VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
STUDIJŲ PROGRAMA: INFORMATIKA

Orų prognozavimas naudojant dirbtinius neuroninius tinklus
Weather Forecasting Using Artificial Neural Networks

Baigiamasis bakalauro darbas

Atliko: Deividas Žemeckas
VU el. p.: deividas.zemeckas@mif.stud.vu.lt

Vadovas: Linas Litvinas

Vilnius
2023

Turinys

Įvadas 4
1. Orų prognozavimas 5
1.1. Tradiciniai orų prognozavimo metodai 5
1.1.1. Orų žemėlapio prognozė 5
1.1.2. Prognozavimas remiantis orų radaru 5
1.1.3. Skaitinis orų numatymas 5
1.2. Orų prognozavimas neuroniniais tinklais 6
1.2.1. Tiesinio skleidimo neuronių tinklų naudojimas 6
1.2.2. Rekurentinių neuroninių tinklų naudojimas 6
1.2.3. Konvoliucinių neuroninių tinklų naudojimas 6
2. Dirbtiniai neuroniniai tinklai 7
2.1. Žmogaus smegenys 7
2.2. Neuronas 7
2.3. Neuroninių tinklų tipai 8
2.3.1. Tiesinio skleidimo neuroniniai tinklai 9
2.3.2. Rekurentiniai neuroniniai tinklai 10
2.3.3. Konvoliuciniai neuroniniai tinklai 11
2.4. Aktyvacijos funkcijos 11
2.4.1. Sigmoido funkcija 11
2.4.2. Binarinė žingsnio funkcija 12
2.4.3. Linijinė aktyvacijos funkcija 12
2.4.4. Hiperbolinė tangento funkcija (TANH) 13
2.4.5. ReLU aktyvacijos funkcija 13
2.4.6. Švytuojanti aktyvacijos funkcija 14
3. Tyrimai 14
3.1. Bandymų aprašymas 14
3.2. Orų prognozavimas nenaudojant neuroninių tinklų 15
3.2.1. Spėjimo būdo prognozė 15
3.2.1.1. Pirmas eksperimentas 15
3.2.2. Tiesinė regresija 15
3.2.2.1. Antras eksperimentas 16
3.2.2.2. Trečias eksperimentas 16

3.3. Orų prognozavimas naudojant neuroninius tinklus 17
3.3.1. Pasiruošimas tyrimams 17
3.3.1.1. Neuroninio tinklo pasirinkimas 18
3.3.1.2. Duomenų normalizavimas 18
3.3.1.3. Neuroninio tinklo persimokymas (angl. *overfitting*) 18
3.3.2. Tiesinio skleidimo neuroninis tinklas 20
3.3.2.1. Ketvirtas eksperimentas 20
3.3.2.2. Penktas eksperimentas 23
3.3.2.3. Šeštas eksperimentas 26
3.3.2.4. Septintas eksperimentas 30
3.3.2.5. Aštuntas eksperimentas 33
3.3.3. Neuroninio tinklo treniravimo optimizavimas 37
3.3.3.1. Devintas eksperimentas 37
3.3.3.2. Dešimtas eksperimentas 40
3.3.4. Rekurentinis neuroninis tinklas. 44
3.3.4.1. Vienuoliktas eksperimentas 44
3.3.4.2. Dvyliktas eksperimentas 47
3.4. Bandymų palyginimas 50
Išvados 54
Literatūra 56

Įvadas

Šiomis dienomis sparčiai populiarėja dirbtiniai neuroniniai tinklai (angl. artificial neural networks) dėl jų gebėjimo mokytis ir atpažinti sudėtingus modelius, pasitelkiant didelius duomenų rinkinius jų apmokymui. Dirbtiniai neuroniniai tinklai pasisėmė įkvėpimo tiesiai iš žmogaus smegenų, jų veiklos ir struktūros. Neuroniniai tinklai šiuo metu yra pritaikomi įvairiose srityse įskaitant vaizdo ir kalbos atpažinimą, natūralios kalbos apdorojimą, finansus, sveikatos priežiūrą ir daugelyje kitų. Taip pat, dirbtiniai neuroniniai tinklai tapo populiaria priemone ir orų prognozavimo srityje. Orų prognozavimas yra sudėtinga užduotis dėl atmosferos procesų sudėtingumo ir pastovaus kintamumo. Tikslios orų prognozės gali turėti didelės įtakos įvairioms pramonės šakoms, įskaitant žemės ūkį, transportą, energetiką ir įvairių krizių valdymą. Tradiciniai orų prognozavimo metodai remiasi skaitiniai modeliais, kuriem yra reikalingi dideli apskaičiavimo ištekliai. Vis gi, šiems modeliams sunkiai sekasi užfiksuoti visas sąsajas tarp įvairių atmosferos ir orų kintamųjų.

Dirbtiniai neuroniniai tinklai suteikia alternatyvų būdą pažvelgti į orų prognozavimą, pasinaudojant mašininio mokymosi metodus identifikuojant ir išmokstant skirtingus modelius, bei sąryšius tarp įvairių kintamųjų. Neuroninis tinklas imituoja žmogaus smegenų veiklą, kai informacija yra perduodama per tarpusavyje sujungtus taškus ar neuronus.

Šio darbo tikslas yra ištirti neuroninių tinklų panaudojimą prognozuojant oro sąlygas ir išanalizuoti kiek efektyvus yra neuroninio tinklo panaudojimas tokiam uždaviniui. Mes analizuosime įvairių tipų orų duomenis: oro temperatūrą, slėgį, vėjo greitį ir drėgmę. Taip pat, ištirsime dirbtinio neuroninio tinklo našumą, bei įvairių parametrų įtaką rezultatams. Darbe tiriama tiesioginio skleidimo neuroninio tinklo (angl. „feed-forward neural network") gebėjimas prognozuoti oro temperatūrą remiantis istoriniais duomenimis. Tiriamas neuroninio tinklo našumas keičiant tam tikrus jo parametrus bei keičiant įvesties ir išvesties parametrus. Taip pat, bandysime optimizuoti tinklo apmokymo procesą. Galiausiai bandysime pakeisti neuroninio tinklo architektūrą ir atlikti daugiau bandymų. Gauti rezultatai yra palyginami.

Iškeltam tikslui pasiekti, bus vykdomi šie uždaviniai:

1. Susipažinti su neuroniniais tinklais ir jų savybėmis;
2. Plačiau išnagrinėti tiesioginio skleidimo neuroninį tinklą;
3. Sukurti neuroninį tinklą;
4. Atlikti eksperimentus su skirtingu paslėptų neuronų skaičiumi neuroniniame tinkle;
5. Atlikti eksperimentus su įvairiais įvesties ir išvesties parametrais;
6. Atlikti eksperimentus optimizuojant apsimokymo procesą;
7. Atlikti eksperimentus pakeitus neuroninio tinklo architektūrą;
8. Palyginti tyrimus;
9. Pateikti išvadas apie neuroninio tinklo gebėjimą prognozuoti oro temperatūrą;

Darbo pradžioje bus atliekama literatūros analizė ir apžvelgiami moksliniai darbai apie neuroninius tinklus. Vėliau bus plačiau aprašytas tiesioginio skleidimo neuroninis tinklas, jo ypatybės, parametrai ir t.t. Darbo pabaigoje bus atliekami tyrimai. Bus sukurtas ir apmokytas neuroninis tinklas, atliekamos prognozės, gaunami rezultatai. Galiausiai gauti rezultatai bus palyginami ir pateiktos išvados.

1. Orų prognozavimas

Šiame skyriuje aptarsime tradicinius orų prognozavimo metodus ir orų prognozavimą pasitelkiant dirbtinius neuroninius tinklus.

1.1. Tradiciniai orų prognozavimo metodai

Egzistuoja keli tradiciniai orų prognozavimo metodai. Toliau juos trumpai aptarsime.

1.1.1. Orų žemėlapio prognozė

Šis metodas egzistuoja jau beveik šimtą metų, nuo pat telegrafo atsiradimo. Iš orų žemėlapio buvo išmokta analizuoti orų sistemą ir prognozuoti tolimesnius judėjimus ir intensyvumą. Šiuo metodu yra stebimi orų žemėlapiai, tam, kad nustatyti orų sistemą, kuri sukelia tam tikrus pokyčius ore. Pirmas šio metodo žingsnis yra prognozuoti orą, t.y. šiuo metu egzistuojančią orų sistemą ir jos ateities pokyčius orų žemėlapyje. Toliau, pasitelkiant gamtinių dujų prognozavimo praktika, meteorologai bando atpažinti ir išveda empirinius orų sistemos judėjimo arba intensyvumo dėsnius [FXS+21]. Orų žemėlapyje yra pavaizduoti atmosferos slėgio, vėjo, temperatūros ir drėgmės pasiskirstymas skirtinguose atmosferos lygiuose. Yra dviejų tipų orų žemėlapiai: paviršiaus ir viršutinio oro. Duomenys pavaizduoti paviršiaus žemėlapyje yra analizuojami izobariškai. Tai reiškia, kad tas pats atmosferos slėgis skirtingose vietose yra sujungtas linija, atsižvelgiant į vėjo kryptį. Atliekant šią analizę, galima nustatyti ir nubrėžti oro sistemas arba vadinamuosius veikimo centrus, tokius kaip aukšto ir žemo slėgio zonos, atogrąžų ciklonai, šaltieji ir šiltieji frontai, tarptropinė konvergencijos zona. Tuo tarpu duomenys pavaizduoti viršutinio oro žemėlapyje yra analizuojami naudojant supaprastintą analizę. Yra nubrėžiamos linijos pavaizduojančios vėjo srautą. Taikant tokią analizę, galima atpažinti anticiklonus, aukšto slėgio sritis ir ciklonus arba žemo slėgio sritis [Phi23].

1.1.2. Prognozavimas remiantis orų radaru

Radaras gali skleisti trumpo ilgio radijo bangas. Kai jis tolumoje aptinka kokį nors oro reiškinį, kaip taifūną, perkūniją ar liūtį, jo radijo bangos atsispindės ir bus rodomos radaro ekrane. Todėl ekrane galima matyti šių reiškinių išvaizdą ir vidinę struktūrą [FXS+21]. Orų radaras atlieka trimatį atmosferos skenavimą skleisdama greitą aukšto dažnio elektromagnetinių impulsų seką ir priima jų aidus. Skenavimui yra naudojama parabolinė antena, kuri tam tikrame aukštyje atlieka pilną apsisukimą. Per šį apsisukimą yra surenkami visi duomenys. Tada ši procedūra kartojama iškėlus radarą į skirtingą aukštį. Iš esmės iš gaunamo aido stiprumo, vadinamojo radaro atspindžiu, galima spręsti apie kritulių kiekį. Atsižvelgiant į laiko tarpą nuo šviesos greičiu sklindančio impulso siuntimo iki aido gavimo, taip pat antenos orientacijos, nustatoma kritulių vieta. Tačiau, orų radarų sistemos sulaukia ne tik kritulių aidus, bet ir daugelio kitų atmosferoje esančių objektų, pvz., lėktuvų, paukščių, vabzdžių, kalnų, laivų, pastatų ar vėjo turbinų aidus [Deu23].

1.1.3. Skaitinis orų numatymas

Orų prognozavimas visada yra grindžiamas meteorologijos principais. Tobulėjant skaičiavimo ir aptikimo technologijoms, be tradicinio orų žemėlapio metodo ir orų radaro metodo, buvo išrasti skaitinio prognozavimo metodai. Šiuo metodu galima numatyti fizinį atmosferos procesą, nustatant atmosferos masės ir energijos principus. Skaitmeninis orų prognozavimo modelis yra tipiška netiesinė sistema su daugybe skaičiavimų. Kad visi skaičiavimai būtų atlikti per trumpesnį laiką nei faktinė orų raida, didelės spartos kompiuteriai tapo pagrindinė ir svarbiausia technologija [FXS+21]. Kartu su kompiuterinių technologijų

pažanga, skaitmeninė orų prognozė tapo pagrindiniu orų prognozavimo komponentu. Pavyzdžiui, Jungtinėse Valstijose kasdienė orų prognozė yra atliekama superkompiuteriu esančiu Vašingtone. Europoje didžiausias pasaulyje skaitmeninių orų prognozių centras esantis Jungtinėje karalystėje, teikia pažangias orų prognozes visoms Europos Sąjungos šalims narėms [PK20].

1.2. Orų prognozavimas neuroniniais tinklais

Trumpai aptarsime kelis orų prognozavimo metodus pasitelkiant dirbtinius neuroninius tinklus.

1.2.1. Tiesinio skleidimo neuronių tinklų naudojimas

Šio neuroninio tinklo struktūra yra: įvesties sluoksnis, paslėptasis sluoksnis ir išvesties sluoksnis [FXS+21]. Jei kalbant apie orų prognozavimą, paprastai neuroninis tinklas yra apmokomas istoriniais orų duomenimis tokiais, kaip temperatūra, drėgmė, vėjo greitis ir atmosferos slėgis, kartu su atitinkamomis faktinėmis reikšmėmis. Prognozės tikslumas labai priklauso nuo žinių apie vyraujančias oro sąlygas plačioje teritorijoje. Oras yra nelinijinis ir dinamiškas procesas, kuris kasdien kinta, net ir kiekvieną minutę. Kadangi klimato duomenų rinkinys yra netiesinis, dirbtinis neuroninis tinklas gali būti naudojamas orų prognozavimui ir klasifikavimui [NP12].

1.2.2. Rekurentinių neuroninių tinklų naudojimas

Rekurentinis neuroninis tinklas yra tam tikras nukreiptas ciklas, kurį sudaro tarpusavyje sujungti neuronai. Šis tinklas nuo tiesinio skleidimo skiriasi tuo, jog jis apdoroja sekos duomenis. Palyginti su tradiciniu neuroniniu tinklu, rekurentinis gali geriau susidoroti su laiko eilučių duomenų prognozavimu. Būtent šis metodas yra taikomas Weifang vandens stotyje, Kinijoje. Jis prognozuoja vandens padidėjimą, o rezultatai rodo, jog palyginti su tiesinio skleidimo neuroniniu tinklu, rekurentinis tinklas gali gauti geresnius prognozavimo rezultatus su mažesnėmis paklaidomis [FXS+21]. 2009 metais Indijos mokslininkai nustatė, kad musoniniai krituliai virš vieno Indijos regiono priklauso ne tik nuo įvairių išorinių jėgų, bet ir nuo laiko eilutės kintamumo. Kitaip tariant, ateities krituliai priklauso nuo praeityje buvusių kritulių ir jų pasirodymo tam tikru laiku, tam tikra seka. Taigi, rekurantinis neuroninis tinklas puikiai sugeba išspręsti šią problemą ir net prognozuoti kritulius tolimoje ateityje [SKG+12].

1.2.3. Konvoliucinių neuroninių tinklų naudojimas

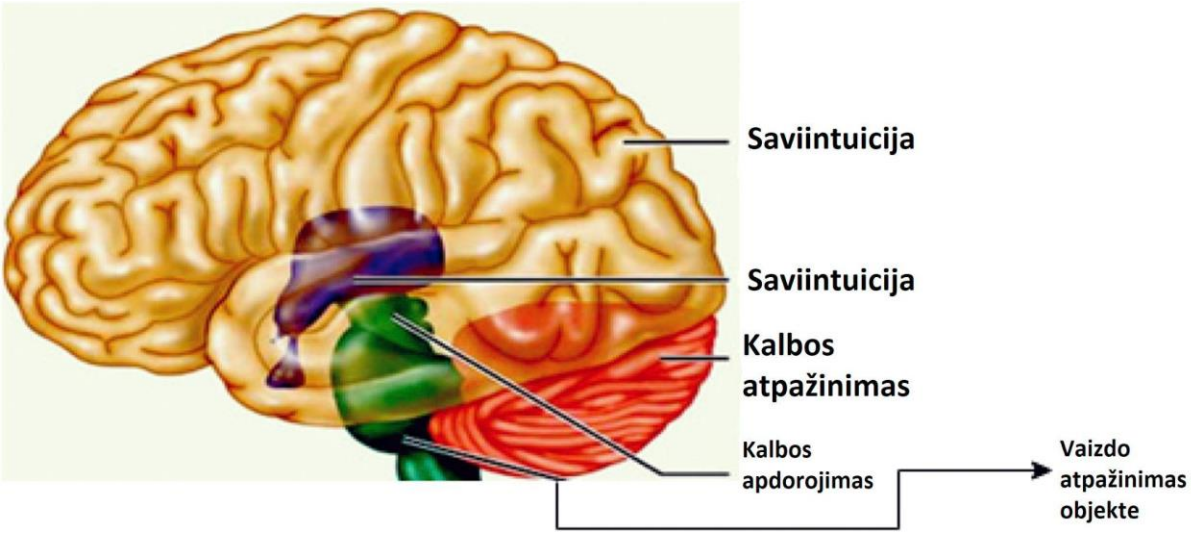
Konvoliuciniai neuroniniai tinklai gali ištraukti reikalingus duomenis ar savybes iš originalios informacijos be sudėtingo išankstinio apdorojimo. Šie tinklai buvo pritaikyti daugelyje meteorologinių prognozavimo sričių. Taip jie yra labai naudingi prognozuojant ekstremalius orus. Gerą konvoliucinio neuroninio tinklo algoritmo tikslumą lemia didžiulis skaičiavimų sudėtingumas. Šie tinklai paprastai yra pagreitinami bendrosios paskirties procesoriuose arba grafikos apdorojimo įrenginiuose [FXS+21]. Šie neuroniniai tinklai ateityje gali padėti spręsti daugybę problemų klimato moksle, kaip pvz., sąsajų ir šablonų aptikimas. Tiesa, dauguma konvoliucinių neuroninių tinklų tyrimų yra orientuoti į objektų aptikimą ir atpažinimą [KHA21].

2. Dirbtiniai neuroniniai tinklai

Pirmiausia reiktų aptarti, kas yra neuroniniai tinklai, kokie jie būna ir ką jie atlieka. Neuroniniai tinklai yra informacijos valdymo modeliai, veikiantys panašiai kaip žmogaus smegenų biologinė nervų sistema.

2.1. Žmogaus smegenys

Neuroniniai tinklai, tai tarsi mašina sukurta veikti taip, kaip veikia žmogaus smegenys norinčios išspręsti norimą užduotį. Pagrindinis žmogaus smegenų privalumas yra unikalus gebėjimas apdoroti informaciją. Paprasčiausias neuroninio tinklo funkcijos pavyzdys yra žmogaus smegenys, kurios siunčia ir priima signalus, tam, kad žmogus galėtų atlikti norimus veiksmus [AJO+18].

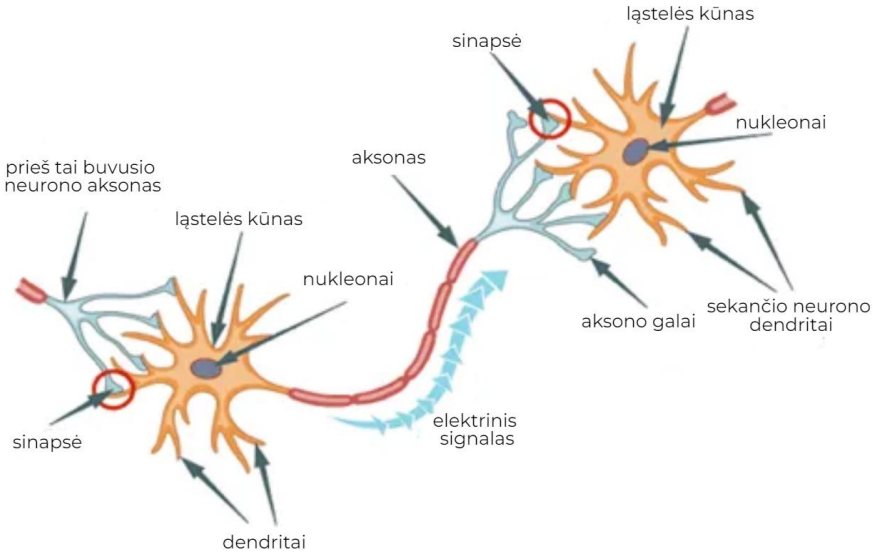


Pav. 1. Tipiška žmogaus smegenų struktūra su operatyvinėmis galimybėmis [AJO+18].

1 paveikslėlyje yra pademonstruotos žmogaus smegenys, kurios sugeba atlikti įvairias samprotavimo funkcijas. Neuronų pagalba jos gali mąstyti tam, kad kažką išsiaiškintų (ieškoti informacijos interneto paieškoje), atpažinti kalbą (pavyzdžiui atskirti pažįstamą žmogų nuo nepažįstamo), atpažinti vaizdą (iš kokio nors objekto), apdoroti kalbą (pavyzdžiui išversti iš vienos kalbos į kitą) ir atlikti kitus veiksmus, kaip valgymas, važiavimas dviračiu (saviintuicija).

2.2. Neuronas

Mūsų smegenyse yra milijardai ląstelių, vadinamų neuronais, kurios apdoroja informaciją elektrinių signalų pavidalu. Išorinę informaciją/dirgiklius priima neurono dendritai, apdorojami neurono ląstelės kūne, paverčiami išėjimu ir per aksoną perduodami kitam neuronui. Kitas neuronas gali pasirinkti jį priimti arba atmesti, priklausomai nuo signalo stiprumo [Mah17].

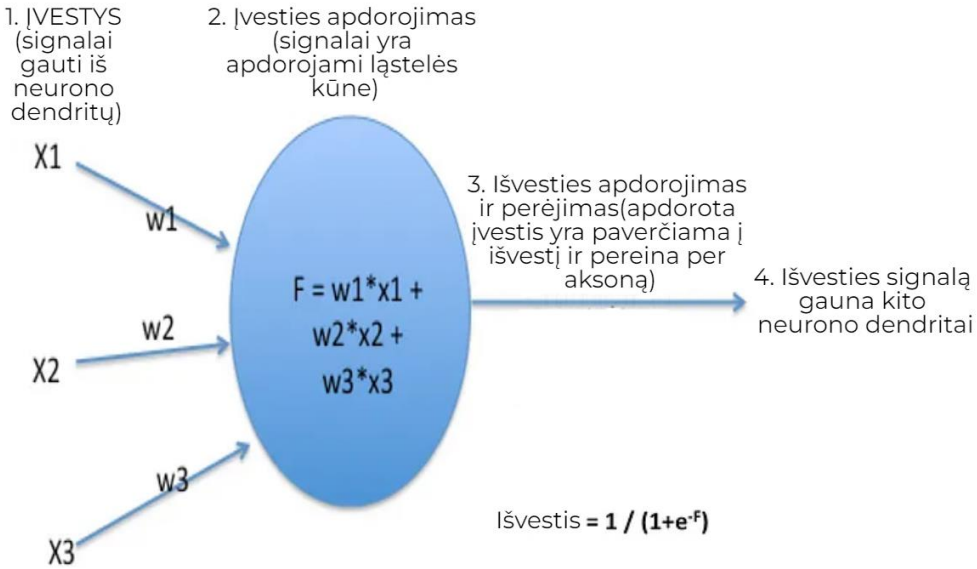


Pav. 2. Neurono ląstelė [Mah17].

Remiantis 2 paveikslėliu galima aptarti kaip signalas keliauja per neuroną:

1. Dendritai aptinka išorinį signalą;
2. Išorinis signalas apdorojamas neurono ląstelės kūne;
3. Apdorotas signalas paverčiamas išvesties signalu ir perleidžiamas per aksoną;
4. Išvesties signalą gauna kito neurono dendritai, per neuronų jungtį.

Dabar aptarkime kaip galėtų atrodyti dirbtinio neuroninio tinklo neuronas.



Pav. 3. Neuroninio tinklo neuronas [Mah17].

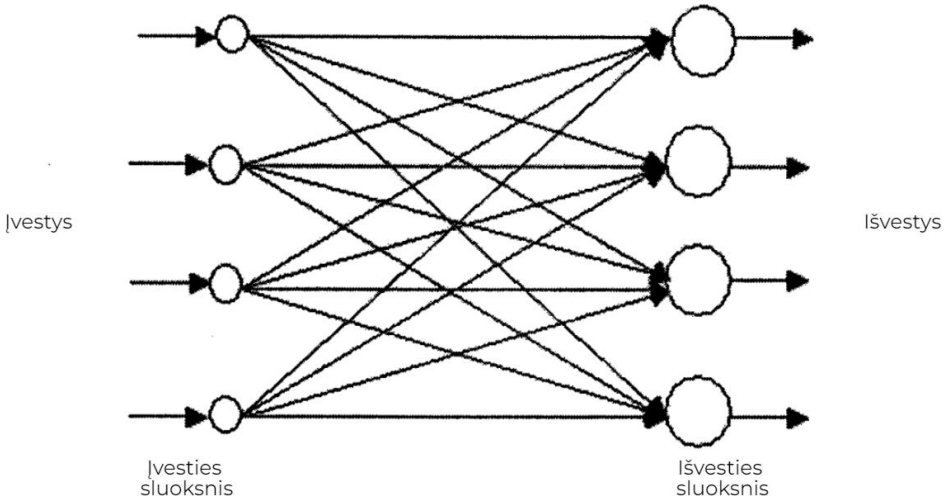
3 paveikslėlyje matomas dirbtinio neuroninio tinklo neuronas, kur w1, w2, w3 nurodo įvesties signalų stiprumą [Mah17]. Kaip matome iš paveikslėlio, dirbtinis neuroninis tinklas yra supaprastintas žmogaus smegenų veikimo modelis.

2.3. Neuroninių tinklų tipai

Yra daug įvairių neuroninių tinklų tipų, skirtų spręsti tam tikras problemas. Toliau panagrinėsime kelis pagrindinius jų tipus.

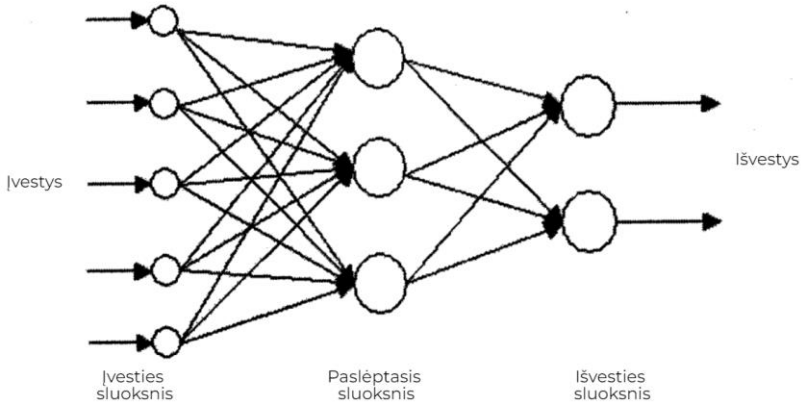
2.3.1. Tiesinio skleidimo neuroniniai tinklai

Tiesinio skleidimo neuroninis tinklas (angl. feed-forward neural network) yra toks tinklas, kuris neturi jokio grįštamojo ryšio iš savo išvesties neuronų atgal į įvesties neuronus. Tiesinis neuroninis tinklas gali būti dviejų kategorijų pagal savo sluoksnius, t.y. arba vieno sluoksnio arba daugiasluoksnis.



Pav. 4. Vieno sluoksnio tiesinio skleidimo neuroninis tinklas [Saz06].

4 paveikslėlyje matome vieno sluoksnio tiesinio skleidimo neuroninį tinklą, kurio kiekvienas įvesties neuronas yra sujungtas su kiekvienu išvesties neuronu. Šis neuroninis tinklas yra tik vieno sluoksnio, kadangi įvesčių mes neskaičiuojame kaip atskiro sluoksnio, nes jos neatlieka jokių skaičiavimo, o tik priima duomenis. Įvesties signalai yra perduodami išvesties sluoksniui su tam tikrais svoriais ir išvesties neuronai apskaičiuoja išvesties signalus.



Pav. 5. Daugiasluoksnis tiesinio skleidimo neuroninis tinklas [Saz06].

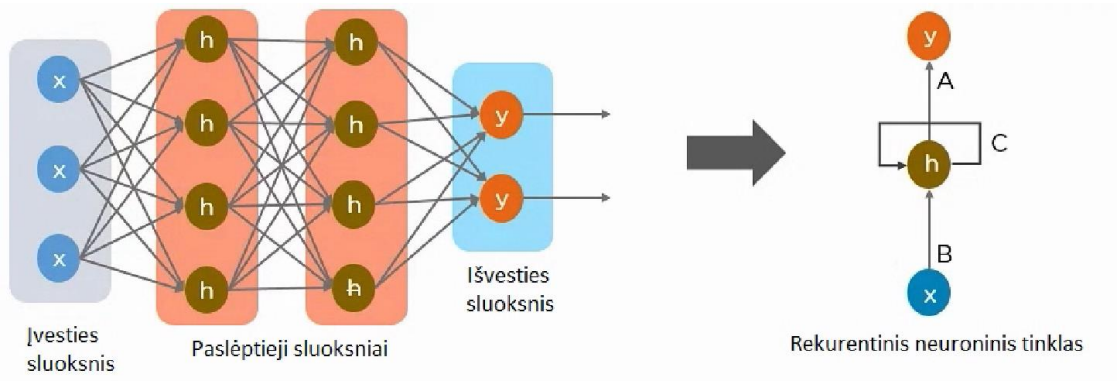
5 paveikslėlyje yra pavaizduotas tiesinio skleidimo neuroninis tinklas su vienu paslėptu sluoksniu (angl. hidden layer). Tai yra daugiasluoksnis neuroninis tinklas, kadangi jis turi bent vieną paslėptąjį sluoksnį tarp įvesties ir išvesties sluoksnių. Paslėptojo sluoksnio tikslas yra įsiterpti tarp išorinės įvesties ir tinklo išvesties, tam, kad išvesti būtų tikslesnė. Vienas ar daugiau paslėptųjų sluoksnių įgalina neuroninį tinklą geriau spręsti sudėtingesnius uždavinius. Tiek pirmame tiek antrame paveikslėlyje tinklai yra pilnai sujungti, nes kiekvienas neuronas

visuose sluoksniuose yra sujungtas su kiekvienu neuronu kitame esančiame sluoksnyje. Jeigu sujungimai būtų ne visi, neuroninis tinklas būtų vadinamas dalinai sujungtu [Saz06].

Tiesinio skleidimo neuroniniai tinklai yra paprasčiausios architektūros iš visų kitų tipų, kadangi informacija juo juda tik į priekį. Jie dažniausiai yra naudojami klasifikacijos ar regresijos uždaviniams. Klasifikacijos uždaviniai gali būti tokie kaip nuotraukos atpažinimas ar jame yra kažkoks specifinis objektas ar kiti uždaviniai, kaip pavyzdžiui elektroninio laiško atpažinimas ar tai svarbus laiškas ar reklama. Regresijos uždavinio pavyzdys būtų kažkokio skaičiaus spėjimas. Pavyzdžiui namo kainos prognozavimas pagal vietą, dydį ir kitas jo ypatybes. Taip pat, tai gali būti ir oro temperatūros prognozavimas pagal vėjo greitį, slėgį, drėgmę ir metų dieną. Tiek klasifikavimo tiek regresijos uždaviniuose neuroninis tinklas mokosi turėdamas įvesties duomenis kurie yra suporuoti su išvesties duomenimis. Tada algoritmas naudodamasis šiais ryšiais išmoksta sąryšius tarp šių duomenų ir gali atlikti sėjimus jau su naujais, nematytais duomenimis.

2.3.2. Rekurentiniai neuroniniai tinklai

Rekurentinis neuroninis tinklas (angl. recurrent neural network) yra specialus neuroninio tinklo tipas adaptuotas veiktu su laiko eilutėmis arba duomenimis kuriuose yra įvairios sekos [Sae22]. Šio tinklo veikimo principas yra išsaugoti tam tikro sluoksnio išvestį ir perduoti jį atgal į įvestį, tam, kad nuspėti kokia turi būti išvestis tame sluoksnyje.



Pav. 6. Paprastas rekurentinis neuroninis tinklas [Bis23].

6 paveikslėlyje matome, kaip tiesinio skleidimo neuroninį tinklą galima paversti į rekurentinį. Čia „x“ yra įvesties sluoksnis, „h“ paslėpstasis sluoksnis, o „y“ yra išvesties sluoksnis. „A“, „B“ ir „C“ yra neuroninio tinklo parametrai naudojami pagerinti modelio tikslumą. Bet kuriuo laiko momentu t, neuroninio tinklo įvestis yra įvesčių kombinacija x(t) ir x(t-1). Išvestis bet kuriuo laiko momentu yra grąžinama atgal neuroninį tinklą, tam, kad būtų pagerintas tikslumas ir gauta tikslesnė išvestis.

Rekurentiniai neuroniniai tinklai buvo sukurti, nes tiesinio skleidimo tinklai turi šias problemas:

- Negali apdoroti duomenų sekos
- Priima tik dabartinę įvestį
- Negali įsiminti prieš tai buvusių įvesčių

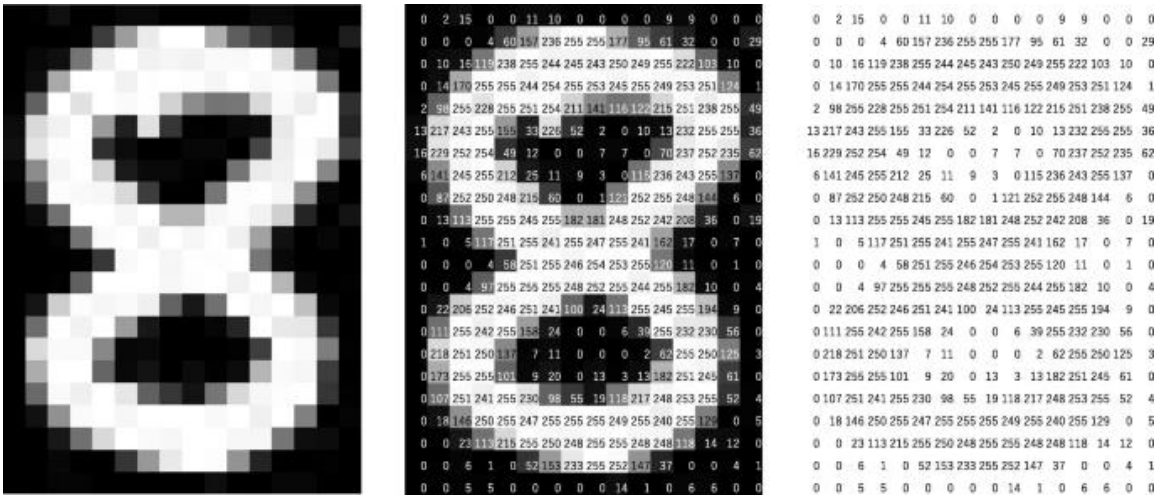
Šias problemas išsprendžia rekurentinis neuroninis tinklas. Jis gali apdoroti duomenų sekas, priimdamas dabartinę įvestį ir prieš tai gautas įvestis. Taip pat jie gali įsiminti prieš tai buvusias įvestis dėl savo turimos vidinės atminties.

Šiuose tinkluose informacija juda ciklu viduriniame sluoksnyje. Jis gali turėti kelis paslėptuosius sluoksnius, kur kiekvienas turi savo aktyvacijos funkcijas ir svorius.

Rekurentiniai neuroniniai tinklai yra dažnai naudojami kalbos atpažinimui ir teksto generavimui. Jie taip pat tinkamu bet kokiai laiko eilučių problemai, kaip pavyzdžiui akcijų kainų prognozavimas tam tikrą mėnesį [Bis23].

2.3.3. Konvoliuciniai neuroniniai tinklai

Konvoliuciniai neuroniniai tinklai (angl. convolutional neural networks) yra neuroninių tinlų klasė, kuri tapo viena iš dominuojančių tam tikruose uždaviniuose, ji vis susilaukia didelio susidomėjimo įvairiuose srityse, tokiose kaip radiologija. Šis modelis skirtas apdoroti duomenis, kurie turi tinklelio šabloną, tokius kaip paveikslėliai. Konvoliuciniai neuroniniai tinklai paprastai susideda iš trijų sluoksnių: konvoliucinis sluoksnis, sutelkimo (angl. pooling) ir pilnai sujungtų sluoksnių (angl. fully-connected layers). Pirmieji du, konvoliucinis ir sutelkimo sluoksniai atlieka savybių ištraukimą iš įvesties, o trečiasis sujungia ištrauktas savybes į galutinę išvestį. Jeigų imtume skaitmeninį paveikslėlį, jo pikselių reikšmės yra laikomos dviejų dimensijų tinklelyje, pavyzdžiui skaičių masyvas (kaip parodyta 7 paveikslėlyje) ir mažas parametrų tinklelis, vadinamas branduoliu. Tada, kiekvienai paveiklėlio pozicijai yra ištraukiama reikalinga savybė (šiuo atveju pikselio ryškumas), kas padaro konvulcinius neuroninius tinlus labai efektyvius paveikslėlių apdorojimo uždaviniui, nes reikiama savybė gali būti bet kurioje paveikslėlio vietoje [Yam18].



Pav. 7. Neuroninis tinklas paveikslėlį atpažįsta, kaip skaičių masyvą [Yam18]

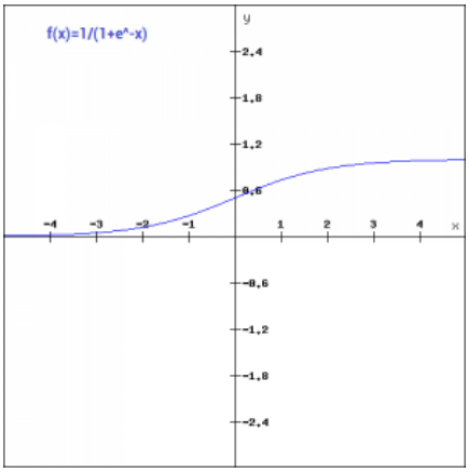
2.4. Aktyvacijos funkcijos

Aktyvacijos funkcijos yra svarbus neuroninių tinlų komponentas. Tai yra matematinės funkcijos, kurios neuroninio tinklo išvestyje įgalina netiesiškumą, kas leidžia neuroninį tinklą išmokti sudėtingus šablonus ir sąryšius esančius tarp duomenų. Netiesiškumas atsispindi tame, kad neuroninio tinklo išvestis nėra tiesiogiai proporcinga įvesčiai. Kitaip tariant, jeigu padvigubinsime įvestį, išvestis nebūtinai pasidvigubins. Be netiesiškumo neuroninis tinklas būtų tiesiog tiesinė įvesties funkcija. Toliau aptarsime skirtingas aktyvacijos funkcijas.

2.4.1. Sigmoido funkcija

Tai yra plačiausiai naudojama aktyvacijos funkcija, kadangi tai yra ne linijinė funkcija. Sigmoido funkcija transformuoja reikšmes nuo 0 iki 1. Ji gali būti apibrėžta, kaip f(x) = 1/e-x. Sigmoido funkcija yra nuolat diferencijuojama. Jos grafikas yra S raidės formos. Jos išvestinė

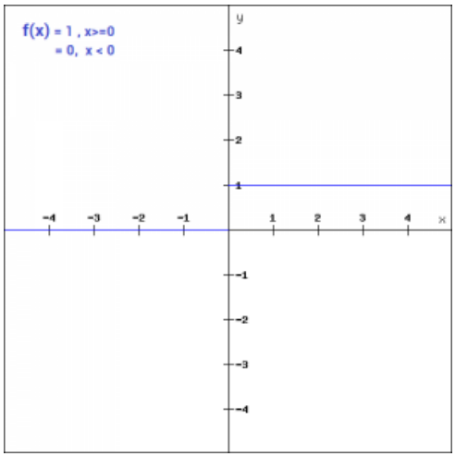
yra: f’(x) = 1-sigmoid(x). Taip pat ji yra ne simetriška ties 0, todėl visų neuronų išvesties reikšmės turės tuos pačius ženklus [SSA20].



Pav. 8. Sigmoido funkcija [SSA20].

2.4.2. Binarinė žingsnio funkcija

Tai yra paprasčiausia aktyvacijos funkcija, kuri egzistuoja ir kuri gali paprasčiausiai būti aprašyta „if-else“ sąlygomis, kokioje nors programavimo kalboje. Norit sukurti binarinį klasifikatorių paprasčiausiai yra naudojama binarinė žingsnio funkcija. Matematiškai ji yra apibrėžiama taip: f(x) = 1, x>=0; f(x)= 0, x<0 [SSA20].

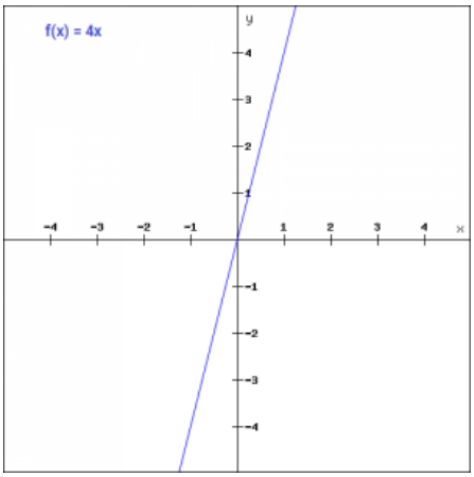


Pav. 9. Binarinė žingsnio funkcija [SSA20].

2.4.3. Linijinė aktyvacijos funkcija

Ši funkcija yra tiesiogiai proporcinga įvesčiai. Pagrindinis binarinio žingsnio funkcijos trūkumas buvo tas, kad ji turi nulinį gradientą, nes toje funkcijoje nėra komponento „x“. Tam, kad to išvengti gali būti naudojama linijinė funkcija. Ji gali būti apibrėžta taip: F(x) = ax. Kintamojo reikšmė gali būti pasirinkta bet kokia konstanta.

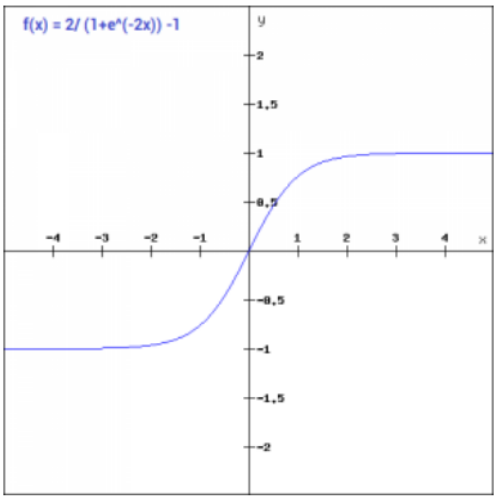
Tiesa, daug privalumų naudoti šią funkciją nėra, kadangi neuroninis tinklas nepagerins savo tikslumo dėl tos pačios gradianto reikšmės kiekvienoje iteracijoje. Taip pat, tinklas negalės identifikuoti sudėtingų modelių. Būtent todėl ši funkcija yra labiau naudojama paprastesniems uždaviniams [SSA20].



Pav. 10. Linijinė aktyvacijos funkcija [SSA20].

2.4.4. Hiperbolinė tangento funkcija (TANH)

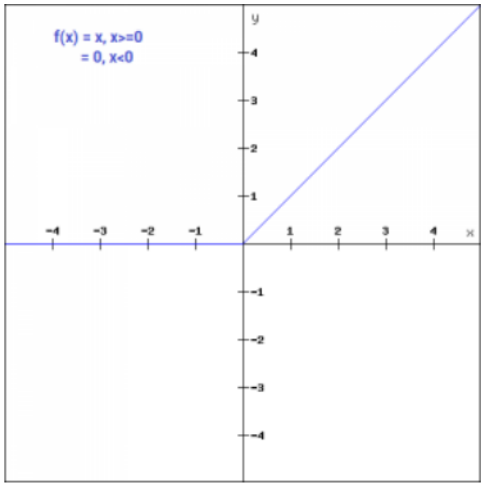
Ši funkcija yra panaši į Sigmoido funkciją, tačiau ji yra simetriška. Ji apibrėžiama taip: f(x) = 2sigmoid(2x)-1. Ji transformuoja reikšmes nuo -1 iki 1. Lyginant su Sigmoido funkcija jos gradientas yra statesnis. Ši funkcija kai kuriais atvejais yra labiau naudojama už Sigmoido, kadangi jos gradientas nėra apribotas judėti tik viena kryptimi ir jis yra centruotas ties 0 [SSA20].



Pav. 11. Hiperbolinė tangento aktyvacijos funkcija [SSA20].

2.4.5. ReLU aktyvacijos funkcija

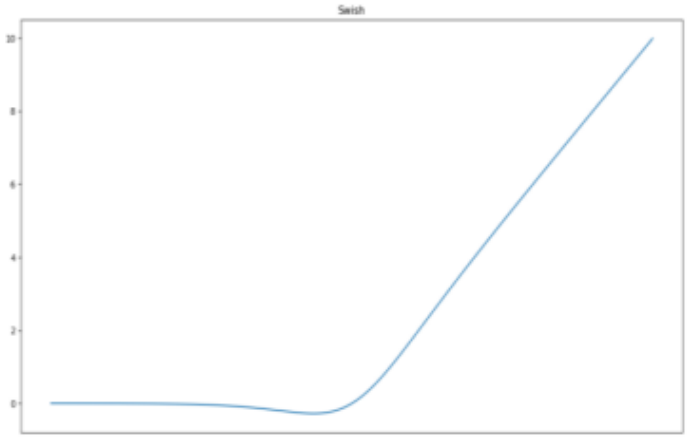
ReLU (angl. Rectified Linear Unit) reiškia „dalinis tiesinis vienetas“. Tai yra nelinijinė funkcija, dažnai naudojama neuroniniuose tinkluose. Pagrindinis šios funkcijos privalumas yra tas, jog visi neuronai nėra aktyvuojami vienu metu. Tai reiškia, kad neuronas bus deaktyvuotas tik tada kai tiesinės transformacijos išvestis bus lygus nuliui. Matematiškai ji apibrėžiama taip: f(x) = max(0,x). ReLU yra našesnė funkcija, kadangi tam tikras skaičius neuronų yra aktyvuojamas vienu momentu, bet nėra sktyvuojami visi [SSA20].



Pav. 12. ReLU aktyvacijos funkcija [SSA20].

2.4.6. Švytuojanti aktyvacijos funkcija

Švytuojanti (angl. Swish) aktyvacijos funkcija yra ganėtinai nauja funkcija. Ji išsiskiria tuo, jog ji yra monotoniška, kas reiškia, kad funkcijos reikšmė gali mažėti, nors ir įvesties reikšmės didėja. Tam tikrais atvejais švytuojanti funkcija nugali net ReLU funkciją. Jos matematinė išraiška yra: f(x) = x * sigmoid(x); $f(x) = x/(1 - e^{-x})$.



Pav. 13. Švytuojanti aktyvacijos funkcija [SSA20].

3. Tyrimai

Tyrimus atliksime keliais metodais. Juos aprašysime tolimesniuose skyreliuose. Iš pradžių nenaudosime neuroninio tinklo, o likusiuose bandymuose – naudosime.

3.1. Bandymų aprašymas

Turimi duomenys: Šveicarijos miesto, Bazelio, oro temperatūra, slėgis, oro drėgnumas ir vėjo greitis kas valandą nuo 2000 m. sausio 1d. 00:00 val. iki 2022 m. gegužės 16 d. 23:00 val. (iš viso 196128 valandos duomenų).

Tikslas: žinant praeities oro temperatūrą ir kitus duomenis prognozuoti kokia bus oro temperatūra po 3 val. Palyginti prognozę su faktine oro temperatūra.

Bandymai: bandysime prognozuoti kokia bus oro temperatūra kiekvienų metų kovo 7 – 21 dienomis, 06:00 val., t.y. imsime kiekvienos kovo 7 – 21 dienos, 00:00 val., 01:00 val., 02:00

val. ir 03:00 val. oro temperatūras ir kitus duomenis, prognozuosime kokia temperatūra bus 06:00 val. Tolimesniuose bandymuose imsime ne tik kovo mėnesio duomenis, bet ir visų metų, bei visos paros duomenis. Bandymus atliksime keliais metodais, rezultatus palyginsime.

3.2. Orų prognozavimas nenaudojant neuroninių tinklų

Pirmiausia atlikime tyrimą nenaudodami neuroninio tinklo. Atlikę tokį tyrimą, rezultatus galėsime palyginti su neuroninių tinklų tyrimų rezultatais.

3.2.1. Spėjimo būdo prognozė

Pirmiausia bandysime atlikti pačią paprasčiausią prognozę, t.y. spėsime. Palyginsime savo spėjima su faktine oro temperatūra.

3.2.1.1. Pirmas eksperimentas

Tarkime, kad po 3 val. oro temperatūra bus tokia pati kaip ir prieš 3 valandas.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras.
2. Testavimui imkime paskutinius 20% gautų duomenų (oro temperatūrą nuo 2018 m. kovo 7 d. iki 2022 m. kovo 21 d. imtinai).
3. Iš turimų duomenų imkime kiekvienų metų kovo 7 – 21 dienų 03:00 val. oro temperatūrą ir teikime, kad po 3 valandų (06:00 val.) ji bus tokia pati. Taip pat pasiimkime faktinę kiekvienų kovo 7 – 21 dienų 06:00 val. temperatūrą, tam, kad galėtume palyginti su prognoze.
4. Prognozuokime, kad visų turimų metų kovo 7 – 21 dieną, 06:00 val. oro temperatūra bus tokia pati kaip ir buvo prieš 3 valandas (03:00 val.).
5. Pasinaudoję matlab programavimo kalba palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
6. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

Gauti rezultatai:

	Testavimas
Prognozių skč.	75
Min. absoliuti paklaida	0.01
Max. absoliuti paklaida	3.85
Absoliutinių paklaidų vidurkis	**1.0161**
Vidutinė kvadratinė paklaida (Mean Squared Error)	1.5618
Paklaidų standartinis nuokrypis	0.7324

Lent. 1. 1 eksperimento rezultatai.

1 lentelėje matome, kad visų absoliutinių paklaidų vidurkis 1.016 °C, kas yra gana didelis skaičius. Tolimesniuose bandymuose bandysime sumažinti šį vidurkį.

3.2.2. Tiesinė regresija

Tiesinė regresija, tai statistikos moksle sukurtas modelis, kuris skirtas suprasti ir tirti ryšį tarp įvesties ir išvesties skaitinių parametrų. Tai yra tiesinis modelis, kuris pvz. Daro prielaidą, jog įvesties kintamieji yra tiesiškai susiję su išvesties kintamuoju. Tai gali būti tiesiog tiesinė

lygtis, sujungianti tam tikrą įvesties reikšmių rinkinį, kurios sprendimas būtų pronozuojama išvestis. Tiek įvesties reikšmės, tiek išvesties yra skaitinės [Bro20].

3.2.2.1. Antras eksperimentas

Pabandykime prognozuoti kokia oro temperatūra bus po 3 val. pasitelkiant tiesinės regresijos (angl. linear regression) modelį. Kaip įvesties duomenis naudokime ne tik kiekvienos valandos temperatūrą, bet ir oro drėgnumą, slėgį, vėjo greitį. Šio modelio išvestis bus prognozuojama oro temperatūra. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus tiesinės regresijos modelio treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Mūsų modelis turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų kovo 7 – 21 dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūrą atitinkamomis dienomis 06:00 val.
5. Apmokykime modelį su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *fitlm*.
6. Kai tiesinės regresijos modelis jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per šį modelį paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
7. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
8. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

Gauti rezultatai:

	Treniravimas	**Testavimas**
Prognozių skč.	270	75
Min. absoliuti paklaida	0.004	0.006
Max. absoliuti paklaida	2.998	2.995
Absoliutinių paklaidų vidurkis	**0.541**	**0.532**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.541	0.678
Paklaidų standartinis nuokrypis	0.532	0.663

Lent. 2. 2 eksperimento rezultatai.

2 lentelėje matome, kad testavimo absoliutinių paklaidų vidurkis yra 0.532 °C, taigi mum pavyko šį skaičių sumažinti po pirmojo eksperimento. Vis gi vidurkis yra ganėtinai aukštas, todėl tolimesniuose bandymuose bandysime jį sumažinti.

3.2.2.2. Trečias eksperimentas

Pabandykime prognozuoti kokia oro temperatūra bus po 3 val. pasitelkiant tiesinės regresijos modelį. Kaip įvesties duomenis naudokime ne kovo 7 – 21 dienų, o visų metų

duomenis. Šio modelio išvestis bus prognozuojama oro temperatūra. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime kiekvienos dienos, 00:00 val., 01:00 val., 02:00 val., 03:00 val. ir 06:00 val. oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus tiesinės regresijos modelio treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1.
4. Mūsų modelis turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų, kiekvienos dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūrą atitinkamomis dienomis 06:00 val.
5. Apmokykime modelį su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *fitlm*.
6. Kai tiesinės regresijos modelis jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per šį modelį paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
7. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
8. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

Gauti rezultatai:

	Treniravimas	**Testavimas**
Prognozių skč.	6407	1765
Min. absoliuti paklaida	0	0
Max. absoliuti paklaida	5.737	4.811
Absoliutinių paklaidų vidurkis	**0.548**	**0.581**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.581	0.698
Paklaidų standartinis nuokrypis	0.566	0.637

Lent. 3. 3 eksperimento rezultatai.

3 lentelėje matome, kad testavimo absoliutinių paklaidų vidurkis yra 0.581 °C, taigi panaudojus daugiau duomenų tiesinės regresijos įvestyje mum nepavyko sumažinti paklaidos. Tolimesniuose bandymuose bandysime naudoti neuroninius tinklus ir gauti geresnius rezultatus.

3.3. Orų prognozavimas naudojant neuroninius tinklus

Toliau ištreniruosime neuroninį tinklą ir naudosime jį prognozavimui.

3.3.1. Pasiruošimas tyrimams

Prieš pradedant tyrimus su neuroniniais tinklais aptarkime kaip normalizuosime visus duomenis ir kaip stengsimės, kad neuroninis tinklas būtų efektyvus ir neįvyktų persimokymas.

3.3.1.1. Neuroninio tinklo pasirinkimas

Iš jau aptartų tinklų tipų imsime - tiesinio skleidimo neuroninį tinklą, kadangi tai yra paprasčiausias tinklas, kuris puikiai tiks vieno ar kelių skaičių prognozavimui.

Taip pat reikia pasirinkti aktyvacijos funkciją, kadangi neuroninis tinklas be aktyvacijos funkcijos veiktų kaip tiesinės regresijos modelis (jį naudojome 3.2.2 skyrelyje). Tokiu atveju neuroninis tinklas turėtų limituotą našumą ir tikslumą. Mes norime, kad neuroninis tinklas sugebėtų ne tik apskaičiuoti tiesines funkcijas, bet ir atlikti sudėtingesnius skaičiavimus [SSA20]. Kadangi mūsų neuroninis tinklas yra tiesinio skleidimo, pasirinksime Sigmoido aktyvacijos funkciją. Ši funkcija naudojama su vieno paslėptojo sluoksnio tiesinio skleidimo neuroniniu tinklu gali aproksimuoti bet kokią tolydžią funkciją tam tikru tikslumo laipsniumi, priklausomai nuo neuronų esančių paslėptąjame sluoksnyje. Kitaip tariant tiesinio skleidimo neuroninis tinklas su vienu paslėptuoju sluoksniu yra universalus aproksimatorius. Jis gali apskaičiuoti sudėtingas funkcijas aukštu tikslumu [Cyb89].

Taigi, naudosime tiesinio skleidimo neuroninį tinklą, su vienu paslėptuoju sluoksniu, įvairiu neuronų skaičiu jame ir Sigmoido aktyvacijos funkcija.

3.3.1.2. Duomenų normalizavimas

Prieš pradedant treniruoti mūsų neuroninį tinklą normalizuosime visus duomenis. Normalizavimas yra metodas, dažnai taikomas ruošiant duomenis mašininiam mokymuisi. Normalizavimo tikslas – pakeisti skaitinių stulpelių reikšmes duomenų rinkinyje į bendrą skalę, neiškraipant reikšmių diapazono [Jai18]. Nenormalizuoti duomenys gali nulemti lėtą arba nestabilų neuroninio tinklo treniravimo procesą. Neuroninio tinklo įvesties kintamieji gali turėti skirtingus vienetus (pvz., pėdas, kilometrus ir valandas), o tai savo ruožtu gali reikšti, kad kintamieji turi skirtingą skalę. Įvesties kintamųjų skalių skirtumai gali apsunkinti modeliuojamo problemos sprendimą [Bro19]. Kadangi mes turime skirtingus vienetus, t.y. celcijus, paskalius, metrus per sekundę ir t.t., normalizavimas mums padės visus šiuos duomenis pakeisti į tam tikrą skalę, bet tuo pačiu išsaugoti reikšmių skirtumus. Tai turėtų padėti pagerinti mūsų neuroninio tinklo darbą. Duomenų normalizavimui naudosime matlab funkciją - *mapminmax* į kurią, kaip parametrą, perduosime duomenis kuriuos norime normalizuoti kartu, taip pat paduosime intervalą, t.y. nuo -1 iki 1. Taip mes normalizuosime visas reikšmes, nuo -1 iki 1. Norėdami denormalizuoti, naudosime tą pačią funkciją, tik su parametru *reverse*. Šio proceso metu, 00:00 val., 01:00 val., 02:00 val., 03:00 val. ir 06:00 valandos oro temperatūras normalizuosime kartu, kadangi tai yra to pačio mato duomenys (laipsniai celcijaus), o oro slėgį, vėjo greitį ir drėgnumą normalizuosime visus atskirai, kadangi tai yra skirtingų matmenų duomenys.

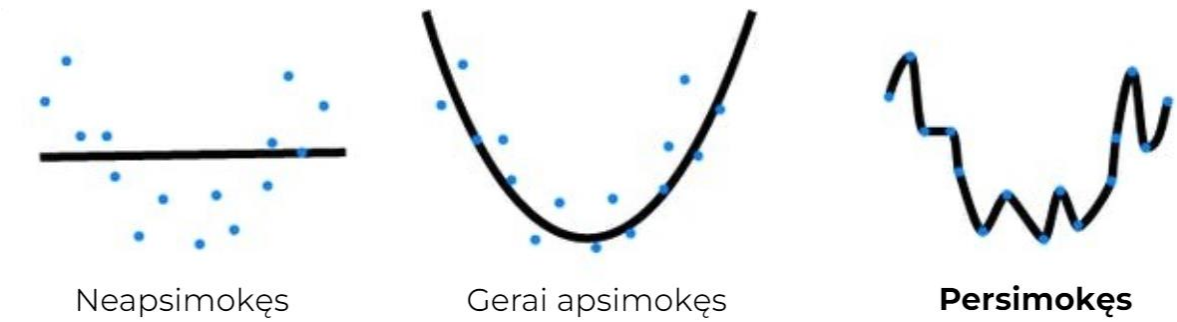
3.3.1.3. Neuroninio tinklo persimokymas (angl. *overfitting*)

Prieš pradedant atlikti bandymus su neuroniniais tinklais reikia išmokti įvertinti ir atskirti kada mūsų neuroninio tinklo apmokymas pavyksta, kada jis neapsimoko arba kada jis persimoko. Neuroninio tinklo tikslas yra turėti galutinį modelį, kuris gerai veiktų tiek su duomenims, kuriuos naudojome mokydami (pvz., mokymo duomenų rinkinį), tiek su naujais duomenimis, pagal kuriuos modelis bus naudojamas prognozėms. [Bro18] Taigi, egzistuoja šie neuroninio tinklo apsimokymo modeliai:

- **Neapsimokęs modelis (angl. Underfit Model)**. Modelis, kuris nesugeba pakankamai išmokti problemos ir prastai veikia mokymo duomenų rinkinyje, taip pat prastai veikia ir validavimo duomenų rinkinyje.
- **Persimokęs modelis (angl. Overfit Model)**. Modelis, kuris per gerai įsisavina mokymo duomenų rinkinį, gerai veikia mokymo duomenų rinkinyje, bet neveikia gerai su naujais, nematytais duomenimis.

- **Gero apsimokymo modelis (angl. Good Fit Model)**. Modelis, kuris tinkamai išmoksta mokymo duomenų rinkinį ir gerai atlieką darbą su naujais, nematytais duomenimis [Bro18].

14 paveikslėlyje galima matyti kuo skiriasi minėti modeliai, pagal tikrus ir prognozuojamus duomenis.

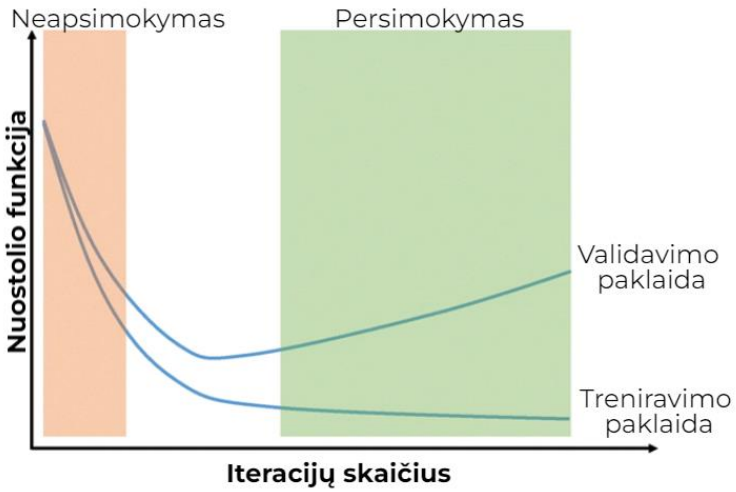


Pav. 14. Neuroninio tinklo mokymosi modeliai [Des18].

Vienas iš labiausiai paplitusių būdų, kaip aptikti persimokymą, yra nubrėžti treniruočių ir validavimo paklaidų kreives treniruotės metu. Jei treniruočių paklaida mažėja, o validavimo paklaida didėja arba nesikeičia, tai yra persimokymo požymis. Yra du būdai, kaip išvengti arba sumažinti persimokymą:

- Apmokyti neuroninį tinklą su daugiau duomenų.
- Keisti neuroninio tinklo sudėtingumą [Bro18].
- Tam, kad sumažinti persimokymo tikimybę mūsų programoje naudosime:
- Daug apmokymo duomenų (22 metų duomenis);
- Uždėsime neuroninio tinklo apsimokymo stabdymo sąlygą - 100 iteracijų;
- Tam, kad išvengti efektyvumo prastėjimo su validavimo duomenimis mes uždėsime stabdymo sąlyga tada, kai vidutinė paklaida nebemažėja;
- Stabdysime treniravimą, kai vidutinė paklaida pasiekia tam tikrą reikšmę;
- Nustatysime minimalią reikšmę, kuria turi pagerėti validavimo duomenų tikslumas, tam, kad būtų tęsiamas treniravimas. Taigi, jeigu validavimo duomenų tikslumas ims gerėti labai mažai - stabdysime treniravimą.

Visi šie stabdymo kriterijai yra svarbūs, tam, kad išvengtumę persimokymo ir neuroninio tinklo treniravimas sustotų tinkamu laiku. Viso to išpildymui padės matlab aplinkoje esanti *trainParam* struktūra. Taip pat po kiekvieno bandymo nubrėšime treniravimo ir validavimo paklaidų grafikus bei palyginsi juos su 15 paveikslėliu, kuriame yra pavaizduota kaip tokiame grafike atrodo neapsimokęs arba persimokęs modelis.



Pav. 15. Treniravimo ir validavimo paklaidų grafikas.

3.3.2. Tiesinio skleidimo neuroninis tinklas

Toliau sukursime ir bandymams naudosime mūsų pasirinktą tiesinio skleidimo neuroninį tinklą.

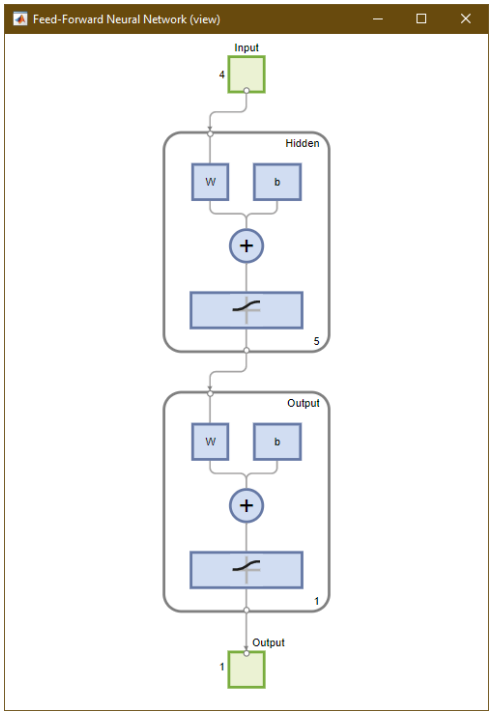
3.3.2.1. Ketvirtas eksperimentas

Pabandykime prognozuoti kokia oro temperatūra bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą, naudojant matlab programavimo kalbą. Kaip neuroninio tinklo įvestį imkime tik oro temperatūras, kitų duomenų kol kas neimkime.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui (tuos pačius duomenis kaip ir 1 metode).
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.
5. Mūsų neuroninis tinklas turės 4 įvestis ir 1 išvestį. Įvestys bus kiekvienų metų kovo 7 – 21 dienos temperatūra 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūrą atitinkamomis dienomis 06:00 val.
6. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

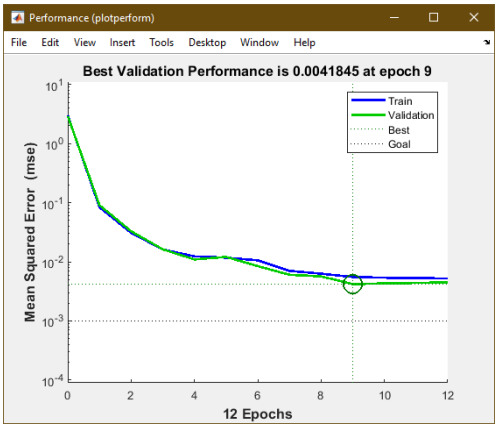
Gauti rezultatai:



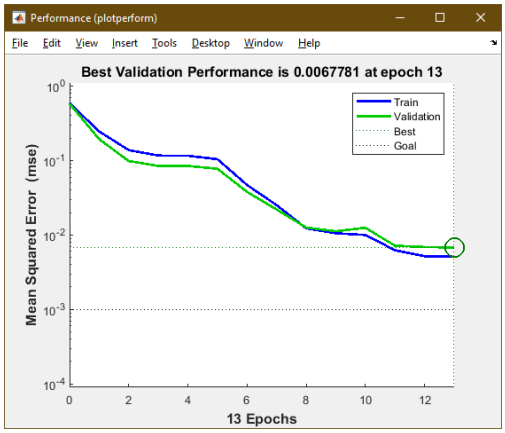
Pav. 16. Neuroninio tinklo diagrama.

16 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 4 įvesties parametrus ir 1 išvesties parametrą.

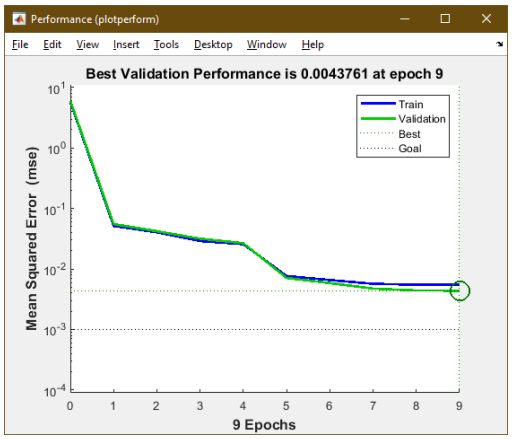
Treniravimo ir validavimo tikslumo grafikai:



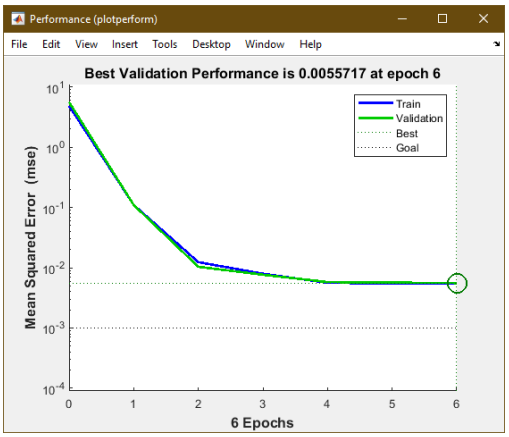
Pav. 17. 5 paslėptieji neuronai.



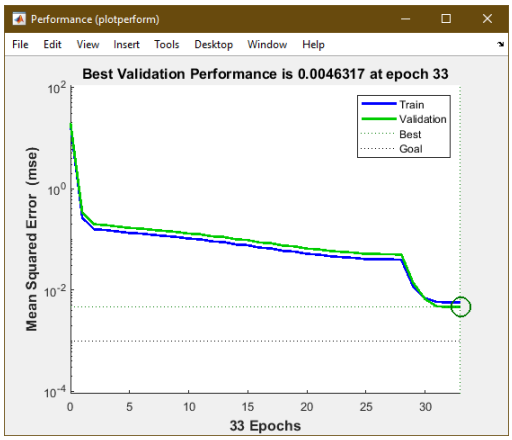
Pav. 18. 10 paslėptųjų neuronų.



Pav. 19. 15 paslėptųjų neuronų.



Pav. 20. 30 paslėptųjų neuronų.



Pav. 21. 50 paslėptųjų neuronų.

17 – 21 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	270	270	270	270	270
Min. absoliuti paklaida	0.0125	0.0055	0.0008	0.0007	0.0112
Max. absoliuti paklaida	2.8921	3.007	2.8968	3.1094	2.9355
Absoliutinių paklaidų vidurkis	**0.5074**	**0.5092**	**0.507**	**0.5068**	**0.5303**

Vidutinė kvadratinė paklaida (Mean Squared Error)	0.4973	0.5127	0.4946	0.514	0.5139
Paklaidų standartinis nuokrypis	0.4906	0.5043	0.4883	0.5079	0.4832

Lent. 4. 4 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	75	75	75	75	75
Min. absoliuti paklaida	0.0176	0.0026	0.0075	0.0092	0.0182
Max. absoliuti paklaida	2.9108	2.9882	3.043	2.9581	2.8671
Absoliutinių paklaidų vidurkis	**0.5019**	**0.4944**	**0.5225**	**0.4987**	**0.5358**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.6341	0.6466	0.6761	0.6450	0.6486
Paklaidų standartinis nuokrypis	0.6223	0.6384	0.6391	0.6338	0.6052

Lent. 5. 4 eksperimento testavimo rezultatai.

Remiantis 4 ir 5 lentelėmis galime teigti, kad naudoti tiesinio skleidimo neuroninį tinklą jau yra efektyviau negu tiesinę regresiją. Toliau bandykime padidinti įvesties parametrų skaičių ir taip dar pagerinti neuroninio tinklo tikslumą.

3.3.2.2. Penktas eksperimentas

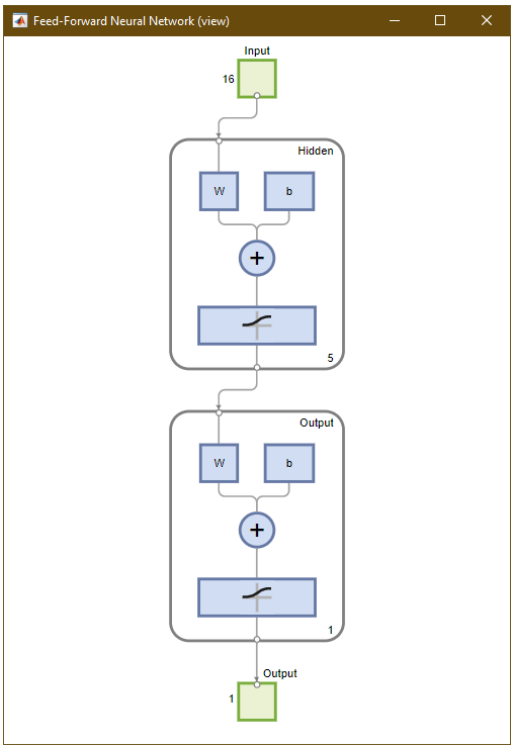
Pabandykime prognozuoti kokia oro temperatūra bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime ne tik kiekvienos valandos temperatūrą, bet ir oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo išvestis lieka ta pati - prognozuojama oro temperatūra. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.

5. Mūsų neuroninis tinklas turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų kovo 7 – 21 dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūrą atitinkamomis dienomis 06:00 val.
6. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

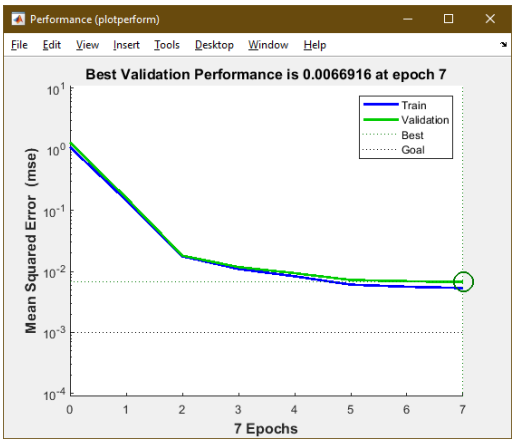
Gauti rezultatai:



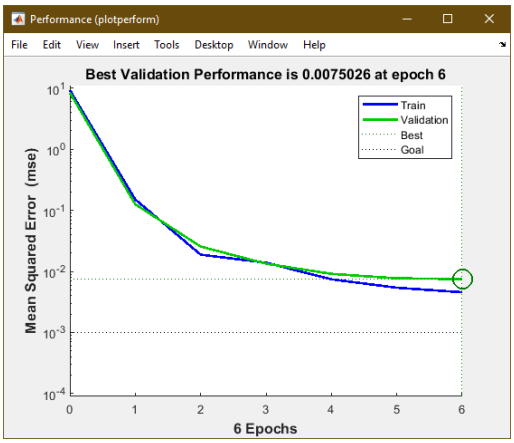
Pav. 22. Neuroninio tinklo diagrama.

22 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 1 išvesties parametrą.

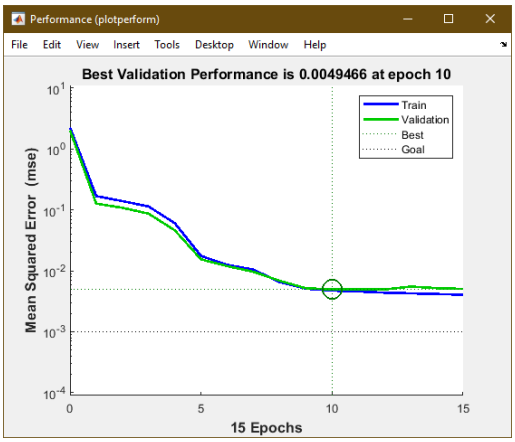
Treniravimo ir validavimo tikslumo grafikai:



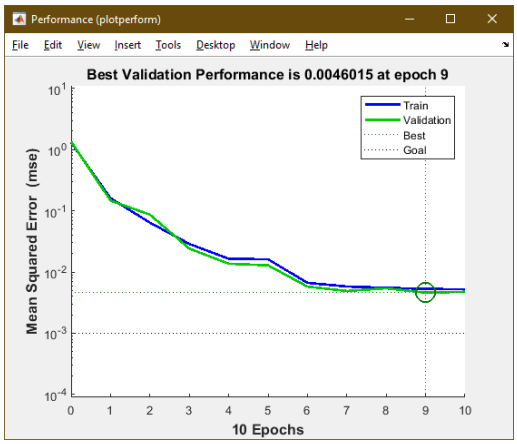
Pav. 23. 5 paslėptieji neuronai.



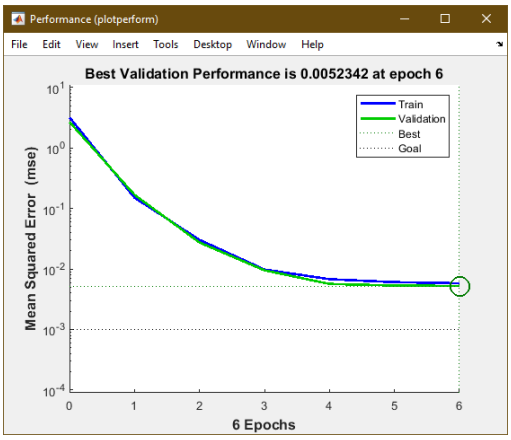
Pav. 24. 10 paslėptųjų neuronų.



Pav. 25. 15 paslėptųjų neuronų.



Pav. 26. 30 paslėptųjų neuronų.



Pav. 27. 50 paslėptųjų neuronų.

23 – 27 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	10 neuronų	15 neuronų	30 neuronų	50 neuronų
Prognozių skč.	270	270	270	270	270
Min. absoliuti paklaida	0.0023	0.0010	0	0.0051	0.0037
Max. absoliuti paklaida	3.715	3.1283	2.8584	3.0735	3.6675
Absoliutinių paklaidų vidurkis	**0.5207**	**0.5039**	**0.4871**	**0.5084**	**0.5250**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.5301	0.4857	0.4506	0.4875	0.5365
Paklaidų standartinis nuokrypis	0.5097	0.4823	0.4627	0.4795	0.5116

Lent. 6. 5 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	10 neuronų	15 neuronų	30 neuronų	50 neuronų
Prognozių skč.	75	75	75	75	75
Min. absoliuti paklaida	0.0061	0.0167	0.0018	0.0029	0.0040
Max. absoliuti paklaida	3.254	3.3227	3.1585	3.1283	3.2112
Absoliutinių paklaidų vidurkis	**0.5058**	**0.4764**	**0.4954**	**0.4857**	**0.5139**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.6161	0.5814	0.5824	0.5841	0.6061
Paklaidų standartinis nuokrypis	0.6042	0.5993	0.5843	0.5940	0.5887

Lent. 7. 5 eksperimento testavimo rezultatai.

Remiantis 6 ir 7 lentelėmis galime teigti, jog didesnis įvesties parametrų skaičius padėjo pagerinti neuroninio tinklo prognozavimo tikslumą. Toliau bandykime išvestyje pridėti daugiau parametrų ir rezultatus palyginti su šiuo bandymu.

3.3.2.3. Šeštas eksperimentas

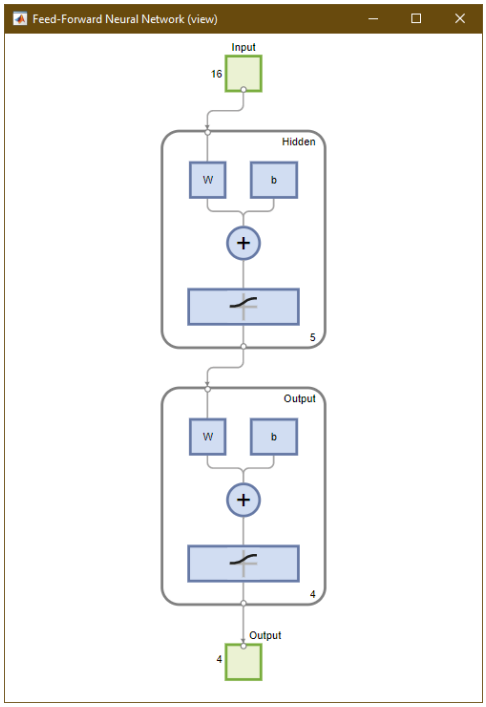
Pabandykime prognozuoti kokia oro temperatūra, slėgis, oro drėgmė ir vejo greitis bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo

išvestyje prognozuokime, ne tik kokia bus oro temperatūra, bet ir 3 papildomi faktoriai: slėgis, oro drėgmė ir vejo greitis. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.
5. Mūsų neuroninis tinklas turės 16 įvesčių ir 4 išvestis. Įvestys bus kiekvienų metų kovo 7 – 21 dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestys - pronozuojama oro temperatūra, slėgis, drėgnumas ir vėjo greitis atitinkamomis dienomis 06:00 val.
6. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį (imame tik prognozuojamą oro temperatūrą).

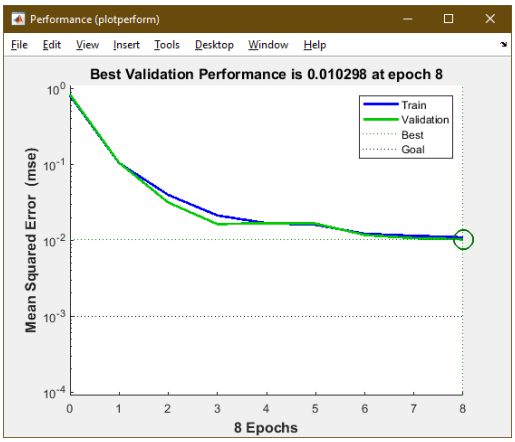
Gauti rezultatai:



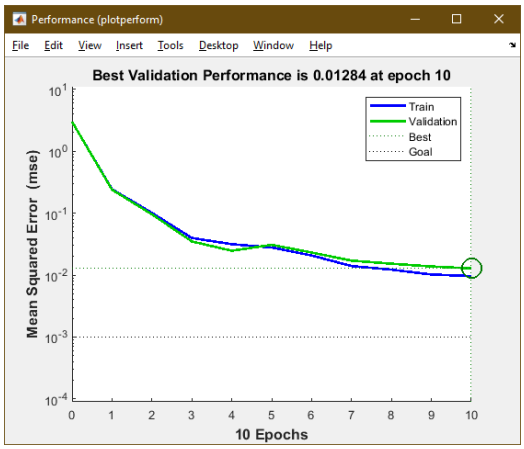
Pav. 28. Neuroninio tinklo diagrama.

28 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 4 išvesties parametrus.

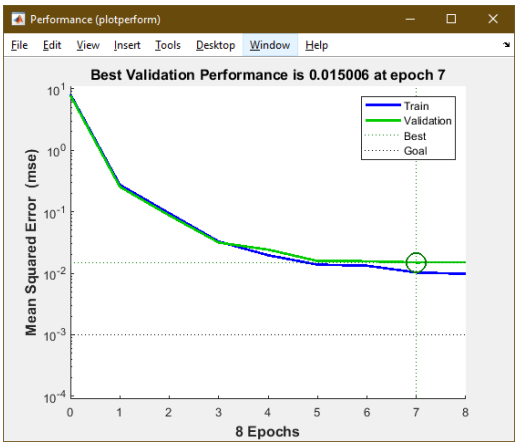
Treniravimo ir validavimo tikslumo grafikai:



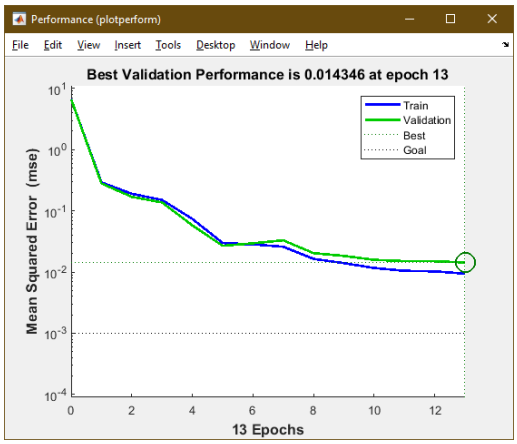
Pav. 29. 5 paslėptieji neuronai.



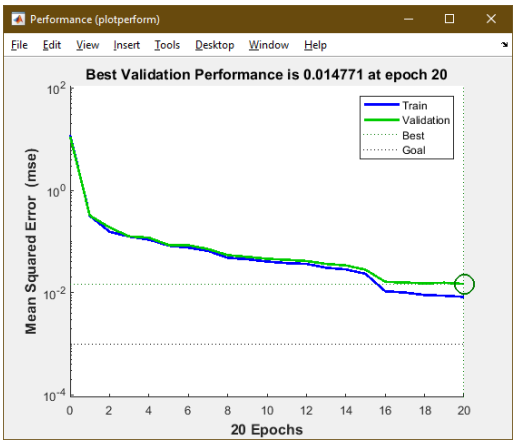
Pav. 30. 10 paslėptųjų neuronų.



Pav. 31. 15 paslėptųjų neuronų.



Pav. 32. 30 paslėptųjų neuronų.



Pav. 33. 50 paslėptųjų neuronų.

29 – 33 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	270	270	270	270	270
Min. absoliuti paklaida	0.0003	0.0038	0.0038	0.0028	0.0009
Max. absoliuti paklaida	3.7123	3.1966	3.2028	2.6283	2.6183
Absoliutinių paklaidų vidurkis	**0.4968**	**0.5055**	**0.5507**	**0.4941**	**0.4845**

Vidutinė kvadratinė paklaida (Mean Squared Error)	0.4930	0.4797	0.5266	0.4626	0.4480
Paklaidų standartinis nuokrypis	0.4971	0.4743	0.4734	0.4682	0.4626

Lent. 8. 6 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	75	75	75	75	75
Min. absoliuti paklaida	0.0086	0.0008	0.0079	0.0053	0.0042
Max. absoliuti paklaida	3.2649	3.1772	3.0824	2.9101	2.8315
Absoliutinių paklaidų vidurkis	**0.5058**	**0.5278**	**0.5511**	**0.5013**	**0.5002**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.6108	0.6231	0.6383	0.6262	0.5830
Paklaidų standartinis nuokrypis	0.5997	0.5909	0.5823	0.5837	0.58

Lent. 9. 6 eksperimento testavimo rezultatai.

Remiantis 8 ir 9 lentelėmis galime teigti, jog didesnis išvesties parametrų kiekis labai minimaliai parodė geresnius rezultatus. Tolimesniuose eksperimentuose naudokime vieną išvesties parametrą – oro temperatūrą, tam, kad neuroninis tinklas būtų paprastesnis.

3.3.2.4. Septintas eksperimentas

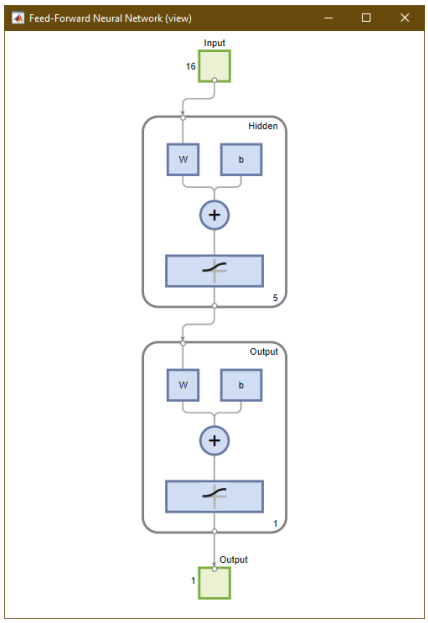
Pabndykime imti daugiau duomenų. Prognozuokime kokia oro temperatūra, bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo išvestyje prognozuokime, kokia bus oro temperatūra. Imkime ne kovo 7 – 21 dienų, o visų metų duomenis. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime kiekvienos dienos, 00:00 val., 01:00 val., 02:00 val., 03:00 val. ir 06:00 val. oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.

4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.
5. Mūsų neuroninis tinklas turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų, kiekvienos dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūrą atitinkamomis dienomis 06:00 val.
6. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

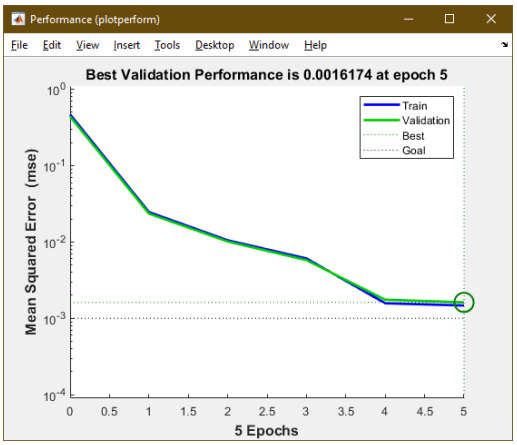
Gauti rezultatai:



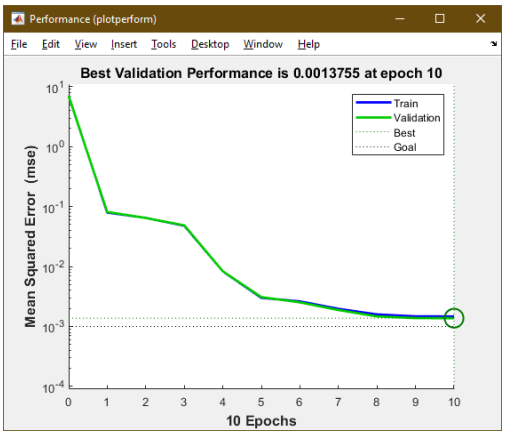
Pav. 34. Neuroninio tinklo diagrama.

34 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 1 išvesties parametrą.

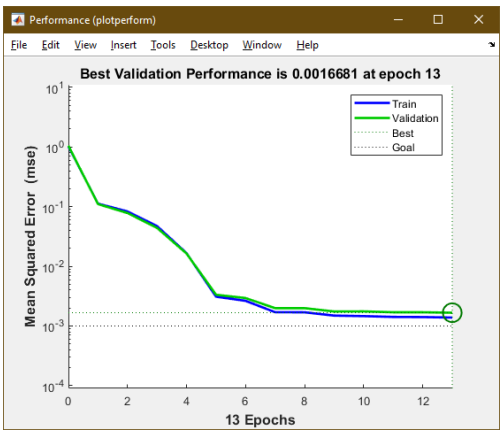
Treniravimo ir validavimo tikslumo grafikai:



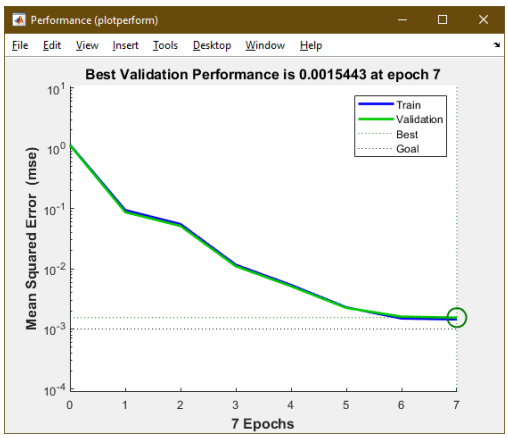
Pav. 35. 5 paslėptieji neuronai.



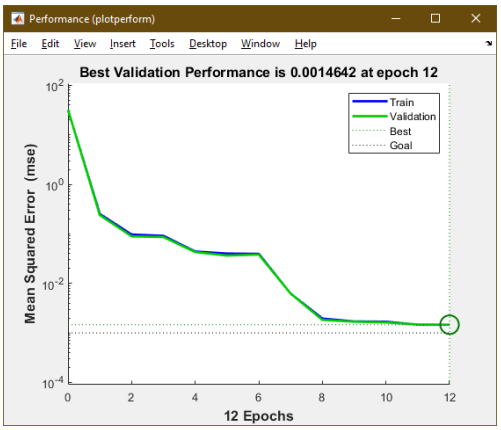
Pav. 36. 10 paslėptųjų neuronų.



Pav. 37. 15 paslėptųjų neuronų.



Pav. 38. 30 paslėptųjų neuronų.



Pav. 39. 50 paslėptųjų neuronų.

35 – 39 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	6407	6407	6407	6407	6407
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	5.482	5.29	5.12	5.35	5.209
Absoliutinių paklaidų vidurkis	**0.509**	**0.478**	**0.455**	**0.433**	**0.427**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.544	0.501	0.488	0.455	0.44
Paklaidų standartinis nuokrypis	0.535	0.499	0.481	0.459	0.447

Lent. 10. 7 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	1765	1765	1765	1765	1765
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	4.923	5.104	5.408	5.2	5.32
Absoliutinių paklaidų vidurkis	**0.507**	**0.5**	**0.49**	**0.478**	**0.466**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.667	0.601	0.589	0.561	0.548
Paklaidų standartinis nuokrypis	0.606	0.589	0.577	0.561	0.545

Lent. 11. 7 eksperimento testavimo rezultatai.

Remiantis 10 ir 11 lentelėmis galime teigti, jog didesnis duomenų kiekis padėjo pagerinti neuroninio tinklo prognozavimo rezultatus.

3.3.2.5. Aštuntas eksperimentas

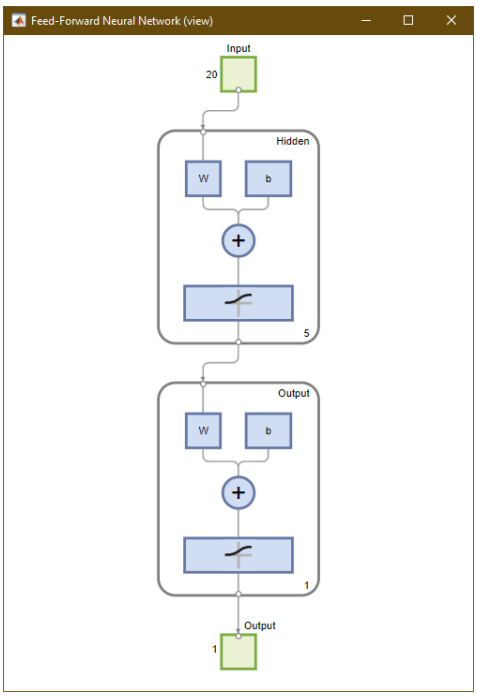
Pabandykime parametrizuoti paros laiką. Prognozuokime kokia oro temperatūra, bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį ir paros valandą. Neuroninio tinklo išvestyje prognozuokime, kokia bus oro temperatūra. Vėl

imkime kovo 7 – 21 dienas, tačiau šį kartą prognozuokime kokia oro temperatūra bus ne tik 06:00 valandą, bet kiekvieną paros valandą. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime kovo 7 – 21 dienų, kiekvienos valandos oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis išskyrus paros valandą nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.
5. Mūsų neuroninis tinklas turės 20 įvesčių ir 1 išvestį. Įvestys bus kovo 7 – 21 dienų, kiekvienos valandos temperatūra, oro drėgnumas, slegis, vėjo greitis ir valanda, o išvestis - pronozuojama oro temperatūrą po 3 valandų.
6. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

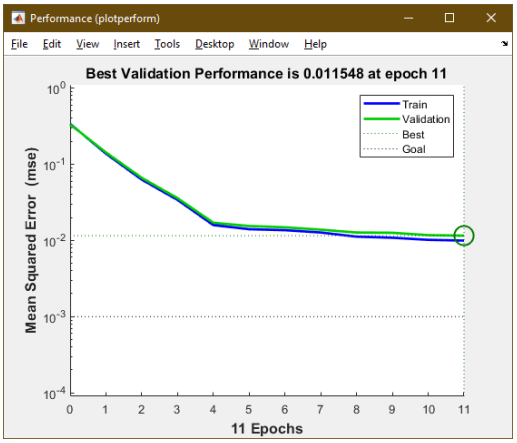
Gauti rezultatai:



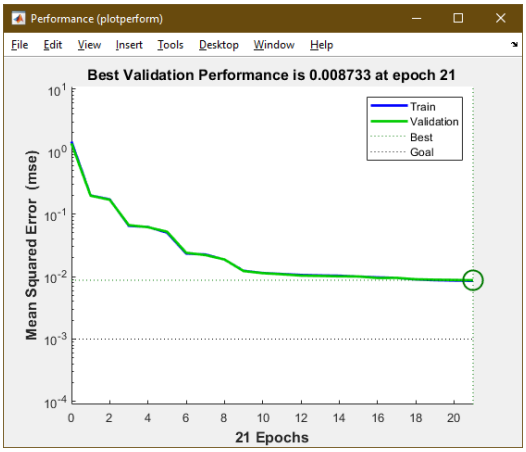
Pav. 40. Neuroninio tinklo diagrama.

40 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 20 įvesties parametrų ir 1 išvesties parametrą.

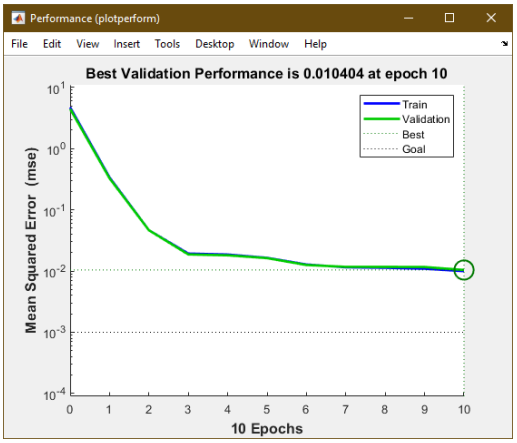
Treniravimo ir validavimo tikslumo grafikai:



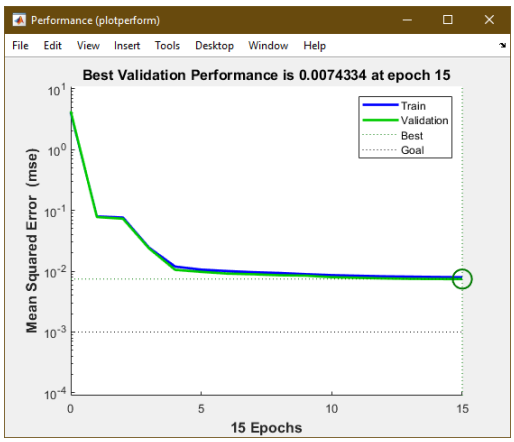
Pav. 41. 5 paslėptieji neuronai.



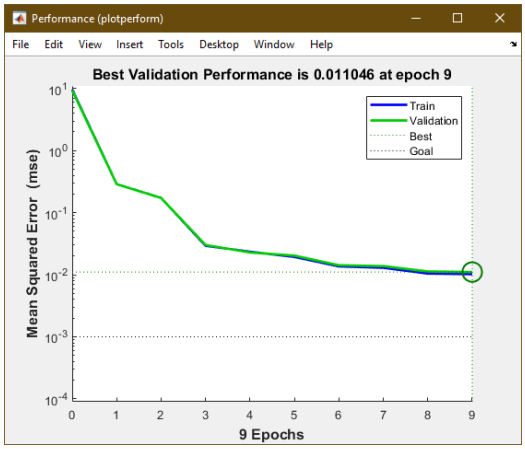
Pav. 42. 10 paslėptųjų neuronų.



Pav. 43. 15 paslėptųjų neuronų.



Pav. 44. 30 paslėptųjų neuronų.



Pav. 45. 50 paslėptųjų neuronų.

41 – 45 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas

nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	6487	6487	6487	6487	6487
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	5.36	5.11	5.09	4.99	4.87
Absoliutinių paklaidų vidurkis	**0.499**	**0.444**	**0.438**	**0.431**	**0.411**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.521	0.478	0.458	0.452	0.429
Paklaidų standartinis nuokrypis	0.511	0.481	0.465	0.451	0.437

Lent. 12. 8 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	1787	1787	1787	1787	1787
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	4.781	4.9	4.99	4.8	4.88
Absoliutinių paklaidų vidurkis	**0.529**	**0.512**	**0.484**	**0.472**	**0.464**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.649	0.599	0.562	0.551	0.532
Paklaidų standartinis nuokrypis	0.597	0.58	0.561	0.559	0.54

Lent. 13. 8 eksperimento testavimo rezultatai.

Remiantis 12 ir 13 lentelėmis galime teigti, jog neuroninis tinklas apsimokė gana gerai ir naudojant 50 paslėptųjų neuronų pavyko pasiekti 0.464 °C absoliutinių paklaidų vidurkį. Toliau bandykime optimizuoti treniravimosi procesą ir taip pasiekti dar geresnius rezultatus.

3.3.3. Neuroninio tinklo treniravimo optimizavimas

Kadangi mūsų neuroninio tinklo algoritmo tikslas yra surasti globalų minimumą, kartais gali atsitikti taip, kad jis gali užstrigti lokaliame minimume. Kad to neatsitiktų pasitelksimę kelių paleidimų algoritmą matlab aplinkoje. Kelių paleidimų algoritmas ypač naudingas sprendžiant optimizavimo problemas, kurios gali turėti kelis lokalius minimumus arba maksimumus, o tikslas yra rasti globalų minimumą arba maksimumą. Matlab kelių paleidimų algoritmas veikia pakartotinai inicijuodamas optimizavimo procesą iš skirtingų pradžios taškų ir atlikdamas lokalų optimizavimą iš kiekvieno iš šių pradinių taškų. Ištyrus skirtingus paieškos erdvės regionus, padidėja tikimybė rasti optimaliausią reikšmę.

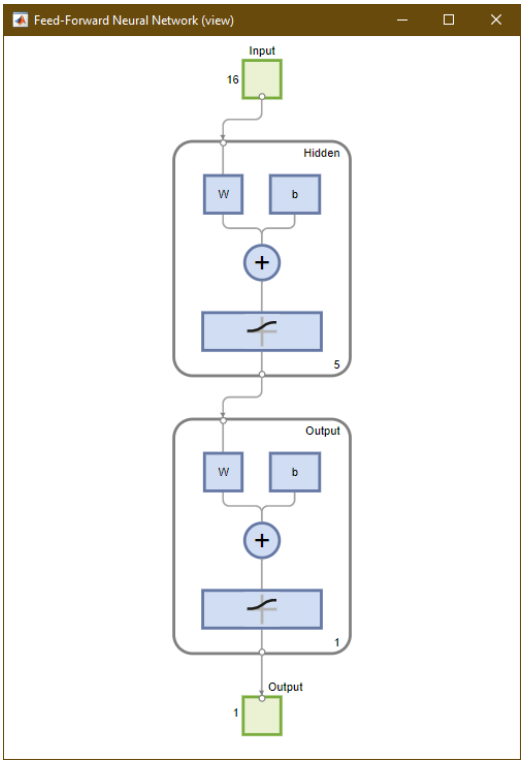
3.3.3.1. Devintas eksperimentas

Pabandykime optimizuoti neuroninio tinklo apsimokymo procesą. Prognozuokime kokia oro temperatūra, bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kovo 7 – 21 dienų, kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo išvestyje prognozuokime, kokia bus oro temperatūra. Treniravimo optimizavimui pasitelkime kelių kartų treniravimo metodą (angl. multistart). Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.
5. Mūsų neuroninis tinklas turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų kovo 7 – 21 dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūra, atitinkamomis dienomis 06:00 val.
6. Panaudokime matlab funkciją *Multistart* ir bandykime 10 kartų apmokyti tinklą ir rasti geriausiai apsimokiusį modelį. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį (imame tik prognozuojamą oro temperatūrą).

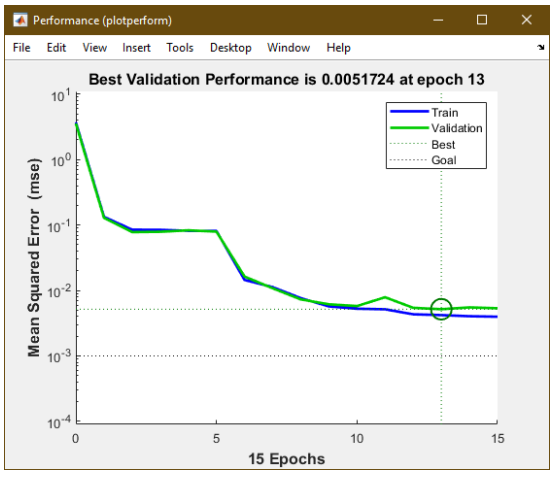
Gauti rezultatai:



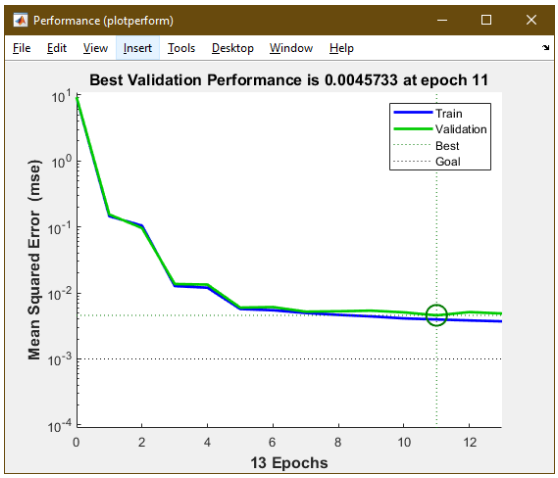
Pav. 46. Neuroninio tinklo diagrama.

46 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 1 išvesties parametrą. Jo struktūra išliko ta pati, kaip ir ankstesniuose eksperimentuose, kadangi mes jos nekeitėme, o tiesiog apmokėme tą patį neuroninį tinklą kelis kartus.

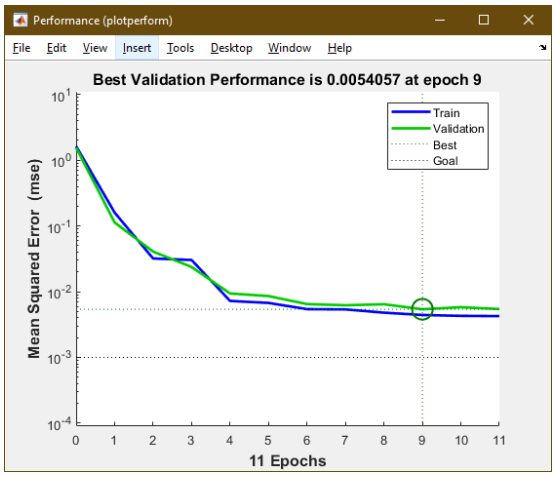
Treniravimo ir validavimo tikslumo grafikai:



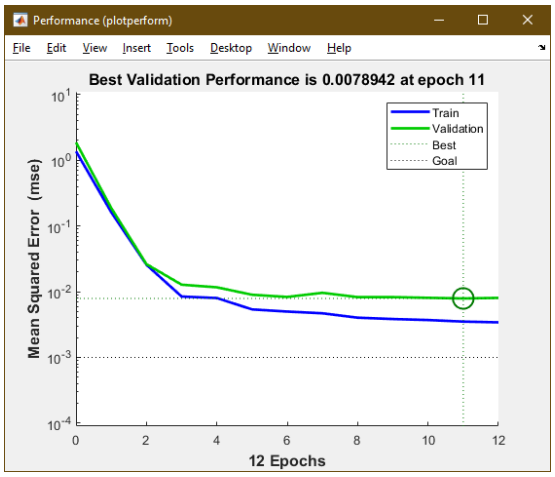
Pav. 47. 5 paslėptieji neuronai.



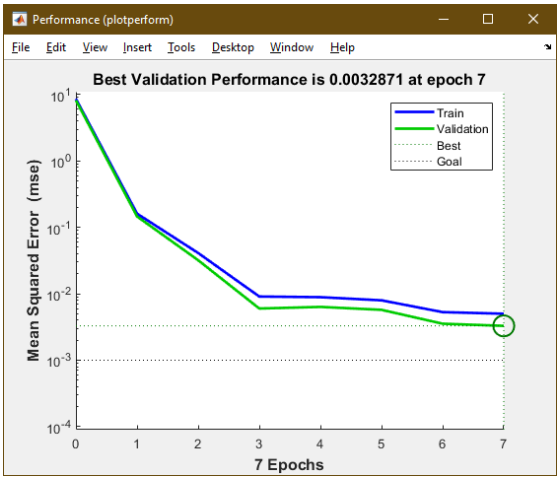
Pav. 48. 10 paslėptųjų neuronų.



Pav. 49. 15 paslėptųjų neuronų.



Pav. 50. 30 paslėptųjų neuronų.



Pav. 51. 50 paslėptųjų neuronų.

47 – 51 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	270	270	270	270	270
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	2.62	2.51	3.28	2.45	2.69
Absoliutinių paklaidų vidurkis	**0.478**	**0.454**	**0.475**	**0.47**	**0.484**

Vidutinė kvadratinė paklaida (Mean Squared Error)	0.415	0.386	0.436	0.414	0.44
Paklaidų standartinis nuokrypis	0.432	0.423	0.459	0.44	0.454

Lent. 14. 9 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	75	75	75	75	75
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	3.53	3	3.29	2.6	2.9
Absoliutinių paklaidų vidurkis	**0.547**	**0.479**	**0.477**	**0.455**	**0.46**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.67	0.583	0.6	0.534	0.53
Paklaidų standartinis nuokrypis	0.585	0.58	0.573	0.567	0.557

Lent. 15. 9 eksperimento testavimo rezultatai.

Remiantis 14 ir 15 lentelėmis galime teigti, jog su ganėtinai mažai apmokymo duomenų pavyko geriau optimizuoti treniravimosi procesą ir pasiekti ganėtinai mažą paklaidą. Toliau bandykime imti daugiau apsimokymo ir testavimo duomenų.

3.3.3.2. Dešimtas eksperimentas

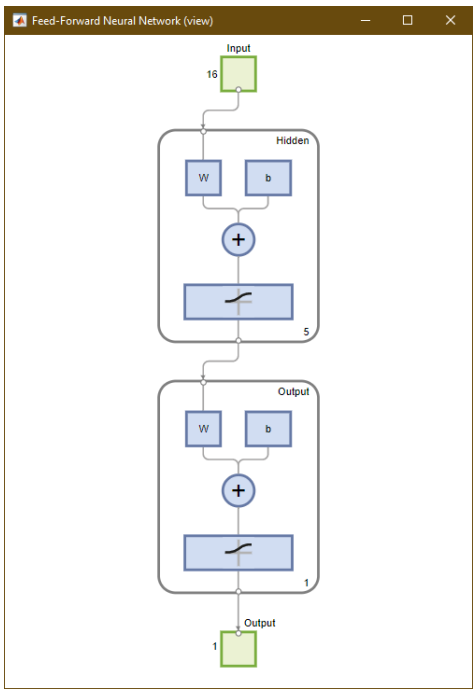
Pabandykime optimizuoti neuroninio tinklo apsimokymo procesą ir naudokime daugiau įvesties duomenų. Prognozuokime kokia oro temperatūra, bus po 3 val. pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo išvestyje prognozuokime, kokia bus oro temperatūra. Imkime ne kovo 7 – 21 dienų, o visų metų duomenis. Treniravimo optimizavimui pasitelkime kelių kartų treniravimo metodą (angl. multistart). Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime kiekvienos dienos, 00:00 val., 01:00 val., 02:00 val., 03:00 val. ir 06:00 val. oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.

4. Matlab aplinkoje sukurkime *feedforwardnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų, 30 neuronų ir 50 neuronų.
5. Mūsų neuroninis tinklas turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų, kiekvienos dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūrą atitinkamomis dienomis 06:00 val.
6. Panaudokime matlab funkciją *Multistart* ir bandykime 10 kartų apmokyti tinklą ir rasti geriausiai apsimokiusį modelį. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
7. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
8. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
9. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį.

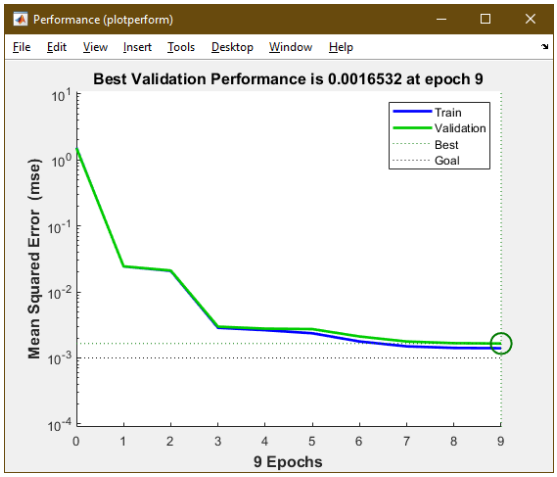
Gauti rezultatai:



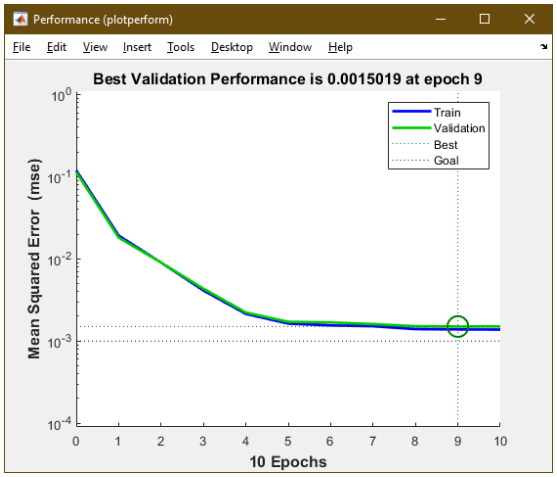
Pav. 52. Neuroninio tinklo diagrama.

52 paveikslėlyje matome, kad matlab aplinkoje sukūrėme tiesinio skleidimo neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 1 išvesties parametrą. Jo struktūra išliko ta pati, kaip ir ankstesniuose eksperimentuose, kadangi mes jos nekeitėme, o tiesiog apmokėme tą patį neuroninį tinklą kelis kartus.

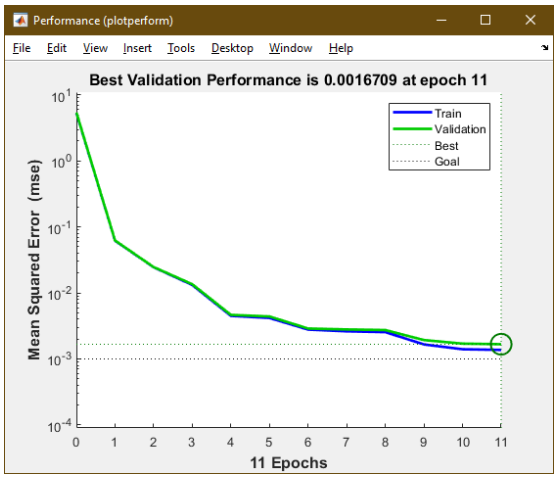
Treniravimo ir validavimo tikslumo grafikai:



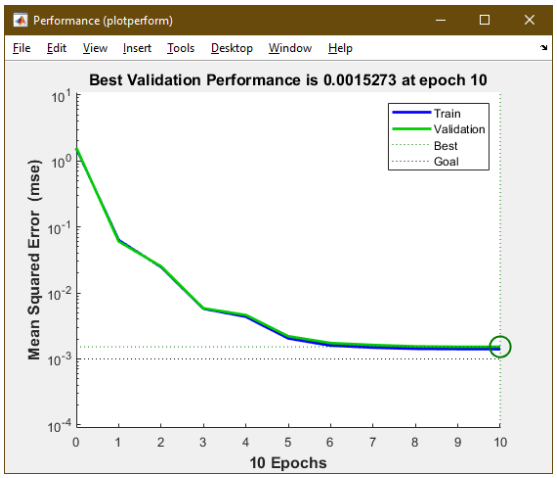
Pav. 53. 5 paslėptieji neuronai.



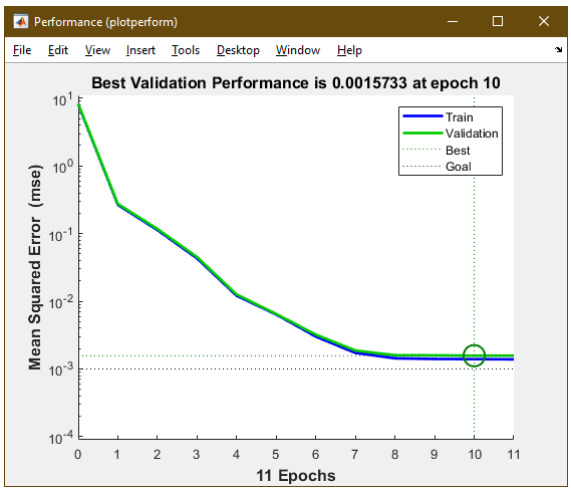
Pav. 54. 10 paslėptųjų neuronų.



Pav. 55. 15 paslėptųjų neuronų.



Pav. 56. 30 paslėptųjų neuronų.



Pav. 57. 50 paslėptųjų neuronų.

53 – 57 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15, 30 ir 50 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas.

Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	6407	6407	6407	6407	6407
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	5.2	5.07	5.22	5.21	5.2
Absoliutinių paklaidų vidurkis	**0.498**	**0.495**	**0.498**	**0.471**	**0.465**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.521	0.515	0.518	0.499	0.495
Paklaidų standartinis nuokrypis	0.523	0.52	0.519	0.502	0.489

Lent. 16. 10 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	1765	1765	1765	1765	1765
Min. absoliuti paklaida	0	0	0	0	0
Max. absoliuti paklaida	5.17	5.25	5.2	5	5.1
Absoliutinių paklaidų vidurkis	**0.512**	**0.485**	**0.481**	**0.462**	**0.451**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.649	0.63	0.611	0.594	0.601
Paklaidų standartinis nuokrypis	0.596	0.575	0.562	0.551	0.5

Lent. 17. 10 eksperimento testavimo rezultatai.

Remiantis 16 ir 17 lentelėmis galime teigti, jog padidinus duomenų kiekį ir vėl optimizavus neuroninio tinklo apsimokymo procesą pavyko pagerinti tikslumą. Toliau bandykime keisti neuroninio tinklo architektūrą.

3.3.4. Rekurentinis neuroninis tinklas.

Pabandykime pakeisti esamą tiesinio skleidimo neuroninį tinklą į rekurentinį. Rekurentinio tinklo įvesties rezultatas bus naudojamas kitoje įvestyje. Tokiu būdu neuroninis tinklas turės trumpalaikę atmintį. Toks tinklas puikiai tiks apdoroti trumpas duomenų sekas [YLW18]. Prieš tai buvusiuose eksperimentuose sukurtą tiesinio skleidimo neuroninį tinklą paversime į rekurentinį pridėję laiko delsimo įvestis (angl. time-delayed inputs). Tai yra toks mechanizmas, kuris leidžia įtraukti laikiną informaciją iš prieš tai buvusių laiko žingsnių į dabartinę prognozę. Kitaip tariant mūsų neuroninis tinklas dabar turės trumpalaikę atmintį. Mūsų naujas tinklo išvesties sluoksnis turės rekurentinį sujungimą atgal su paslėptuoju sluoksniu. Tokia neuroninio tinklo architektūra leis traktuosi istorinius duomenis, kaip duomenų seką. Kiekvienas laiko delsimas reiškia konkretų laiko žingsnį praeityje, leidžiantį tinklui užfiksuoti laikinas duomenų priklausomybes. Suteikdami savo tinklui tokias praeities žingsnių įvestis, mes ištirsime kiek efektyvus yra toks prognozavimas, kai yra svarbi praeities duomenų sekos tvarka. Kitaip tariant, kadangi orų temperatūra dažnai priklauso ir koreliuoja su prieš tai buvusių dienų temperatūra, toks tinklas turėtų tikti mūsų sprendžiamam uždaviniui.

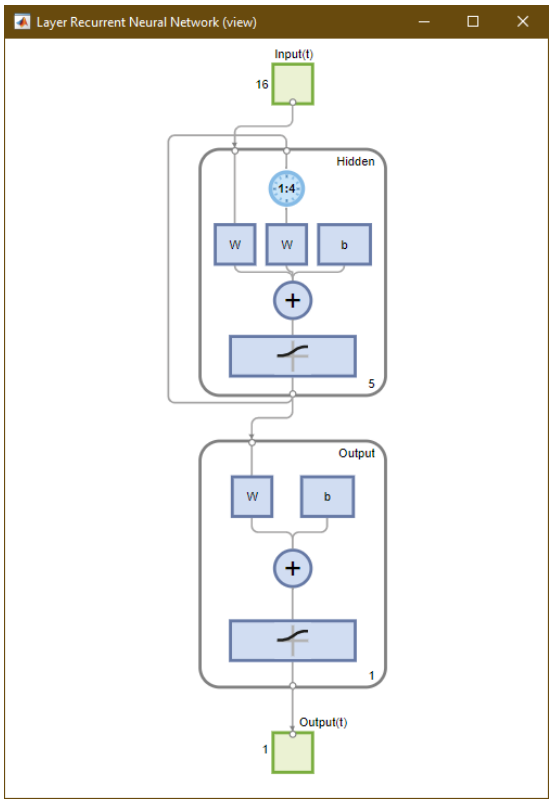
3.3.4.1. Vienuoliktas eksperimentas

Prognozuokime kokia oro temperatūra, bus po 3 val. pasitelkiant rekurentinį dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kovo 7 – 21 dienų, kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo išvestyje prognozuokime, kokia bus oro temperatūra. Naudosime matlab programavimo kalbą.

Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų kovo 7 – 21 dienų oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *layrecnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų ir 30 neuronų neuronų.
5. Mūsų neuroninis tinklas turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų kovo 7 – 21 dienos temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūra, atitinkamomis dienomis 06:00 val.
6. *Layrecnet* neuroninis tinklas kiekviename rekurentiniame žingsnyje, kaip įvestį ims duomenų seką, t.y. dabartinę įvestį ir dar prieš tai buvusias 3 įvestis.
7. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
8. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.
9. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
10. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį (imame tik prognozuojamą oro temperatūrą).

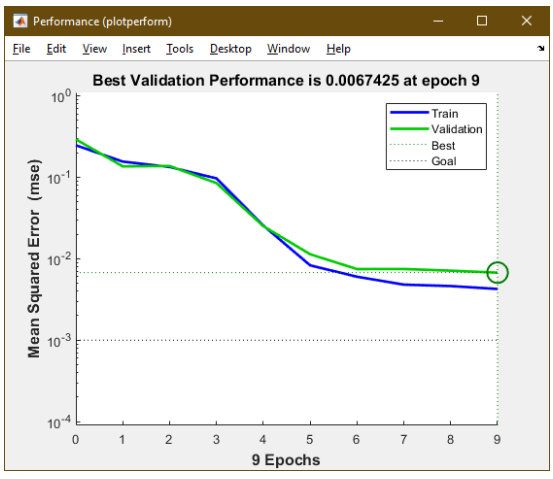
Gauti rezultatai:



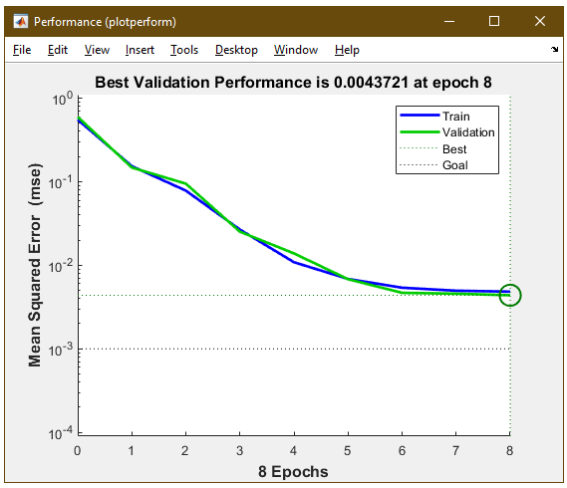
Pav. 58. Neuroninio tinklo diagrama.

58 paveikslėlyje matome, kad matlab aplinkoje sukūrėme rekurentinį neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 1 išvesties parametrą. Šis tinklas kiekviename rekurentiniame žingsnyje, kaip įvestį ima duomenų seką, t.y. dabartinę įvestį ir dar prieš tai buvusias 3 įvestis.

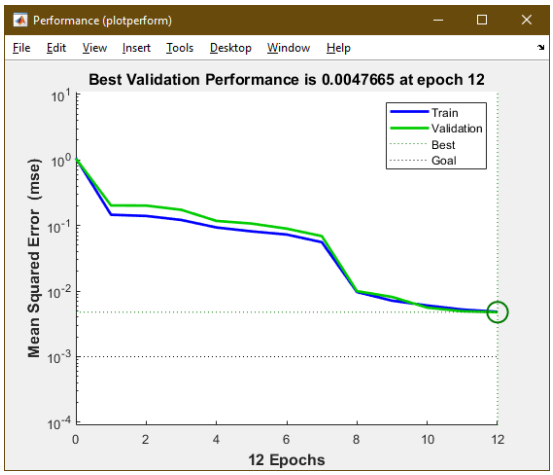
Treniravimo ir validavimo tikslumo grafikai:



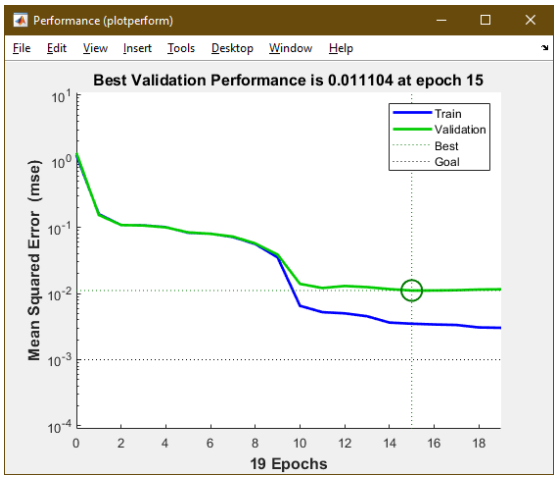
Pav. 59. 5 paslėptieji neuronai.



Pav. 60. 10 paslėptųjų neuronų.



Pav. 61. 15 paslėptųjų neuronų.



Pav. 62. 30 paslėptųjų neuronų.

59 – 62 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15 ir 30 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	270	270	270	270	-
Min. absoliuti paklaida	0	0	0	0	-
Max. absoliuti paklaida	2.86	2.98	3.07	3.49	-
Absoliutinių paklaidų vidurkis	**0.496**	**0.48**	**0.466**	**0.456**	-
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.448	0.437	0.435	0.423	-
Paklaidų standartinis nuokrypis	0.45	0.445	0.448	0.437	-

Lent. 18. 11 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**

Prognozių skč.	75	75	75	75	-
Min. absoliuti paklaida	0	0	0	0	-
Max. absoliuti paklaida	3.28	3.12	3.14	2.96	-
Absoliutinių paklaidų vidurkis	**0.489**	**0.503**	**0.437**	**0.44**	**-**
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.555	0.565	0.527	0.546	-
Paklaidų standartinis nuokrypis	0.566	0.552	0.546	0.53	-

Lent. 19. 11 eksperimento testavimo rezultatai.

Remiantis 18 ir 19 lentelėmis galime teigti, jog su ganėtinai mažai apmokymo duomenų pavyko geriau apmokyti neuroninį tinklą būtent naudojant pakeistą tiesinio skleidimo tinklą į rekurentinį. Toliau naudokime didesnį duomenų kiekį ir taip bandykime dar labiau pagerinti rekurentinio neuroninio tinklo tikslumą.

3.3.4.2. Dvyliktas eksperimentas

Prognozuokime kokia oro temperatūra, bus po 3 val. pasitelkiant rekurentinį dirbtinį neuroninį tinklą. Kaip įvesties duomenis naudokime kiekvienos valandos oro temperatūrą, oro drėgnumą, slėgį, vėjo greitį. Neuroninio tinklo išvestyje prognozuokime, kokia bus oro temperatūra. Imkime ne kovo 7 – 21 dienų, o visų metų duomenis. Naudosime matlab programavimo kalbą.

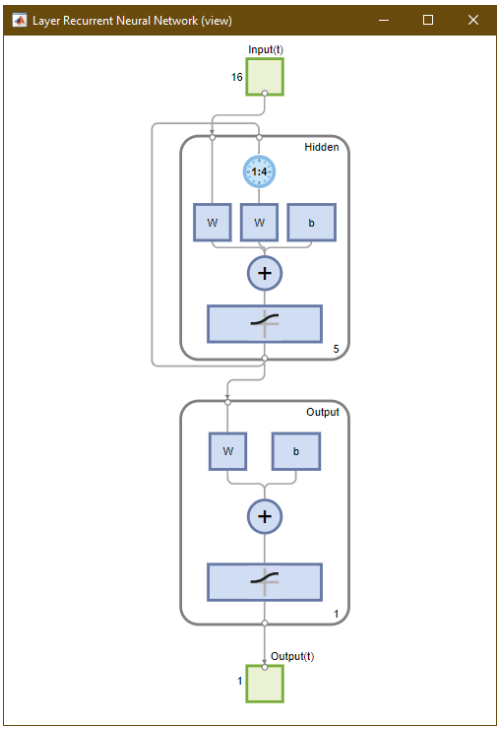
Darbo eiga:

1. Iš visų turimų duomenų imkime tik kiekvienų metų oro temperatūras, oro drėgnumą, slegį ir vėjo greitį.
2. Visus duomenis suskirstysime į dvi dalis. 80% duomenų bus neuroninio tinklo treniravimui, o 20% duomenų skirkime testavimui.
3. Normalizuokime visus duomenis nuo -1 iki 1 pasinaudodami savo sukurta matlab funkcija.
4. Matlab aplinkoje sukurkime *layrecnet* neuroninį tinklą. Naudokime vieną paslėptąjį sluoksnį su skirtingu neuronų skaičiumi kiekviename bandyme. Bandysime su 5 neuronais, 10 neuronų, 15 neuronų ir 30 neuronų neuronų.
5. Mūsų neuroninis tinklas turės 16 įvesčių ir 1 išvestį. Įvestys bus kiekvienų metų oro temperatūra, oro drėgnumas, slegis ir vėjo greitis 00:00 val., 01:00 val., 02:00 val. ir 03:00 val., o išvestis - pronozuojama oro temperatūra, atitinkamomis dienomis 06:00 val.
6. *Layrecnet* neuroninis tinklas kiekviename rekurentiniame žingsnyje, kaip įvestį ims duomenų seką, t.y. dabartinę įvestį ir dar prieš tai buvusias 3 įvestis.
7. Apmokykime neuroninį tinklą su įvesties ir išvesties duomenimis. Naudosime matlab funkciją *train*.
8. Kai neuroninis tinklas jau apmokytas galime naudoti likusius 20% duomenų testavimui. Per neuroninį tinklą paleidžiame kiekvienos savaitės 4 valandų

temperatūros, oro drėgnumo, slegio ir vėjo greičio duomenis ir išvestyje gauname prognozę - kokia oro temperatūra bus po 3 valandų.

9. Palyginkime visas prognozes su faktine oro temperatūra. Apskaičiuokime absoliutinę paklaidą tarp kiekvienos prognozės ir realios temperatūros.
10. Apskaičiuojame paklaidų vidurkį, standartinį nuokrypį ir vidutinę kvadratinę šaknį (imame tik prognozuojamą oro temperatūrą).

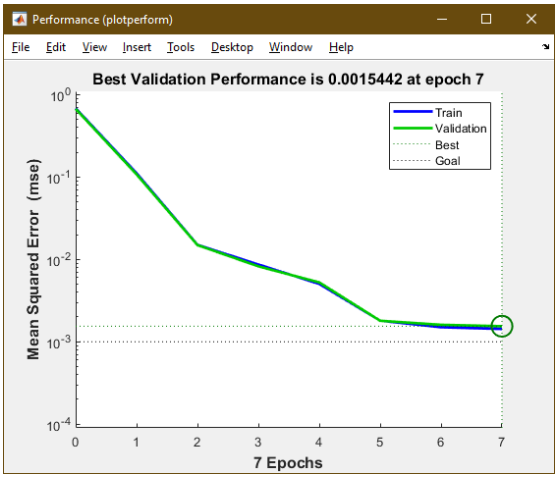
Gauti rezultatai:



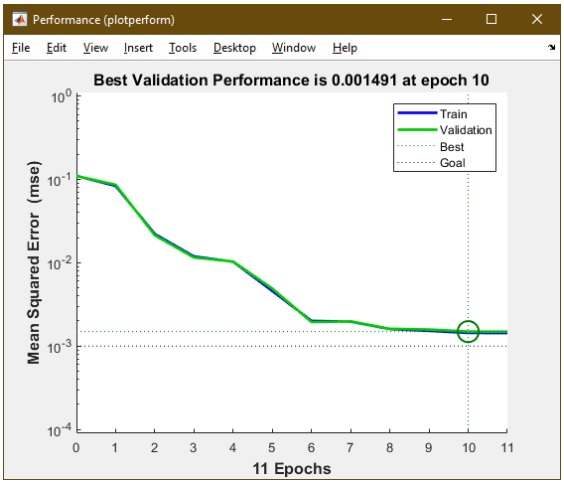
Pav. 63. Neuroninio tinklo diagrama.

63 paveikslėlyje matome, kad matlab aplinkoje sukūrėme rekurentinį neuroninį tinklą, kuris turi vieną paslėptąjį sluoksnį ir naudoja Sigmoido aktyvacijos funkciją. Šis neuroninis tinklas turi 16 įvesties parametrų ir 1 išvesties parametrą. Šis tinklas kiekviename rekurentiniame žingsnyje, kaip įvestį ima duomenų seką, t.y. dabartinę įvestį ir dar prieš tai buvusias 3 įvestis.

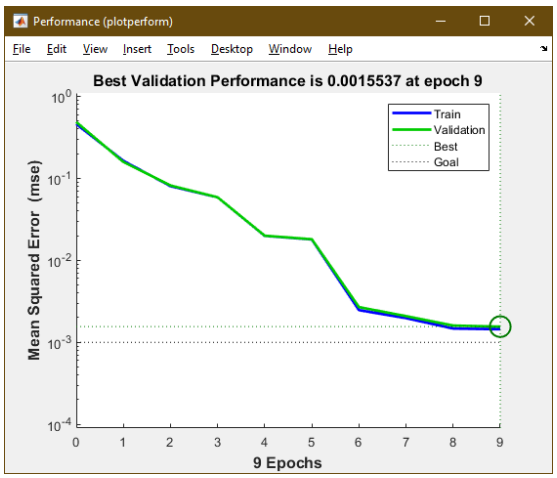
Treniravimo ir validavimo tikslumo grafikai:



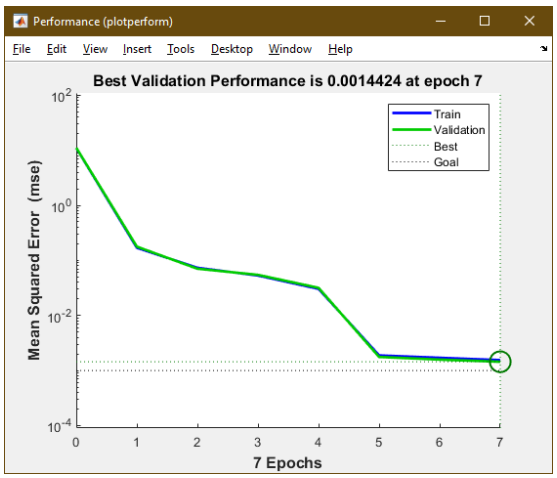
Pav. 64. 5 paslėptieji neuronai.



Pav. 65. 10 paslėptųjų neuronų.



Pav. 66. 15 paslėptųjų neuronų.



Pav. 67. 30 paslėptųjų neuronų.

64 – 67 paveikslėliuose matome mūsų sukurto neuroninio tinklo treniravimo ir validavimo tikslumo grafikus. Šiuo neuroniniu tinklu buvo atlikti penki bandymai, t.y. su 5, 10, 15 ir 30 paslėptųjų neuronų skaičiumi. Kiekvienam bandymui yra pateiktas šis tikslumo grafikas. Šiuos grafikus palygine su pavyzdiniu 15 paveikslėliu matome, jog validavimo tikslumas nepradėjo sparčiai greitėti, dėl mūsų nustatytų neuroninio tinklo treniravimo stabdymo sąlygų. Iš to galime teigti, jog neįvyko persimokymas ir mūsų tinklas apsimokė gerai.

Neuroninio tinklo treniravimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	6407	6407	6407	6407	-
Min. absoliuti paklaida	0	0	0	0	-
Max. absoliuti paklaida	5.26	5.28	5.38	5.46	-

Absoliutinių paklaidų vidurkis	**0.478**	**0.466**	**0.45**	**0.443**	-
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.507	0.499	0.488	0.478	-
Paklaidų standartinis nuokrypis	0.507	0.5	0.487	0.48	-

Lent. 20. 12 eksperimento treniravimo rezultatai.

Neuroninio tinklo testavimas

	5 neuronai	**10 neuronų**	**15 neuronų**	**30 neuronų**	**50 neuronų**
Prognozių skč.	1765	1765	1765	1765	-
Min. absoliuti paklaida	0	0	0	0	-
Max. absoliuti paklaida	5.45	5.67	5.28	4.91	-
Absoliutinių paklaidų vidurkis	**0.47**	**0.467**	**0.45**	**0.442**	-
Vidutinė kvadratinė paklaida (Mean Squared Error)	0.599	0.585	0.571	0.56	-
Paklaidų standartinis nuokrypis	0.54	0.531	0.511	0.493	-

Lent. 21. 12 eksperimento testavimo rezultatai.

Remiantis 20 ir 21 lentelėmis galime teigti, jog rekurentinis neuroninis tinklo tikslumas yra geresnis nei tiesinio skleidimo tinklo. Toliau gautus rezultatus palyginkime.

3.4. Bandymų palyginimas

Žemiau pateiktose 22 ir 23 lentelėse apibendrinome visus eksperimentus. Visi rezultatai yra suskirstyti į modelio apmokymo ir testavimo. Mum aktualiausi yra testavimo rezultatai, kadangi jie parodo kaip modelis veikia su nematytais duomenimis. Kaip prognozavimo tikslumo matą ėmėme absoliutinių paklaidų vidurkį. Matome kad prasčiausia prognozė buvo mūsų spėjimas, kaip ir buvo galima tikėtis. Tiesinės regresijos modelis pasirodė geriau nei mūsų spėjimas. Toliau buvo naudojamas tiesinio skleidimo neuroninis tinklas. Įvestyje imant tik oro temperatūrą neuroninis tinklas pasirodė šiek tiek geriau už tiesinę regresiją. Padidinus įvesties parametrų kiekį, neuronino tinklo tikslumas pagerėjo. Vėliau pabandžius ir išvestyje prognozuoti daugiau parametru neuroninio tinklo tikslumas beveik nepakito, o pats neuroninis tinklas tapo sudėtingesnis, todėl toliau išvestyje naudojome tik vieną parametrą. Toliau padidinome apsimokymo duomenų kiekį ir tas mum padėjo dar labiau pagerinti neuroninio

tinklo prognozavimo tikslumą. Pabandžius prognozuoti oro temperatūra ne tik 06:00 valandą, o kiekvieną paros valandą – neuroninis tinklas parodė geresnį prognozavimo tikslumą. Toliau optimizavome neuroninio tinklo apsimokymo procesą, t.y. apmokėme jį kelis kartus ir naudojome geriausią rezultatą. Tas padėjo dar labiau pagerinti prognozavimo tikslumą su naujais duomenimis. Na, ir galiausiai pavertėme savo tiesinio skleidimo tinklą į rekurentinį. Šis naujas neurninis tinklas pasirodė geriausiai iš visų eksperimentų ir su daug apmokymo duomenų pasiekė apie 0.46 °C absoliutinį vidurkį.

Treniravimas

Bandymo nr.	Trumpas aprašymas	Prognozių skč.	Įvest. param. skč.	Išv. param. skč.	Maž. Vidurkis	Didž. vidurkis	**Vid. Vidurkis**
1	Tarkime, kad po 3 val. oro temperatūra bus tokia pati kaip ir prieš 3 valandas.	-	1	1	-	-	**-**
2	Pasitelkiant tiesinės regresijos (angl. linear regression) modelį.	270	16	1	0.541	0.541	**0.541**
3	Pasitelkiant tiesinės regresijos (angl. linear regression) modelį.	6407	16	1	0.548	0.548	**0.548**
4	Pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Įvestis tik oro temperatūra.	270	4	1	0.5068	0.5303	**0.5122**
5	Pasitelkiant *feedforward* tinklą. Įvestis: oro temperatūra, oro drėgnumas, slėgis, vėjo greitis.	270	16	1	0.4871	0.525	**0.50902**
6	Pasitelkiant *feedforward* tinklą. Išvestis: oro temperatūra, slėgis, oro drėgmė ir vejo greitis.	270	16	4	0.4845	0.5507	**0.50632**
7	Pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Išvestis tik oro temperatūra.	6407	16	1	0.427	0.509	**0.4604**
8	Pasitelkiant *feedforward* tinklą. Prognozuojama oro temperatūra kiekvieną paros valandą.	6487	20	1	0.411	0.499	**0.4446**
9	Pasitelkiant *feedforward* tinklą. Treniravimo optimizavimui pasitelkime kelių kartų treniravimo metodą (angl. multistart).	270	16	1	0.454	0.484	**0.4722**
10	Pasitelkiant *feedforward* tinklą. Treniravimo optimizavimui pasitelkime	6407	16	1	0.465	0.498	**0.4854**

	kelių kartų treniravimo metodą (angl. multistart).						
11	Pasitelkiant rekurentinį neuroninį tinklą.	270	16	1	0.456	0.496	**0.4745**
12	Pasitelkiant rekurentinį neuroninį tinklą.	6407	16	1	0.443	0.478	**0.45925**

Lent. 22. Bandymų treniravimo palyginimas.

Testavimas

Bandymo nr.	Trumpas aprašymas	Prognozių skč.	Įvest. param. skč.	Išv. param. skč.	Maž. Vidurkis	Didž. vidurkis	**Vid. Vidurkis**
1	Tarkime, kad po 3 val. oro temperatūra bus tokia pati kaip ir prieš 3 valandas.	75	1	1	1.0161	1.0161	**1.0161**
2	Pasitelkiant tiesinės regresijos (angl. linear regression) modelį.	75	16	1	0.532	0.532	**0.532**
3	Pasitelkiant tiesinės regresijos (angl. linear regression) modelį.	1765	16	1	0.581	0.581	**0.581**
4	Pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Įvestis tik oro temperatūra.	75	4	1	0.4944	0.5358	**0.5107**
5	Pasitelkiant *feedforward* tinklą. Įvestis: oro temperatūra, oro drėgnumas, slėgis, vėjo greitis.	75	16	1	0.4764	0.5139	**0.4955**
6	Pasitelkiant *feedforward* tinklą. Išvestis: oro temperatūra, slėgis, oro drėgmė ir vejo greitis.	75	16	4	0.5002	0.5511	**0.5173**
7	Pasitelkiant *feedforward* dirbtinį neuroninį tinklą. Išvestis tik oro temperatūra.	1765	16	1	0.466	0.507	**0.4882**
8	Pasitelkiant *feedforward* tinklą. Prognozuojama oro temperatūra kiekvieną paros valandą.	1787	20	1	0.464	0.529	**0.4922**
9	Pasitelkiant *feedforward* tinklą. Treniravimo optimizavimui pasitelkime kelių kartų treniravimo metodą (angl. multistart).	75	16	1	0.455	0.547	**0.4836**
10	Pasitelkiant *feedforward* tinklą. Treniravimo optimizavimui pasitelkime	1765	16	1	0.451	0.512	**0.4782**

	kelių kartų treniravimo metodą (angl. multistart).						
11	Pasitelkiant rekurentinį neuroninį tinklą.	75	16	1	0.437	0.503	**0.4673**
12	Pasitelkiant rekurentinį neuroninį tinklą.	1765	16	1	0.442	0.47	**0.4572**

Lent. 23. Bandymų testavimo palyginimas.

Išvados

Šiame darbe buvo siekiama susipažinti su neuroniniais tinklais, jų tipais ir pasirinkus vieną iš tipų, atlikti prognozavimo bandymus. Pasirinktas tikslas buvo įvertinti kiek efektyvus yra tiesinio skleidimo neuroninis tinklas oro temperatūros prognozavimo uždaviniui. Šiam tikslui įgyvendinti buvo atliktos šios užduotys:

- Susipažinta su neuroniniais tinklais ir jų savybėmis;
- Susipažinta su skirtingomis aktyvacijos funkcijomis;
- Plačiau išnagrinėtas tiesioginio skleidimo neuroninis tinklas;
- Sukurtas neuroninį tinklas matlab aplinkoje;
- Atlikti eksperimentai su skirtingu paslėptų neuronų skaičiumi neuroniniame tinkle;
- Atlikti eksperimentai su įvairiais įvesties ir išvesties parametrais;
- Gauti ir palyginti rezultatai.

Iš pradžių buvo atlikti eksperimentai nenaudojant neuroninio tinklo. Buvo atlikta spėjimo prognozė ir palyginta su faktine informacija. Tada buvo atlikta prognozė naudojant tiesinę regresiją. Po šių bandymu, prognozavimui buvo pasitelkti neuroniniai tinklai. Eksperimentų metu buvo sukurtas vieno paslėptojo sluoksnio tiesioginio skleidimo neuroninis tinklas. Paslėptojo sluoksnio neuronų skaičius varijavo nuo 5 iki 50. Tada neuroninis tinklas buvo apmokytas įvairiais duomenimis. Kaip įvestis iš pradžių buvo imama tik tam tikro laikotarpio oro temperatūra, o vėliau įvesta ir daugiau papildomų parametrų, kaip atmosferos slėgis, drėgmė ir vėjo greitis. Taip pat, tolimesniuose bandymuose buvo didinamas treniravimo ir testavimo duomenų kiekis. Tada, iš pradžių pasirinktas tiesinio skleidimo neuroninis tinklas buvo paverstas į rekurentinį neuroninį tinklą ir atlikti keli eksperimentai keičiant apmokymo duomenų kiekį. Galiausiai palyginome gautus rezultatus. Kaip prognozavimo tikslumo matą ėmėme absoliutinių paklaidų vidurkį.

Taigi, iš atliktų tyrimų akivaizdžiai matosi, kad net ir pats paprasčiausias, t.y. tiesioginio skleidimo neuroninis tinklas yra efektyvus modelis prognozuoti oro temperetūrą, remiantis istoriniais orų duomenimis. Taip galime teigti, nes jis net ir kaip įvestį naudojant tik oro temperatūrą buvo tikslesnis, nei logika paremtas spėjimas. Kalbant apie tiesinę regresiją tai neuroninis tinklas su Sigmoido aktyvacijos funkcija pasirodė geriau, kadangi oras yra nelinijinis ir dinamiškas procesas, kuris kasdien kinta, todėl tiesinė lygtis šiam uždaviniui ne visai tinka. Kadangi klimato duomenų rinkinys yra netiesinis, dirbtinis neuroninis tinklas orų prognozavimo uždavinyje parodė geresnį tikslumą. Na, o rekurentinis neuroninis tinklas pasirodė geriau už paprastą tiesinio skleidimo neuroninį tinklą, kadangi jis turi trumpalaikę atmintį ir gali traktuoti istorinius orų duomenis, kaip duomenų seką, o akivaizdu, jog ateities oro temperatūra yra priklausoma ir nuo praėjusių dienų sekos įvairių parametrų.

Taip pat iš atliktų eksperimentų galima pastebėti, kad kalbant apie šį orų prognozavimo uždavinį – neuronų kiekis paslėptąjame sluoksnyje turi įtakos. Vidutiniškai kuo daugiau neuronų buvo naudojama neuroninio tinklo paslėptąjame sluoksnyje, tuo geriau apsimokydavo modelis ir testavimo metu buvo gaunamas geresnis prognozavimo tikslumas. Efektyviausias paslėptųjų neuronų kiekis šiam uždaviniui buvo – penkiasdešimt. Taip galėjo įvykti todėl, kad daugumoje eksperimentų naudojome ganėtinai ne mažą įvesties parametrų skaičių ir daugiau paslėptųjų neuronų sugebėjo geriau išmokti jų koreleaciją ir priklausomybę su išvestimi.

Didelis įvesties parametrų skaičius nebūtinai gali pagerinti neuroninio tinklo prognozavimo tikslumą, tačiau mūsų atveju naudojant apie 16 įvesties parametrų (4 valandų oro temperatūrą, slėgį, drėgmę ir vėjo greitį) pavyko pasiekti geresnius rezultatus, nei naudojant 4 parametrus (4 valandų oro temperatūrą). Taip galėjo būti todėl, kad daugiau įvesties parametrų suteikė neuroniniui tinklui daugiau informacijos apie modeliuojamą problemą ir atitinkamai jis geriau apsimokė, bei testavimo metu parodė mažesnę paklaidą.

Dar buvo galima pastebėti, kad naudojant daugiau istorinių duomenų – neuroninio tinklo tikslumas buvo geresnis. Tiesa apmokymo paklaidos buvo šiek tiek prastesnės, nei naudojant mažiau duomenų, tačiau kalbant apie naujus, t.y. nematytus duomenis, neuroninis tinklas pasirodė žymiai geriau. Geresnį tikslumą naudojant daugiau istorinių duomenų galėjo lemti tas, kad mūsų neuroninis tinklas buvo treniruojamas su įvairesniais duomenimis ir taip geriau apsimokė. Tinklas taip pat galėjo geriau generalizuoti duomenis ir atpažinti įvairesnius scenarijus, bei variantus, o tinklas, kuris geriau generalizuoja duomenis gali geriau veikti testavimo metu, t.y. naudojant nematytus duomenis.

Taigi, tiesinio skleidimo neuroninis tinklas yra ganėtinai efektyvus modelis prognozuoti oro temperatūrą, remeintis istoriniais duomenimis. Tiesa, norint gauti tikslesnę prognozę reikėtų naudoti sudėtingesnį, t.y. rekurentinį neuroninį tinklą su trumpalaikę atmintimi.

Literatūra

[Jai18] Urvashi Jaitley. Why Data Normalization is necessary for Machine Learning models. https://towardsdatascience.com/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029, 78.14KB, 2018.

[Bro19] Jason Brownlee. How to Use Data Scaling Improve Deep Learning Model Stability and Performance. https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/, 680.59KB, 2019.

[Des18] Julien Despois. Overfitting & How to Fix it. https://hackernoon.com/memorizing-is-not-learning-6-tricks-to-prevent-overfitting-in-machine-learning-820b091dc42, 185.93KB, 2018.

[Bro18] Jason Brownlee. How to Avoid Overfitting in Deep Learning Neural Networks. https://machinelearningmastery.com/introduction-to-regularization-to-reduce-overfitting-and-improve-generalization-error/, 250.3KB, 2018.

[AJO+18] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. Heliyon. 938 pp. 2018.

[Mah17] Jahnavi Mahanta. Introduction to Neural Networks, Advantages and Applications. https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207, 167.06KB, 2017.

[Saz06] Murat H. Sazli. A Brief Review of Feed-forward Neural Networks. Ankara University. 11-17 pp. 2006.

[Sae22] Mehreen Saeed. An Introduction to Recurrent Neural Networks and the Math That Powers Them. https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them, 207.86KB, 2022.

[Yam18] Rikiya Yamashita. Convolutional neural networks: an overview and application in radiology. Insights into Imaging 9, 611-629 pp. 2018.

[Bis23] Avijeet Biswal. Recurrent Neural Network(RNN) Tutorial: Types, Examples, LSTM and More. https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn, 385.39KB, 2023.

[SSA20] Siddharth Sharma, Simone Sharma and Anidhya Athaiya. Activation Functions in Neural Networks. International Journal of Engineering Applied Sciences and Technology. 4, 310-316 pp. 2020.

[Cyb89] G. Cybenko. Approximation by Superpositions of a Sigmoidal Function. Math. Control Signals Systems. 2, 303-314 pp. 1989.

[FXS+21] Wei Fang, Qiongying Xue, Liang Shen and Victor S. Sheng. Survey on the Application of Deep Learning in Extreme Weather Prediction. *Atmosphere. 12(6), 661 pp. 2021.*

[YLW18] Xiaofeng Yuan, Lin Li and Yalin Wang, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, 1551-3203 pp. 2018.

[Bro20] Jason Brownlee. Linear Regression for Machine Learning. https://machinelearningmastery.com/linear-regression-for-machine-learning/, 368.41KB, 2020.

[Phi23] Philippine Atmospheric, Geophysical and Astronomical Services Administration. How a Weather Forecast is Made. https://www.pagasa.dost.gov.ph/learning-tools/how-weather-forecast-made, 139.41KB, 2023.

[Deu23] Deutscher Wetterdienst. Radar Data Quality Control. https://www.dwd.de/EN/research/weatherforecasting/met_applications/radar_data_applications/radar_data_quality_control_node.html, 52.64KB, 2023.

[PK20] Zhaoxia Pu and Eugenia Kalnay. Numerical Weather Prediction Basics: Models, Numerical Methods, and Data Assimilation. Handbook of Hydrometeorological Ensemble Forecasting. Springer Berlin, Heidelberg, 1 – 31 pp., 2020.

[SKG+12] Gyanesh Shrivastava, Sanjeev Karmakar, Pulak Guhathakurta and Manoj Kumar Kowar. International Journal of Computer Applications. 51(18), 2012.

[NP12] Arti R. Naik and S.K.Pathan. Weather Classification and Forecasting using Back Propagation Feed-forward Neural Network. International Journal of Scientific and Research Publications. 2(12), 2012.

[KHA21] Shahab Wahhab Kareem, Zhala Jameel Hamad and Shavan Askar. An evaluation of CNN and ANN in prediction weather forecasting: A review. Sustainable Engineering and Innovation. 3(2), pp.148-159. 2021.