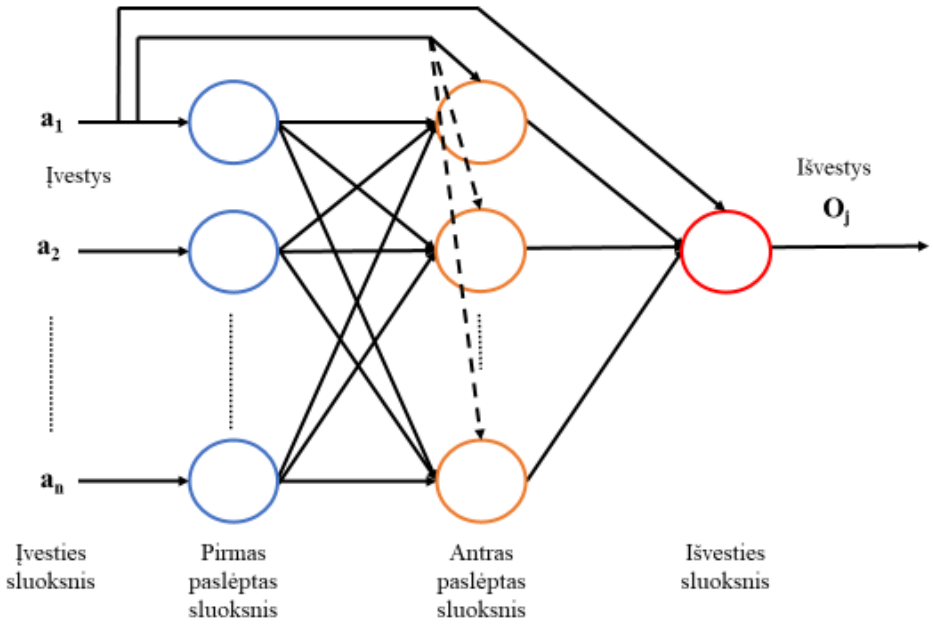
panaudotas prognozavimo užduotims labiau tinkantis mokymo su mokytoju neuroninio tinklo modelis.

Kadangi darbe analizuojama kompleksinė ir didelė duomenų imtis, be to, sprendžiamas prognozavimo uždavinys, vieno sluoksnio tiesioginio sklidimo neuroninis tinklas, dar vadinamas parceptronu (aprašytas šio darbo 1.2.2 skyriuje), nėra tinkamas. Todėl prognozuojant gamybos efektyvumą išbandyti įvairaus sklidimo daugiasluoksniai neuroniniai tinklai su įvairiomis sluoksnių sudėtimis ir skirtingu neuronų kiekiu. *Matlab* programinėje platformoje dažniausiai naudojami trijų sklidimo architektūrų dirbtiniai neuroniniai tinklai:

- Tiesioginio sklidimo tinklai;
- Kaskadinio sklidimo tinklai;
- *Patternnet* funkcija.

Paprastai praktikoje naudojami tiesioginio sklidimo struktūros tinklai, juose signalai iš įvesčių pirmyn keliauja per visus paslėptus sluoksnius tik viena kryptimi ir taip pasiekia išvesties taškus. Tokia struktūra ypač vertinama praktikoje, kadangi yra itin stabili [29].

Nors įrodyta, kad bent dviejų sluoksnių tiesioginio sklidimo neuroninis tinklas potencialiai įgalus nustatyti ir užfiksuoti bet kurių įvesčių–išvesčių tarpusavio ryšį, tiesioginio sklidimo tinklai turintys didesnį skaičių paslėptų sluoksnių gali būti greičiau išmokyti spręsti kompleksiškai sudėtingesnes problemas.



4 pav. Kaskadinio sklidimo DNT modelis

Matlab platformoje prognozavimo problemoms spręsti naudojamas ir kaskadinio sklidimo (angl. cascade-forward) tinklas, tai panaši į tiesioginio sklidimo neuroninio tinklo architektūrą, tačiau turinti svorių jungtį nuo įvesties taškų tiesiogiai į visus sluoksnius, ir kartu nuo kiekvieno sluoksnio į sekantį (4 pav.). Pavyzdžiui, trijų sluoksnių tinklas turės jungtį iš pirmo sluoksnio į antrą, antras sluoksnis į trečią ir pirmas sluoksnis į trečią. Be to, šis trijų sluoksnių tinklas turės jungtį nuo įvesčių į kiekvieną tinklą atskirai [30]. Šios papildomos jungtys gali smarkiai pagerinti tinklo ryšių mokymosi greitį. Kaskadinio sklidimo architektūra yra viena iš rekurentinių (turinčių atgalinių jungčių iš tolesnių neuronų) tinklo rūšių.

Patternet funkcijos architektūra, nors ir analogiška tiesioginio sklidimo architektūrai, savo paskutiniame sluoksnyje naudoja *tansig* perdavimo funkciją, ir savo prigimtimi labiau tinkama modelio ar tendencingumo atpažinimo uždaviniams spręsti.

2.4. Dirbtinio neuroninio tinklo mokymas

Kai dirbtinio neuroninio tinklo svoriai ir slenksčiai yra inicijuoti, galimas tinklo mokymas. Nesvarbu ar dirbtinio neuroninio tinklo mokymo tikslas yra funkcijos aproksimacija (netiesinei regresijai) ar struktūros / formos (angl. pattern) atpažinimas, mokymo procesas reikalauja pavyzdinio duomenų rinkinio, kuris indikuotų, kada neuroninis tinklas pasiekė pageidautiną tikslumą ir yra pajėgus įgyvendinti užbrėžtus tikslus.

DNT mokymo proceso metu tinklo svoriai ir slenksčiai optimizuojami siekiant geriausių įmanomų tinklo charakteristikų, nusakytų neuroninio tinklo pradinės tikslo funkcijos. Tačiau dirbtinio neuroninio tinklo mokymo proceso tikslas nėra analogiškai reprezentuoti duomenų imtį. Tikslas yra suformuoti ir imituoti statistinį proceso modelį, pagal kurį suformuotas tinklo mokymosi rinkinys ir naujai generuojami duomenys savo reikšmėmis būtų kuo artimesni jam. DNT pagal mokymo duomenų imtį turi sugebėti tinkamai įvertinti, suvokti ir apibendrinti naujus duomenis. Ši tinklo savybė vadinama generalizacija (angl. generalization).

Daugiasluoksnio dirbtinio neuroninio tinklo mokymo procese naudojamas klaidos skidimo atgal algoritmas (angl. error back propagation), jis išskaido klaidą tarp neuroninio tinklo elementų nuo viršaus tolyn į apačią dviem žingsniais:

- Įvesties reikšmių sklidimas pirmyn nuo įvesties į išvesties sluoksnius;
- Susidariusios paklaidos sklidimas atgal iš išvesties sluoksnio į įvesties.

Klaidos sklidimo atgal algoritme pritaikyta mokymo su mokytoju koncepcija, todėl įvesties duomenys yra mokymo rinkinio vektorius ir norimų reikšmių vektorius. Kaip aprašyta šio darbo 2.2

skyriuje, tiesioginio sklidimo metu, pagal funkcijos formulę, kiekvienas elementas skaičiuoja svorinę įėjimo elementų sumą:

$$a = w_1x_1 + w_2x_2 + \ldots + w_nx_n = \sum_{k=1}^{n} w_kx_k \qquad (5)$$

Čia:

$x_1, x_2, x_3, \ldots, x_n$ yra neurono įėjimo reikšmės;

$w_1, w_2, w_3, \ldots, w_n$ – įėjimo perdavimo svoris (w_0 – slenksčio reikšmė).

Gauta suma transformuojama netiesine perdavimo funkcija:

1. Įvesties kintamuosiuose suma: $a_j = \sum_i w_{ji}x_i$ ir funkcija: $z_j = f(a_j)$ *(6), (7)*
2. Paslėptuose sluoksniuose suma: $a_l = \sum_j w_{lj}z_j$ ir funkcija: $z_l = f(a_1)$ *(8), (9)*
3. Tinklo išvesties kintamuosiuose suma: $a_k = \sum_l w_{kl}x_l$ ir funkcija: $y_k = f(a_k)$ *(10), (11)*

Čia:

a_j, a_l, a_k – visų neuronų išvesčių svorinė suma;

x_i, z_j, z_l – įvesties kintamieji arba prieš tai buvusiame sluoksnyje esančio neurono išvesties reikšmė;

w_{ji}, w_{lj}, w_{kl} – jungčių svoriai tarp nagrinėjamo sluoksnio ir prieš tai buvusio sluoksnio neuronų;

z_j, z_l, y_k – neuronų išvesties reikšmės.

Yra du skirtingi DNT mokymo diegimo būdai: inkrementinis (angl. incremental mode) ir paketinis (angl. batch mode). Inkrementiniu būdu (inicijuojamas *Matlab* komanda *adapt*) gradientas paskaičiuojamas ir svoriai yra atnaujinami po kiekvienos įvesties diegimo į tinklą. Paketiniu būdu (inicijuojamas *Matlab* komanda *train*) visos įvestys mokymo rinkinyje yra pritaikytos tinklui prieš svorių atnaujinimą. Kadangi inkrementinis būdas yra dinamiškai adaptyvus ir labiau naudojamas, kai tinklui keliamas reikalavimas atsinaujinti proceso eigoje, šiame darbe naudojama paketinis mokymo diegimas būdas.

2.5. Tinklo tikrinimas ir vertinimas

DNT mokymas pagrįstas klaidos minimizavimo metodais. Optimali minimizuojamos klaidos reikšmė ir tipas parenkamas pagal neuroninio tinklo rūšį, sprendžiamus uždavinius ar duomenų imtį.

Viena dažniausiai *Matlab* programinėje aplinkoje naudojamų klaidos funkcijų yra suminė kvadratinė klaida (angl. sum squared error), ypač tinkanti regresijos uždavinių sprendimams. Suminė kvadratinė

klaida (angl. trumpinys SSE) apskaičiuojama sumuojant visos duomenų imties klaidas visuose tinklo išėjimuose:

$$E(w) = \frac{1}{2}\sum_{n=1}^{N} \sum_{k=1}^{C}\{y_k(x^n, w) - t_k^n\}^2 \quad (12)$$

Čia N – įėjimo vektorių skaičius; c – tinklo išėjimų skaičius; $y_k(x^n,w)$ – k-asis tinklo išėjimas, apskaičiuotas, kaip n-ojo įėjimo vektoriaus x^n ir svorių w funkcija [5].

Standartinė tiesioginio sklidimo neuroninio tinklo vertinimo funkcija *Matlab* programinėje aplinkoje yra vidutinės kvadratinės klaidos rodiklis (angl. mean square error, trumpinys – MSE), nurodantis vidutinę kvadratinę klaidą tarp tinklo realiu laiku generuojamų išvesčių ir tikslo išvesčių. Ji aprašoma:

$$F = \frac{1}{N}\sum_{i=1}^{N}(e_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i)^2 \quad (13)$$

Tai vidutinė kvadratinė klaida tarp tinklo išvesčių a ir tikslo išvesčių t. Čia kaip ir statistikoje vidutinės kvadratinės klaidos nustatymas kiekybiškai apibrėžia skirtumą tarp tikrosios vertės ir apskaičiuotosios. Šis skirtumas yra natūraliai atsitiktinis ir susidaro dėl objektyvaus informacijos trūkumo, kuris užtikrintų tikslesnius skaičiavimus.

Jau sukurto dirbtinio neuroninio tinklo svarbus validacijos proceso rodiklis yra tinklo regresijos (R) koeficiento vertė, kuri indikuoja ryšį tarp išvesties duomenų ir tikslo. Jeigu R=1, galima daryti išvadą, kad yra tikslus tiesinis ryšys tarp gautų rezultatų ir tikslo reikšmių. Ir atvirkščiai, jeigu R vertė artima nuliui – tiesinis ryšys tarp išvesčių ir tikslo neegzistuoja, kas reiškia, kad DNT modelis yra sukurtas netinkamai. *Matlab* programinėje aplinkoje dažnai naudojama grafinė regresijos išraiška, kuri reprezentuoja santykį tarp išvesčių ir tinklo tikslinių reikšmių. Apskaičiuojami ir nubraižomi trys atskiri regresijos grafikai visiems trims (mokymo, validacijos ir testavimo) rinkiniams.

2.6. Mokymo funkcijos pasirinkimas

Dauguma neuroninių tinklų kūrimo programų automatiškai priskiria arba iš karto naudoja konkrečias įvesties ir išvesties duomenų apdorojimo funkcijas. Šios funkcijos transformuoja pateiktas įvesties ir tikslo reikšmes į reikšmes, kurios labiau atitinka tinklo mokymo poreikius. Daugeliu atvejų kuriant tinklą galima pasirinkti įvesties ir išvesties apdorojimo funkcijas pagal tikslines tinklo savybes.

Pasirinkimas, kuris mokymo algoritmas optimaliausiai atitinka tinklo poreikius ir bus greičiausias tiriamiems duomenims apdoroti, yra sudėtingas procesas, susidedantis iš daugybės dedamųjų [27]:

- Problemos kompleksija;
- Duomenų taškų skaičius mokymo rinkinyje;

- Slenksčių ir svorių skaičius tinkle;
- Tikslinė paklaida;
- Neuroninio tinklo panaudojimo tikslas (ar modelio struktūros atpažinimui (diskriminanto analizei) ar funkcijos aproksimacijai, tai yra regresijai).

Levenberg-Marquardt (*trainlm*) algoritmas yra standartinė metodika netiesinėms mažiausių kvadratų problemoms spręsti. Mažiausių kvadratų uždavinių tikslas – parametrizuotos funkcijos pritaikymas išmatuojamų duomenų taškų rinkinyje, minimizuojant paklaidų kvadratų sumą tarp duomenų taškų ir funkcijos [31]. Funkcijų aproksimacijos uždaviniams spręsti, ypač neuroniniams tinklams, sudarytiems iš iki kelių šimtų svorių, Levenberg-Marquardt algoritmas *Matlab* aplinkoje turės greičiausią konvergenciją. Metodas rekomenduojamas naudoti, kai siekiama itin tikslaus tinklo mokymo. *Trainlm* pasižymi įgyjama žemesne vidutine kvadratine klaida (angl. mean square error), tačiau išaugus svorių (angl. weights) tinkle skaičiui, metodo pranašumas akivaizdžiai mažėja. Taip pat reikėtų pastebėti, kad Levenberg-Marquardt prastai sprendžia modelio (angl. pattern) atpažinimo užduotis.

Lankstaus atgalinio sklidimo (angl. resilient backpropagation) (*trainrp*) algoritmas yra greičiausias, sprendžiant struktūros / formos (angl. pattern) atpažinimo uždavinius, tačiau prastai tinka funkcijos aproksimacijos problemoms spręsti. Algoritmo rezultatai prastėja mažinant klaidos tikslą.

Konjugacijos gradiento grąžinimo algoritmas (angl. conjugate gradient algorithm) (*trainscg)* yra gana universalus metodas neuroniniams tinklams su dideliu svorių skaičiumi. Kaip ir Levenberg-Marquardt algoritmas, yra gana greitas sprendžiant funkcijų aproksimacijos uždavinius (didelės struktūros tinkluose netgi greitesnis) ir panašiai greitas kaip atgalinio sklidimo algoritmas modelio atpažinimo uždaviniuose, nes *trainscg* greičio charakteristikos nesuprastėja taip greitai, kaip *trainrp* efektyvumas, kai klaidos koeficientas mažinamas.

Quasi-Newton algoritmas (*trainbfg) Matlab* aplinkoje veikia panašiai kaip *trainlm* algoritmas, generuoja mažiau informacijos duomenų, todėl užima mažiau vietos, tačiau skaičiavimų kiekis didėja geometrine progresija paraleliai neuroninio tinklo dydžiui, kadangi su kiekviena iteracija apskaičiuojamas atvirkštinė matrica.

Kintamojo mokymosi lygio grąžinimo algoritmas *(traingdx)* yra natūraliai lėtesnis nei kiti *Matlab* metodai, tačiau gali būti naudojamas specifiniams uždaviniams spręsti, ypač situacijose, reikalaujančiose lėtesnės konvergencijos. Pavyzdžiui, kai naudojant ankstyvą tinklo generavimo sustabdymą galimi nenuoseklūs rezultatai naudojant itin greitai konverguojančius tinklus ir sunku tiksliai sustabdyti tinklą ties minimaliausia klaidos validacija [27].

2.7. Dirbtinio neuroninio tinklo tobulinimas

Įprastai daugiasluoksnių neuroninių tinklų generalizacija tobulinama ankstyvo stabdymo metodu (angl. early stopping). *Matlab* programinėje skaičiavimo aplinkoje ši technika yra automatiškai nustatoma visiems mokymo su mokytoju tinklams, o taip pat klaidos sklidimo atgal modelių kūrimo funkcijoms. Šita technika jau aprašyta šio darbo 2.1 skyriuje „Pradinių duomenų surinkimas, apdorojimas ir padalinimas į funkcines grupes“.

Viena iš problemų, galinčių iškilti DNT mokymo metu, yra itin didelis tinklo adaptyvumas (angl. overfitting), kai nors tinklo mokymo rinkinys generuoja minimalią klaidą, tačiau pateikus jam naujus duomenis klaida smarkiai didėja. Tokiu atveju galima daryti išvadą, kad tinklas gerai įsimena mokymo pavyzdžius, bet nėra įgalus dirbti su naujomis, dar nematytomis jam, situacijomis. Vienas iš metodų šiai problemai spręsti, tai yra gerinti tinklo generalizaciją, tai yra padidinti tinklo apimtis. Kuo didesnis naudojamas tinklas, tuo kompleksiškai sudėtingesne funkcija jis operuoja. Pakankamai mažas tinklas neturės užtektinai potencialo adaptuotis prie duomenų.

Jeigu parametrų skaičius tinkle yra daug mažesnis nei bendras taškų skaičius mokymo rinkinyje, yra minimali tikimybė susidaryti per dideliam tinklo adaptyvumui, todėl esant galimybei rekomenduojama surinkti daugiau duomenų ir padidinti mokymosi imtį. Tačiau, kai tokios galimybės nėra, generalizacijai gerinti naudojami reguliarizacijos metodai. Įprastai kiekviena klaidos sklidimo atgal mokymo sesija pradedama vis kitomis svorių ir slenksčių reikšmėmis bei vis kitais duomenų imties padalinimais į mokymo, validacijos ir testavimo rinkinius. Šios skirtingos pradinės sąlygos dažnai yra visiškai skirtingų sprendimų tai pačiai problemai spręsti priežastis. Todėl DNT su mažomis pradinių duomenų imtimis rekomenduojama mokinti daugiau nei vieną tinklą, siekiant užfiksuoti geriausios generalizacijos modelį.

Kitas generalizacijos tobulinimo įrankis, ypač kai įtariama, kad pradiniai įvesties duomenys sudaryti iš per mažos imties arba gali būti paveikti triukšmų, yra kelių neuroninių tinklų mokymas ir jų išvesčių vidurkio skaičiavimas [2]. Išvesčių vidurkio vidutinė kvadratinė paklaida dažniausiai bus mažesnė nei daugelis individualių modelių rezultatų, tačiau galimos išimtys. Ypač sudėtingoms problemoms galima mokyti šimtus tinklų, skaičiuojant jų išvesčių vidurkį bet kurioms įvestims ir toks modelis turėtų geriau generalizuotis dirbant su naujais papildomais duomenimis.

3. Dirbtinio neuroninio tinklo pritaikymas gamybos efektyvumo prognozavimui

Vienas iš šio darbo tikslų yra sukurti DNT modelį gamybos efektyvumo prognozavimui, naudojant realius gamybinės įmonės duomenis, taip pat atlikti sukurto modelio testavimą ir patikrinimą realiomis sąlygomis. Šiems tikslams panaudoti UAB „Mars Lietuva“ (toliau „Mars Lietuva“) trylikos mėnesių gamybos procese sukauptų realių gamybos proceso efektyvumo duomenys.

3.1. Duomenų surinkimas ir paruošimas

Kaip aprašyta šio darbo 2.1 skyriuje „Pradinių duomenų surinkimas ir apdorojimas“, teisingas pradinių duomenų surinkimas, apdorojimas ir interpretavimas yra fundamentali dirbtinio neuroninio tinklo kūrimo proceso dalis, tiesiogiai veikianti visus sekančius modelio kūrimo etapus. Siekiant kokybiškai adaptuoti duomenų imtį DNT ir apibrėžti santykį tarp įvesčių ir išvesčių, būtinas geras nagrinėjamo proceso supratimas (šiuo atveju „Mars Lietuva“ gamybos proceso).

3.1.1. Įmonės gamybos procesų aprašymas

„Mars Lietuva” yra Lietuvoje veikianti įmonė, JAV korporacijos Mars, Inc. dalis. Virš 22 tūkst. kvadratinių metrų „Mars Lietuva“ gamykla priklauso Mars gyvūnų augintinių maisto segmentui, joje dirba apie 800 pastovių ir 200 laikinai samdomų darbuotojų. Kasmet čia pagaminama apie 80 tūkst. tonų, per 400 rūšių ir formatų konservuoto maisto gyvūnams augintiniams. Beveik visa Lietuvoje pagaminama produkcija eksportuojama į 32 Europos šalis ir Australiją, todėl „Mars Lietuva“ gamykla yra viena didžiausių šalies eksportuotojų. Produkcijos kiekį skaičiuojant pakuotėmis, per metus įmonė pagamina virš 1 mlrd. vienetų gyvūnų augintinių maisto maišelių. Per parą (2 pamainos po 12 valandų) įmonėje pagaminama apie 300 tonų vadinamo šlapio kačių ar šunų ėdalo maišeliuose po 50, 85 ar 100g formatuose. „Mars Lietuva“ gaminamo konservuoto kačių ir šunų maisto maišeliuose (angl. „pouches“) pagrindiniai prekės ženklai: WHISKAS®, KITEKAT®, SHEBA®; EXCELCAT®; DINE®, PEDIGREE® ir kiti.

Įmonėje gamyba organizuojama pagal patvirtintas, aiškiai standartizuotas ir aprašytas procedūras, laikantis visų maisto pramonei taikomų kokybės ir maisto saugos reikalavimų. Pagrindiniai „Mars Lietuva“ technologiniai gamybos procesai:

- Šaldytos mėsinės, birios ir skysto pavidalo žaliavos bei kiti priedai priimami ir sandėliuojami įmonėje, pagal nustatytus reikalavimus individualiai. Šaldytos mėsos žaliavų blokai paruošiami gamybai – susmulkinami pramoniniais trupintuvais.
- Sutrupintos žaliavos pagal receptūrą transportuojamos į mėsos emulsijos paruošimo maišykles, kur į produktą pridedami reikalingi priedai – birios žaliavos, reikiamos spalvos tirpalas,

maistiniai priedai. Viskas gerai susmulkinama ir sumaišoma dviejų smulkinimo etapų metu. Gauta masė pašildoma iki nustatytos temperatūros ir paruošiama gabaliukų kepimui.

- Tuo pačiu metu ruošiamas kita produkto dalis – mėsinės struktūros gabaliukų užpilas.
- Iš paruoštos emulsijos suformuojamos juostos, kurios iškepamos specialiose garo krosnyse, atvėsinamos, susmulkinamos į gabaliukus, kurie sukaupiami specialiuose bunkeriuose, paruošiant dozavimui į maišelius.
- Gauti gabaliukai sveriami, dozuojami į maišelius ir užpilamas paruoštas užpilas. Užlydoma maišelio viršutinė siūlė. Gargždų gamykloje naudojami trys pagrindiniai maišelių dydžio formatai – 50g, 85g ir 100g. Ant pirminės pakuotės užpurškiama galiojimo data – 2 metai nuo pagaminimo datos.
- Užpildyti maišeliai sudedami į padėklus ir apie 60 minučių sterilizuojami specialiuose autoklavuose virimo temperatūros vandenyje aukštame slėgyje.
- Maišeliai nuimami nuo padėklų, nusausinami ir sudedami į antrinę pakuotę – kartonines dėžutes, pagal suplanuotą formatą.
- Pagaminta produkcija sukraunama ant medinių padėklų, sandėliuojama ir išvežama į priskirtos rinkos tarpinį sandėlį.

„Mars Lietuva“ gamykla nepertraukiamai dirba keturiomis pamainomis. Vienoje pamainoje vienu metu dirba apie 230-250 darbuotojų: apie 160 pastovių Mars, Inc. bendradarbių ir likusi dalis užpildoma darbo nuomos kompanijų darbuotojais. Pamainos dirba 12 valandų grafiku, tai yra - dvi rytines, dvi naktines pamainas, po to seka 3,5 paros (96 valandos) išeiginių. Gamyba organizuojama keturiose pagrindinėse linijose (vidiniai pavadinimai atitinkamai – PL1, PL2, PL3 ir PL4). Tiek ir bendras gamybos efektyvumas, tiek ir atskirų linijų efektyvumas skaičiuojamas pagal užpildymo įrengimų greitį, kuris yra fiksuotas – 105 maišeliai per minutę (100proc.). Šiuo metu įmonėje veikia 24 užpildymo įrengimai: PL1 linija – 8 užpildymo įrengimai, PL2 linija – 8 įrengimai, PL3 linija – 4 įrengimai ir PL4 linija – 4 įrengimai.

3.1.2. Įmonės procesų efektyvumo vertinimo metodika

Visoje Mars, Inc. gamybos efektyvumas skaičiuojamas pagal maišelių užpildymo įrenginių teorinį pajėgumą. Bet kurioje gamybos proceso stadijoje patirtos prastovos skirstomos į darančias tiesioginę įtaką efektyvumui ir ne. Nedarančios įtakos efektyvumui prastovos fiksuojamos, kai įvyksta proceso stadijose su gamybiniu rezervu ir pakankamai trumpą laiką, todėl nesustabdomi maišelių užpildymo įrengimai. Darančios įtaką prastovos fiksuojamos, kai yra stabdomi arba pristabdomi užpildymo įrengimai ir nepagaminamas suplanuotas kiekis. Kadangi maišelių dozavimo greitis yra reglamentuotas griežtais centralizuotais standartais, taip sąmoningai suformuojamas vadinamasis

„butelio kakliukas“, kuris padeda aiškiau valdyti viso fabriko gamybos procesus. Svarbu pažymėti, kad stabdymo priežastys gali įvykti tiek prieš maišelių užpildymą (pavyzdžiui dėl mėsos struktūros gabaliukų ar užpilo trūkumo), tiek ir po (pavyzdžiui dėl pakavimo ar sterilizavimo įrengimų gedimo). Vidinė įmonės pagaminto kiekio ir kartu efektyvumo apskaita vykdoma po produkto užpildymo stadijos, prieš sterilizacijos proceso pradžią, kai sudėjus produktą į pirminę pakuotę užlydoma maišelio viršutinė siūlė, patikrinamas svoris, uždedama galiojimo data ir transporteriais keliauja į sudėjimo į specialius padėklus įrengimą. Skaičiuojamas maksimalus galimų pagaminti maišelių kiekio ir realiai užpildytų maišelių kiekio santykis, taip gaunant efektyvumą procentais (14 formulė).

$$E = \frac{Q_r}{Q_m} \times 100\% \tag{14}$$

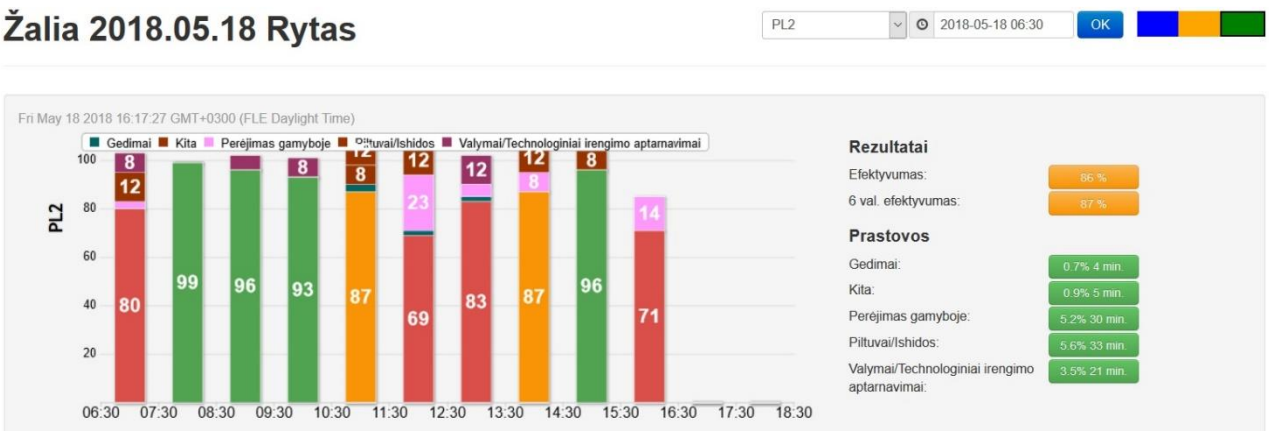
Čia E – gamybos efektyvumas procentais, Qr – realus pagamintas kiekis, Qm – maksimalus kiekis (laiko vienetas padaugintas iš linijos greičio vnt/min). Pavyzdžiui, jeigu vienas užpildymo įrengimas per valandą 100 procentų gamybos efektyvumu gali užpildyti maksimaliai 6300 maišelius (105 vnt/min greičiu) ir pagamina 5670 vnt. maišelių, vadinasi fiksuojamas konkrečios valandos gamybos efektyvumas yra 90 procentų. Atsakingas operatorius konvertuoja nepagamintą kiekį į laiką minutėmis ir užfiksuoja sistemoje prastovos priežastį.

$$T_p = T_m \times (1 - E) \tag{15}$$

Konkrečiu atveju būtų:

$$60\ min \times (1 - 0{,}9) = 6\ min$$

Taip pamatuojamas pamainos darbo, konkretaus produkto ar bendras gamyklos gamybos efektyvumo rezultatas norimu laiko intervalu. Šiuo metu įmonės vidaus apskaitos sistema ne tik fiksuoja, bet ir atspindi pagamintos produkcijos kiekį, gamybos proceso efektyvumą realiu laiku bei pagrindines prastovų priežastis (5 pav.).



5 pav. „Mars Lietuva“ gamybos stebėsenos sistemos fragmentas

Aprašyta proceso efektyvumo vertinimo metodika vienija visas Mars, Inc. gamyklas, todėl šia metodika galima objektyviai nustatyti bendrą korporacijos, jos atskirų segmentų ar funkcinių vienetų gamybos procesų eigą.

Gamybos efektyvumo duomenys yra gyvybiškai svarbūs ne tik vertinant gamybos rezultatus, tačiau ir planuojant būsimą gamybą. Tikslus gamybos efektyvumo prognozavimas leidžia:

- Optimaliai paskirstyti gaminamos produkcijos kiekius skirtinguose fabrikuose skirtingose šalyse;
- Optimaliai paskirstyti gamybos išteklius (operatorių skaičių; papildomą darbo jėgą, pinigų srautus ir t.t.);
- Maksimaliai įvykdyti įsipareigojimus rinkoms ir vartotojams;
- Minimizuoti pagamintos produkcijos logistikos kaštus ir pristatymo laiką;
- Optimaliai prognozuoti žaliavų ir pakuotės tiekimo planavimą;
- Racionaliai pagrįsti priimtus sprendimus.

„Mars Lietuva“ visų keturių pagrindinių gamybinių linijų gamybos planas rengiamas dviem savaitėms į priekį. Prognozuojamas efektyvumas apskaičiuojamas paprastu statistinio imties vidurkio metodu, išvestu buvusių gamybos rezultatų metiniu vidurkiu, nereglamentuotai atsižvelgiant į gaminamo produkto rūšį ir pakuotės formatą, tačiau neatsižvelgiant į bendras gamybos tendencijas, dedamąsias, potencialią įtaką ir t.t. Gamybinis planas su minimaliais nuokrypiais yra įgyvendinamas pagal nurodytą eiliškumą ir suplanuotais kiekiais, bet dėl gamybos efektyvumo apskaičiavimo metodo netobulumo visada fiksuojamas neatitikimas tarp plano ir realių gamybos procesų. Kartais net iki 12 valandų. Siekiant tikslumo ir proceso sklandumo visi einamosios gamybos planai perskaičiuojami kas 12 valandų pamainų pasikeitimo metu, nustatant ir užfiksuojant svarbiausią informaciją – preliminarų perėjimų gamyboje arba pakavime laiką pamainos eigoje, taip suvaldant žmogiškuosius išteklius ir nustatant žaliavų poreikį.

3.1.3. Įmonės gamybos duomenų apdorojimas

Siekiant sukurti tiksliai prognozuojantį gamybos procesų modelį, reikalingas kuo didesnis jau sukauptų gamybos eigos duomenų kiekis. Kaip aprašyta šio darbo 3.1.2 skyriuje, įmonėje visi gamybos rezultatai, prastovos ir jų priežastys yra fiksuojamos bei saugomos specialioje vidaus apskaitos sistemoje. Naudojama lygiai vienerių metų „Mars Lietuva“ pagrindinių gamybinių linijų gamybos duomenų imtis (nuo 2016 metų liepos 1 d. iki 2017 metų birželio 30 d.), kuri iš stebėsenos sistemos sugeneruota į *Excel* skaičiuoklės programinės įrangos *.xlsx* failą (6 pav.). Duomenys sistemoje užfiksuojami kaip vienos produkto rūšies gamyba, tai yra pagaminus tam tikrą kiekį

suplanuotos produkcijos (tonomis ir vienetais), pažymimas per tą laiko intervalą maksimaliai galimas pagaminti kiekis vienetais ir išskaičiuojamas gamybos efektyvumas procentais. Tada pereinama į kitą produkto gamybą ir taip per visas keturias gamybos linijas.

Pamaina: Raudona, Oranžinė, Mėlyna, Žalia

Efektyvumai pagal linijas

nuo: 2016-07-01
iki: 2017-07-01

Formatas / Pakuotė	Pamaina	Pagaminta (T.)	Pajėgumas (tūkst.)	Pagaminta (tūkst.)	Efektyvumas
KEK 100g	Raudona 2017.06.26 Rytas	21,240	259,200	212,400	81,94%
KEK 100g	Raudona 2017.06.25 Rytas	38,640	460,800	386,400	83,85%
KEK 100g	Žalia 2017.06.25 Diena	56,280	691,200	562,800	81,42%
KEK 100g	Mėlyna 2017.06.23 Diena	2,160	25,900	21,600	83,40%
KEK 100g	Žalia 2017.06.23 Rytas	51,480	633,600	514,800	81,25%
KEK 100g	Mėlyna 2017.06.22 Diena	57,240	662,400	572,400	86,41%
KEK 100g	Žalia 2017.06.22 Rytas	54,960	691,200	549,600	79,51%
KEK 100g	Oranžinė 2017.06.21 Diena	25,200	302,400	252,000	83,33%
KEK 100g	Mėlyna 2017.06.14 Diena	44,760	516,100	447,600	86,73%
KEK 100g	Žalia 2017.06.14 Rytas	55,560	691,200	555,600	80,38%
KEK 100g	Mėlyna 2017.06.13 Rytas	58,920	662,400	589,200	88,95%
KEK 100g	Oranžinė 2017.06.13 Diena	56,880	691,200	568,800	82,29%
KEK 100g	Oranžinė 2017.06.12 Diena	8,760	115,200	87,600	76,04%
KEK 100g	Raudona 2017.06.08 Rytas	57,120	691,200	571,200	82,64%
KEK 100g	Žalia 2017.06.08 Diena	29,280	337,000	292,800	86,88%
KEK 100g	Mėlyna 2017.06.07 Diena	38,520	455,400	385,200	84,58%
KEK 100g	Oranžinė 2017.06.03 Rytas	46,920	633,600	469,200	74,05%
KEK 100g	Raudona 2017.06.03 Diena	9,480	115,200	94,800	82,29%
KEK 100g	Oranžinė 2017.06.02 Rytas	56,160	691,200	561,600	81,25%
KEK 100g	Raudona 2017.06.02 Diena	54,010	691,200	540,100	78,14%
KEK 100g	Žalia 2017.06.01 Diena	33,000	385,900	330,000	85,51%
KEK 100g	Raudona 2017.05.23 Rytas	56,310	691,200	563,100	81,47%
KEK 100g	Žalia 2017.05.23 Diena	13,560	155,500	135,600	87,20%
KEK 100g	Mėlyna 2017.05.22 Diena	58,440	662,400	584,400	88,22%
KEK 100g	Oranžinė 2017.05.18 Rytas	5,940	72,000	59,400	82,50%
KEK 100g	Oranžinė 2017.05.17 Rytas	55,440	691,200	554,400	80,21%
KEK 100g	Žalia 2017.05.17 Diena	56,260	691,200	562,600	81,39%
KEK 100g	Žalia 2017.05.16 Diena	48,720	612,900	487,200	79,49%
KEK 100g	Mėlyna 2017.05.11 Rytas	21,720	259,200	217,200	83,80%
KEK 100g	Oranžinė 2017.05.10 Rytas	48,120	691,200	481,200	69,62%
KEK 100g	Raudona 2017.05.10 Diena	55,090	691,200	550,900	79,70%
KEK 100g	Raudona 2017.05.09 Diena	33,960	429,100	339,600	79,14%
KEK 100g	Mėlyna 2017.05.04 Rytas	31,610	386,400	316,100	81,81%

6 pav. Gamybos rezultatų duomenų *.xlsx* formatu fragmentas

Labai svarbu pažymėti, kad pagal dabartinę įmonės stebėsenos sistemos specifiką, gamybos rezultatų skaičiavimas užbaigiamas ir pradedamas iš naujo pamainų pasikeitimo metu, kas 12 valandų. Tai gali neigiamai veikti galutinį prognozavimo modelį, kadangi trumpa gamyba iškreipia vienetinius rezultatus dėl iš dalies didelių gamybinio perėjimo iš vieno produkto į kitą laiko sąnaudų. Šiame darbe aprašomo modelio tikslas – kuo tikslesnis gamybos efektyvumo suprognozavimas, atsižvelgiant į gaminamą produkto rūšį, formatą ir gamybos kiekį, o ne pamainos bendro 12 valandų darbo laiko efektyvumo prognozavimas.

Iš gautų duomenų matoma, kad per nustatytą laikotarpį „Mars Lietuve“ fabrike iš viso pagaminta daugiau nei 76,8 tūkst. tonų produkcijos, tai yra daugiau nei 1,01 milijardo užpildytų maišelių gyvūnų augintinių maisto. Bendras keturių pagrindinių gamybinių linijų statistinis efektyvumas – 86,84 procentai.

Kaip aprašyta anksčiau, teisingas pradinių duomenų atrinkimas ir paruošimas neuroniniams tinklams yra viena svarbiausių proceso dalių, vėliau tiesiogiai veikiančių visų toliau sekančių procesų kokybę. Šiuo atveju, siekiant pritaikyti surinktus statistinius duomenis *Matlab* programinei platformai, pradiniai duomenys iš gamybos eigos stebėsenos sistemos perkeliami į *Excel* skaičiuoklės failą ir ten apdorojami:

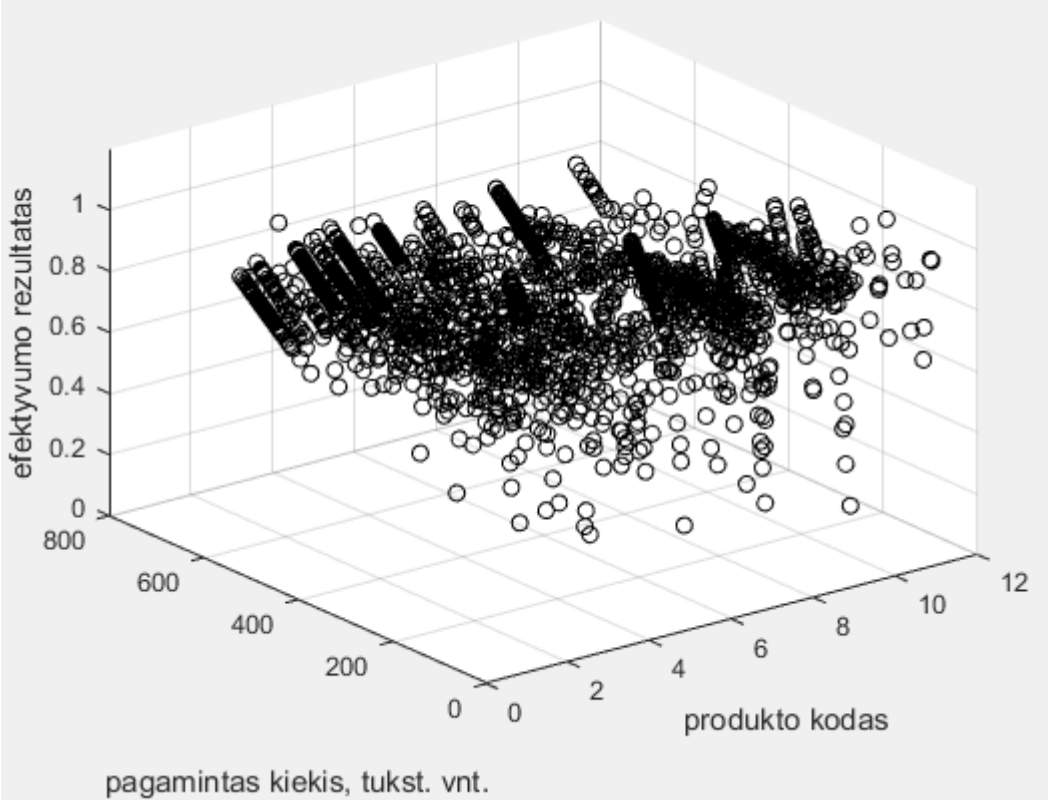
1. Priskirtas gamybinės linijos numeris: PL1 – 1; PL2 – 2; PL3 – 3; PL4 – 4;
2. Kiekvienam produktui priskirtas skaitinis eilės numeris: Kitekat – 1; Sheba RA – 2; WHISKAS Archi – 3 ir t.t. Viso 11 produkto rūšių;
3. Atskirti galimi produkcijos svorio formatai: 50g – 50; 85g – 85 ir 100g – 100;
4. Priskirti pamainos numeriai: mėlyna pamaina – 1, oranžinė – 2, raudona – 3 ir žalia – 4;
5. Išskirtas realiai pagamintas kiekis tūkst. vienetų;
6. Išskirtas užduoties įvykdymo efektyvumas procentais.

Linijos kodas	Produkto kodas	Formatas	Pamainos kodas	Pagaminta (tūkst.)	Efektyvumas
1	1	100	1	21,600	83,40%
1	1	100	1	572,400	86,41%
1	1	100	1	447,600	86,73%
1	1	100	1	589,200	88,95%
1	1	100	1	385,200	84,58%
1	1	100	1	584,400	88,22%
1	1	100	1	217,200	83,80%
1	1	100	1	316,100	81,81%
1	1	100	1	426,000	82,98%
1	1	100	1	563,900	81,58%
1	1	100	1	592,800	85,76%
1	1	100	1	585,600	84,72%
1	1	100	1	590,100	85,37%
1	1	100	1	13,900	89,10%
1	1	100	1	316,800	80,84%
1	1	100	1	608,100	87,98%
1	1	100	1	592,800	85,76%
1	1	100	1	589,800	85,33%
1	1	100	1	278,400	80,56%
1	1	100	1	193,200	82,81%
1	1	100	1	242,300	77,91%
1	1	100	1	523,800	75,78%
1	1	100	1	217,100	76,93%
1	1	100	1	433,500	80,07%
1	1	100	1	530,000	76,68%
1	1	100	1	166,800	77,22%
1	1	100	1	61,200	81,71%
1	1	100	1	538,700	77,94%
1	1	100	1	531,100	76,84%
1	1	100	1	329,800	70,68%

7 pav. Adaptuotų gamybos rezultatų duomenų fragmentas

Gautos 3150 duomenų eilutės su šešių kintamųjų imtimi (7 pav.). Duomenys atvaizduoti trijų dimensijų grafike, kur išskirtas produkto rūšis, pagamintas kiekis tūkst. vnt. ir gamybos efektyvumas (8 pav.).

Modelio prognozavimo tikslumo patikrinimui analogiškai apdorota vieno mėnesio gamybos efektyvumo rodiklio duomenys nuo 2017 metų liepos 1 dienos iki 2017 metų liepos 31 dienos. Gauta 278 eilučių po 6 stulpelius matrica.



8 pav. Bendras duomenų pasiskirstymas *x, y, z* dimensijų grafike

Kadangi siekiama sukurti įmonės gamybos efektyvumo prognozavimo modelį dirbtiniais neuroniniais tinklais, nustatoma, kad pirmi penki kintamieji (gamybos linija, produktas, gaminio formatas, pamaina, gamybos kiekis) yra įvesties reikšmės tinkle. Tikslo reikšmė – pagaminto produkto efektyvumas procentais, apskaičiuojamas pagal 3.1.2 skyriaus „Įmonės procesų efektyvumo vertinimo metodika“ nurodytą formulę.

3.2. Dirbtinio neuroninio tinklo mokymas gamybos efektyvumui prognozuoti

Kaip aprašyta šio darbo 2.6 skyriuje „DNT mokymo funkcijos pasirinkimas“, siekiant kuo tikslesnio galutinio prognozės modelio, labai svarbus yra įvesties ir išvesties duomenų apdorojimo funkcijos pasirinkimas. Šios funkcijos transformuoja pateiktas įvesties ir tikslo reikšmes į reikšmes, kurios tiksliau atitinka tinklo mokymo poreikius.

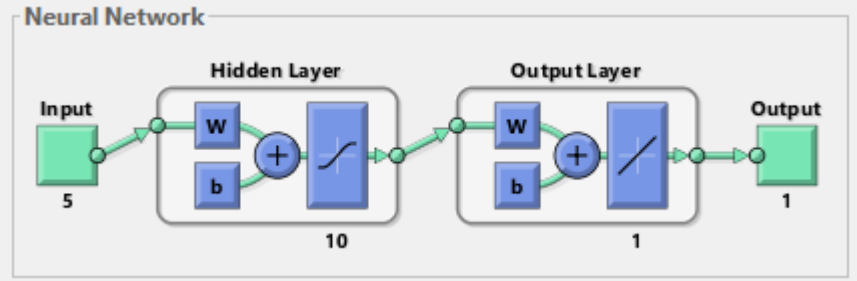
Norint atrinkti šiam neuroninių tinklų prognozės modeliui geriausiai tinkamą mokymo funkciją ir tinklo architektūrą, *Matlab* programinėje platformoje sukurti tiesioginio sklidimo neuroniniai tinklai (angl. feedforward) su visais penkiais 2.6 skyriuje „DNT mokymo funkcijos pasirinkimas“ aprašytais algoritmais: Levenberg-Marquardt (*trainlm*); lankstaus atgalinio sklidimo (angl. resilient

backpropagation) (*trainrp*); konjugacijos gradiento grąžinimo (angl. conjugate gradient algorithm) (*trainscg)*; Quasi-Newton (*trainbfg)* ir kintamojo mokymosi lygio grąžinimo *(traingdx)*. Po to šie penki algoritmais ištestuoti kaskadinio sklidimo (angl. cascade forward) tinklo architektūroje.

Kaip aprašyta šio darbo 2.5 skyriuje „Neuroninio tinklo tikrinimas ir vertinimas“, sukurtų dirbtinių neuroninių tinklų charakteristikos vertinamos pirmiausia pagal vidutinę kvadratinę paklaidą (MSE) ir suminę kvadratinę paklaidą (SSE) nuo tryliktojo mėnesio realių gamybos efektyvumo rezultatų. Siekiant kuo objektyviau įvertinti gautų DNT modelių patikimumą, šiame darbe taip pat įvertintas individualus kiekvieno tinklo bendras regresijos koeficientas bei vidutinė paklaidą, apskaičiuota tarp realaus ir suprognozuoto kiekvieno individualaus rezultato. Pastarasis rodiklis skaitomas svarbiausiu, kadangi vienas iš šio darbo tikslų yra sukurti DNT modelį gamybos efektyvumo prognozavimui naudojant realius gamybinės įmonės duomenis ir susitelkiant į kiekvieno suprognozuoto rezultato vertę lyginant su realia. Be to, rezultatai yra sulyginami su dabar „Mars Lietuva“ naudojamu statistiniu imties vidurkio modeliu.

3.2.1. Tiesioginio sklidimo neuroninis tinklas

Pradiniams bandymams naudojama standartinė *Matlab* tiesioginio sklidimo neuroninio tinklo struktūra, kai naudojami du sluoksniai su 10 neuronų paslėptame sluoksnyje. Standartinė perdavimo (angl. transfer) funkcija paslėptiems sluoksniams yra *tansig*, o standartinė funkcija išvesties sluoksnyje yra *purelin* (9 pav.).



9 pav. Tiesioginio sklidimo neuroninio tinklo struktūra

Kadangi duomenų imtis yra gana didelė, panaudota *dividerand* duomenų rinkinių padalinimo funkcija, kai duomenys atsitiktine tvarka padalinti į tris pogrupius – mokymo, validacijos ir testavimo. Naudotos standartinės proporcijos: 70 proc. duomenų kiekio mokymui ir po 15 proc. validacijai bei testavimui.

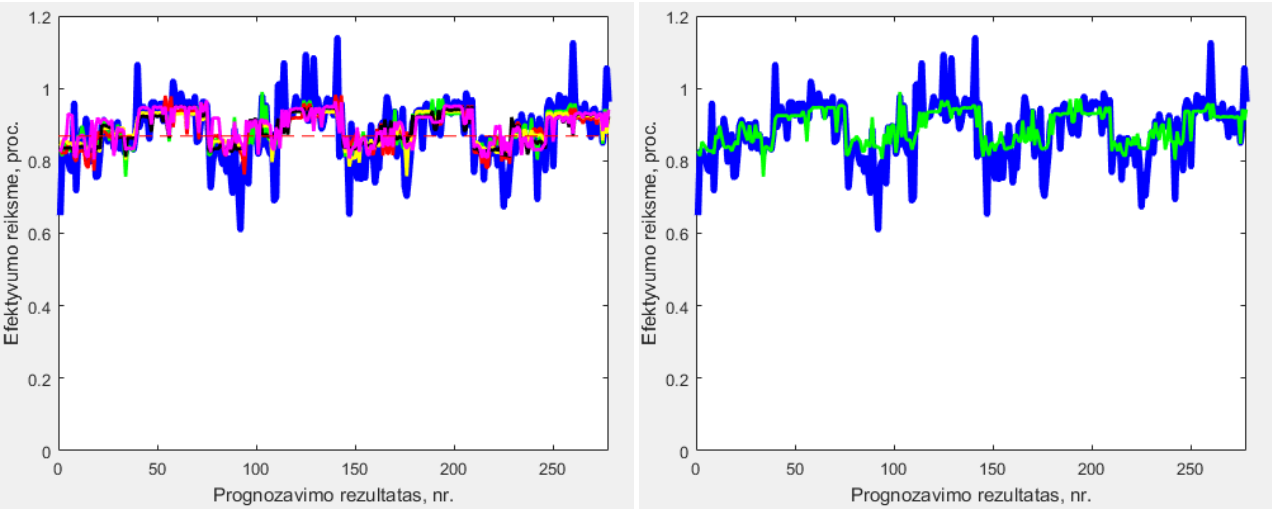
Tinklas yra patikrintas su penkiais pagrindiniais *Matlab* programinės aplinkos algoritmais, lyginant suprognozuotas vertes su realiomis tryliktojo mėnesio vertėmis (programinis kodas pateiktas 1 priede). Gauti rezultatai pateikti 1 lentelėje. Iš jų matosi, kad geriausia dirbtinio neuroninio tinklo struktūra, vertinant visais keturiais parametrais, gauta naudojant Levenberg-Marquardt (*trainlm*)

algoritmą. Suprognozuotas rezultatas skiriasi nuo realaus vidutiniškai ± 4,10 proc. ir, galima teigti, kad lyginant su dabar „Mars Lietuva“ naudojamu rezultatų imties vidurkiu (paklaida ± 6,9 proc.), yra 40,6 proc. patikimesnis. Be to, lyginant su kitais penkiais tinklų modeliais, prasčiausius rezultatus sugeneraveęs *traingdx* modelis pasižymi ± 4,93 proc. vidutine paklaidą, kas vėlgi yra 28,6 proc. patikimiau už dabar naudojamą rezultatų imties vidurkio metodą.

1 lentelė. Tiesioginio sklidimo tinklo su skirtingais algoritmais rezultatai

	Matlab funkcija	Algoritmo pavadinimas	Bendras regresijos koeficientas	Suminė kvadratinė klaida (SSE)	Vidutinė kvadratinė klaida (MSE)	Vidutinis rezultatas, proc.	Paklaidų nuo realių rezultatų vidurkis
Y1_lm	*trainlm*	Levenberg-Marquardt	0.579	0.9604	0.0035	89.05%	4.10%
Y2_rp	*trainrp*	Lankstaus atgalinio sklidimo	0.50609	1.1742	0.0042	88.64%	4.54%
Y3_scg	*trainscg*	Konjugacijos gradiento grąžinimo	0.49995	1.1071	0.004	88.79%	4.42%
Y4_bfg	*trainbfg*	Quasi-Newton	0.4935	1.1208	0.004	88.58%	4.37%
Y5_gdx	*traingdx*	Kintamojo mokymosi lygio grąžinimo	0.45315	1.3472	0.0048	89.27%	4.93%
Yx_i Rezultatų imties vidurkis	-	-	-	-	-	86.84%	6.90%

Siekiant iliustruoti rezultatus, nubraižyti 278 realių efektyvumo rezultatų procentais kreivės ir keliais algoritmais suprognozuotų išvesčių kreivių grafikas (10 pav.). Šalia pavaizduoti geriausius rezultatus sugeneravusio DNT modelio su Levenberg-Marquardt (*trainlm*) algoritmu efektyvumo rezultatai.



10 pav. Realių rezultatų ir prognozuojamų reikšmių kreivės

Iš šių grafikų matosi, kad visi DNT modeliai gerai suvokia prognozuojamų reikšmių tendencijas. Sunkiausiai suprantami nestandartiniai nuokrypiai, kai dėl tam tikrų gamybinio proceso priežasčių efektyvumo rezultatas smarkiai nukrypsta nuo bendrų tendencijų. Pavyzdžiui, nenatūraliai aukšti rezultatai dėl vėlesnio nei pagal planą gamybos stabdymo prieš gamyklos plovimą ar remonto pamainas (rezultatai virš 100 proc.), ar itin maži efektyvumo rezultatai (mažesni nei 70 proc.), dažniausiai sugeneruojami po pamainų pasikeitimo, kai gamybos apskaitos sistema pradeda skaičiuoti efektyvumo rezultatus iš naujo.

3.2.2. Dirbtinio neuroninio tinklo generalizacijos patikrinimas

Siekiant patikrinti ar pradinių duomenų apimtis yra pakankamai didelė gerai tinklo generalizacijai ir ištestuota kiek įmanoma optimaliausia prognozavimo modelio struktūra, pradinė duomenų imtis (3150 stulpeliai po 6 eilutes) sumaišyta ir dar kartą padalinta į 90 proc. ir 10 proc. dalis. Kaip aprašyta šio darbo 2.7 skyriuje, kiekviena klaidos sklidimo atgal mokymo sesija pradedama skirtingais pradiniais svoriais ir slenksčiais, be to, suformuojami skirtingi duomenų mokymo, validacijos ir testavimo rinkiniai. Šios, kartais visiškai skirtingos pradinės sąlygos, dažnai suponuoja vis kitus sprendimus tai pačiai problemai.

Kiekviena prieš tai sukurta DNT architektūra mokoma dešimt kartų, remiantis pirma duomenų dalimi (2834 stulpeliai po 6 eilutes) ir su kiekvieno tinklo vidutine kvadratine klaida (angl. mean square error) įvertinta pagal antrąją dalį (315 stulpeliai po 6 eilutes).

Tokiu būdu užtikrinama, kad kiekvienas suprojektuotas neuroninis tinklas yra mokomas pradedant skirtingais pradiniais svoriais ir slenksčiais, o kartu ir skirtingu 90 proc. duomenų dalies padalinimu į tris duomenų rinkinius (mokymo, validacijos ir testavimo). Taip kiekvieną kartą panaudojami kiti imties testavimo rinkiniai (Matlab programinis kodas pateiktas 2 priede). Be to, išvengiama vieno tinklo testavimo duomenų rinkinio panaudojimo, kaip kito tinklo mokymo ar validacijos rinkinio, rizikos [27].

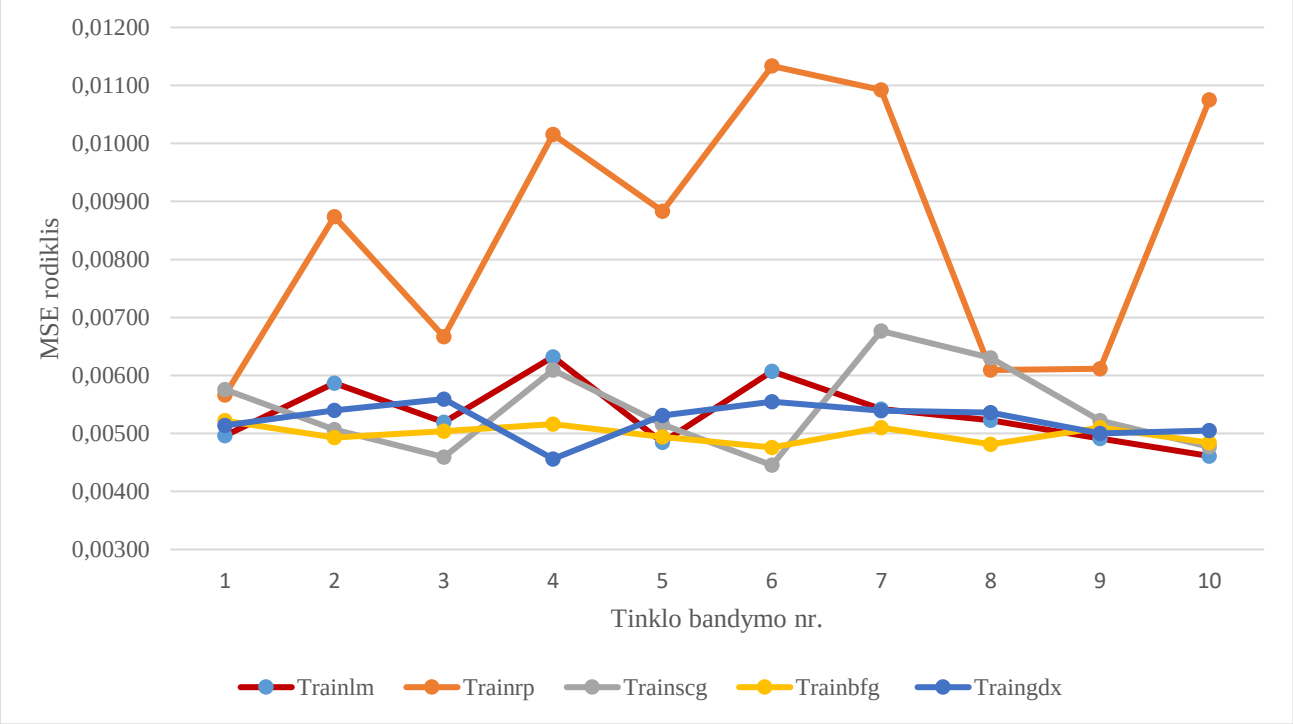
Jeigu tos pačios architektūros ir to pačio algoritmo tinklo vidutinė kvadratinė klaida per visus dešimt bandymų smarkiai varijuos, galima teigti, kad sukurtas tinklas nepakankamai generalizuojasi ir reikalingi papildomi veiksmai: duomenų imties didinimas, reguliarizacijos arba ankstyvaus stabdymo metodai.

Gauti rezultatai pateikti 2 lentelėje ir atvaizduoti grafiškai (11 pav.). Iš jų matosi, kad vidutinės kvadratinės klaidos (MSE) rodiklis per visus bandymus išliko gana tolygus, o ypač Levenberg-Marquardt (*trainlm*) algoritme (rodiklio svyravimo amplitudė nuo 0,00461 reikšmės iki 0,00607).

2 lentelė. DNT modelių vidutinės kvadratinės paklaidos

Matlab funkcija	Algoritmo pavadinimas	Vidutinė kvadratinė klaida (MSE)									
		1	2	3	4	5	6	7	8	9	10
Trainlm	Levenberg-Marquardt	0,00496	0,00587	0,00520	0,00632	0,00485	0,00607	0,00542	0,00523	0,00491	0,00461
Trainrp	Lankstaus atgalinio sklidimo	0,00566	0,00874	0,00667	0,01015	0,00883	0,01133	0,01092	0,00609	0,00611	0,01075
Trainscg	Konjugacijos gradiento grąžinimo	0,00576	0,00507	0,00459	0,00609	0,00515	0,00445	0,00677	0,00630	0,00522	0,00477
Trainbfg	Quasi-Newton	0,00522	0,00493	0,00504	0,00516	0,00494	0,00476	0,00510	0,00481	0,00510	0,00484
Traingdx	Kintamojo mokymosi lygio grąžinimo	0,00514	0,00540	0,00559	0,00456	0,00531	0,00555	0,00539	0,00536	0,00500	0,00505

Geriausias individualus tinklo rezultatas buvo pasiektas *trainscg* – 0,00445, tačiau per dešimt bandymų svyravimo amplitudė buvo didesnė. Didžiausia vidutinės kvadratinės klaidos rezultatų sklaida per dešimt sukurtų tinklo modelių buvo užfiksuota naudojant *trainrp* algoritmą.



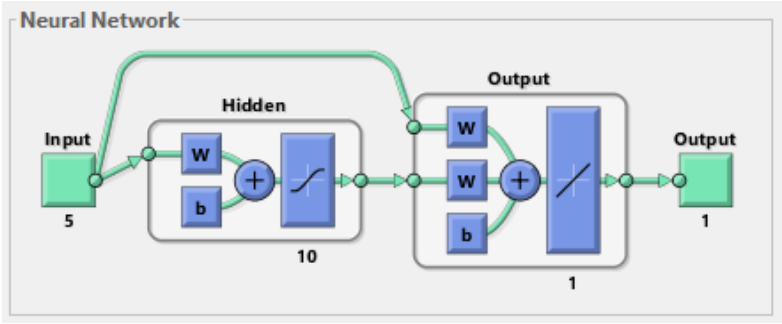
11 pav. Skirtingų algoritmų *MSE* rodiklio rezultatai per 10 bandymų

Remiantis šiais rezultatais galima daryti išvadą, kad dirbtinių neuroninių tinklų modeliai prognozavimo uždaviniams spręsti yra sudaryti naudojant pakankamą parametrų skaičių ir gautų modelių adaptyvumo lygis yra ne per didelis, todėl papildomi tinklo generalizacijos gerinimo veiksmai, aprašyti šio darbo 2.7 skyriuje, yra nereikalingi.

3.2.3. Kaskadinio sklidimo neuroninis tinklas

Kaip aprašyta šio darbo 2.3 skyriuje „DNT architektūros parinkimas“, kaskadinio sklidimo neuroninis tinklas (angl. cascade-forward net) yra klaidos sklidimo atgal DNT, savo veikimo principu panašus į tiesioginio sklidimo neuroninį tinklą, tačiau turintis svorio jungtį iš įvesties ir iš kiekvieno prieš tai esančio sluoksnio į sekantį. Iš *Matlab* pateikiamos kaskadinės tinklo architektūros (12 pav.) matosi, kad pirmas sluoksnis turi svorio jungtį į antrą, antras sluoksnis į trečią, o kartu perduodama svorio jungtis iš įvesčių į visus tris sekančius sluoksnius.

Programuojant tinklo modelį vėlgi panaudotas standartinis *dividerand* duomenų padalinimas – 70 proc. mokymui ir po 15 proc. validacijai bei testavimui. Kartu kaskadinio tinklo modelis patikrinamas su penkiais pagrindiniai *Matlab* programinės skaičiavimo platformos algoritmais (programinis kodas pateiktas 3 priede).



12 pav. Kaskadinio sklidimo neuroninio tinklo architektūra

Gauti rezultatai pateikti 3 lentelėje. Kaip ir tiesioginio sklidimo DNT, vertinant visais keturiais parametrais, kaskadinio sklidimo architektūroje geriausias dirbtinio neuroninio tinklo modelis gautas naudojant Levenberg-Marquardt (trainlm) algoritmą (± 4,21 proc.).

3 lentelė. Kaskadinio sklidimo tinklo su skirtingais algoritmais rezultatai

	Matlab funkcija	Algoritmo pavadinimas	Bendras regresijos koeficientas	Suminė kvadratinė klaida (SSE)	Vidutinė kvadratinė klaida (MSE)	Vidutinis rezultatas, proc.	Paklaidų nuo realių rezultatų vidurkis
Y6_CF_lm	*trainlm*	Levenberg-Marquardt	0.56545	0.9814	0.0035	89.07%	4.21%
Y7_CF_rp	*trainrp*	Lankstaus atgalinio sklidimo	0.49431	1.2712	0.0046	88.77%	4.73%
Y8_CF_scg	*trainscg*	Konjugacijos gradiento grąžinimo	0.4625	1.1688	0.0042	88.52%	4.67%
Y9_CF_bfg	*trainbfg*	Quasi-Newton	0.52161	1.2092	0.0043	89.00%	4.58%
Y10_CF_gdx	*traingdx*	Kintamojo mokymosi lygio grąžinimo	0.4793	1.2632	0.0045	88.65%	4.85%
Yx_i Rezultatų imties vidurkis	-	-	-	-	-	86.84%	6.90%

Visi modeliai prognozavo tiksliau nei dabar „Mars Lietuva“ naudojamas rezultatų vidurkių metodas, tačiau, lyginant su tiesioginio sklidimo paklaidomis nuo realių rezultatų, kaskadinio sklidimo tinklo paklaidos yra didesnės. Todėl galima daryti išvadą, kad papildomi svoriai, perduodami iš įvesčių sluoksnio į kiekvieną toliau einantį sluoksnį atskirai, teigiamos įtakos dirbtinio neuroninio tinklo struktūrai nedaro.

Tikrinti kaskadinio sklidimo tinklo modelių generalizacijos nėra tikslo, kadangi architektūros struktūra yra itin panaši į tiesioginio sklidimo modelius, be to, 3 lentelėje matosi, kad vidutinės kvadratinės klaidos rodiklis (MSE) yra itin artimas prieš tai gautiems ir svyruoja sąlyginai nežymiai.

3.3. Atrinkto neuroninio tinklo optimizavimas

Iš sukurtų dirbtinių neuroninių tinklų modelių atrinktas geriausią rezultatą sugeneravusi struktūra – tiesioginio sklidimo Levenberg-Marquardt (*trainlm*) algoritmo modelis, kurio suprognozuotas procentinis rezultatas individualiai skiriasi nuo realios vertės vidutiniškai 4,10 proc.

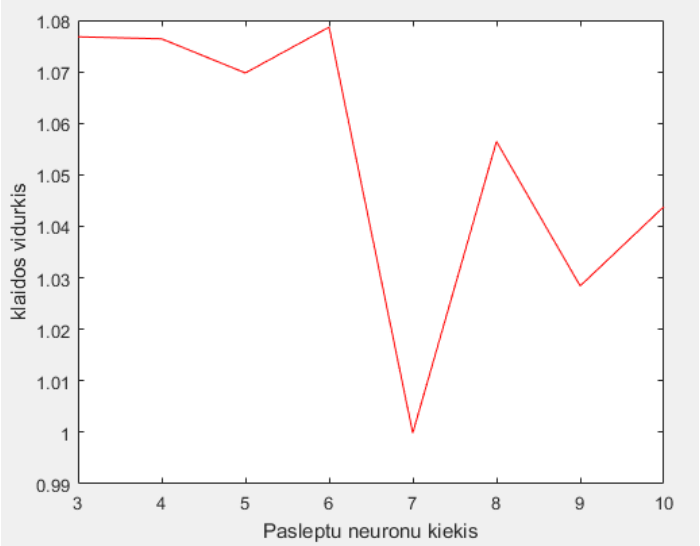
Norint patobulinti atrinktą dirbtinį neuroninį tinklą ir siekiant tikslesnių prognozavimo rezultatų, patikrinama tiesioginio sklidimo *trainlm* modelis su skirtingu neuronų paslėptame sluoksnyje skaičiumi (*Matlab* programinis kodas 4 priede). Neuronų skaičius įvesties sluoksnyje yra lygus įvesčių parametrų skaičiui, o neuronų skaičius išvesčių sluoksnyje yra lygus pageidaujamų išvesčių skaičiui (šiuo atveju procentais apibrėžtam gamybos proceso efektyvumo rodikliui). Tuo tarpu, nustatyti DNT optimalų neuronų skaičių paslėptame sluoksnyje yra daug sudėtingesnis procesas. Kuo neuronų daugiau, tuo daugiau tinklas turi svorių jungčių ir tuo yra sudėtingesnis [32]. Iš kitos pusės, per daug sudėtingas tinklas yra sunkiai apdorojamas ir gali eigoje pasidaryti pernelyg adaptyvus, o kartu prarasti improvizavimo su naujomis duomenų imtimis sugebėjimus.

4 lentelė. DNT su skirtingu neuronų skaičiumi SSE rodiklio rezultatai

Neuronų skaičius	Suminė kvadratinė klaida (SSE)										Vidurkis
	1	2	3	4	5	6	7	8	9	10	
3	1.0975	1.1132	1.0155	1.0589	1.0649	1.0878	1.0787	1.1055	1.0688	1.0771	1.0768
4	1.0289	1.1104	0.9832	1.1098	1.0969	1.0653	1.0780	1.0841	1.0810	1.1263	1.0764
5	1.0306	1.0638	1.0850	1.0311	1.0385	1.0475	1.0459	1.0682	1.0292	1.2579	1.0698
6	1.1008	1.1412	1.0454	1.0428	1.1027	1.0618	1.1090	1.0052	1.1055	1.0721	1.0787
7	1.0212	1.0247	0.9219	0.9181	1.0451	1.0454	1.0400	1.1161	0.9398	0.9264	0.9999
8	1.1397	1.1352	1.0283	1.0725	1.0771	1.0040	1.0518	0.9743	0.9775	1.1042	1.0565
9	0.9464	0.9820	1.0525	1.1052	1.0312	0.9746	1.1085	0.9767	1.0629	1.0442	1.0284
10	0.9788	1.1523	1.0713	0.9879	1.0560	1.0398	1.0920	0.9227	1.0243	1.1131	1.0438

Paskaičiuotas SSE rezultato vidurkis ir nubraižytas jo priklausomybės nuo neuronų skaičiaus grafikas (13 pav.), kuriame matosi, kad geriausią suminės kvadratinės klaidos vidurkio rezultatą, artimiausią 1 vertei, sugeneravo DNT su septyniais neuronais paslėptame sluoksnyje.

Siekiant optimalios DNT neuronų konstrukcijos, *Matlab* programinėje platformoje paruoštas neuroninio tinklo kodas sugeneruoja po dešimt dirbtinių daugiasluoksnių neuroninių tiesioginio sklidimo tinklų su *trainlm* algoritmu, susidedančių iš nuo trijų iki dešimties neuronų paslėptame sluoksnyje. Sukurti modeliai mokyti, validuoti, testuoti po dešimt kartų ir vertinti pagal suminės kvadratinės klaidos (SSE) rezultatą (4 lentelė).



13 pav. Suminė kvadratinės klaidos (SSE) vidurkis

Atitinkamai paruoštas tiesioginio sklidimo Levenberg-Marquardt (*trainlm*) algoritmo modelis su septyniais neuronais paslėptame sluoksnyje. Be to, panaudotas tinklo generalizacijos gerinimo būdas – ankstyvas stabdymas. Čia apibrėžiamas tinklo mokymo stabdymas, kai SSE klaida pasiekia iš anksto apibrėžtas ribas. Programos kode suminės kvadratinės klaidos maksimalios ir minimalios vertės pasirinktos atitinkamai kaip didžiausia ir mažiausia vidutinė SSE vertės iš 4 lentelės. Tinklui nustatytas 100 mokymo epochų ciklas.

5 lentelė. DNT su septyniais neuronais paslėptame sluoksnyje rezultatai

	Matlab funkcija	Algoritmo pavadinimas	Bendras regresijos koeficientas	Suminė kvadratinė klaida (SSE)	Vidutinė kvadratinė klaida (MSE)	Vidutinis rezultatas, proc.	Paklaidų nuo realių rezultatų vidurkis
Y11_lm	*trainlm*	Levenberg-Marquardt	0.56729	0.925	0.0033	88.99%	4.01%

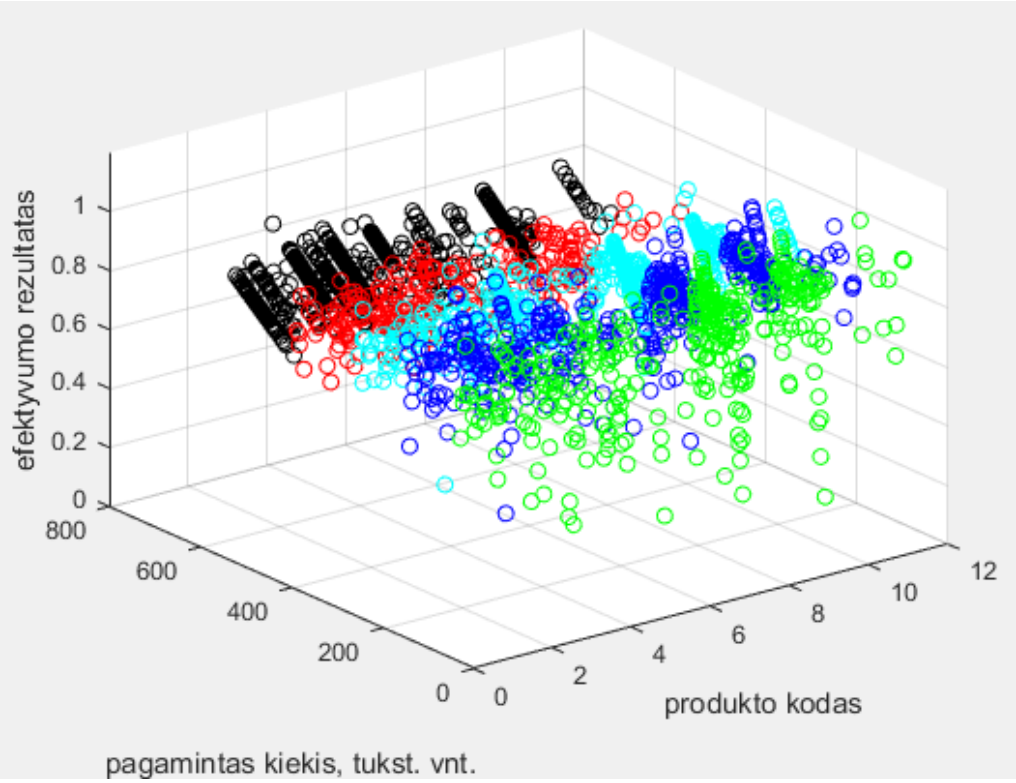
Gauti sukurto dirbtinio neuroninio tinklo prognozavimo rodikliai (bendras regresijos ir vidutinės kvadratinės klaidos koeficientai) yra geresni už prieš tai buvusio pirminio modelio, o kartu šio tinklo

suprognozuotas procentinis rezultatas individualiai skiriasi nuo realios vertės vidutiniškai 4,01 proc. (5 lentelė).

3.4. Duomenų imties triukšmo šalinimas klasterizuojant pradinę imtį

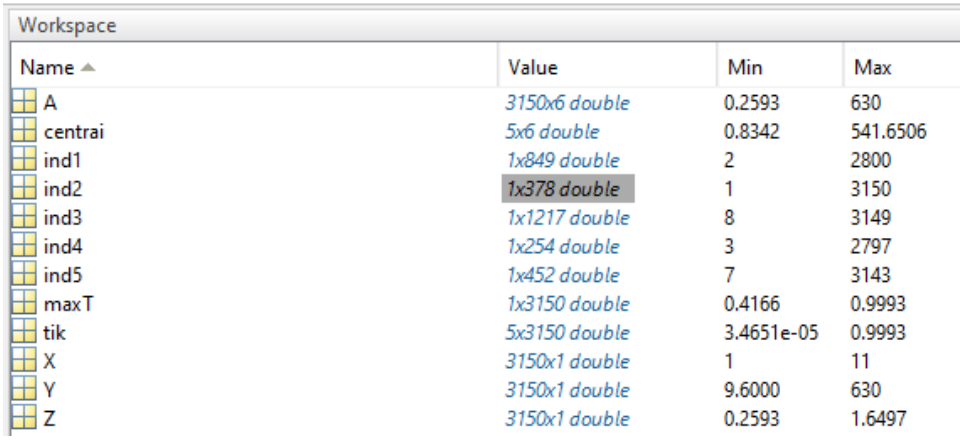
Kaip aprašyta šio darbo 3.2.2 skyriuje, atliekant dirbtinių neuroninių tinklų modelių, sukurtų iš tos pačios duomenų imties, analizę, nustatyta, kad pradinių duomenų imtis yra pakankama gera tinklo generalizacijai. Iš kitos pusės, modelis, paruoštas iš itin didelio parametrų kiekio, yra linkęs prisitaikyti prie duomenų imtyje esančių triukšmų. Kadangi šiuo metu „Mars Lietuva“ gamybos efektyvumo rezultatų surinkimo būdas iškraipo galutinius rezultatus, nes du kartus per parą pasikeičiant pamainoms sustabdomas skaičiavimas visose linijose ir pradedamas skaičiuoti pagal naujai dirbančią pamainą, galima daryti prielaidą, kad taip susidaro mažos apimties duomenys, tai yra duomenų triukšmo šaltinis, kuris iškraipo bendrą duomenų imtį ir blogina prognozavimo kokybę. Dirbtinis neuroninis tinklas nesuvokia ir neužčiuopia šios tendencijos, kadangi neturi pradinių įvesčių, susietų su pamainų darbo laiku, ir turinčių pakankamą koreliaciją.

Siekiant atrinkti imties triukšmą generuojančiu duomenis, *Matlab* programinėje skaičiavimo platformoje *Fuzzy c-means* (*Matlab* funkcija – fcm) klasterizavimo metodu imtis suskirstyta į penkis atskirus klasterius (programinis kodas 5 priede). *Fuzzy c-means* tai klasterizavimo metodas leidžiantis kiekvieną duomenų tašką priskirti keliems klasteriams apibrėžiant skirtingu priklausomybės laipsniu [33]. Gauti klasteriai iliustruoti skirtingomis spalvomis grafike (14 pav.).



14 pav. fcm metodu klasterizuotos imties atvaizdavimas *x*, *y*, *z* dimensijų grafike

Metodo pagalba iš bendros 3150x6 duomenų imties eliminuota kraštinė grafike žaliai pavaizduota duomenų imtis *ind2*, kurią sudaro 378 nariai (15 pav.). Analizuojant klasterizavimo rezultatus matosi, kad modelis suponuoja eliminuoti duomenis iš mažiausių gamybos periodų, vertinant vienetais.



15 pav. C-means klasterizuotų duomenų ind2 imtis

Tai yra logiška iš praktinės pusės ir patvirtina prieš tai padarytą prielaidą, kad nesavalaikis gamybos skaičiavimo stabdymas dėl pamainų pasikeitimo blogai veikia bendrą duomenų imties kokybę. Gauti DNT modelio rezultatai surašyti į 6 lentelę, kur matyti, kad paklaidų nuo realių rezultatų vidurkis yra dar mažesnis ir siekia ± 3,69 proc.

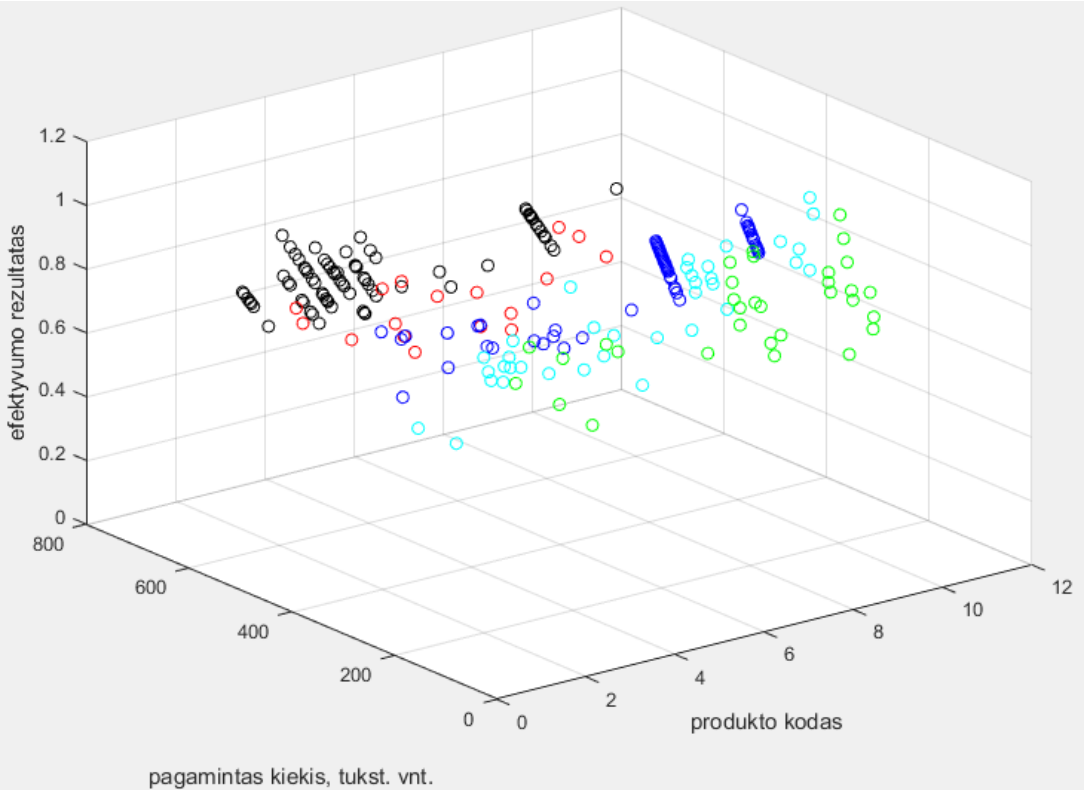
6 lentelė. DNT su klasterizuota mokymo imtimi rezultatai

	Matlab funkcija	Algoritmo pavadinimas	Bendras regresijos koeficientas	Suminė kvadratinė klaida (SSE)	Vidutinė kvadratinė klaida (MSE)	Vidutinis rezultatas, proc.	Paklaidų nuo realių rezultatų vidurkis
Y12_lm	*trainlm*	Levenberg-Marquardt	0.659	0.9298	0.0033	88.83%	3.69%

Remiantis šiais rezultatais, galima daryti išvadą, kad surinkus gamybos rezultatų duomenis tik per gaminamos produkto rūšies prizmę ir stabdant produkto gamybos fiksavimą kartu su perėjimu į kitą rūšį, o ne kartu su pamainos pasikeitimu, gamybos efektyvumo skaičiavimo metodika taptų objektyvesnė, o kartu atsirastų galimybė tikslesniam prognozavimui.

3.5. Tikslo duomenų imties tobulinimas

Galima daryti išvadą, kad tikslinė rezultatų imtis yra taip pat veikiama duomenų surinkimo metu atsirandančio imties triukšmo. Todėl naudojant tą pačią *Fuzzy c-means* klasterizavimos metodologiją 276 tryliktojo tiriamojo laikotarpio mėnesio duomenys yra analogiškai suskirstyti į penkis klasterius ir atskirta kraštinė mažiausios fiksuotos gamybos duomenų grupė (iki 150 tūkst. gaminamo produkto vienetų).



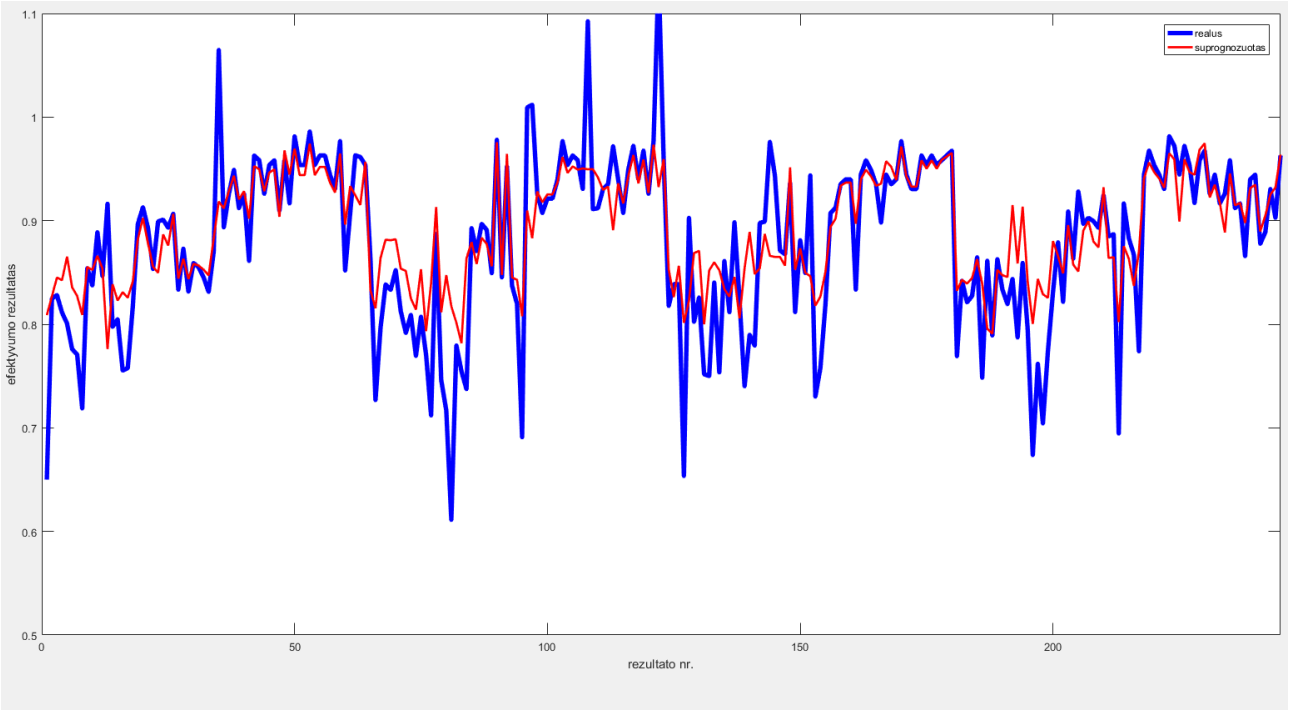
16 pav. fcm metodu klasterizuotos rezultatų imties atvaizdavimas *x, y, z* dimensijų grafike

Gauti klasteriai atvaizduoti skirtingomis spalvomis *x, y, z* dimensijų grafike (16 pav.) ir iš imties eliminuotas žaliai pavaizduotas *ind2* klasteris, kurį sudaro 33 vienetai gamybos rezultatų (*Matlab* programinis kodas pateiktas 6 priede). Tada, naudojant modifikuotą 2772x6 pradinę imtį, vėl sukurtas tiesioginio sklidimo *trainlm* algoritmo modelis su septyniais neuronais paslėptame sluoksnyje, pasitelkiant ankstyvą stabdymą ir nustačius pradinį 100 mokymo epochų ciklą, tačiau šį kartą bandant suprognozuoti tik atrinktus 276 tryliktojo mėnesio gamybos efektyvumo duomenis.

7 lentelė. DNT su klasterizuota mokymo ir tikslo imtimis rezultatai

	Matlab funkcija	Algoritmo pavadinimas	Bendras regresijos koeficientas	Suminė kvadratinė klaida (SSE)	Vidutinė kvadratinė klaida (MSE)	Vidutinis rezultatas, proc.	Paklaidų nuo realių rezultatų vidurkis
Y13_lm	*trainlm*	Levenberg-Marquardt	0.66307	0.6173	0.0025	89.11%	3.17%

Iš gautų rezultatų matosi (7 lentelė), kad sukurto daugiasluoksnio dirbtinio neuroninio tinklo modelio prognozuojamas procentinis rezultatas individualiai skiriasi nuo realios vertės vidutiniškai 3,17 procento ir yra 54 proc. tikslesnis lyginant su dabar „Mars Lietuva“ naudojamu rezultatų imties vidurkiu (paklaida ± 6,9 proc.).



17 pav. galutinio DNT modelio prognozuojamų ir realių rezultatų kreivės

Nubraižius DNT suprognozuotų ir realių reikšmių gamybos efektyvumo rezultatų kreivių grafiką (17 pav.), matosi, kad tinklas gerai suvokia gamybos efektyvumo bendrąsias tendencijas, todėl galima daryti išvadą, kad surinktų įvesčių duomenų variacija koreliuoja su tikslo duomenimis.

3.6. Tinklo tobulinimo galimybės ir rekomendacijos įmonei

Kaip įrodyta klasterizuojant pradinę duomenų imtį, norint turėti dar tikslesnį gamybos efektyvumo prognozavimo modelį, reikėtų rinkti gamybos rezultatų duomenis tik per gaminamos produkto rūšies prizmę ir stabdyti produkto gamybos rezultatų fiksavimą kartu su perėjimu į kitą rūšį, o ne kartu su pamainos pasikeitimu, kaip yra daroma šiuo metu. Dar viena priemonė rezultatų imties gerinimui ir kartu dirbtinio neuraninio tinklo prognozavimo kokybės tobulinimui, būtų duomenų bazėje registruoti ne tik produkto rūšį, tačiau recepto kodą, pakuotės ir pakavimo būdą, kadangi tai yra itin svarbūs faktoriai, darantys tiesioginę įtaką fabriko darbo efektyvumui. Šie papildomi parametrai smarkiai prasiplėstų tinklo įvesčių skaičių, o kartu suteiktų tinklui dabar trūkstamos informacijos, bandant nuspėti dabar esančius didžiausias svyravimo priežastis.

Automatizavus ir įdiegus aprašytą prognozavimo modelį į UAB Mars Lietuva vidinės gamybos apskaitos programą, pagerėtų įmonės procesų valdymas, būtų aiškiau nusakoma gamybos eiga realiu laiku ir ženkliai tiksliau nuspėjami ateities procesai.

Išvados

1. Atlikus dirbtinių neuroninių tinklų panaudojimo galimybių gamybinėje praktikoje analizę, nustatyta, kad plačiausiai DNT naudojami optimizavimo, modeliavimas, kokybės kontrolės, vaizdų atpažinimo ir prognozavimo problemų sprendimui.
2. Išanalizavus dirbtinio neuroninio tinklo kūrimo *Matlab* programinėje aplinkoje procesą, išskirti ir išanalizuoti pagrindiniai jo etapai: duomenų surinkimas, apdorojimas ir padalinimas į funkcines grupes; tinklo struktūros ir architektūros parinkimas; optimalios DNT mokymo funkcijos pasirinkimas; neuroninio tinklo mokymas, tikrinimas, vertinimas bei tobulinimas.
3. Naudojant realius „Mars Lietuva“ gamybos proceso generuojamus duomenis, sukurtas DNT modelis gamybos efektyvumo prognozavimui pagal kiekvieno fabrike gaminamo produkto rūšį, formatą, gamybos kiekį, pamainą ir gamybos liniją. Atlikus daugiau nei šimtą sukurtų įvairių DNT architektūrų modelių testavimus ir patikrinimus, lyginant su realiais „Mars Lietuva“ gamybos duomenimis, nustatyta, kad geriausia dirbtinio neuroninio tinklo struktūra, vertinant visais keturiais parametrais, gauta naudojant tiesioginio sklidimo DNT su Levenberg-Marquardt (trainlm) algoritmu, septyniais neuronais paslėptame sluoksnyje ir naudojant ankstyvo stabdymo metodiką.
4. Sukurto daugiasluoksnio dirbtinio neuroninio tinklo modelio prognozuojamas procentinis rezultatas individualiai skiriasi nuo realios vertės vidutiniškai 3,17 procento ir yra 54 proc. tikslesnis lyginant su dabar įmonės naudojamu rezultatų imties vidurkiu (paklaida ± 6,9 proc.).

Literatūros sąrašas

1. Gurney, K. (2004). An introduction to neural networks. London: Taylor & Francis e-Library.
2. Hagan, M. T., Demuth, H. B., Beale, M. H.,& De Jesus, O. (2014). Neural Network Design, 2nd Edition, Stillwater, Oklahoma: Oklahoma State University.
3. Smith, K. A.,& Gupta, J. N. (2000). Neural networks in business: techniques and applications for the operations researcher. Computers & Operations Research, pp. 1023-1044.
4. Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V.,& Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. Computational and Structural Biotechnology Journal 13, pp. 8-17.
5. Verikas, A. ir and Gelžinis, A. (2003). Neuroniniai tinklai ir neuroniniai skaičiavimai. Kaunas: KTU leidykla Technologija.
6. Ata, R. (2015). Artificial neural networks applications in wind energy systems: a review. Renewable and Sustainable Energy Reviews (49), pp. 534-562.
7. Hu, X.,& An, R. (2011). Modeling and Simulation of Manufacturing Systems in Unstable Environment. Proceedings of the World Congress on Engineering, Vol I.
8. Ghamarian, N., Azmah Hanim, M. A., Nahavandi, M., Zainal, Z.,& Ngee Lim, H. (2017). Application of Artificial Neural Networks for the Optimisation of Wetting Contact Angle for Lead Free Bi-Ag Soldering Alloys. Pertanika J. Sci. & Technol..
9. Betiku, E.,& Taiwo, A. E. (2015). Modeling and optimization of bioethanol production from breadfruit starch hydrolyzate vis-à-vis response surface methodology and artificial neural network. Renewable Energy, v. 74., pp. 87-94.
10. Pfrommer, J., Zimmerling, C., Liu, J., Kärger, L., Henning. F.,& Beyerera, J. (2018). Optimisation of manufacturing process parameters using deep neural networks as surrogate models. 51st CIRP Conference on Manufacturing Systems.
11. Lechevalier, D., Ak, R., Lee, T. Y., Hudak. S.,& Foufou, S. (2015). A Neural Network Meta-Model and its Application for Manufacturing. 2015 IEEE International Conference on Big Data.
12. Bamdad, K., Cholette, M. E., Guan, L.,& Bell, J. (2017). Building Energy Optimisation Using Artificial Neural Network and Ant Colony Optimisation. AIRAH and IBPSA's Australasian Building Simulation 2017 Conference, Melbourne.
13. Li, Y. (2013). Application of Improved SOM Neural Network in Manufacturing Process Quality Control. School of Mechanic and Electronic Engineering, Wuhan University of Technology.
14. Bahlmann, C., Ritter. H.,& Heidemann, G. (1999). Artificial Neural Networks for Automated Quality Control of Textile Seams. AG Neuroinformatik, Technische Fakultät, Universität Bielefeld.

15. Kant, G.,& Sangwan, K. S. (2015). Predictive Modelling and Optimization of Machining Parameters to Minimize Surface Roughness using Artificial Neural Network Coupled with Genetic Algorithm. 15th CIRP Conference on Modelling of Machining Operations, Catcliffe.
16. Sun, T. H., Tien, F. C., Tien, F. C.,& Kuo, R. J. (2016). Automated thermal fuse inspection using machine vision and artificial neural networks. Journal of Intelligent Manufacturing, Volume 27, p. 639–651.
17. Scrime, L.,& Beuth, J. (2018). A multi-scale convolutional neural network for autonomous anomaly detection and classification in a laser powder bed fusion additive manufacturing process. Additive Manufacturing, pp. 273-286.
18. Markovič, V., Černašėjus, O. ir Prokopovič, V. (2013). Lazerinės miltelių sukepinimo technologijos analizė. Vilnius: Vilniaus Gedimino technikos universitetas.
19. Vainienė, R. (2005). Ekonomikos terminų žodynas. Vilnius: Tyto alba.
20. Street, N. W. (1998). A Neural Network Model for Prognostic Prediction. ICML.
21. Vasiliauskas, A. (2007). Strateginis Valdymas. Kaunas: KTU leidykla Technologija.
22. Bak, C.,& Son, H. (2017). Development of Prediction Model for Root Industry Production Process Using Artificial Neural Network. J. Korean Soc. Precis. Eng.
23. Drouillet, C., Karandikar, J., Nath, C., Journeaux, A. C., Mansori, M. E.,& Kurfess, T. (2016). Tool life predictions in milling using spindle power with the neural network technique. Journal of Manufacturing Processes, pp. 161-168.
24. Crone, R., Nath, C., Mansori, M. E.,& Kurfess, T. (2017). Study of spindle power data with neural network for predicting real-time tool wear/breakage during inconel drilling. Journal of Manufacturing Systems.
25. Kutyłowska, M. (2014). Neural network approach for failure rate prediction. Engineering Failure Analysis 47, p. 41–48.
26. Fried, H. O., Lovell, C. K.,& Schmidt, S. S. (2008). The Measurement of Productive Efficiency and Productivity Growth. New York: Oxford University Press.
27. Beale, M. H., Hagan, M. T.,& Demuth, H. M. (2015). Neural Network Toolbox™ User's Guide. The MathWorks, Inc.
28. Nasr, M. S., Moustafa, M. A., Seif, H. A.,& Kobrosy, G. E. (2012). Application of Artificial Neural Network (ANN) for the prediction of EL-AGAMY wastewater treatment plant performance-EGYPT. Alexandria Engineering Journal (2012) 51, p. 37–43.
29. Medvedev, V. (2007). Tiesioginio sklidimo neuroninių tinklų taikymo daugiamačiams duomenims vizualizuoti tyrimai. Vilnius: VGTU leidykla TECHNIKA.
30. Narad, S.,& Chavan, P. (2016). Cascade Forward Back-propagation Neural Network Based Group Authentication Using (n, n) Secret Sharing Scheme. Procedia Computer Science 78, p. 185 – 191.

31. Gavin, H. P. (2017). The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. Department of Civil and Environmental Engineering, Duke University.
32. Gad, A. (2018). How Many Hidden Layers/Neurons to Use in Artificial Neural Networks? Towards Data Science. [žiūrėta 2018-12-12]. Prieiga per internetą https://towardsdatascience.com/beginners-ask-how-many-hidden-layers-neurons-to-use-in-artificial-neural-networks-51466afa0d3e.
33. Dembélé, D.,& Kastner, P. (2003). Fuzzy C-means method for clustering microarray data. Bioinformatics, Volume 19, Issue 8, p. 973–980.

Priedai

1 priedas

Tiesioginio sklidimo tinklo *Matlab* programinis kodas

Levenberg-Marquardt *(train):*

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = newff(tranponuota_datax,tranponuota_datay, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - train ir pirmus 12 menesiu
duomenis:
[net, tr] = train(net, tranponuota_datax, tranponuota_datay);
net1lm = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y1lm = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida1SSElm = sse(Y1lm-transponuota_datay2);
klaida1MSElm = mse(Y1lm-transponuota_datay2);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(1)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y1lm, '-g','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Lankstaus atgalinio sklidimo (*trainrp*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
```

```
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = newff(tranponuota_datax,tranponuota_datay, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainrp ir pirmus 12 menesiu
duomenis:
[net, tr] = trainrp(net, tranponuota_datax, tranponuota_datay);
net2rp = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y2rp = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida2SSErp = sse(Y2rp-transponuota_datay2);
klaida2MSErp = mse(Y2rp-transponuota_datay2);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(1)
plot(Y2rp, '-r', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Konjugacijos gradiento grąžinimo (*trainscg*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
% 1-5 stulpelis - uzduoties saveduomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = newff(tranponuota_datax,tranponuota_datay, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainscg ir pirmus 12 menesiu
duomenis:
[net, tr] = trainscg(net, tranponuota_datax, tranponuota_datay);
net3scg = net;
```

```
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y3scg = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida3SSEscg = sse(Y3scg-transponuota_datay2);
klaida3MSEscg = mse(Y3scg-transponuota_datay2)
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(1)
plot(Y3scg, '-y', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Quasi-Newton (*trainbfg*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = newff(tranponuota_datax,tranponuota_datay, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainbfg ir pirmus 12 menesiu
duomenis:
[net, tr] = trainbfg(net, tranponuota_datax, tranponuota_datay);
net4bfg = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y4bfg = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida4SSEbfg = sse(Y4bfg-transponuota_datay2);
klaida4MSEbfg = mse(Y4bfg-transponuota_datay2)
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(1)
plot(Y4bfg, '-k', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Kintamojo mokymosi lygio grąžinimo (*traingdx*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
```

```
B=importdata('13men.xlsx')
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net“
% su 10 neuronu pasleptame sluoksnyje.
net = newff(tranponuota_datax,tranponuota_datay, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - traingdx ir pirmus 12 menesiu
duomenis:
[net, tr] = traingdx(net, tranponuota_datax, tranponuota_datay);
net5gdx = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y5gdx = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida5SSEgdx = sse(Y5gdx-transponuota_datay2);
klaida5MSEgdx = mse(Y5gdx-transponuota_datay2);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(1)
plot(Y5gdx, '-m','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Grafikai:

```
close all
% dabartinis statistinio vidurkio prognozavimo modelis:
x=[0,278];
y=[0.8684;0.8684]
% Grafikas tarp realiu ir suprgnozuotu verciu:
figure(2)
plot(data13y,'b','LineWidth',4,'DisplayName','realus rezultatai');
hold on
plot(Y1lm, '-g','LineWidth',2,'DisplayName','trainlm');
plot(Y2rp, '-r', 'LineWidth',2,'DisplayName','trainrp');
plot(Y3scg, '-y', 'LineWidth',2,'DisplayName','trainscg');
plot(Y4bfg, '-k', 'LineWidth',2,'DisplayName','trainbfg');
plot(Y5gdx, '-m','LineWidth',2,'DisplayName','traingdx');
plot (x,y, '--r','DisplayName','dabartinis modelis');
axis([0 278 0.0 1.2]);
xlabel('Prognozavimo rezultatas, nr.');
ylabel('Efektyvumo reiksme, proc.')
hold on
% Grafikas tarp realiu ir suprgnozuotu verciu:
figure(3)
plot(data13y,'b','LineWidth',4,'DisplayName','realus rezultatai');
hold on
plot(Y1lm, '-g','LineWidth',2,'DisplayName','trainlm');
axis([0 278 0.0 1.2]);
xlabel('Prognozavimo rezultatas, nr.');
ylabel('Efektyvumo reiksme, proc.')
hold on
```

Tinklo imties generalizacijos *Matlab* patikrinimo kodas

Levenberg-Marquardt (*train*)*:*

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2835)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2835)
%tinklo tikrinimui like 10proc. duomenu
% 1-5 eilutes - uzduoties duomenys
data12x10=data12men(1:5,2836:3150)
%6 eilutes - tikslo duomenys.
data12y10=data12men(6,2836:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net“
% su 10 neuronu pasleptame sluoksnyje.
net = newff(data12x90,data12y90, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - train ir pirmus 12 menesiu
duomenis:
numNN = 10;
NN = cell(1,numNN);
perfs_lm = zeros(1,numNN);
for i=1:numNN
 disp(['Training ' num2str(i) '/' num2str(numNN)])
 NN{i} = train(net,data12x90, data12y90);
 Y1_10_lm = NN{i}(data12x10);
 perfs_lm(i) = mse(net,data12y10,Y1_10_lm);
end
net1lm = net;
% Tinklu suprognozuojami 13to menesio efektyvumo duomenys:
Y1_lm = sim(NN{i},data13x);
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu sumine kvadratine
paklaida:
klaida1_SSE_lm = sse(Y1_lm-data13y);
klaida1_MSE_lm = mse(Y1_lm-data13y);
figure(4)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y1_lm, '-g','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Lankstaus atgalinio sklidimo (*trainrp*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2835)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2835)
%tinklo tikrinimui like 10proc. duomenu
% 1-5 eilutes - uzduoties duomenys
data12x10=data12men(1:5,2836:3150)
%6 eilutes - tikslo duomenys.
data12y10=data12men(6,2836:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = newff(data12x90,data12y90, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainrp ir pirmus 12 menesiu
duomenis:
numNN = 10;
NN = cell(1,numNN);
perfs_rp = zeros(1,numNN);
for i=1:numNN
 disp(['Training ' num2str(i) '/' num2str(numNN)])
 NN{i} = trainrp(net,data12x90, data12y90);
 Y1_10_rp = NN{i}(data12x10);
 perfs_rp(i) = mse(net,data12y10,Y1_10_rp);
end
net2rp = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y2_rp = sim(NN{i},data13x);
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida2_SSE_rp = sse(Y2_rp-data13y);
klaida2_MSE_rp = mse(Y2_rp-data13y);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(4)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y2_rp, '-r', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Konjugacijos gradiento grąžinimo (*trainscg*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
```

```
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2835)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2835)
%tinklo tikrinimui like 10proc. duomenu
% 1-5 eilutes - uzduoties duomenys
data12x10=data12men(1:5,2836:3150)
%6 eilutes - tikslo duomenys.
data12y10=data12men(6,2836:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net“
% su 10 neuronu pasleptame sluoksnyje.
net = newff(data12x90,data12y90, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainscg ir pirmus 12 menesiu
duomenis:
numNN = 10;
NN = cell(1,numNN);
perfs_scg = zeros(1,numNN);
for i=1:numNN
 disp(['Training ' num2str(i) '/' num2str(numNN)])
 NN{i} = trainscg(net,data12x90, data12y90);
 Y3_10_scg = NN{i}(data12x10);
 perfs_scg(i) = mse(net,data12y10,Y3_10_scg);
end
net3scg = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y3_scg = sim(NN{i},data13x);
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida3_SSE_scg = sse(Y3_scg-data13y);
klaida3_MSE_scg = mse(Y3_scg-data13y);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(4)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y3_scg, '-y', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Quasi-Newton (*trainbfg*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2835)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2835)
%tinklo tikrinimui like 10proc. duomenu
% 1-5 eilutes - uzduoties duomenys
data12x10=data12men(1:5,2836:3150)
```

```
%6 eilutes - tikslo duomenys.
data12y10=data12men(6,2836:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - realus efektyvumo duomenys
data13y=data13(6,:)
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net“
% su 10 neuronu pasleptame sluoksnyje.
net = newff(data12x90,data12y90, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainbfg ir pirmus 12 menesiu
duomenis:
numNN = 10;
NN = cell(1,numNN);
perfs_bfg = zeros(1,numNN);
for i=1:numNN
 disp(['Training ' num2str(i) '/' num2str(numNN)])
 NN{i} = trainbfg(net,data12x90, data12y90);
 Y4_10_bfg = NN{i}(data12x10);
 perfs_bfg(i)  = mse(net,data12y10,Y4_10_bfg);
end
net4bfg = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y4_bfg = sim(NN{i},data13x);
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida4SSEbfg = sse(Y4_bfg-data13y);
klaida4MSEbfg = mse(Y4_bfg-data13y);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(4)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y4_bfg, '-k', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Kintamojo mokymosi lygio grąžinimo (*traingdx*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12men.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13men.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2835)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2835)
%tinklo tikrinimui like 10proc. duomenu
% 1-5 eilutes - uzduoties duomenys
data12x10=data12men(1:5,2836:3150)
%6 eilutes - tikslo duomenys.
data12y10=data12men(6,2836:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
```

```
% Sukurtas dvieju sluoksniu tiesioginio sklidimo (angl. feed-forward network)
neuroninis tinklas „net“
% su 10 neuronu pasleptame sluoksnyje.
net = newff(data12x90,data12y90, 10);
% Apmokytas sukurtas tinklas naudojant funkcija - traingdx ir pirmus 12 menesiu
duomenis:
numNN = 10;
NN = cell(1,numNN);
perfs_gdx = zeros(1,numNN);
for i=1:numNN
 disp(['Training ' num2str(i) '/' num2str(numNN)])
 NN{i} = traingdx(net,data12x90, data12y90);
 Y5_20_gdx = NN{i}(data12x10);
 perfs_gdx(i) = mse(net,data12y10,Y5_20_gdx);
end
net5gdx = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y5_gdx = sim(NN{i},data13x);
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida5SSEgdx = sse(Y5_gdx-data13y);
klaida5MSEgdx = mse(Y5_gdx-data13y);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(4)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y5_gdx, '-m','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Kaskadinio sklidimo tinklo *Matlab* programinis kodas

Levenberg-Marquardt (*train*):

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('aaa.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('bbb2.xlsx')
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu kaskadinio sklidimo (angl. cascade-forward network)
neuroninis tinklas „net“
% su 10 neuronu pasleptame sluoksnyje.
net = cascadeforwardnet(10);
% Apmokytas sukurtas tinklas naudojant funkcija - train ir pirmus 12 menesiu
duomenis:
[net, tr] = train(net, tranponuota_datax, tranponuota_datay);
net6CFlm = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y6CF_lm = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida6cfSSElm = sse(Y6CF_lm-transponuota_datay2);
klaida6cfMSElm = mse(Y6CF_lm-transponuota_datay2);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(5)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y6CF_lm, '-g','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Lankstaus atgalinio sklidimo (*trainrp*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('aaa.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('bbb2.xlsx')
data13=B
```

```
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu kaskadinio sklidimo (angl. cascade-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = cascadeforwardnet(10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainrp ir pirmus 12 menesiu
duomenis:
[net, tr] = trainrp(net, tranponuota_datax, tranponuota_datay);
net7CFrp = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y7CF_rp = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida7cfSSErp = sse(Y7CF_rp-transponuota_datay2);
klaida7cfMSErp = mse(Y7CF_rp-transponuota_datay2);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(5)
plot(Y7CF_rp, '-r', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Konjugacijos gradiento grąžinimo (*trainscg*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('aaa.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('bbb2.xlsx')
data13=B
% 1-5 stulpelis - uzduoties saveduomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu kaskadinio sklidimo (angl. cascade-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = cascadeforwardnet(10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainscg ir pirmus 12 menesiu
duomenis:
[net, tr] = trainscg(net, tranponuota_datax, tranponuota_datay);
net8CFscg = net;
```

```
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y8CF_scg = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida8cfSSEscg = sse(Y8CF_scg-transponuota_datay2);
klaida8cfMSEscg = mse(Y8CF_scg-transponuota_datay2)
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(5)
plot(Y8CF_scg, '-y', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Quasi-Newton (*trainbfg*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('aaa.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('bbb2.xlsx')
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu kaskadinio sklidimo (angl. cascade-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = cascadeforwardnet(10);
% Apmokytas sukurtas tinklas naudojant funkcija - trainbfg ir pirmus 12 menesiu
duomenis:
[net, tr] = trainbfg(net, tranponuota_datax, tranponuota_datay);
net9CFbfg = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y9CF_bfg = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida9cfSSEbfg = sse(Y9CF_bfg-transponuota_datay2);
klaida9cfMSEbfg = mse(Y9CF_bfg-transponuota_datay2)
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(5)
plot(Y9CF_bfg, '-k', 'LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Kintamojo mokymosi lygio grąžinimo (*traingdx*):

```
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('aaa.xlsx')
data12men=A
```

```
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('bbb2.xlsx')
data13=B
% 1-5 stulpelis - uzduoties duomenys
data12x=data12men(:,1:5)
%6 stulpelis - tikslo duomenys.
data12y=data12men(:,6)
% 13-to men. 1-5 stulpeliai - uzduoties duomenys
data13x=data13(:,1:5)
% 13-to men. 6 stulpelis - relus efektyvumo duomenys
data13y=data13(:,6)
tranponuota_datax=transpose(data12x)
tranponuota_datay=transpose(data12y)
transponuota_datax2=transpose(data13x)
transponuota_datay2=transpose(data13y)
% Sukurtas dvieju sluoksniu kaskadinio sklidimo (angl. cascade-forward network)
neuroninis tinklas „net"
% su 10 neuronu pasleptame sluoksnyje.
net = cascadeforwardnet(10);
% Apmokytas sukurtas tinklas naudojant funkcija - traingdx ir pirmus 12 menesiu
duomenis:
[net, tr] = traingdx(net, tranponuota_datax, tranponuota_datay);
net_10CFgdx = net;
% Tinklo suprognozuoti 13to menesio efektyvumo duomenys:
Y_10cfgdx = sim(net,transponuota_datax2)
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu suminis kvadratin?
paklaida:
klaida_10cfSSEgdx = sse(Y_10cfgdx-transponuota_datay2);
klaida_10cfMSEgdx = mse(Y_10cfgdx-transponuota_datay2);
% Nubraizomas grafikas tarp realiu ir suprgnozuotu verciu:
figure(5)
plot(Y_10cfgdx, '-m','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Grafikas:

```
close all
% dabartinis statistinio vidurkio prognozavimo modelis:
x=[0,278];
y=[0.8684;0.8684]
% Grafikas tarp realiu ir suprgnozuotu verciu:
figure(6)
plot(data13y,'b','LineWidth',4,'DisplayName','realus rezultatai');
hold on
plot(Y6CF_lm, '-g','LineWidth',2,'DisplayName','trainlm');
plot(Y7CF_rp, '-r', 'LineWidth',2,'DisplayName','trainrp');
plot(Y8CF_scg, '-y', 'LineWidth',2,'DisplayName','trainscg');
plot(Y9CF_bfg, '-k', 'LineWidth',2,'DisplayName','trainbfg');
plot(Y_10cfgdx, '-m','LineWidth',2,'DisplayName','traingdx');
plot (x,y, '--r','DisplayName','dabartinis modelis');
axis([0 278 0.0 1.2]);
xlabel('Prognozavimo rezultatas, nr.');
ylabel('Efektyvumo reiksme, proc.')
hold on
```

DNT tobulinimo *Matlab* programinis kodas

Neuronų skaičiaus optimizavimas:

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12menTR.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13menTR.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:3150)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
for Hid=3:10 % paslepti neuronai
% ciklas tinklas su neuronu kiekiu pasleptame sluoksnyje nuo 3 iki 10
for i=1:10 % pakartojimai
net = newff(data12x90,data12y90,[Hid]);
% sukuriamas neuroninis tinklas
net = init(net);
net.trainParam.epochs = 100;
net = train(net,data12x90,data12y90);
% apmokomoas sukurtas neuroninis modelis
Y12 = sim(net,data13x);
%skaiciuojama sumine kvadratine paklaida tarp uzduoties ir atsakymo verciu
Klaida(Hid,i)=sse(Y12-data13y);
end
end
```

Grafikas:

```
figure(7)
plot(3:10, mean(Klaida(3:end,:)'),'r');
hold on
ylabel('klaidos vidurkis')
xlabel('Pasleptu neuronu kiekis')
figure(3)
plot(3:10,std(Klaida(3:end,:)'),'g');
ylabel('std')
xlabel('Pasleptu neuronu kiekis')
```

Optimizuotas tinklas su 7 neuronais:

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
```

```
A=importdata('12menTR.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13menTR.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:3150)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:3150)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
for Hid=7; % paslepti neuronai
KL=[1.0787] % naujas kintamasis, kuris didesni už max klaid?
 for i=1:100 % pakartojimai
 if KL>=[0.95]; %Tikriname ar kintamasis KL daugiau arba lygus mažiausiai
klaidai
 net = newff(data12x90,data12y90,[Hid]);
 % sukuriamas neuroninis tinklas
 net = init(net);
 net.trainParam.epochs = 100;
 net = train(net,data12x90,data12y90);
 % apmokomoas sukurtas neuroninis modelis
Y11 = sim(net,data13x);
%skaiciuojama sumine kvadratine paklaida tarp uzduoties ir atsakymo verciu
Klaida(Hid,i)=sse(Y11-data13y);
KL=Klaida(Hid,i); %Priskiriame klaid? KL kintamajam
 end
 end
end
net12_7n = net;
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu sumine kvadratine
paklaida:
klaida_11_SSE_lm = sse(Y11-data13y);
klaida_11_MSE_lm = mse(Y11-data13y);
figure(8)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y11, '-g','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Duomenų imties tobulinimo *Matlab* programinis kodas

Pradinės duomenų imties kokybės gerinimas Fuzzy C-means metodu:

```
clc
clear all
close all
A=importdata('aaa.xlsx')
X=A(:,2);
Y=A(:,5);
Z=A(:,6);
figure (9)
scatter3(X,Y,Z, 'ok')
axis([0 12 0 800 0 1.2])
ylabel('pagamintas kiekis, tukst. vnt.');
xlabel('produkto kodas');
zlabel ('efektyvumo rezultatas');
[centrai, tik]=fcm(A, 5);
maxT=max(tik);
ind1=find(tik(1,:)==maxT);
ind2=find(tik(2,:)==maxT);
ind3=find(tik(3,:)==maxT);
ind4=find(tik(4,:)==maxT);
ind5=find(tik(5,:)==maxT);
figure (10)
scatter3(A(ind1,2),A(ind1,5),A(ind1,6), 'or')
hold on
scatter3(A(ind2,2),A(ind2,5),A(ind2,6), 'og')
hold on
scatter3(A(ind3,2),A(ind3,5),A(ind3,6), 'oc')
hold on
scatter3(A(ind4,2),A(ind4,5),A(ind4,6), 'ok')
hold on
scatter3(A(ind5,2),A(ind5,5),A(ind5,6), 'ob')
hold on
axis([0 12 0 800 0 1.2])
ylabel('pagamintas kiekis, tukst. vnt.');
xlabel('produkto kodas');
zlabel ('efektyvumo rezultatas');
```

Modelis su atrinkta pradinių duomenų imtimi:

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12menCmeanTR.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('13menTR.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2772)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2772)
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
```

```
data13y=data13(6,:)
for Hid=7; % paslepti neuronai
KL=[1.0787] % naujas kintamasis, kuris didesni už max klaid?
 for i=1:100 % pakartojimai
 if KL>=[1.0212]; %Tikriname ar kintamasis KL daugiau arba lygus mažiausiai
klaidai
 net = newff(data12x90,data12y90,[Hid]);
 % sukuriamas neuroninis tinklas
 net = init(net);
 net.trainParam.epochs = 100;
 net = train(net,data12x90,data12y90);
 % apmokomoas sukurtas neuroninis modelis
Y12 = sim(net,data13x);
%skaiciuojama sumine kvadratine paklaida tarp uzduoties ir atsakymo verciu
Klaida(Hid,i)=sse(Y12-data13y);
KL=Klaida(Hid,i); %Priskiriame klaid? KL kintamajam
 end
 end
end
net12_7n_atrinktas = net;
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu sumine kvadratine
paklaida:
klaida_12_SSE_lm = sse(Y12-data13y);
klaida_12_MSE_lm = mse(Y12-data13y);
figure(11)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y12, '-g','LineWidth',2);
axis([0 278 0.5 1.1])
hold on
```

Tikslo duomenų imties tobulinimas *Matlab* programinis kodas

Tikslo duomenų imties kokybės gerinimas Fuzzy C-means metodu:

```
clc
clear all
close all
A=importdata('bbb2.xlsx')
X=A(:,2);
Y=A(:,5);
Z=A(:,6);
figure (9)
scatter3(X,Y,Z, 'ok')
axis([0 12 0 800 0 1.2])
ylabel('pagamintas kiekis, tukst. vnt.');
xlabel('produkto kodas');
zlabel ('efektyvumo rezultatas');
[centrai, tik]=fcm(A, 5);
maxT=max(tik);
ind1=find(tik(1,:)==maxT);
ind2=find(tik(2,:)==maxT);
ind3=find(tik(3,:)==maxT);
ind4=find(tik(4,:)==maxT);
ind5=find(tik(5,:)==maxT);
figure (12)
scatter3(A(ind1,2),A(ind1,5),A(ind1,6), 'or')
hold on
scatter3(A(ind2,2),A(ind2,5),A(ind2,6), 'og')
hold on
scatter3(A(ind3,2),A(ind3,5),A(ind3,6), 'oc')
hold on
scatter3(A(ind4,2),A(ind4,5),A(ind4,6), 'ok')
hold on
scatter3(A(ind5,2),A(ind5,5),A(ind5,6), 'ob')
hold on
axis([0 12 0 800 0 1.2])
ylabel('pagamintas kiekis, tukst. vnt.');
xlabel('produkto kodas');
zlabel ('efektyvumo rezultatas');
```

Modelis su atrinkta pradininių duomenų imtimi ir tikslo duomenų imtimi:

```
clc
clear all
close all
% Duomenys:
% 12 menesiu duomenys - neuroniniam tiknlui suformuoti
A=importdata('12menCmeanTR.xlsx')
data12men=A
% 13-tas menuo - modelio tikslumo patikrinimui
B=importdata('bbb3.xlsx')
data13=B
%Tinklo apmokymui 90proc. duomenu:
% 1-5 eilutes - uzduoties duomenys
data12x90=data12men(1:5,1:2772)
%6 eilutes - tikslo duomenys.
data12y90=data12men(6,1:2772)
```

```
% 13-to men. 1-5 eilutes - uzduoties duomenys
data13x=data13(1:5,:)
% 13-to men. 6 eilutes - relus efektyvumo duomenys
data13y=data13(6,:)
for Hid=7; % paslepti neuronai
KL=[1.0787] % naujas kintamasis, kuris didesni už max klaid?
 for i=1:100 % pakartojimai
 if KL>=[0.950]; %Tikriname ar kintamasis KL daugiau arba lygus mažiausiai
klaidai
 net = newff(data12x90,data12y90,[Hid]);
 % sukuriamas neuroninis tinklas
 net = init(net);
 net.trainParam.epochs = 100;
 net = train(net,data12x90,data12y90);
 % apmokomoas sukurtas neuroninis modelis
Y12 = sim(net,data13x);
%skaiciuojama sumine kvadratine paklaida tarp uzduoties ir atsakymo verciu
Klaida(Hid,i)=sse(Y12-data13y);
KL=Klaida(Hid,i); %Priskiriame klaid? KL kintamajam
 end
 end
end
net13_atrinkta_final = net;
% paskaiciuojama sukurto modelio rezultatu ir realiu duomenu sumine kvadratine
paklaida:
klaida_13_SSE_lm = sse(Y12-data13y);
klaida_13_MSE_lm = mse(Y12-data13y);
figure(13)
plot(data13y,'b','LineWidth',4);
hold on
plot(Y12, '-g','LineWidth',2);
axis([0 245 0.5 1.1])
hold on
```