Vilniaus universitetas

Fizikos fakultetas

Taikomosios elektrodinamikos ir telekomunikacijų institutas

Andrius Lengvinas

INTERNETO SRAUTO ANOMALIJŲ TYRIMAS TAIKANT NEURONINIUS TINKLUS

Magistratūros studijų baigiamasis darbas

Telekomunikacijų fizikos ir elektronikos studijų programa

Studentas	Andrius Lengvinas
Leista ginti	2018-05-23
Darbo vadovas	doc. Feliksas Kuliešius
Instituto direktorius	prof. Jonas Matukas

Vilnius 2018

Turinys

Įvadas

Šiuo metu internetas prieinamas kiek daugiau nei 3,9 milijardo žmonių [1]. Interneto prieinamumas ir pasiekiamumas gerėja ir mažiau išsivysčiusiose šalyse. Internetas vis plačiau naudojamas ir kasdieniniame gyvenime. Viena pirmųjų sistemų, panaši į dabartinę buvo sukurta sujungiant kelių universitetų kompiuterius tarpusavyje. Tokios sistemos perduodamų duomenų kiekis, savaime suprantama, buvo minimalus. Šiandienos pasaulyje internetas naudojamas praktiškai visose srityse: laiškai, vaizdo konferencijos, filmai ir t. t. Taip pat internetas naudojamas ir virtualiose sistemose („cloud“). Vis daugiau žmonių saugo savo duomenis nutolusiuose serveriuose (virtualiose sistemose), o norėdami pasiekti, atnaujinti ar pakeisti šią informaciją naudojasi internetu. Reikėtų dar paminėti ir pastaraisiais metais populiarėjantį rūko („fog“) tinklą, kurį sudaro mūsų namų buities išmanieji įrenginiai, kuriuos galima valdyti internetu. Prie tinklo prijungiant vis didesnę įvairovę įrenginių, siekiant išvengti sutrikimų yra svarbu prognozuoti pokyčius sraute.

Internetas vartotojams suteikia „neribotas“ galimybes, tačiau internetinėje erdvėje gausu kenksmingų failų, apgaulingų svetainių ir kibernetinių atakų. Vienos iš populiariausių kibernetinių atakų yra atsisakymo aptarnauti – DoS (*Denial of service*) ir DDoS (*Distributed Denial of Service*) – atakos [2]. Šiomis atakomis siekiama sutrikdyti aukos veiklą apkraunant jos įrangą didžiuliu paketų srautu [3]. Tai nėra vienintelė DDoS atakų pritaikymo galimybė: DDoS ataka, kurios intensyvumas mažesnis negu puolamos įrangos srauto apdorojimo galimybės gali būti pritaikyta įsilaužimo maskavimui. Tokio tipo atakas galima aptikti analizuojat anomalijas interneto sraute.

Interneto srauto padidėjimus gali sukelti įvairūs veiksniai. Pavyzdžiui čempionatų ar kitos veiklos stebėjimas dideliu mastu, naudojantis internetu, sukelia anomaliją, nes padidėja tinklo apkrova. Kai žinoma tik dalis aplinkybių, atskirti natūralų srauto nuokrypį nuo kenksmingos veiklos sukelto nuokrypio yra sudėtinga. Vienas iš metodų yra palyginimas su srauto duomenų prognoze. Prognozuojamos vertės turėtų atkartoti legalių vartotojų inicijuoto srauto nuokrypius, o lyginant realų srautą su prognozuojamomis vertėmis galima išskirti internetinių atakų sukeltus nukrypimus.

Daugeliu atveju srauto prognozė atliekama taikant tiesinės ir netiesinės dinamikos metodus, plačiai taikomus laiko sekų analizei (ekonomikoje, orų kaitoje, energetikoje). Alternatyvus sprendimas paminėtiems metodams yra neuroniniai tinklai. Neuroniniai tinklai buvo pristatyti daugiau kaip prieš pusę šimto metų, tačiau tiesinės ir netiesinės dinamikos metodai yra pasirenkami dažniau. Tai lemia didelis skaičiavimo resursų poreikis, reikalingas naudojant neuroninių tinklų metodą. Netiesinės dinamikos metodus galima optimizuoti parenkant atitinkamas parametrų vertes priklausomai nuo

pritaikymo paskirties. Kompiuterinių skaičiavimų technologijoms pasistūmėjus į priekį neuroninių tinklų taikymo sritys išsiplėtė, viena iš jų – chaotinių laiko eilučių prognozė.

Šio darbo tikslas buvo ištirti neuroninio tinklo taikymo galimybes interneto srauto duomenų sekos prognozavimui. Siekiant šio tikslo darbo metu iškelti ir sprendžiami šie uždaviniai:

1. sudaryti neuroninį tinklą, tinkamą interneto srauto prognozei;
2. išanalizuoti gautus rezultatus ir įvertinti pritaikymo galimybes interneto srauto anomalijoms tirti;
3. patikrinti netiesinės dinamikos metodų skirtų parametrams nustatyti pritaikymo galimybes neuroninio tinklo struktūros parametrams parinkti;
4. palyginti neuroninių tinklų ir kitų plačiai taikomų alternatyvių metodų pritaikymo interneto srauto prognozei galimybes.

Atakų ir atakų aptikimo metodų apžvalga

DoS ir DDoS atakos

Kibernetiniai nusikaltimai kasdieniame gyvenime tokie dažni, kad jų negalima ignoruoti. Praktiškai kiekvieną savaitę galima perskaityti po kelis straipsnius apie elektroninių įsilaužimų atvejus, metinėse ir ketvirčių ataskaitose pranešama apie didelių įmonių arba organizacijų prarastus ar neteisėtai atskleistus duomenis, pasisavintas lėšas. Internetinių nusikaltėlių labai pamėgta atsisakymo aptarnauti ("Denial of Service") DoS ataka. DoS yra viena iš labiausiai paplitusių elektroninių nusikaltimų atakų. Perskaičius atakos pavadinimą galima pasakyti, kad tam tikras turinys bus nepasiekiamas vartotojams. Dauguma DoS tipo išpuolių yra nukreipti prieš interneto svetainę ar kitą internete pasiekiamą objektą.

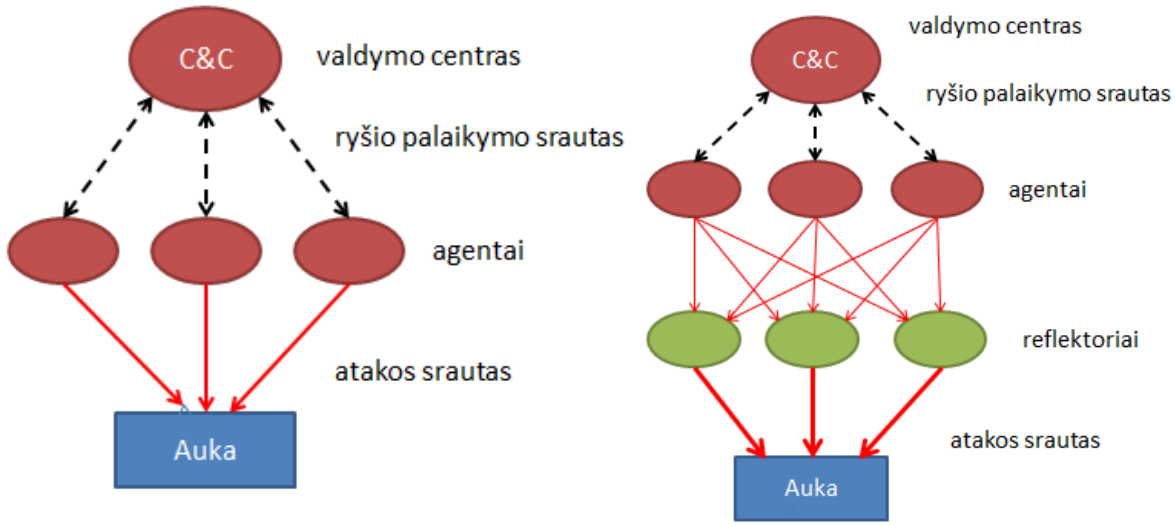
Nusikaltėliai DoS ataką gali sugeneruoti įvairiausiais būdais. Juos riboja tik jų pačių žinios ir vaizduotė. DoS atakos skiriasi nuo daugelio kitų kibernetinių išpuolių, kurie paprastai naudojami siekiant gauti ilgalaikę prieigą prie sistemų ir taip palaipsniui išgauti svarbią informaciją. DoS atakos nėra skirtos apeiti sistemos saugumo perimetrą. DoS atakų veikimas nukreiptas tiesiai į pačią sistemų fizinę ir programinę infrastruktūrą, išnaudoja prieigos prie programų spragas ir informacijos perdavimo protokolus [3]. Šių išpuolių tikslas - tinklalapius ir serverius padaryti neprieinamus teisėtiems naudotojams. Sėkminga DoS ataka iškart aptinkama, nes jis paveikia visus vartotojus, naudojančius serverį. Dėl jau išvardintų pagrindinių priežasčių DoS atakos yra labai populiarios tarp kibernetinių nusikaltėlių. Pasitaiko atvejų, kai DoS išpuoliai naudojami kaip maskuojamieji veiksmai siekiant paslėpti kitas neteisėtas veiklas, pvz., išeikvoti internetinės ugniasienės darbinius resursus, skirtus paketų turinio inspekcijai.

DoS atakos dažnai trunka kelias dienas, savaites ar net mėnesius, todėl jos yra labai veiksmingos prieš bet kokią internetinę organizaciją. Taip pat gausu ir trumpų, lėtų arba labai didelio srauto atakų. Kai kurių atakų intensyvumas kinta laike, tai priklauso nuo atakos srautą generuojančių įrenginių skaičiaus tinkle arba generuojamą srautą tenka papildomai padidinti, nes atakuojama sistema yra apsaugota srautą valančiais serveriais. Atakuojamos organizacijos negalėdamos panaudoti turimais resursais praranda pajamas ir vartotojų pasitikėjimą. Be to, internetiniai nusikaltėliai reikalauja didelių vienkartinių išpirkų už atakos nutraukimą, o sutikus su reikalavimais ir atstačius sistemos veikimą vis tiek neišvengiami nuostoliai dėl atakos metu sukeltų servisų veiklos sutrikimų.

Skirtumas tarp DoS ir DDoS (Distributed Dinial of Service) išpuolių yra labai mažas. DoS atakos atveju, užpuolikas naudoja tik vieną žiniatinklio sąsają ar vieną įrenginį atakos srautui, kuris skirtas serverių išteklių išnaudojimui ar programų pažeidžiamų spragų išnaudojimui, sugeneruoti. DDoS

išpuolius generuoja keli įrenginiai, kurie gali būti išsklaidyti visame pasaulyje. Įrenginių skaičius neribojamas, todėl šimtai tūkstančių įrenginių gali generuoti srautą vienam konkrečiam išpuoliui. Kadangi naudojama daug prietaisų ir generuojamą atakos srautą galima paskirstyti įrenginiams įvairiai, išpuolio šaltinį aptikti yra daug sudėtingiau. Be to, dėl didelio IP adresų skaičiaus, kuris susimaišo su legalių naudotojų srautu, tokios atakos įrenginius identifikuoti yra sunkiau. Skirtingai nei vieno šaltinio DoS atakos, DDoS tipo atakų išpuoliai nukreipti prieš interneto infrastruktūrą, kurią norima perpildyti dideliu beprasmių paketų kiekiu.

Svarbus skirtumas tarp "DDoS" ir "DoS" išpuolių yra jų įgyvendinimas. DoS atakos yra realizuojamos naudojant kodus, kuriuos parašė atakos inicijuotojas arba panaudojo viešai prieinamomis priemonės (pvz.,DoS „Low Orbit lin Canon“) [4], kurios visą atakos darbą automatizuoja ir belieka nurodyti puolamos sistemos adresą. DDoS atakoms reikalingas robotizuotas kompiuterių tinklas (*botnet*). Tai didelis kompiuterinių rinkinys, kuriame yra įrašyta kenkėjiška programinė įranga. Tokiam tinklui valdyti nusikaltėliai naudoja kontrolės centrą (Comand and Contol, C & C), kurio pagalba perduoda robotams instrukcijas apie tikslą, IP adresą, atakos pradžios laiką ir puolimo seką. Jei agentas pasiekia internetą atakoje nurodytu laiku, jis generuoja DDoS paketus pagal atakos algoritmą. Pateikiama tokio išpuolio schema (1 pav., kairėje). DRDoS (Distributed Reflection Denial of Service) ataka yra pavaizduota (1 pav., dešinėje): šiuo atveju atakos srauto sustiprinimui pasitelkiami reflektoriai. Agentai siunčia užklausas su aukos IP adresu IP paketo antraštėje nurodančioje paketo siuntėją į reflektorių, o reflektoriai, gavę taip pakeistas paketų užklausas mano, kad jos buvo išsiųstos iš aukos ir išsiunčia atsaką į kiekvieną tokią užklausą aukos IP adresu. Tai padidina atakos srautą. Reflektoriai gali būti įvairūs tinklo įrenginiai: kompiuteriai, serveriai ir kt.



1 pav. DDoS (kairėje) ir DRDoS (dešinėje) atakų pavyzdžiai.

Robotizuotų kompiuterių tinklas

Robotų kompiuterių tinklas – kenkėjiškomis programomis užkrėstas kompiuterinis tinklas, kurį valdo vienas arba organizuota internetinių nusikaltėlių grupė, žinoma kaip valdymo centras (*bot-header*) [5]. Šiam tinklui priklausančios mašinos vadinamos robotais (*bot*). Visas tinklas valdomas iš vieno centrinio kompiuterio, todėl nusikaltėliai gali koordinuoti kibernetines atakas. Užkrėstų kompiuterių, sudarančių robotų kompiuterių tinklą, skaičius neribojamas, dauguma šių tinklų sudaryta iš milijonų įrenginių. Tokio dydžio tinklai įgalina pažeidėją organizuoti ir įgyvendinti didelio masto išpuolius, kurie būtų neįmanomi įgyvendinti DoS tipo atakomis. Nusikaltėliai išlaiko nuotolinę prieigą prie užkrėstų kompiuterių, kuri yra reikalinga užduočių perkėlimui ir tam tikrais atvejais kenksmingo kodo atnaujinimui. Kai kurie nusikaltėliai užsiima visų ar dalies jų robotizuotų kompiuterių tinklo nuoma „juodojoje rinkoje“.

Robotizuotų kompiuterių tinklas apima ne tik kompiuterius, bet ir mobiliuosius telefonus, planšetinius kompiuterius, daiktų interneto objektus. Šiuo metu mūsų namuose vis daugiau protingų prietaisų, tokių kaip kavos aparatai, skalbimo mašinos, kurių gamintojai dažnai nepakankamai rimtai žiūri į įrenginių kibernetinę saugą. Dėl to internetiniai nusikaltėliai, aptikę saugumo spragas, įrašo savo kenkėjišką kodą ir taip dar labiau plečia robotų tinklą. Būtent, pasinaudojus viena iš tokių saugos spragų, IP kamerų sugeneruotas DDoS srautas viršijo 1 TB / s [6].

Didelis skaičius tinklą sudarančių įrenginių sukuria dar didesnį galimų komunikacijos jungčių tinklą. Pasinaudojus komunikacijos jungčių tinklu, komunikaciją tarp robotų ir valdymo centro tampa praktiškai nesusekama. Botus galima atnaujinti iš valdymo centro ir taip pakeisti jų funkcionalumą. Pakeitimais tinklas pritaikomas prie puolamos sistemos saugumo spragų.

Populiariausių srauto analizės metodų aptarimas

Norint apsisaugoti nuo dažnai nusikaltėlių naudojamų DoS ir DDoS atakų svarbiausias žingsnis yra greitas atakų aptikimas. Keičiantis interneto prieinamumui ir tobulėjant atakų algoritmams, keitėsi ir atakų aptikimui taikomos metodikos.

Pirmosios DDoS atakos buvo labai paprastos palyginus su šių dienų internete aptinkamomis kibernetinėmis atakomis. Tokias atakas būdavo nesunku aptikti. Paprasčiausias aptikimo procesas paremtas naujų šaltinio IP adresų atsiradimu sraute, kurių skaičius apibrėžtame laiko intervale padidėja kelis kartus, priklausomai nuo atakos intensyvumo [7, 8]. Kiek sudėtingesnis metodas veikė skaičiuojant srauto entropiją, kuri parodo sistemos tvarkingumą [9]. Atakos metu pasikeitus srauto balansui, IP adresams, portams ar paketų dydžiui (priklausomai nuo stebimo atributo) pasikeičia ir

srauto entropijos vertė. Abiem paminėtais atvejais reikia nustatyti slenkstines ribas, kurios indikuoja vykstančią ataką. Tačiau tai yra viena iš problemų – kintant naudotojų poreikiams dėl įvairių veiksnių legalūs vartotojai gali sugeneruoti srautą, panašų į DDoS ataką. Įvardintas atakų aptikimo metodikas galima priskirti atakos elgsenos grupei, reikia paminėti ir kitą grupę žinomą kaip šabloniniai metodai. Šabloninių metodų veikimas pagrįstas atakų profiliavimu [10, 11]. Atakos aptikimas atliekamas lyginant žinomų atakų kontrolinius kodus (*fingerprints*) [12] su stebimu srautu. Deja nusikaltėliai nuolatos atnaujina ir tobulina atakų algoritmus siekdami jas padaryti kuo labiau panašias į legalių vartotojų sugeneruotą srautą. Todėl praktiškai neįmano suklasifikuoti visų nuolatos keičiamų atakų, kad taikant šabloninius metodus būtų galima jas nesunkiai aptikti.

Laiko sekų prognozėje dažniausiai naudojami tiesiniai metodai. Vienas iš jų yra ARIMA (Autoregressive integrated moving average). Tiesinių metodų populiarumą lėmė jų paprastumas ir nesudėtingas pritaikymas daugumai problemų spręsti. Tačiau ARIMA modelis turi ir trūkumų: vienas iš jų, įvesties duomenys turi būti stacionarizuoti. Laiko eilutę, pasižyminčią stipriomis chaotinėmis savybėmis, stacionarizuoti galima ne visada. Kitas trūkumas – reikalingas didelis duomenų rinkinys prognozės modeliui sudaryti. Taip pat reikia parinkti tinkamo dydžio duomenų rinkinį, kad modelis netaptų „permokytas“. Tokiu atveju rezultatai atspindės apmokymo duomenis, o ne sekos kitimo tendencijas.

Taikant įprastas metodikas identifikuoti DDoS atakas yra sudėtinga, todėl ieškant naujų sprendimų buvo pradėta interneto srautą nagrinėti, kaip chaotinę sistemą. Chaotinėmis vadinamos tarpinės sistemos – tarp gerai prognozuojamų ir sudarytų iš atsitiktinių būsenų sistemų, kurios atitinka keletą papildomų sąlygų. Chaotinė sistema turi tenkinti tokias sąlygas 1) sistema yra jautri pradinėms sistemos sąlygoms, 2) topologiškai maišosi, 3) sistemos periodinė orbita yra koncentruota (turi atraktorių) [13]. Chaotinės sistemos analizuojamos taikant netiesinės dinamikos metodus, kurių pagalba analizuojami sistemos pokyčiai (iš kurių galima nustatyti ir DDoS atakas). Ryšio linijomis persiunčiamų duomenų ar juos talpinančių paketų atributų aprašymas deterministiniais metodais yra neįmanomas, nors pats srautas nėra atsitiktinis [13]. Sistemos analizė gali būti atliekama analizuojant duomenų sekas remiantis Taikenso teorema [14]. Remiantis Taikenso teorema galima suformuluoti teiginį, kad fazinė erdvė yra menama ir joje yra visos galimos sistemos būsenos [M1] Unikalūs taškai atvaizduoti menamoje erdvėje sutampa su sekos būsenomis realioje erdvėje. Siekiant sudaryti sekos rekonstruotą fazinę erdvę reikalingi du parametrai: laiko delsa τ ir dimensijos skaičius m [15]. Parametrų nustatymas yra svarbiausia rekonstrukcijos dalis, bet kartu ir pati jautriausia. Neteisingai parinkus parametrus prognozės tikslumas mažėja. Parametrų nustatymui siūlomos įvairios metodikos

kelios iš jų yra netikrų artimiausių kaimynų, autokoreliacijos, bendrosios informacijos (*mutual information*) ar Cao algoritmas [13]. Kursinio darbo metu atliktas tyrimas parodė, kad negalima išskirti nei vieno metodo, kuris, parenkant parametrus prognozei, būtų pranašesnis už kitus, o tai apsunkina šios metodikos pritaikymą aptinkant DDoS atakas. Taip pat reikėtų paminėti, kad patys skaičiavimai reikalauja santykinai nemažų laiko resursų. Žinoma, didžiąją skaičiavimų laiko dalį sudaro parametrų nustatymas, tačiau pasikeitus nors vienam internetinio tinklo parametrui skaičiavimus reikėtų kartoti. Dar vienas netiesinės dinamikos metodikos trūkumas yra tai, kad galima tik trumpalaikė prognozė, kuri neatspindi dienos srauto pokyčių.

Alternatyva netiesinės dinamikos metodams yra neuroninių tinklų pritaikymas interneto srauto prognozei. Nors pirmieji neuroniniai tinklai buvo pristatyti daugiau kaip prieš pusę šimto metų, dažniausiai užduočių sprendimui buvo taikomos tiesinės ir netiesinės dinamikos metodikos. Tai lėmė ilga užduočių sprendimo su neuroniniais tinklais trukmė. Tačiau įvykus kompiuterinių technologijų proveržiui ir technologijoms pasistūmėjus į priekį susidarė geresnės galimybės neuroninių tinklų taikymui.

Neuroniniai tinklai

Žmogaus smegenis galima apibūdinti kaip biologinį neuronų tinklą t. y. tarpusavyje apjungtą neuronų tinklą, perduodantį elektros signalus. Dendritai gauna įvesties signalus ir, remdamiesi šiais įvesties signalais, sugeneruoja išvesties signalą kuris išvedamas per aksoną. Remiantis žmogaus smegenų neuronų funkcionavimo hipotezėmis yra sukurti skaičiavimuose naudojami neuroniniai tinklai [16]. Tai įgalina kurti sudėtingas neuronines sistemas neturint pilno biologinių neuronų veikimo supratimo. Vedami susidomėjimo ir siekdami naujų atradimų kompiuterių srityje‑1943 m. Neurologas Warrenas S. McCullochas ir logistas Walteris Pitsas sukūrė pirmąjį dirbtinio neuroninio tinklo konceptualų modelį. Savo darbe „A logical calculus of the ideas immanent in nervous activity“ [17] jie apibūdino neuroną, kaip vieną ląstelę, gyvuojančią ląstelių tinkle, kuri gauna įvesties vertę, apdoroja įvesties vertes ir sugeneruoja išvesties vertę. Daugumos mokslininkų ir tyrėjų, nagrinėjusių panašias idėjas siekiai nebuvo skirti tiksliai aprašyti, kaip veikia biologinės smegenys. Vietoj to, dirbtinis neuroninis tinklas (ANN – „Artificial Neural Network“) buvo suprojektuotas kaip skaičiavimo modelis, pagrįstas smegenų veikimo principais.

Dirbtiniai neuroniniai tinklai yra matematiniai algoritmai, programinės įrangos modeliai, kurie sudaryti iš vienetinių elementų, vadinamų neuronais. Dirbtiniai neuronai turi įėjimo kelius kaip biologiniai neuronai turi dendrites; taip pat jie turi ir išvesties kelius kuriuos galima sulyginti su

biologinių neuronų aksomomis. Dirbtiniai, ir biologiniai neuronai turi iš anksto nustatytas parametrų vertes, kurios lemia jų išvesties vertes. Neuronas apdoroja įvestas vertes pagal jame nustatytus parametrus, tada prideda šališkumo elementą (*bias*) ir išveda išvesties signalą. Tiek realiuose tiek dirbtiniuose neuronuose vyksta mokymasis, kurio metu pakeičiamas jungčių tarp neuronų stiprumas.

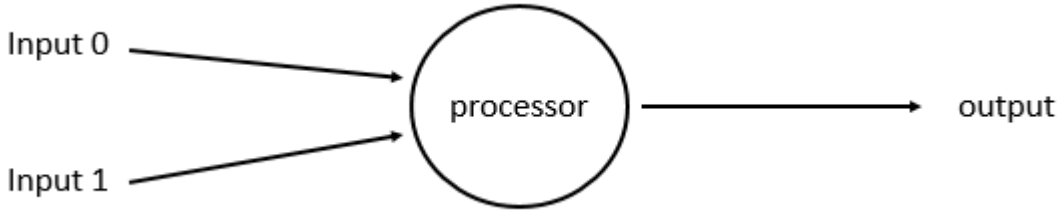
Norint suprasti problemas, kurių sprendimui taikomi neuroniniai tinklai, galima palyginti žmogų su kompiuteriu. Akivaizdu, kad kompiuteriui kai kurios problemos yra neįtikėtinai paprastos, bet žmogui sunkios. Pavyzdžiui, kvadratinė šaknis iš 964324. Optimizuoto programinio kodo eilutė išsprendžia uždavinį ir gauna reikšmę 982. Kompiuteris spręsdamas užduotį užtrunka mažiau nei milisekundę. Kita vertus, yra problemų, kurios yra neįtikėtinai paprastos žmogui, bet ne tokios paprastos kompiuteriui. Parodžius vaikui kačiuko ar šuniuko paveikslėlį jis galės greitai pasakyti, kas yra kas. Žmogui praleidus vos kelias minutes su nepažįstamuoju, jis nesunkiai jį atpažins minioje žmonių. Tačiau norint, kad panašią užduotį atliktų kompiuteris, dalis mokslininkų tam paskyrė visą savo karjerą tiriant ir įgyvendinant sudėtingus kompleksinius sprendimus. Pastarąjį dešimtmetį neuroniniai tinklai dažniausiai yra taikomi siekiant atlikti užduotis, žinomas kaip paprastas žmogui, sudėtingas mašinai (*easy-for-a-human, difficult-for-a-machine*). Viena iš tokių užduočių yra modelio atpažinimas, kurio pritaikymas yra labai platus – nuo simbolių atpažinimo optiniu metodu (paverčiant spausdintą ar ranka parašytą tekstą į skaitmeninį tekstą) iki veido bruožų atpažinimo.

Neuroniniai tinklai (NN) sėkmingai naudojami sudėtingų netiesinių sistemų modeliavimui ir signalų prognozavimui daugelyje inžinerinių programų [18-24]. Neuroniniai tinklai yra viena iš geriausių alternatyvų, galinčių modeliuoti ir prognozuoti įvairiausias sistemas pvz., eismo parametrus, nes taikant neuroninius tinklus galima atkartoti beveik bet kokią funkciją, nepaisant jos netiesiškumo laipsnio be išankstinių žinių apie jos formą [25]. Be to, neuroninių tinklų modeliai buvo sėkmingai pritaikyti siekiant sumodeliuoti sudėtingas netiesines įvesties-išvesties laiko eilutes įvairiuose tyrimuose [20]. Atsižvelgiant į sėkmingą, dirbtinių neuroninių tinklų pritaikymą dinaminių sistemų modeliavimui įvairiose mokslo ir inžinerijos srityse, galima teigti, kad neuroniniai tinklai gali būti veiksmingi ir efektyvūs modeliuojant interneto srauto procesus bei analizuojant, kada ir kokie pokyčiai jame vyksta.

Neuroninio tinklo veikimo principai

Kaip įvesties signalai yra apdorojami dendrituose ir kaip jis yra išvedamas aksiomų aktyvavimui biologiniuose neuronuose lemia gamta. Dirbtiniuose neuroniniuose tinkluose modeliuotojas nustato tinklo procesus ir metodiką. Dar 1957 m. Frank Rosenblatt, Kornelio aeronautikos laboratorijoje, sukūrė perceptroną (2 pav.), kuris yra vieno neurono skaičiavimo modelis. Perceptrono struktūra yra

nesudėtinga ją sudaro vienas ar daugiau įėjimų, procesorius ir viena išvestis. Todėl nagrinėjant perceptroną paprasčiau perprasti sudėtingų struktūrų neuroninių tinklų veikimą.



2 pav. Principinė perceptrono, su dvejomis duomenų įvestimis ir viena išvestimi, schema.

Perceptronas veikia „feed-forward“ modeliu. Duomenys per įvesties taškus siunčiami į neuroną, kuriame jie apdorojami ir išvestyje („output“) gaunamas rezultatas. Pagal 2 pav. tinklas (vienas neuronas) skaitomas iš kairės į dešinę. Taigi 2 pav. pateiktas pavyzdys turi du įeinančius signalus, kurių vertės yra $x_1 = 12$ („input 0“) ir $x_2 = 4$ („input 1“). Duomenys įvedami į neuroną yra padauginami iš svorinių koeficientų 0,5 („input 0“) ir -1 („input 1“) aliekami veiksmai 12* 0,5 = 6 ir 4 * -1= -4. Tada rezultatai yra susumuojami 6 + -4 = 2. Išvesties rezultatas („output“) yra sugeneruojamas sumą perduodant į aktyvacijos funkciją. Nagrinėjamo perceptrono aktyvacijos funkcija yra sign t. y. jei suma teigiama išvestis 1 jei neigiama -1. Atlikus skaičiavimus išvestis yra lygi $\text{sign}(2) = 1$. Perceptrono veikimo algoritmą galima padalinti į tris dalis.

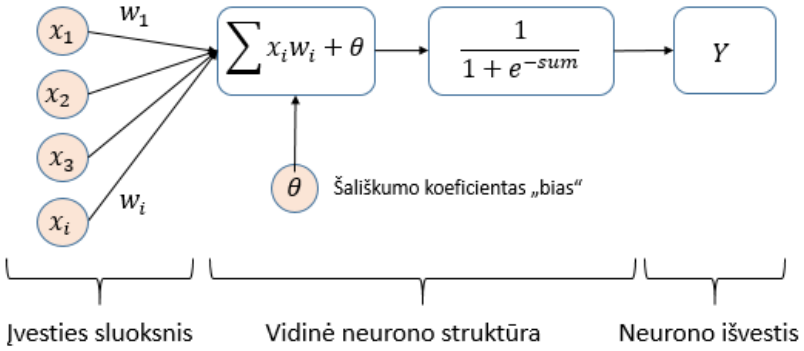
1. Kiekvienas įvesties elementas yra padauginamas iš jam skirto svorinio koeficiento.
2. Sudedami visi įvesties elementai, padauginti iš svorinio koeficiento.
3. Apskaičiuojama išvesties vertė perduodant sumą į aktyvacijos funkciją.

Dirbtinis neuroninis tinklas labai panašus į sujungtus perceptronus. Neuroniniuose tinkluose neurono įvesties kelyje i yra signalas x_i, o kelio stiprumas yra apibūdinimas svorio koeficientu w_i. Neuronų modelį sudaro suminis kelių svoris padaugintas iš įvesties signalo kiekviename kelyje ir pridedamas šališkumo elementas θ [26] (3 pav.). Matematiškai išraiška užrašoma taip:

$$sum = \sum w_i x_i + \theta.$$

Sumos rezultatas perduodamas į aktyvacijos funkciją, kurioje sugeneruojamas išvesties rezultatas Y. Šiuo atveju su logsin [27] formos logistikos funkcija, kurios matematinė išraiška yra

$$Y = \frac{1}{1 + e^{-sum}}.$$



3 pav. Neurono veikimo schema.

Perdavimo funkcijos rūšys

ANN (dirbtinio neuroninio tinklo) elgesys priklauso nuo svorinių koeficientų ir įvesties-išvesties funkcijos (perdavimo funkcijos). Ši funkcija paprastai patenka į vieną iš trijų kategorijų: tiesinė, slenkstinė arba „sigmoid“.

1. Tiesinės funkcijos neuronams išvesties aktyvumas yra proporcingas bendro svorio rezultatui.
2. Neuronų su slenkstine funkcija išvesties vertė priklauso nuo to, ar bendra įvestis yra didesnė arba mažesnė už tam tikrą ribinę vertę.
3. Neuronų su „Sigmoid“ išvesties funkcija, išvesties vertės kinta keičiant įvesties vertes, tačiau pokyčiai nėra tiesiški įvesties verčių pokyčiams. "Sigmoid" elementai yra labiau panašūs į tikrus neuronus nei tiesiniai arba slenkstiniai, tačiau visi trys turi būti laikomi grubiomis aproksimacijomis.

Neuroninio tinklo architektūra

Dažniausiai naudojamas dirbtinis neuroninis tinklas susideda iš trijų grupių arba sluoksnio vienetų, kuriuos sudaro įvesties sluoksnis, išvesties sluoksnis ir vienas arba keli įterpti paslėptieji sluoksniais. Įvesties sluoksnis yra prijungtas prie paslėptojo sluoksnio elementų, kuris yra prijungtas prie išvesties sluoksnio elementų. Kiekvienas sluoksnis susideda iš kelių dirbtinių neuronų, kurie yra sujungti su neuronais esančiais gretimuose sluoksniuose.

1. Įvesties sluoksnis atspindi neapdorotą informaciją paduodamą į neuroninį tinklą.
2. Kiekvieno paslėpto sluoksnio veiklą nulemia įvesties sluoksnio elementai ir svorinių koeficientų vertės tarp įvesties ir paslėptojo sluoksnio elementų.
3. Išvesties sluoksnio elementų elgesys priklauso nuo paslėptojo sluoksnio elementų ir svorinių koeficientų sujungimuose tarp paslėpto ir išvesties elementų.

Šis paprastas tinklo tipas yra įdomus, nes paslėptojo sluoksnio elementai gali laisvai interpretuoti įvesties sluoksnio verčių reikšmes. Svoriniai koeficientai tarp įvesties ir paslėptojo sluoksnio elementų nustato kada ir kuris paslėptasis elementas yra aktyvus, todėl keičiant šiuos svorius, keičiasi paslėptojo sluoksnio elementų atvaizduojamos vertės. Kadangi tinklą sudaro daug tarpusavyje sujungtų neuronų tinklas gali atpažinti ganėtinai sudėtingus reiškinius.

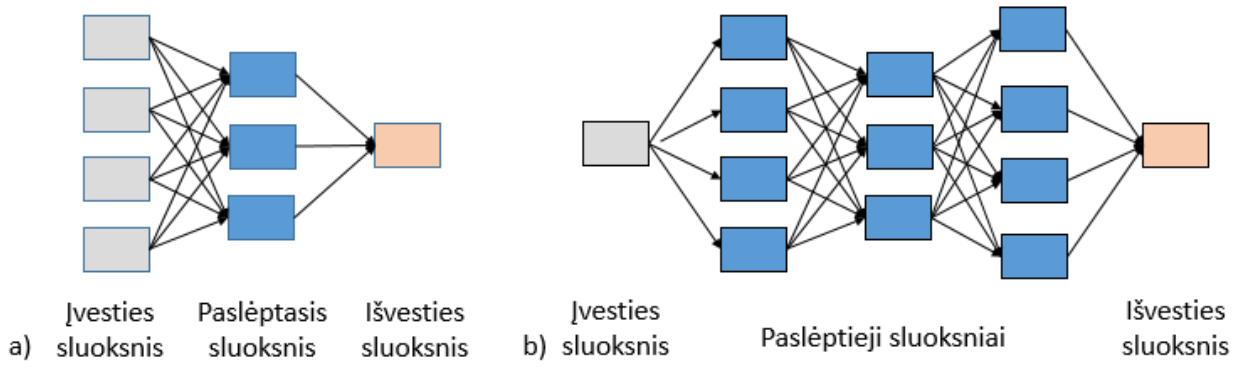
Įvairių tyrimų rezultatai rodo, kad trijų sluoksnių „feed forward“ ANN tinklas gali būti pritaikytas įvairiausių formų ir tipų funkcijų modeliavimui, net kai funkcijos yra sudėtingos, nežinomos ar blogai apibrėžtos formos. Todėl neuroninis tinklas su vienu paslėptu sluoksniu yra universali aproksimacijos sistema netrūkiai funkcijai [28]. Tai ir lėmė šios architektūros paplitimą. Vieno paslėptojo sluoksnio tinklas pasižymi geresne skaičiavimų sparta lyginant su hierarchinės struktūros daugiasluoksniais tinklais. Tačiau norint pritaikyti tokį tinklą reikia išspręsti kelias problemas. Pagrindinė problema yra reikiamo mazgų skaičiaus nustatymas paslėptame sluoksnyje. Jei nagrinėjama sudėtinga funkcija, reikalingas didesnis paslėptų mazgų skaičius. Antroji problema susijusi su optimalių ryšio verčių, vadinamų svoriais nustatymu. Pirmąją problemą galima spręsti tinklą sudarius iš didelio kiekio (šimtų ar tūkstančių) paslėptųjų mazgų paslėptajame sluoksnyje ir palaipsniui sutraukiant tinklą t. y. pašalinant perteklinius mazgus [29]. Tačiau neuroninių tinklų vystymosi laikotarpiu kompiuterių skaičiavimo galimybės buvo labai ribotos, o tokia metodika gali sukelti tinklo apmokymo problemą, daugiadimensiniam tinklui reikalingas ilgas apmokymo laikas, o apmokytas tinklas nebūtinai pateiks geriausius rezultatus. Todėl įprastai taikoma metodika yra priešinga jau paminėtai. Elementų skaičius paslėptajame sluoksnyje yra mažas pradžioje ir palaipsniui didinamas kol pasiekiamas norimas rezultatų tikslumas [30, 31]. Tinkle gali būti naudojamas daugiau nei vienas paslėptas sluoksnis, tokiu atveju tinkle tai pačiai užduočiai atlikti reikalingas mažesnis neuronų skaičius.

Paprastai neuroninio tinklo modeliavimas prasideda nuo tinklo sudarymo, remiantis bendrąja modeliavimo praktika. Vienas bendras principas yra įtraukti kiek įmanoma daugiau pagrįstų prognozuojamų kintamųjų ir netikrų kintamųjų, kurie nepriklauso prognozuojamai duomenų sekai. Taigi, įvesties sluoksnio dydis priklauso nuo prognozuojamų ir netikrų kintamųjų skaičiaus. Laiko eilučių modeliui bendra praktika siūlo užfiksuoti sezoniškumą. Sezoniškumą stengiamasi užfiksuoti turint tiek daug įvesties kintamųjų, kiek yra laiko tarpų viename sezoniniame cikle. Norint sudaryti geresnius modelius, įvesties kintamųjų skaičius yra dažniausiai sumažinamas [26].

Persiuntimo ir atgalinio ryšio neuroniniai tinklai

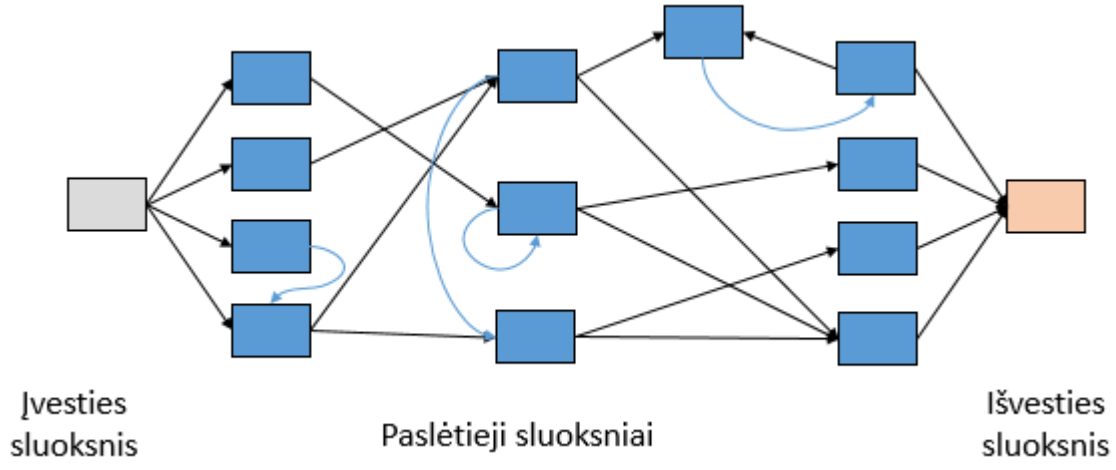
Persiuntimo (*Feed-forward*) neuroninių tinklų grupė yra dažniausiai naudojama sprendžiant įvairias problemas. Tinklas taip pat žinomas kaip „Multi Layer Perceptron Neural Network“ (MLPNN)

[25, 32-34]. Persiuntimo informaciniuose tinkluose (4ab pav.) įvesties signalas gali būti perduodamas tik viena kryptimi; nuo įvesties iki išvesties. Tinkle nėra atgalinio ryšio (kilpų). Tai reiškia, kad bet kurio sluoksnio išvestis neturi įtakos tam pačiam sluoksniui. Persiuntimo informaciniai tinklai dažniausiai yra tiesioginiai tinklai, kurie susieja įvesties duomenis su rezultatais. Jie yra plačiai naudojami modelio atpažinimui. Šios rūšies neuroninių tinklų veikimo principas taip pat vadinamas iš apačios į viršų arba iš viršaus į apačią.



4 pav. Principinės persiuntimo (*Feed-forward*) struktūros neuroninių tinklų schemos.

Atgalinio ryšio (*Feedback*) neuroniniuose tinkluose (5 pav.) gali būti signalų, kurie yra perduodami abiem kryptimis, tinkle pridedant kilpas. Atgalinio ryšio tinklai yra labai naudingi sprendžiant sudėtingas užduotis, tačiau jų struktūra gali būti labai sudėtinga. Atgalinio ryšio tinklai yra dinamiški, jų būsena nuolat keičiasi, kol pasiekiamas pusiausvyros taškas. Tinklas išlieka pusiausvyros taške, kol įvyksta įvesties duomenų pokyčiai, tuomet turi būti nustatyta nauja pusiausvyra. Atgalinio ryšio tinklų architektūros taip pat vadinamos interaktyviomis ar periodiškomis.



5 pav. Principinė atgalinio ryšio neuroninio tinklo schema.

Neuroninio tinklo apmokymas

Vienas svarbiausių neuroninio tinklo elementų yra gebėjimas mokytis. Neuroninis tinklas yra ne tik sudėtinga sistema, o sistema, kuri gali adaptuotis keičiantis sąlygoms. Tai reiškia, kad tinklas gali keisti savo vidinę struktūrą, priklausomai nuo į jį paduodamos informacijos. Paprastai tai pasiekiama reguliuojant svorius. Kiekviena jungtis turi svorį, skaičių, kuris kontroliuoja signalą tarp dviejų neuronų. Jei tinklas sukuria „gerą“ išvesties vertę, svorių koreguoti nereikia. Tačiau, jei tinklas sukuria „prastą“ išvesties vertę, tada sistema prisitaiko, keičiant svorius. Tokiu būdu gerėja tolimesni rezultatai. Neuroninio tinklo apmokymas atliekamas patyrimo principu, kai į tinklą įvedamos nežinomos vertės, tada tinklas gali apibendrinti ankstesnius rezultatus ir išvesti naujus rezultatus [35-37]. Yra dvi tinklo apmokymo metodikos:

- **Prižiūrimas mokymasis** - strategija, kurioje dalyvauja mokytojas, kuris yra protingesnis už patį tinklą. Pavyzdžiui, paimkime veido atpažinimo pavyzdį. Mokytojas rodo tinklui daugybę veidų, taip pat mokytojas žino vardą, susietą su kiekvienu veidu. Tinklas daro prielaidas, tada mokytojas pasako atsakymus tinklui. Sužinojęs atsakymus tinklas gali palyginti savo atsakymus su žinomais „teisingais“ ir atlikti pakeitimus.
- **Neprižiūrimas mokymasis** - reikalingas, kai nėra duomenų pavyzdžio su žinomais atsakymais. Įsivaizduokite, kad ieškote paslėpto modelio duomenų rinkinyje. Tai būtų klasterizavimas – elementų rinkinio dalijimasis į grupes pagal tam tikrą nežinomą modelį.
- **Sustiprintas mokymasis** - strategija, pagrįsta stebėjimu, kuri yra prižiūrimo apmokymo metodikos dalis. Pagalvokite apie mažą pelę, einančią per labirintą. Jei ji pasuko į kairę, ji gauna sūrio gabalėlį; jei ji pasuks į dešinę, ji gaus nedidelį elektros šoką. Manoma, kad pelė per laiką išmoks pasukti į kairę. Neuroninis tinklas priima sprendimą (pasukite į kairę arba į dešinę) ir laikydamasis jo analizuoja rezultatą (sūris ar šokas). Jei analizės rezultatas yra neigiamas, tinklas gali pakoreguoti savo svorius, kad kitą kartą priimtų kitokį sprendimą. Sustiprintas mokymasis yra įprastas robotikoje. Laiko momentu *t* robotas atlieka užduotį ir vertina rezultatus. Ar jis atsitrenkė į sieną arba nukrito nuo stalo? Ar jis nepažeistas?

„Back-propagation“ algoritmas

Norint apmokyti neuroninį tinklą konkrečios užduoties atlikimui, reikia koreguoti kiekvieno mazgo svorius taip, kad būtų sumažinta paklaida tarp norimos išvesties ir faktinės išvesties. Šis

procesas reikalauja, kad neuroninis tinklas apskaičiuotų svorių išvestinę vertę. Kitaip tariant, jis turi apskaičiuoti, kaip pasikeičia paklaida, kai kiekvienas svoris padidėja arba šiek tiek sumažėja. „Back-propogation“ algoritmas yra labiausiai paplitęs svorinių koeficientų nustatymo metodas [38]. Mokymasis vyksta keičiant kelių svorius ir šališkumo elementą. Svoriai koreguojami siekiant sumažinti kvadratinį skirtumą tarp modelio išvesties ir faktinio rezultato analizuojamam duomenų rinkiniui. Kvadratinė paklaida yra skaičiuojama priešinga tinklo veikimui kryptimi ir naudojama svorių ir šališkumo elementui koreguoti. Kvadratinė paklaida yra apskaičiuojam pagal formulę:

$$E = \frac{1}{2}\sum\sum\left(y_{j,c} - d_{j,c}\right)^2,$$

kur c yra indeksas, naudojamas duomenų rinkiniui tinkle įvertinti, j yra išvesties tinklo elemento indeksas , y yra išvesties elementų faktinė būsena, esant tam tikram įvesties verčių rinkiniui ir d yra norima būsena išvestyje esant tam pačiam duomenų rinkiniui įvestyje. Toks koregavimo procesas sukelia gradiento mažėjimą iki minimalios paklaidos. Tačiau šis gradiento mažinimo procesas nėra tobulas, kadangi negalima griežtai pasakyti, kad nustatytas minimumas yra ne vietinis minimumas.

Neuroninių tinklų parametrų parinkimas

Daugeliu atvejų modeliuojant neuroninius tinklus, tinklo parametrai, paslėptų neuronų skaičius ir maksimali laiko delsa, parenkami testavimo ir rezultatų analizavimo būdu. Neuroninio tinklo paslėptų neuronų skaičių reikia parinkti pagal analizuojamos laiko sekos sudėtingumą, o maksimalią laiko delsą siūloma parinkti kelis kartus didesnę už sekos sezoninį ciklą. Sekos sezoninis ciklas yra laiko sekos tendencijų periodinis kitimas. Neuroninių tinklų parametrų parinkimą galima prilyginti netiesinės dinamikos metodo parametrų parinkimui. Siekiant rekonstruoti laiko sekos fazinę erdvę reikia nustatyti dimensijų skaičių ir laiko delsą. Tokiu atveju neuroninio tinklo neuronų skaičius paslėptajame sluoksnyje gali būti nustatytas netikrų artimiausių kaimynų (*false nearest neighbors*)[39] arba Cao [40] metodais, o maksimalios laiko delsos vertę galima nustatyti iš autokoreliacijos ir bendrosios informacijos grafikų [41].

Netikrų artimiausių kaimynų metodas

Netikrų artimiausių kaimynų metodas yra dažniausiai taikomas laiko sekos dimensijų skaičiaus nustatymui [39]. Optimalus dimensijos skaičius m nustatomas vertinant parametrus, kurie turėtų nusistovėti pasiekus reikiamą dimensijų skaičių. Skaičiavimas atliekamas remiantis teiginiu, kad du taškai esantys nedideliu atstumu vienas nuo kito turi išlikti arti vienas kito ir padidinus m. Algoritmas apskaičiuoja atstumus iki artimiausio kaimyno ir taip nustato dimensijų skaičių, kurioje tarp kaimyninių taškų atstumai mažiausi.

Cao algoritmas

Netikrų artimiausių kaimynų metodas yra dažniausiai taikomas sekos dimensijai nustatyti. Tačiau norinti taikyti netikrų artimiausių kaimynų metodą pirmiausia reikia nustatyti delsą, kuri yra vienas iš netikrų artimiausių kaimynų metodo įvesties parametrų. Tiriant interneto srautą, kuris pasižymi stipriomis stochastinėmis savybėmis tiksliai įvertinti optimalią laiko delsą yra sudėtinga [13]. Cao algoritmas, kuris žinomas kaip klaidingų kaimynų vidurkio metodas [40], grąžina tikslesnes vertes. Algoritmas veikia panašiu principu kaip ir netikrų artimiausių kaimynų metodas, vertinamas atstumo pokytis tarp artimiausių kaimynų padidinus dimensiją nuo m iki $m + 1$. Pagrindinis skirtumas nuo netikrų artimiausių kaimynų metodo yra tai, kad Cao metode skaičiuojamas, ne tik dviejų gretimų taškų, o visų taškų atstumo vidurkis.

Autokoreliacija

Duomenų sekos delsą galima įvertinti keliais būdais, dažniausiai naudojamas yra autokoreliacijos metodas. Siūlomi trys būdai parinkti delsos parametrą, taikant autokreliacijos metodą. Pirmasis yra kai sekos autokoreliacijos vertė sumažėja ir kerta $\frac{1}{e}$ tiesę. Antruoju būdu delsa parenkama pagal pirmąjį autokoreliacijos grafiko minimumą. Trečiuoju būdu delsa parenkama pagal pirmą autokoreliacijos grafiko vertę kuri mažesnė už nulį.

Duomenų sekos, kurios elementus pažymėsime $x_i, i = 1, 2, 3, \dots, N$, koreliacija apskaičiuojama pagal formulę:

$$R_j = \sum \frac{(x_i - \bar{x})(x_{i+j} - \bar{x})}{\sum (x_i - \bar{x})^2},$$

čia $\bar{x}$ yra duomenų vidurkis, x_{i+j} pažymėti paslinkti duomenys, o R_j yra apskaičiuotas autokoreliacijos koeficientas, kai duomenys paslinkti per j.

Bendroji informacija

Kitas laiko delsos nustatymo metodas yra bendrosios informacijos (*mutual information*) algoritmas, kuris lyginant su autokoreliacijos metodu yra pranašesnis, nes įvertina tiesinę ir netiesinę koreliaciją tarp duomenų sekos verčių [42]. Bendroji informacija parodo kiek informacijos galima nuspėti apie seką, kai yra žinoma visa kitos sekos informacija.

Bendroji informacija $I(S, Q)$ yra skaičiuojama tarp duomenų sekos $S = \{x(t_1), x(t_2), \dots, x(t_n)\}$ ir tos pačios duomenų sekos laike paslinktų verčių $Q = \{x(t_1 + \tau), x(t_2 + \tau), \dots, x(t_n + \tau)\}$ per laiko delsą τ. $I(S, Q)$ yra išreiškiama taip:

$$I(S, Q) = H(Q) + H(S) - H(S, Q),$$

čia $H(Q)$, $H(S)$ ir $H(S, Q)$ yra sekų entropijos apskaičiuotos pagal formules pateiktas žemiau.

$$H(Q) = -\sum_{s=1}^{n} P(Q_s)\log_2 P(Q_s),$$

$$H(S) = -\sum_{s=1}^{n} P(S_s)\log_2 P(S_s),$$

$$H(Q,S) = H(S,Q) = -\sum_{i,j} P(S_i,Q_j)\log_2 P(S_i,Q_j),$$

čia $P(L_s)$ ir $P(K_i,L_j)$ yra sekos narių tikimybės, tuomet bendrosios informacijos išraiška:

$$I(S,Q) = \sum_{i,j} P(S_i,Q_i)\log_2 \frac{P(S_i,Q_i)}{P(S_i)P(Q_i)}. \quad (14)$$

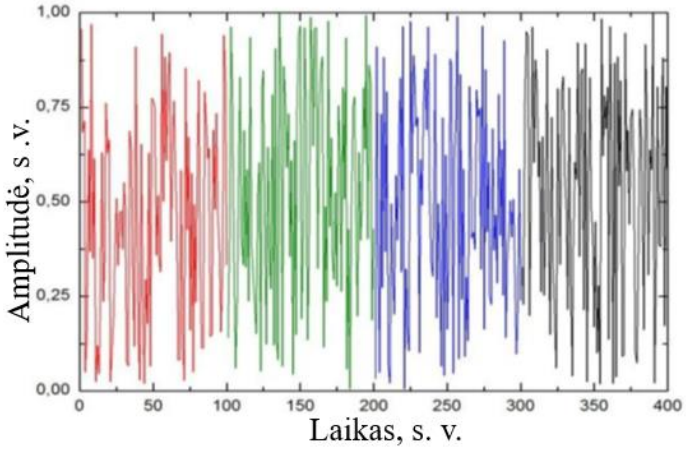
Optimali laiko delsos vertė nustatoma pagal pirmąjį bendrosios informacijos grafiko minimumą [43, 44].

Tyrimo metodika

Siekiant darbo metu išsikelto tikslo suformuluoti uždaviniai buvo padalinti į tris dalis. Pirmojoje dalyje buvo išanalizuotas tiriamajame darbe [45] pristatytas neuroninis tinklas ir skaičiavimų rezultatai. Antrojoje darbo dalyje buvo parašytas programinis kodas su NAR (Nonlinear autoregressive neural network) struktūros neuroniniu tinklu, norint išanalizuoti neuroninių tinklų pritaikymo galimybes. Neuroninis tinklas patikrintas su chaotine Mackey Glass duomenų seka ir atlikta interneto srauto duomenų prognozės analizė. Trečioje darbo dalyje rezultatai, gauti naudojant NN metodikas palyginti su ankstesniais tyrimais [13, 46] ir atliktas įvairių metodų (ARIMA, netiesinės dinamikos ir neuroninių tinklų) pritaikymo galimybių palyginimas.

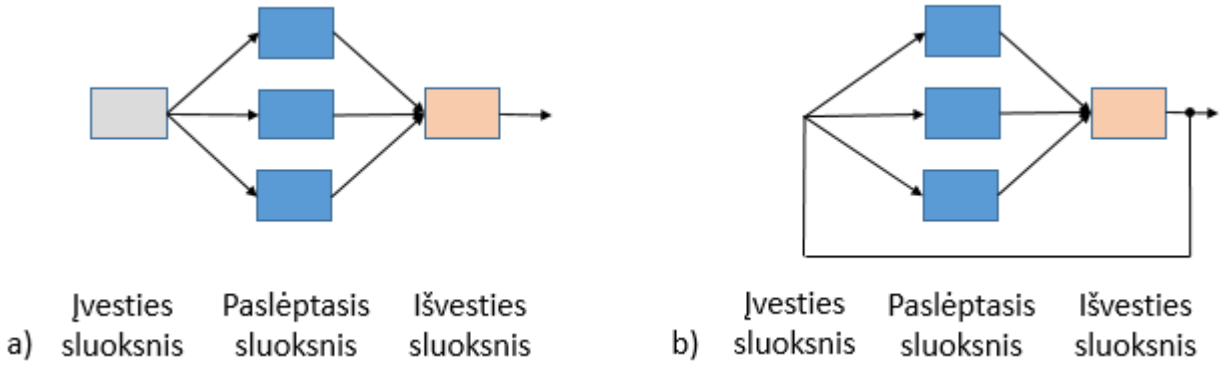
Pirmoje tyrimo dalyje išanalizuotos laiko delsos neuroninio tinklo, pristatyto [45] darbe, pritaikymo galimybės interneto srauto prognozei. Darbo metu išanalizuotas šis metodas ir jo programinis kodas, kuris pateiktas [45] darbo priede. Siekiant patikrinti darbe pristatytus rezultatus ir ištirti pritaikymo galimybes kitiems duomenų srautams, atlikti neesminiai kodo pakeitimai (1 priedas). Pakeistas programinis kodas pavadintas „timedelay“ – pagal neuroninio tinklo funkciją MATLAB programoje. „timedelay“ programa nuskaito nurodytą duomenų failą ir jį padalina į keturias lygias dalis (failą turi sudaryti lyginis duomenų skaičius arba reikia modifikuoti failo interpretavimą programiniame kode). Padalinto failo pirmoji ir antroji duomenų dalys panaudotos tinklo apmokymui. Pirmoji dalis yra kintančių laike verčių seka, kurią neuroninis tinklas interpretuoja kaip žinomus sekos istorinius duomenis, pagal kuriuos tinklas prognozuoja laiko eilutės ateities vertes. Antroji duomenų dalis vadinama tikslų seka t. y. duomenys einantys po istorinių duomenų sekos, kuriuos neuroninis tinklas turi prognozuoti. Trečioji duomenų sekos dalis taip pat yra istoriniai duomenys, kurių ateities vertes apmokytas tinklas prognozuoja pagal pirmąją ir antrąją duomenų sekas. Ketvirtoji dalis yra skirta neuroninio tinklo prognozės rezultatų palyginimui su realiais duomenimis.

Nurodžius „timedelay“ programos pagrindinius parametrus – laiko delsos intervalą ir paslėptojo sluoksnio neuronų skaičių – pradedamas tinklo apmokymas pagal pirmąją ir antrąją duomenų sekas (6 pav.). Apmokymas atliekamas Levenberg–Marquardt metodu. Tinklo mokymasis vyksta keičiant tinklo neuronų ryšių svorinius koeficientus, siekiant sumažinti kvadratinį nuokrypį tarp realių ir prognozuojamų verčių (6 pav. Žalia spalva). „timedelay“ programos neuroniniame tinkle maksimali laiko delsos vertė nurodo kiek praeities verčių įtraukti prognozuojant vieną sekos vertę į priekį. Apmokius tinklą ir nustačius svorinius koeficientus neuroniniam tinklui atliekama prognozė, kurios istoriniai duomenys yra trečioji duomenų dalis (6 pav. mėlyna spalva). Tuomet gauti prognozės rezultatai palyginami su ketvirtąja duomenų sekos dalimi (6 pav. juoda spalva).



6 pav. Interneto srauto duomenų padalinimo į keturias dalis pavyzdys. Primoji ir antroji duomenų dalis (raudona ir žalia kreivės atitinkamai) skirtos tinklo apsimokymui. Trečioji dalis (mėlyna kreivės) skirtas prognozei, o ketvirtoji (juoda kreivė) prognozės verčių patikrinimui.

Antrojoje tyrimo dalyje išanalizuotas NAR („Nonlinear autoregressive neural network“) struktūros laiko delsos neuroninis tinklas (7ab pav.) ir jo pritaikymo galimybės chaotinėms interneto srauto sekoms prognozuoti. Darbo metu parašytas programinis kodas naudojant MATLAB programavimo kalbą. NAR struktūros neuroninis tinklas nuo jau paminėto anksčiau „timedelay“ skiriasi veikimo principu. Pirmiausia įvesties duomenys padalinami į dvi dalis (duomenis galima padalinti į skirtingo dydžio dalis). Pirmoji dalis vadinama apsimokymo duomenimis. Antroji duomenų dalis vadinama patikrinimo duomenimis. Pirmiausia programoje parenkami pagrindiniai parametrai: kiek praeities verčių naudoti prognozei, paslėptųjų sluoksnių skaičius ir neuronų kiekis paslėptuosiuose sluoksniuose. Tada pradedamas tinklo apmokymas, tinklas apmokomas su pirmąja duomenų dalimi. Tinklo apmokymas atliekamas pagal Levenberg–Marquardt [47] algoritmą. Apmokymo metu nustatomi optimalūs svoriniai koeficientai ryšiams tarp neuronų ir šališkumo elemento (*bias*) vertės. Užbaigus tinklo apmokymą tinklo struktūra pakeičiama į uždaro tipo struktūrą 7b pav. Pakeitus neuroninio tinklo struktūrą sugeneruotos ateities vertės grąžinamos į tinklo įvestį. Su uždaros struktūros NAR neuroniniu tinklu, galima atlikti daugiau nei vienos ateities vertės prognozę. Tuomet atliekamas ateities verčių prognozavimas ir gautos vertės palyginamos su antrąja duomenų seka.

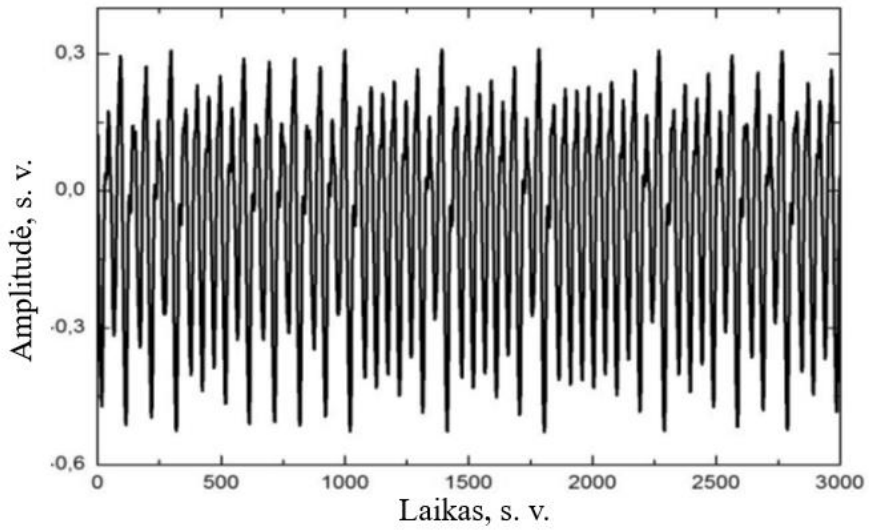


7 pav. NAR neuroninio tinklo atviros (a) ir uždaros (b) struktūros pavyzdžiai.

Tyrimui naudoti duomenys

Siekiant išanalizuoti tiriamų neuroninių tinklų pritaikymo galimybes programos pradiniam veiksmingumui įvertinti buvo naudojama chaotinė seka, kuri buvo panaudota ir netiesinės dinamikos metodų analizės darbuose [13, 45, 46].

Tiriant laiko delsos neuroninio tinklo pritaikymo galimybes darbo metu buvo naudojama plačiai žinoma Mackey Glass laiko seka, kuri pasižymi chaotinėmis savybėmis ir neturi aiškiai apibrėžto periodiškumo. Mackey Glass sekos fragmentas pateiktas 8 pav.



8 pav. Mackey Glass sekos (3000 taškų) fragmentas.

Mackey Glass sekos diskretinė išraiška pateikta žemiau:

$$y(n+1) = y(n) - b * y(n) + c * \frac{n-\tau}{(1+y(n-\tau)^{10})}$$

čia b, c ir τ yra lygties koeficientai. Šiame darbe naudota seka sugeneruota parinkus tokias koeficientų vertės $b = 0{,}1$ $c = 0{,}2$ ir $\tau = 17$.

Darbe analizuoti 1999 metų MIT srauto duomenys, kurie sudaryti fiksuojant universiteto vidinio tinklo serverių srautą [48]. MIT svetainėje yra pateikiami detalūs duomenų rinkinių aprašai, kuriuose yra informacija apie tai, kokio tipo tinkle ir kaip duomenys buvo sugeneruoti arba surinkti, fiksuojant ne simuliuoto, o realiomis sąlygomis naudojamo tinklo srautus. Duomenų rinkiniuose yra pateikiama papildoma informacija apie užfiksuotas ir suklasifikuotas anomalijas bei jų prigimtį. MIT duomenys yra vieni iš dažniausiai interneto srauto analizei naudojamų duomenų rinkinių, todėl galima palyginti skirtingų tyrimų rezultatus.

Antras interneto srauto duomenų rinkinys CICIDS2017 paimtas iš Kanados universiteto University of New Brunswick (UNB). Duomenų rinkinys pristatytas straipsnyje [49]. Lyginant su kitais plačiai prieinamais rinkiniais, tokiais kaip DARPA98 (MIT), KDD99, ISC2012 ir ADFA13, CAIDA(2002-2016) [49], šis duomenų rinkinys yra tinkamesnis interneto srauto analizei. Pagrindiniai išvardintų duomenų rinkinių trūkumai yra šie: senesniuose duomenų rinkiniuose surinkta informacija yra pasenusi ir neatspindi dabartinės interneto srauto struktūros (MIT, KDD99); nepakankamas duomenų rinkinių informacijos dokumentavimas (sunku suprasti, kas yra kas) (ISC2012, ADFA13); daugumos naujų duomenų rinkinių surinktos interneto srauto informacijos kiekis yra nepakankamas detaliems tyrimams (CAIDA (2002-2016)).

CICIDS2017 duomenų rinkinys sugeneruotas sudarius kompiuterių, serverių ir ugniasienių uždarą tinklą, kuriame sukurta legalių vartotojų bendraujančių su serveriais infrastruktūra bei kenkėjiškų įrenginių grupė. Duomenų rinkinį sudaro legalių vartotojų srautas ir įvairiausiomis nusikaltėlių metodikomis sugeneruoti atakų srautai, nukreipti prieš šią tinklo infrastruktūrą. Duomenų rinkinys sugeneruotas 2017 metais. Rinkinyje surinkta penkių dienų srauto informacija. Pirmosios dienos srautas sudarytas tik iš legalių vartotojų paketų. Likusiomis dienomis užfiksuotas interneto srautas yra su įvairių atakų paketais, kurių aprašai pateikiami kartu su duomenų rinkiniu [50]. Darbo metu skaičiavimai atlikti tik su pirmosios dienos duomenimis.

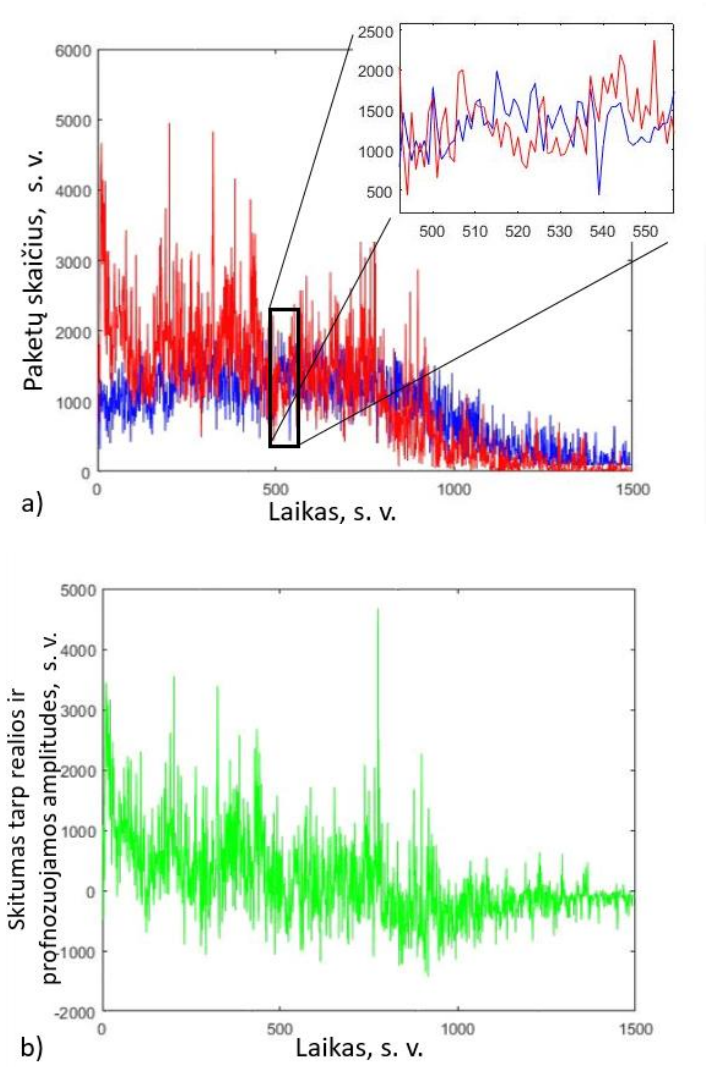
Interneto srauto duomenų apdorojimas

Atsisiųsti duomenų failai su interneto srautu yra įrašyti pcap formatu, tokius failus galima atidaryti su viešai prieinama „WireShark“ programa. Ši programa įgalina vizualizuoti užfiksuoto srauto paketus ir jų informaciją,. Todėl, norint analizuoti turimus duomenis, buvo panaudotas „WireShark“ programos duomenų eksportavimo funkcionalumas ir duomenys buvo eksportuoti į txt failus.

Eksportuotame faile išsaugota ši informacija apie kiekvieną paketą: eilės numeris, paketo laiko žymė, šaltinio IP adresas, paskirties adresas ir protokolas. Darbo metu pagrindinis dėmesys buvo skiriamas srauto intensyvumo nagrinėjimui, todėl svarbiausia informacija šiuo atveju yra paketo laiko žymė. Siekiant įvertinti tinklo apkrovą, duomenis reikia papildomai apdoroti. Duomenys sugrupuoti tam tikrais laiko intervalais. MIT duomenys sugrupuoti 30 s ilgio intervalais siekiant patikrinti [45] darbe pateiktus rezultatus. Kadangi antrajame duomenų rinkinyje CICIDS2017 užfiksuotas daug intensyvesnis srautas, duomenys grupuoti 1 s trukmės intervalais.

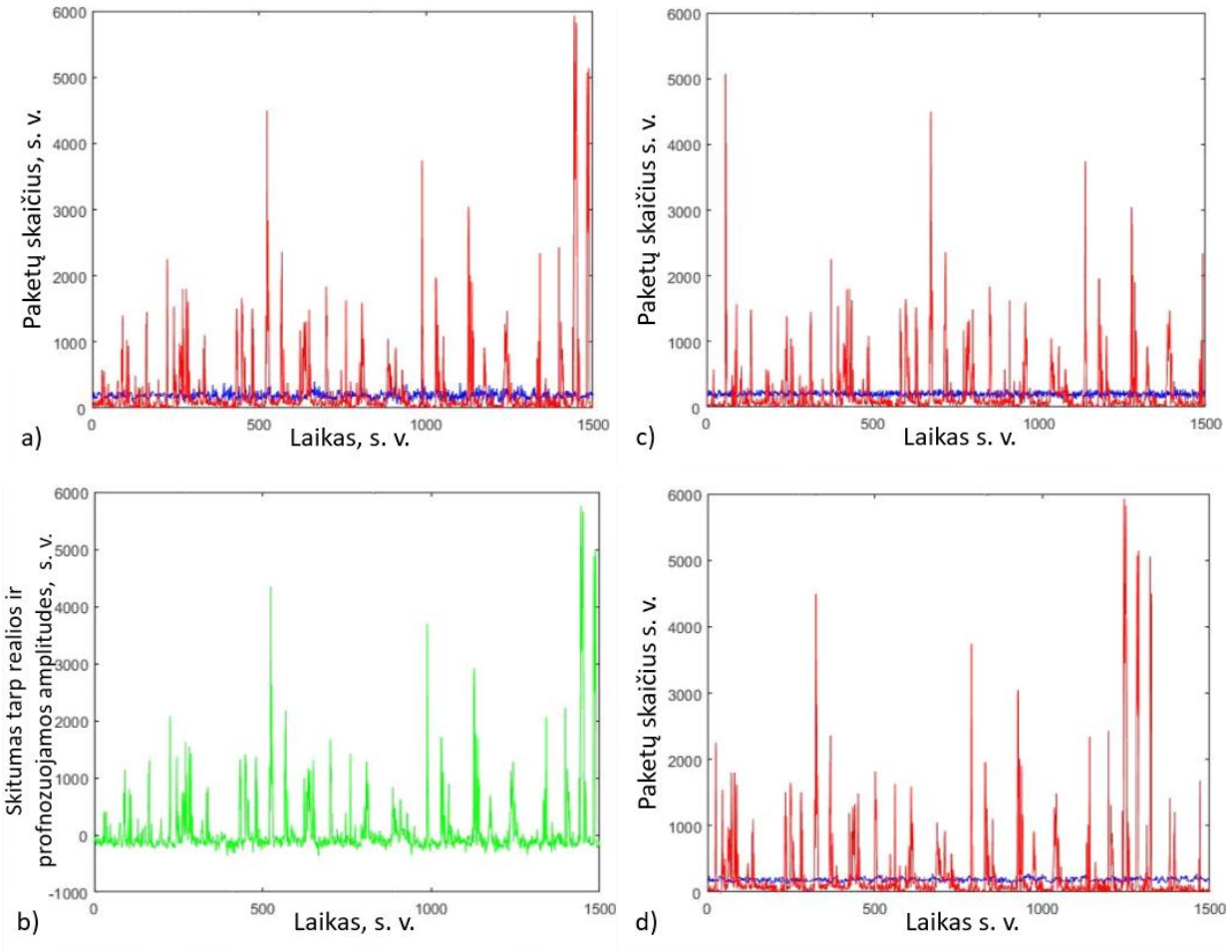
Rezultatai ir jų aptarimas

Darbo metu analizuojant laiko delsos neuroninį tinklą pirmiausia buvo patikrinti [45] darbe pateikti rezultatai su MIT srauto duomenimis, kurie buvo sugrupuoti 30 s trukmės laiko intervalais. Interneto srauto prognozė atlikta nustačius paslėptų neuronų skaičių 10 ir maksimalią delsą 300 s. v. Prognozės rezultatai pateikti 9 pav. (sekos duomenys skirti neuroninio tinklo apmokymui nepateikti). Išanalizavus grafiką galima pasakyti, kad srauto trumpalaikė prognozė nėra tiksli, tačiau atspinti tam tikras kitimo tendencijas ir, vertinant ilgalaikę srauto prognozę gauti rezultatai pakankamai tiksliai atkartoja realių duomenų ilgalaikes tendencijas.

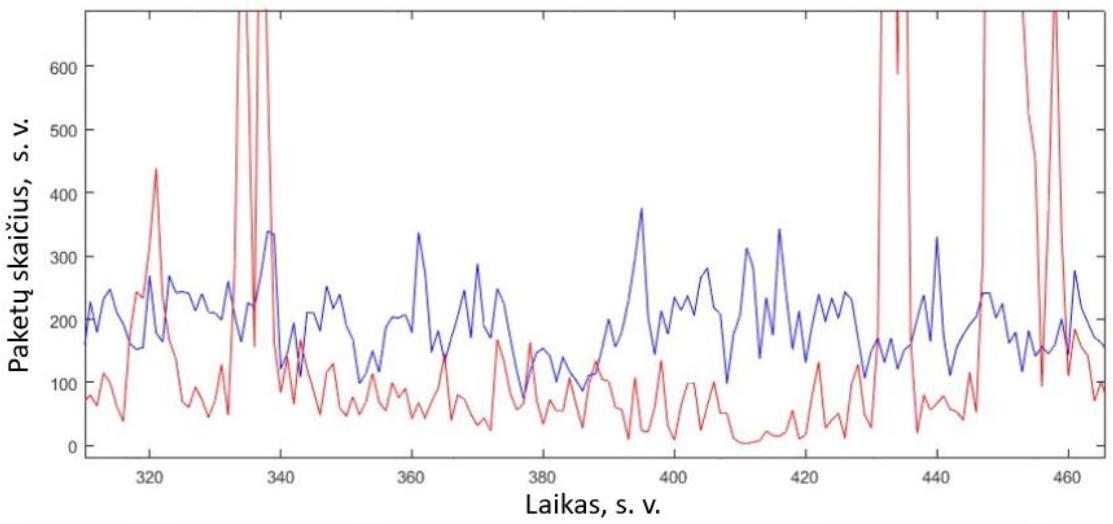


9 pav. MIT interneto srauto paketų skaičiaus (a) – laiko eilutės realūs (raudona kreivė) ir prognozės (mėlyna kreivė) rezultatai , bei skirtumo tarp realių ir prognozuojamų rezultatų paveiksle (b) laikinės priklausomybės. Prognozė atlikta taikant neuroninį tinklą.

Atlikus tinklo srauto prognozės skaičiavimus su MIT duomenų rinkiniu, rezultatai buvo palyginti su rezultatais, pateiktais [45] darbe. Palyginus rezultatus galima pasakyti, kad gauti rezultatai yra labai panašūs į pateiktus [45] darbe. Tolimesniam tyrimui buvo panaudotas antrasis duomenų rinkinys CICIDS2017. Kadangi interneto srautas yra daug intensyvesnis nei pirmajame, antrajame rinkinyje paketai sugrupuoti 1 s trukmės intervalais. Interneto srauto prognozė atliekama parinkus neuroninio tinklo parametrus, neuronų skaičių 10 ir maksimalią laiko delsos vertę 300. Skaičiavimų rezultatai pateikti 10ab ir 11 pav. Išnagrinėjus šiuos grafikus galima teigti, kad šiuo atveju ateities verčių prognozė neatspindi nei trumpalaikių nei ilgalaikių tendencijų, o yra artima tiesei, kurios vertės kinta intervale nuo 100 iki 400. Pakeitus laiko delsos parametrus į 150 ir 500, skaičiavimų rezultatai pateikti 10cd pav. atitinkamai. Palyginus a, c ir d grafikus galima teigti, kad prognozės rezultatai praktiškai nepasikeitė ir visais trimis atvejais kinta nuo 100 iki 400.

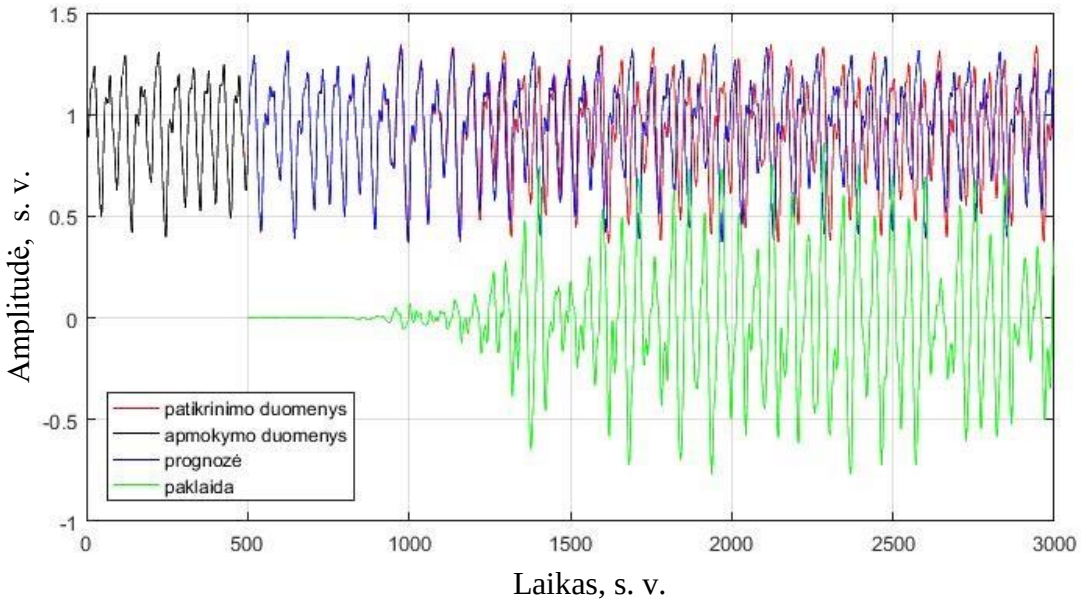


10 pav. CICIDS2017 duomenų paketų skaičiaus prognozės (mėlyna spalva), kai parinkta maksimali laiko delsa yra 300 (a), 150 (c) ir 500 (d) grafikai. Realios vertės (raudona spalva) grafikuose a, c ir d. Skirtumo tarp realių ir prognozuojamų verčių, kai maksimali laiko delsa yra lygi 300 verčių grafikas(b).



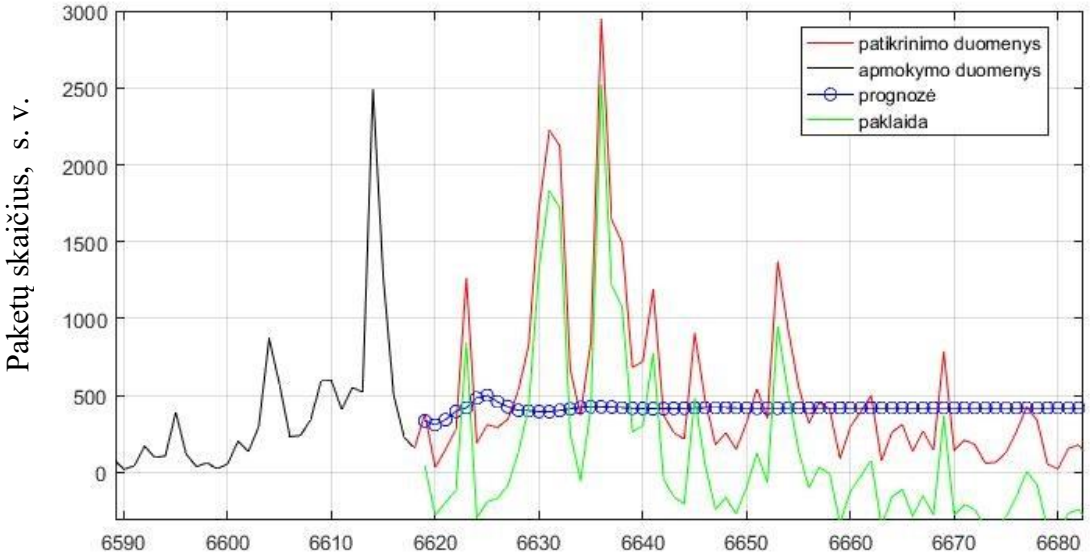
11 pav. CICIDS2017 duomenų paketų skaičiaus prognozės (mėlynai), kai parinkta maksimali laiko delsa yra 300 verčių grafikas.

Tolimesnis tyrimas buvo pratęstas nagrinėjant NAR struktūros laiko delsos neuroninį tinklą. Tyrimo metu parašytas programinis kodas pateiktas antrame priede. Kaip jau minėta tyrimo metodikos skyriuje šis tinklas turi tik vieną duomenų įvestį ir viena išvestį. Siekiant įvertinti ar šis neuroninis tinklas yra tinkamas chaotiškiems interneto srauto duomenims apdoroti, tinklo efektyvumas pirmiausia ištirtas su plačiai taikoma chaotine Mackey Glass duomenų seka. Neuroninis tinklas apmokytas su 500 sekos verčių. Modeliuotas 6 paslėptųjų neuronų tinklas, maksimali laiko delsa 20. Neuroninio tinklo prognozės rezultatai atvaizduoti 12 pav., kuriame juoda spalva pažymėti tinklo apmokymo duomenys, raudona spalva originalūs sekos duomenys, skirti palyginimui su prognozės (pažymėti mėlyna spalva) rezultatais. Skirtumo tarp realių Mackey Glass sekos verčių ir naudojant neuroninį tinklą prognozuotų verčių, kreivė pažymėta žalia spalva. Išnagrinėjus 12 pav. pateiktus rezultatus matoma, kad apmokius tinklą su 500 sekos duomenų ir panaudojus 20 praeities verčių prognozei galima tiksliai prognozuoti apie 400, o apytiksliai daugiau kaip 2500 ateities verčių.

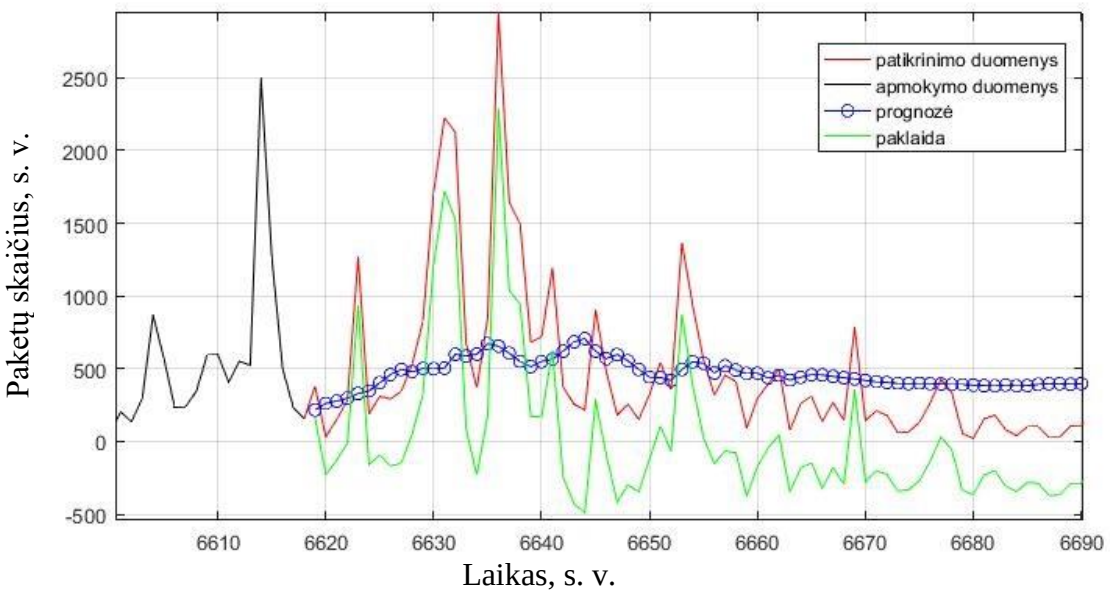


12 pav. Mackey Glass sekos prognozės rezultatai.

Įvertinus neuroninio tinklo pritaikymo galimybes chaotinėms sekoms, tyrimas buvo tęsiamas su CICIDS2017 duomenų rinkiniu. Tačiau interneto srautas lyginant jį su Mackey Glass seka pasižymi daug stipresnėmis stochastinėmis savybėmis. Todėl norint, kad tinklas apsimokymo metu nusistatytų tinkamus svorinius koeficientus ir šališkumo elemento vertes, labai svarbu tinkamai parinkti neuroninio tinklo dydį ir laiko delsos parametrą. Chaotiškiems interneto srauto duomenims nuspėti galimai tinkamus parametrus yra sunku. Tokiu atveju siūloma analizę pradėti nuo nesudėtingos struktūros neuroninio tinklo ir palaipsniui didinti parametrų vertes. Pradiniai parametrai parenkami tokie: neuronų skaičius 10 ir maksimali delsa 10. Apmokius tinklą su ketvirtadaliu duomenų apytiksliai prognozuojamos 9 vertes (13 pav.), pagal kurias galima spręsti apie realių duomenų kitimo tendencijas, tačiau prognozuojamos ateities vertės neatitinka realų duomenų verčių. Padidinus maksimalią laiko delsos vertę skaičiavimo rezultatai pateikti 14 pav. Padidinus laiko delsą iki 50 prognozuojamų rezultatų skaičius padidėja ~40, kurie atspindi sekos trumpalaikes tendencijas. Tolimesnės prognozės vertės nusistovi į tiesę.

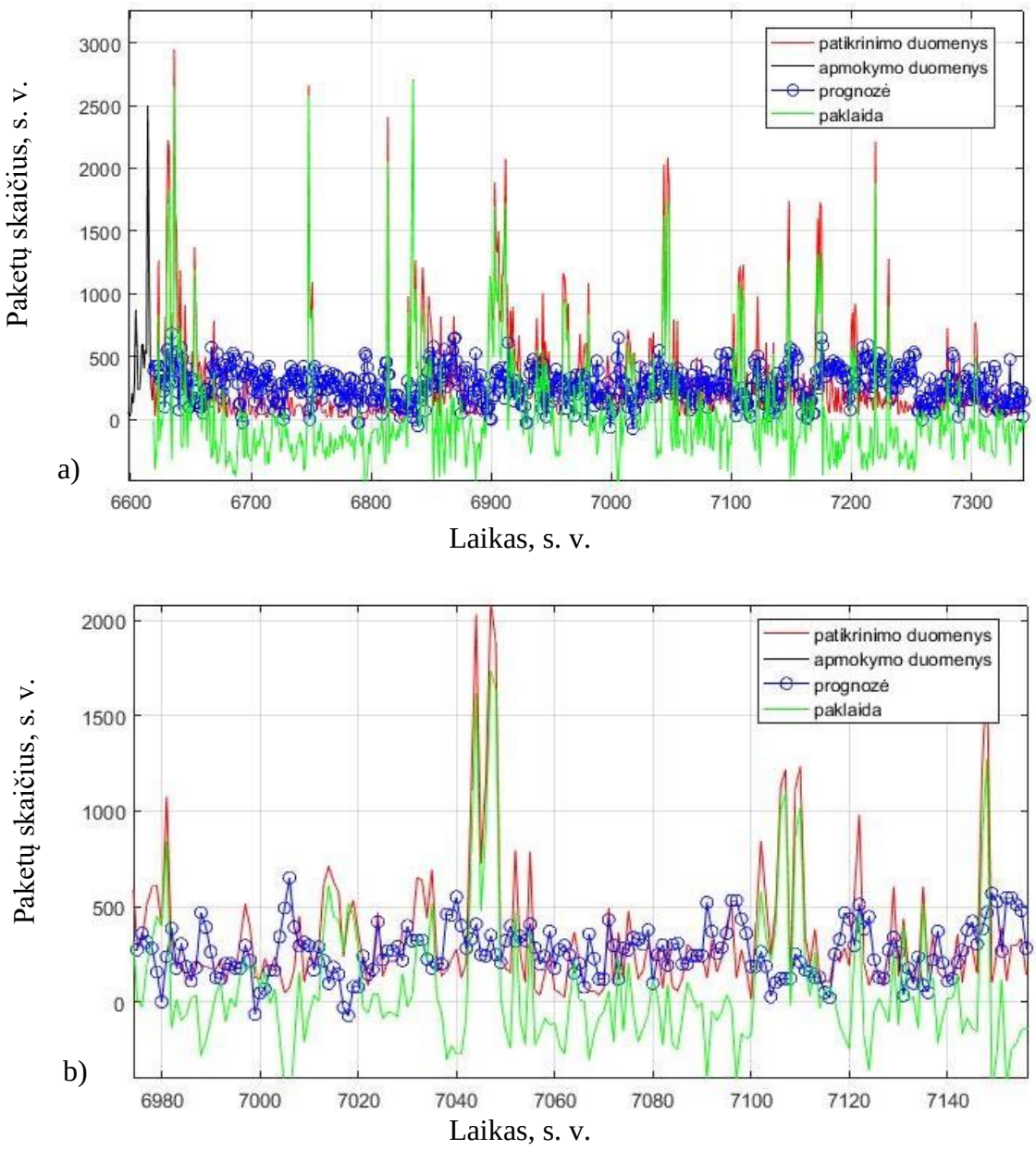


13 pav. CICIDS2017 duomenų paketų skaičiaus prognozės rezultatai, apskaičiuoti naudojant neuroninį tinklą su 10 neuronų paslėptajame sluoksnyje ir maksimalia delsa lygia 10.



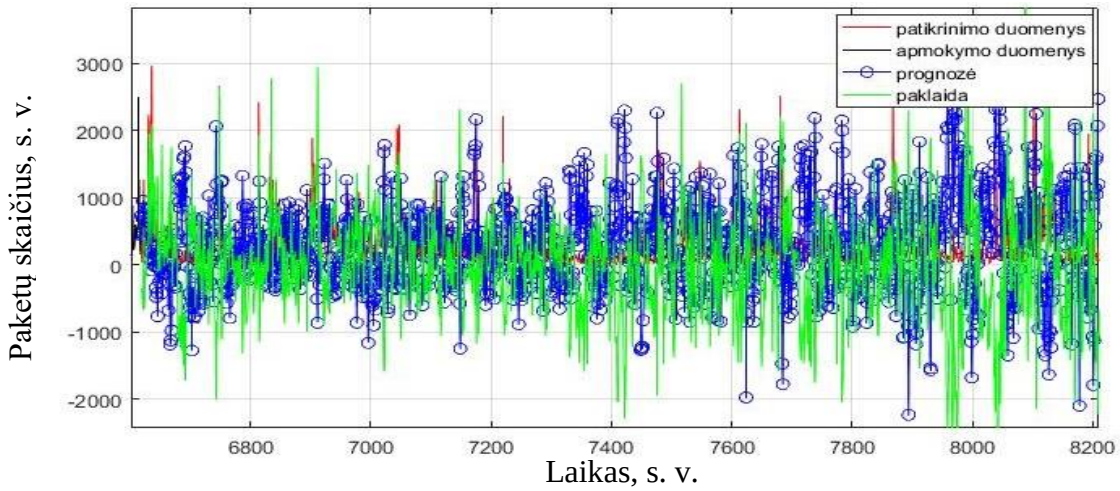
14 pav. CICIDS2017 duomenų paketų skaičiaus prognozės rezultatai. Neuroninio tinklo parametrai yra 10 neuronų paslėptajame sluoksnyje, o maksimali delsa lygi 50.

Padidinus maksimalią delsą iki 1000, modeliavimo, naudojant NAR neuroninį tinklą, rezultatai pateikti 15ab pav. Išanalizavus grafikus galima teigti, kad prognozės rezultatai atspindi realių duomenų tendencijas.



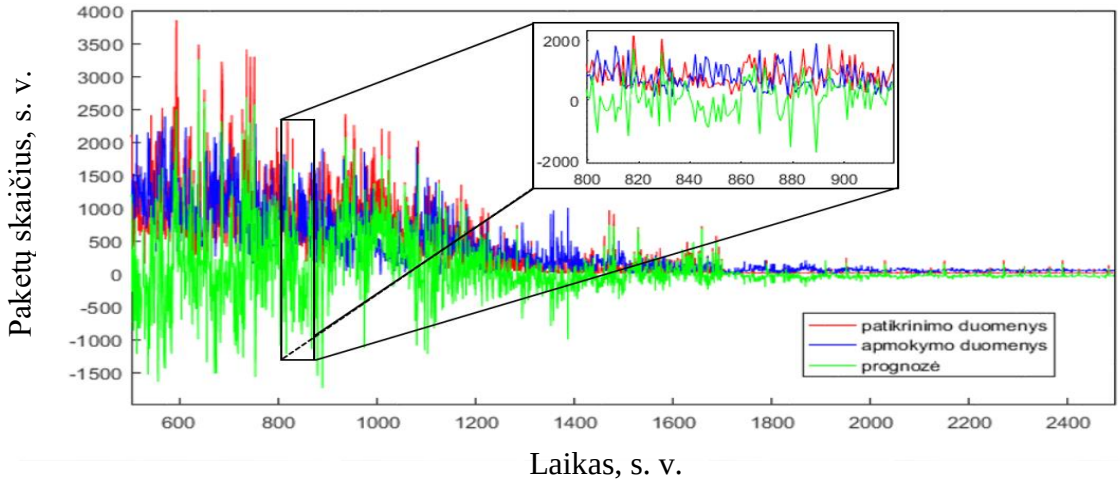
15 pav. CICIDS2017 paketų skaičiaus laikinės priklausomybės prognozės rezultatai. Neuroninio tinklo parametrai yra 10 neuronų paslėptajame sluoksnyje, o maksimali delsa lygi 1000 (b pav. pateiktų rezultatų detalizacija 6977 – 7150 intervale).

Padidinus laiko delsą iki 2000 (16 pav.) įvyksta tinklo persimokymas, todėl prognozės rezultatai nebeatitinka realių sekos duomenų. Toliau kartojant veiksmus su didesniu neuronų kiekiu (20 ir 30) prognozės rezultatai, lyginant su 15 pav. pateiktais rezultatais, yra prastesni.



16 pav. CICIDS2017 paketų skaičiaus laikinės priklausomybės prognozės rezultatai. Neuroninio tinklo parametrai: 10 neuronų paslėptajame sluoksnyje ir maksimali laiko delsa 2000.

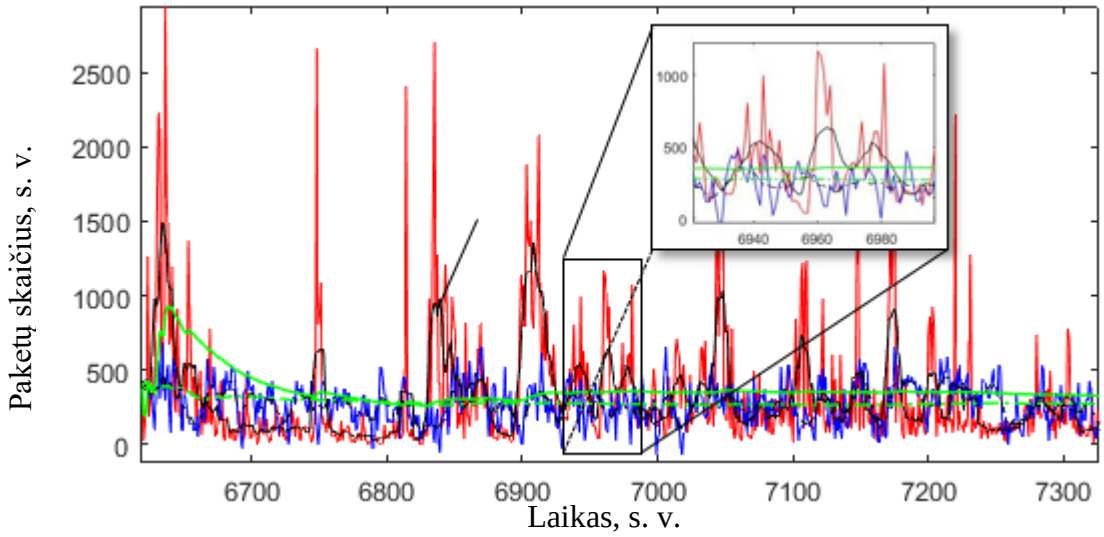
Skaičiavimai buvo pakartoti su MIT duomenų rinkiniu, kai duomenys sugrupuoti 30 s trukmės intervalais. Naudojant NAR struktūros neuroninį tinklą nustatyta, kad parinkus parametrus: paslėptų neuronų skaičių 10 ir maksimalią delsą 500, prognozės rezultatai geriausiai atitinka realius srauto duomenis (17 pav.).



17 pav. MIT paketų skaičiaus laikinės priklausomybės prognozės rezultatai. Neuroninio tinklo parametrai: 10 neuronų paslėptajame sluoksnyje ir maksimali laiko delsa 500.

Siekiant tiksliau įvertinti ar NAR neuroniniu tinklu sugeneruotos sekos prognozės vertės atitinka realius duomenis buvo apskaičiuoti lokalūs ir suminiai vidurkiai CICIDS2017 duomenų rinkiniui. Lokalaus vidurkio skaičiavimas buvo atliktas vidurkį skaičiuojant su 5 taškų iš abiejų pusių nuo skaičiuojamojo taško vidurkio. Suminis vidurkis buvo skaičiuojamas kiekviename taške susumuojamos

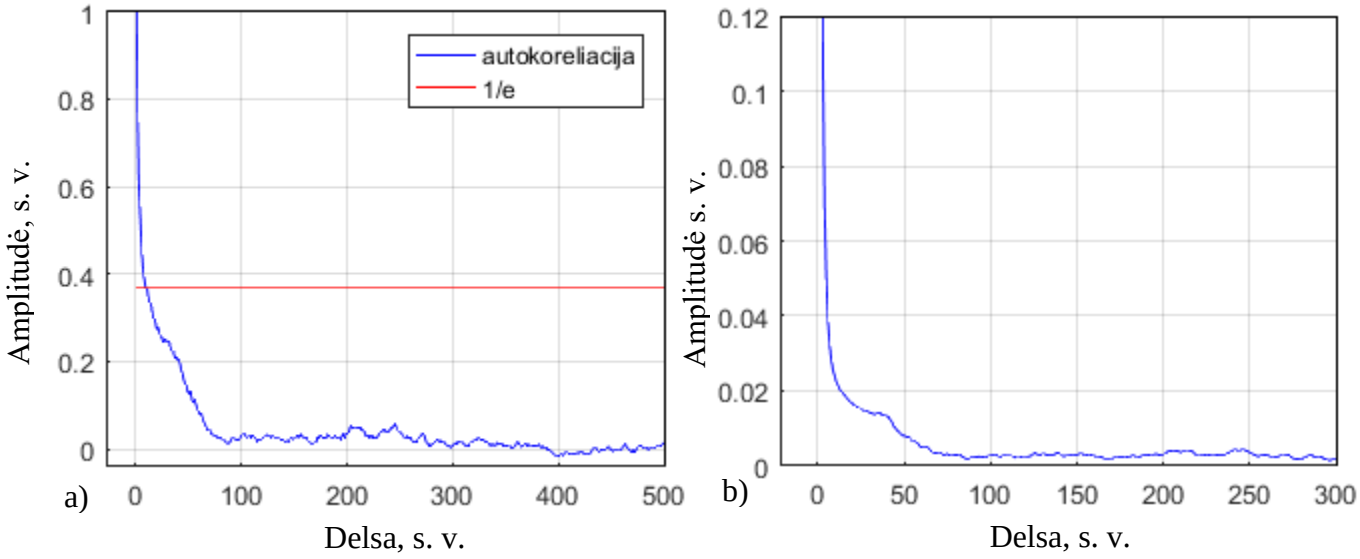
visų praeities taškų vertės ir padalinant sumą iš jų skaičiaus. Išanalizavus rezultatus atvaizduotus 18 pav. galima teigti, kad prognozės rezultatų lokalaus ir suminio vidurkio kreivės yra labai panašios į realių duomenų lokalaus ir suminio vidurkio kreives.



18 pav. CICIDS2017 duomenų paketų skaičiaus (raudona spalva), lokalaus vidurkio (juoda spalva), suminio vidurkio (žalia spalva) ir paketų skaičiaus prognozės (mėlyna spalva), lokalaus vidurkio (juoda spalva, punktyrinė kreivė), ir suminio vidurkio (žalia spalva, punktyrinė kreivė) grafikas.

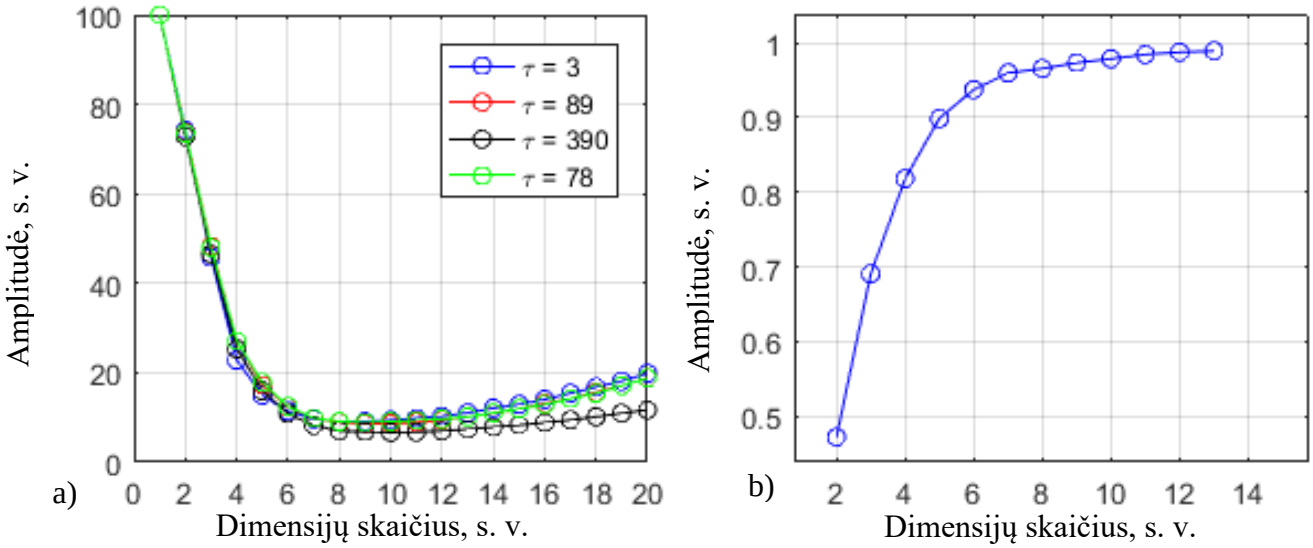
Norint geriau suprasti, kodėl prognozės rezultatai neatspindi interneto sraute matomų pikų (staigių, bet trumpalaikių paketų skaičiaus padidėjimų), buvo apskaičiuota Hurst eksponentė [51] (programos kodas pateiktas 3 priede). Hurst eksponentės vertė yra lygi ~0,016, apskaičiuota reikšmė yra mažesnė už 0,5. Tai reiškia, kad interneto srauto duomenys sugrupuoti 1 s trukmės intervalais svyruoja apie savo vidurkį. Remiantis gautais rezultatais galima teigti, kad duomenų sekoje nėra užkoduotos ilgalaikės informacijos [52-54], kuri leistų prognozės metu nuspėti sekoje matomus paketų skaičiaus staigius padidėjimus.

Siekiant nustatyti ar neuroninio tinklo parametrus, paslėptų neuronų skaičių ir maksimalią delsą, galima nustatyti ne tik parametrų parinkimo ir rezultatų analizės metodu, o taikant metodus skirtus netisinės dinamikos metodo parametrų parinkimui, atlikti skaičiavimai su CICIDS2017 duomenų rinkiniu. Maksimali laiko delsa apskaičiuojama autokoreliacijos ir bendrosios informacijos metodais 19ab pav. Autokoreliacijos metodu laiko delsa gali būti nustatyta pagal tris skirtingus būdus: a) kai autokoreliacijos vertė sumažėja e kartų, b) pagal pirmą autokoreliacijos minimumo vertę ir c) kai autokoreliacijos vertė lygi 0. Iš pateiktų rezultatų 19a pav. nustatytos tokios delsos vertės: 3, 89, 390. Taikant bendrosios informacijos metodą delsa parenkama pagal pirmąjį minimumą ir yra lygi 77 (19b pav.).



19 pav. CICIDS2017 duomenų paketų skaičiaus autokoreliacijos (a) ir bendrosios informacijos (b) grafikai.

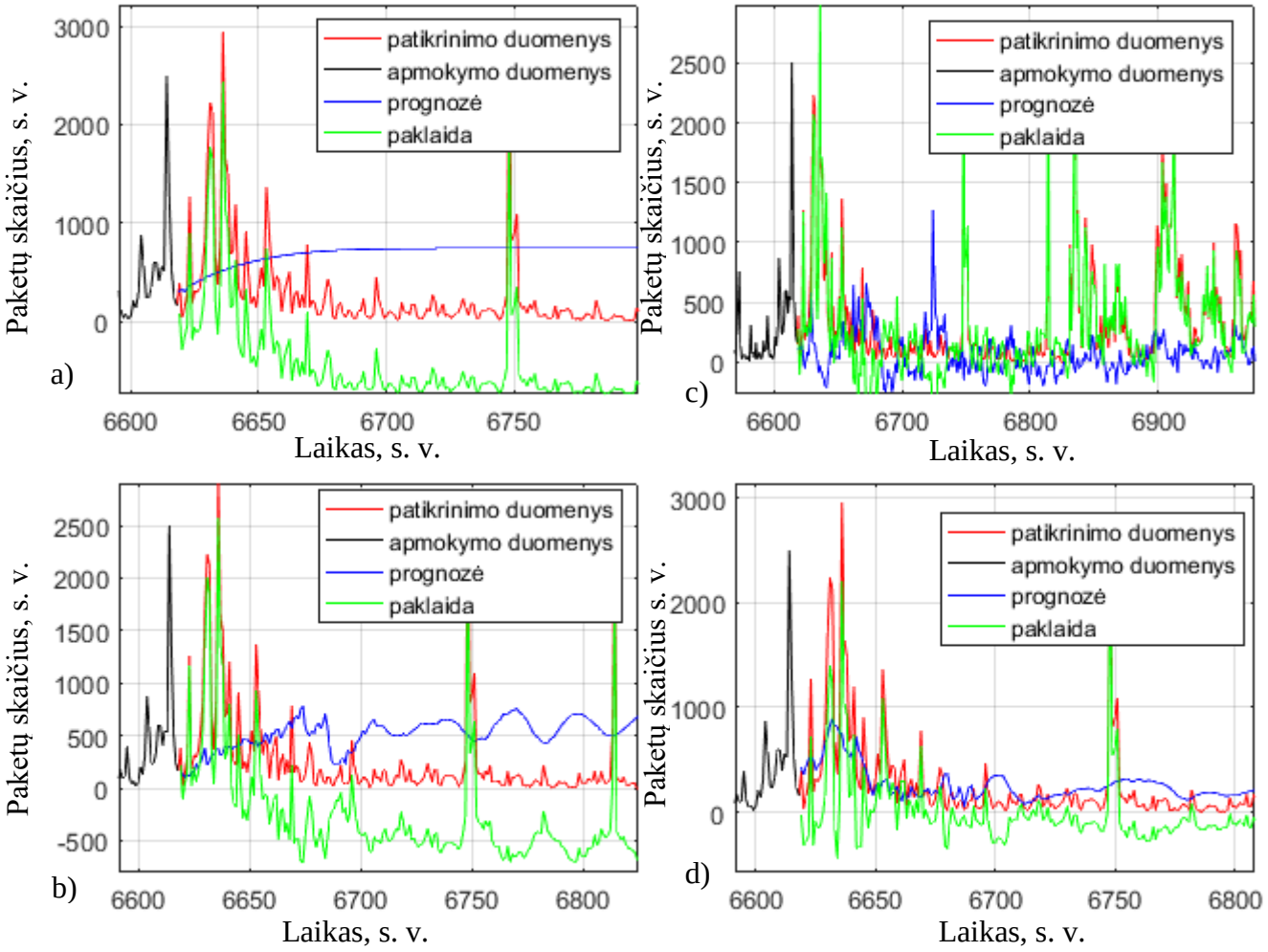
Neuroninio tinklo paslėptųjų neuronų skaičius nustatomas pagal netikrų artimiausių kaimynų minimumo vertę ir Cao metodo įsisotinimo vertę rezultatai atvaizduoti 20ab pav. Išanalizavus rezultatus nustatyta, kad paslėptų neuronų skaičius pagal netikrų artimiausių kaimynų metodą yra 8, o pagal Cao metodą 12.



20 pav. CICIDS2017 duomenų paketų skaičiaus netikrų artimiausių kaimynų (a) ir Cao (b) grafikai.

Nustačius CICIDS2017 duomenų sekos parametrus taikant metodus taikomus netiesinės dinamikos metodo parametrų nustatymui atlikti neuroninio tinklo parametrų pakeitimai. Skaičiavimų

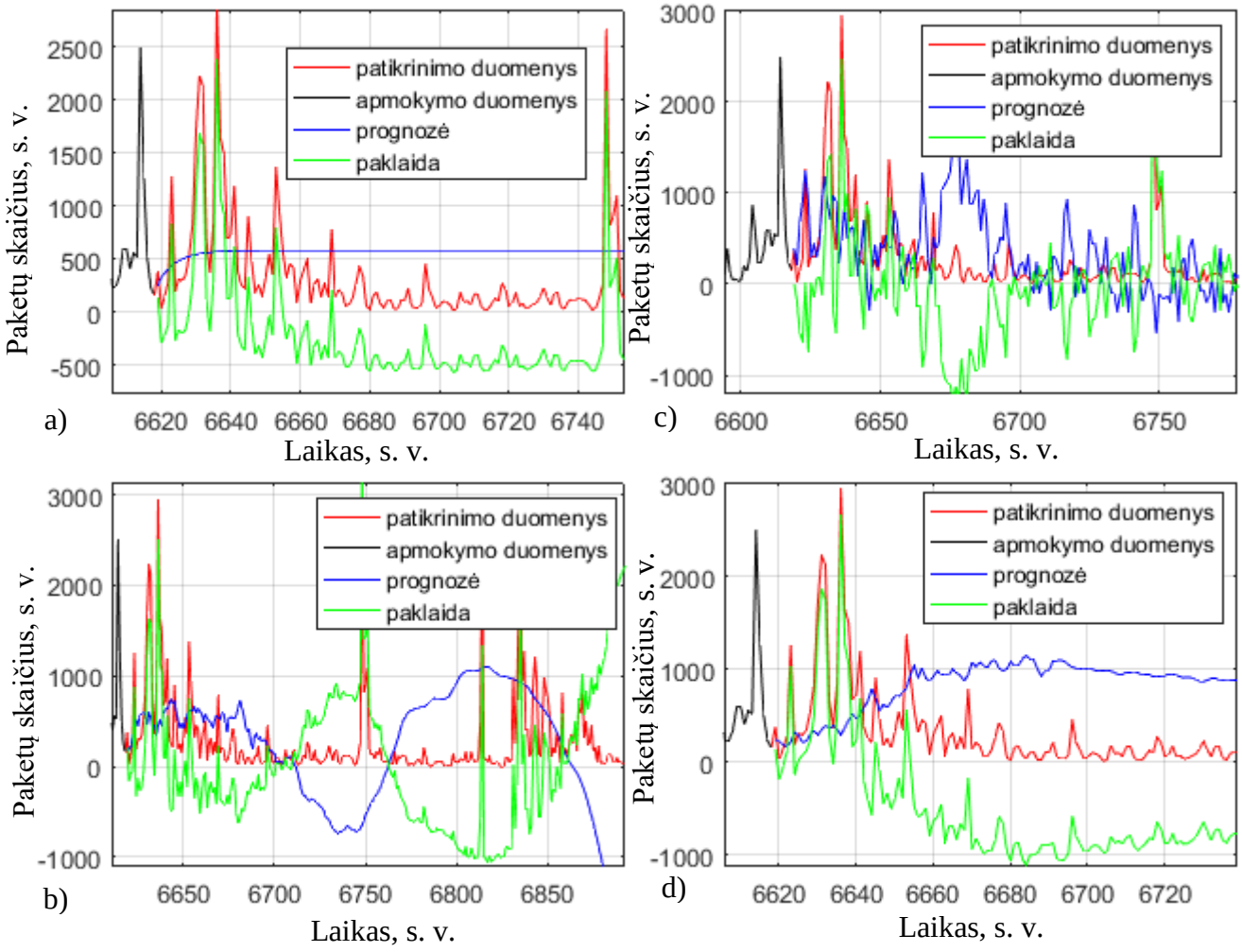
rezultatai, kai paslėptųjų neuronų skaičių parinktas 8, pateikti 21 pav. Išnagrinėjus 21 pav. pateiktus grafikus a, b, c ir d galima pasakyti, kad esant paslėptųjų neuronų skaičiui 8 geriausi prognozės rezultatai (21c pav.) gauti, kai delsos parametras nustatytas pagal bendrosios informacijos grafiko pirmąjį minimumą. Prognozės vertės apytiksliai atitinka daugiau kaip 150 verčių. Kai laiko delsos parametras nustatytas pagal autokoreliacijos sumažėjimą e kartų prognozė neatkartoja realių duomenų sekos verčių, o artėja į tiesę. Kai laiko delsa parenkama pagal autokoreliacijos pirmąjį minimumą ir kai autokoreliacija mažiau už 0 (21bc pav.) prognozės vertės atspindi tik trumpalaikes sekos tendencijas.



21 pav. CICIDS2017 duomenų paketų skaičiaus prognozė. Neuroninio tinklo parametrai: paslėptų neuronų skaičius 8, maksimali delsa 3 (a), 89 (b), 390 (c) ir 78 (d).

Neuroninio tinklo paslėptųjų neuronų skaičių nustačius pagal Cao (20b pav.) įsisotinimo vertę, kuri lygi 12, skaičiavimų rezultatai pateikti 22 pav. Išnagrinėjus 22 pav. pateiktus grafikus a, b, c ir d galima pasakyti, kad esant paslėptųjų neuronų skaičiui 12 geriausi prognozės rezultatai 22bc pav. gauti,

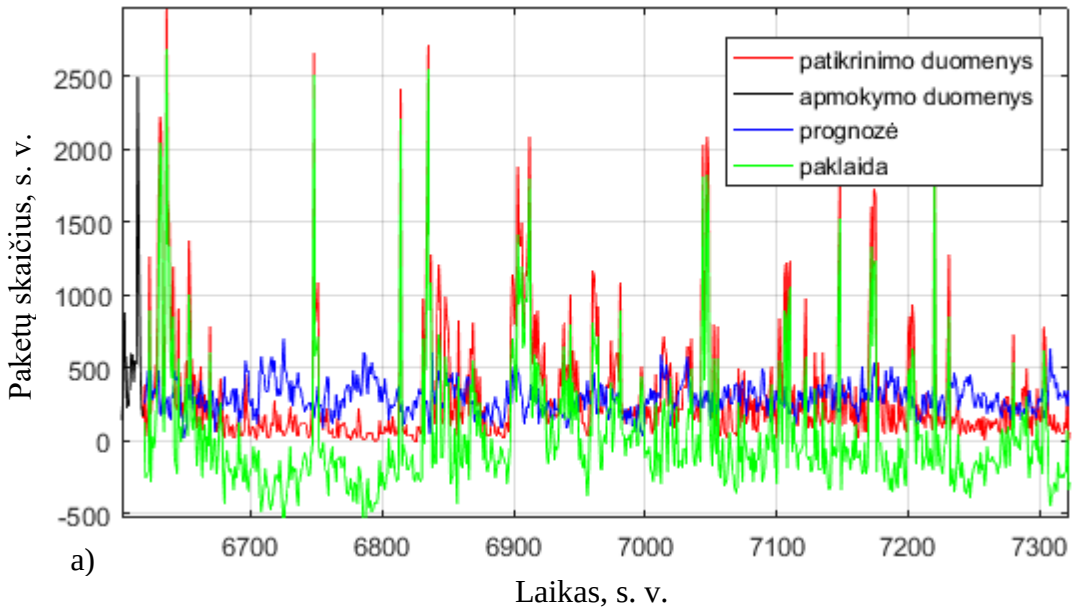
kai delsos parametras nustatytas pagal autokoreliacijos grafiko pirmąjį minimumą ir pirmą vertę kuri mažesnė už 0. Prognozės vertės apytiksliai atitinka daugiau kaip 80 (b) ir daugiau kaip 50 (c) verčių. Kai laiko delsos parametras nustatytas pagal autokoreliacijos sumažėjimą e kartų prognozė neatspindi realių duomenų sekos verčių, o artėja į tiesę. Todėl galima teigti, kad laiko delsos parinkimas pagal autokoreliacijos sumažėjimą e kartų yra netinkamas neuroninių tinklų metodui. Kai laiko delsa parenkama pagal bendrosios informacijos pirmąjį minimumą (22d pav.) prognozės vertės atitinka tik kelis realių duomenų taškus.



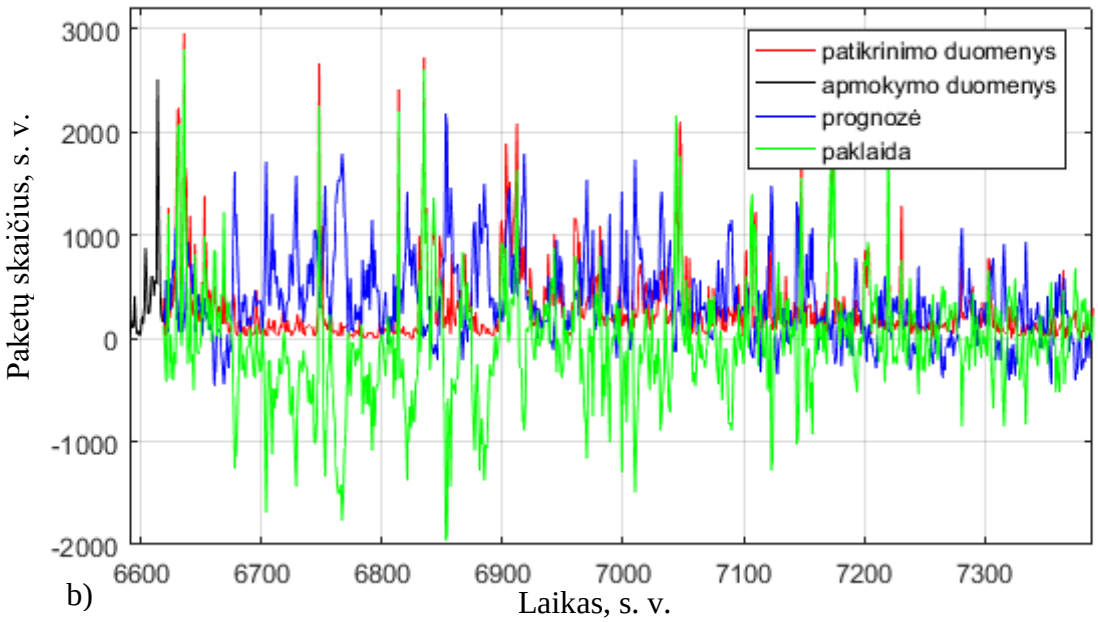
22 pav. CICIDS2017 duomenų paketų skaičiaus prognozė. Neuroninio tinklo parametrai: paslėptų neuronų skaičius 12, maksimali delsa 3 (a), 89 (b), 390 (c) ir 78 (d).

Nustačius delsą pagal autokoreliacijos ar bendrosios informacijos metodus delsos vertė atitinka tik vieną sekos sezoną. Parenkant parametrus neuroninio tinklo apsimokymui siūloma laiko delsą parinkti tokio dydžio, kad joje būtu užfiksuoti keli duomenų sekos sezonai. Todėl tolimesni skaičiavimai buvo atlikti su tris kartus didesne laiko delsa 3*390 = 1170. Skaičiavimo rezultatai pateikti 23ab pav.

Išanalizavus rezultatus, kai paslėptųjų neuronų skaičius yra 8 (23a pav.) prognozės rezultatai apytiksliai atitinka realius duomenis tačiau neatkartoja realių duomenų staigių paketų skaičiaus padidėjimų. Kai paslėptųjų neuronų skaičius yra 12 (23b pav.) prognozės rezultatai apytiksliai atkartoja staigius realių duomenų padidėjimus, tačiau lyginant su 23a pav. pateiktais rezultatais prasčiau atkartoja mažo intensyvumo duomenų sekos vertes.



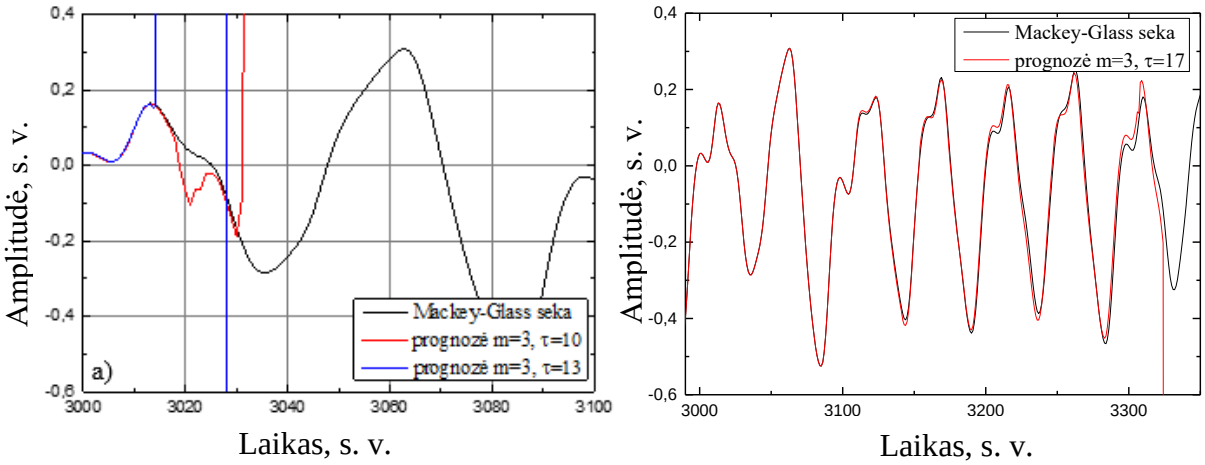
a)



b)

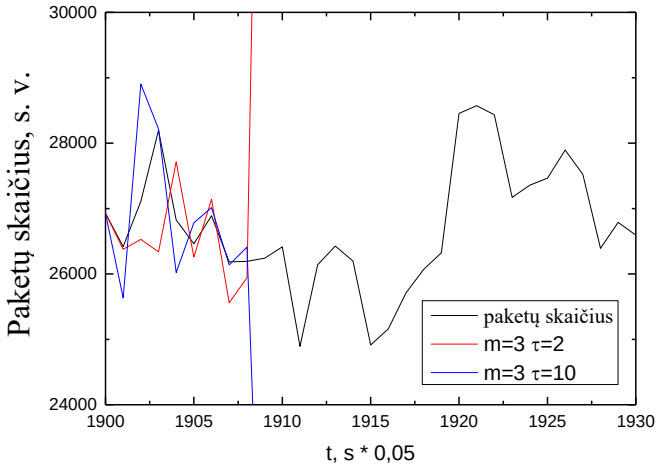
23 pav. CICIDS2017 duomenų paketų skaičiaus prognozė. Neuroninio tinklo parametrai: paslėptų neuronų skaičius8 (a) ir 12 (b), maksimali delsa 1170.

Siekiant visapusiškai įvertinti neuroninių tinklų privalumus, trūkumus ir pritaikymo galimybes interneto srauto analizei tiriamojo darbo metu gauti rezultatai buvo palyginti su rezultatais pristatytais darbuose [13, 46], kuriuose išnagrinėtos netiesinės ir tiesinės dinamikos ARIMA (Autoregresive integrative moving average) metodų taikymo galimybės interneto srauto prognozei. Remiantis rezultatais pristatytais tiriamajame darbe [13], taikant netiesinės dinamikos metodiką chaotinei Mackey Glass sekai prognozuoti, kai prognozei buvo panaudota 3000 sekos verčių, galima ganėtinai tiksliai prognozuoti nuo 15 iki 30 verčių 24a pav. Prognozei būtini parametrai buvo nustatyti iš autokoreliacijos , bendrosios informacijos ir netikrų artimiausių kaimynų grafikų [13]. Pritaikius laiko lango metodą laiko delsos parametrui įvertinti prognozuojamų verčių skaičius padidėja iki 320 24b pav. [13]. Padidinus apmokymo verčių skaičių galima prognozuoti daugiau kaip 1000 verčių [13].



24 pav. Mackey-Glass sekos prognozes naudojant laiko delsą, nustatytą pagal bendrąją informaciją ir autokoreliaciją (a), ir laiko delsą, nustatytą laiko lango metodu (b), grafikai [13].

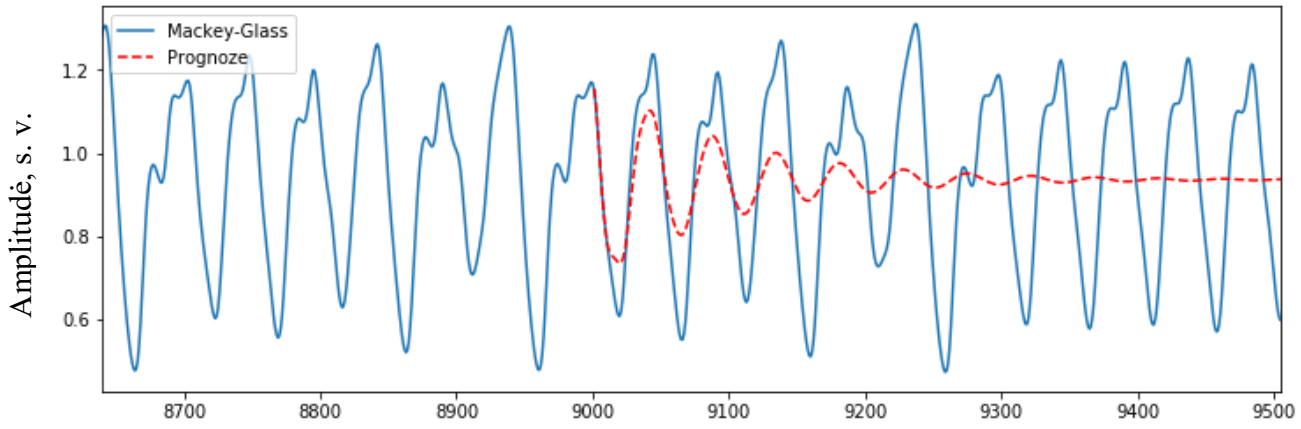
Tiriamajame darbe [13] taip pat išanalizuotas netiesinių dinamikos metodų taikymas interneto srautui prognozuoti. Tyrime buvo naudoti 2016 metais CAIDA (Center for Applied Internet Data Analysis) duomenų centro surinkti duomenys [55]. Duomenys sugrupuoti 0,05 s trukmės intervalais. Prognozei reikalingi parametrai buvo nustatyti iš autokoreliacijos, bendrosios informacijos ir Cao [40] grafikų [13]. Nustačius parametrus galima apytiksliai prognozuoti 15 verčių (25 pav.). Interneto srauto duomenims laiko delsos nustatymas taikant laiko lango metodą negalimas [13].



25 pav. CAIDA fiksuoto paketų skaičiaus per 0,05 s prognozės grafikas [13].

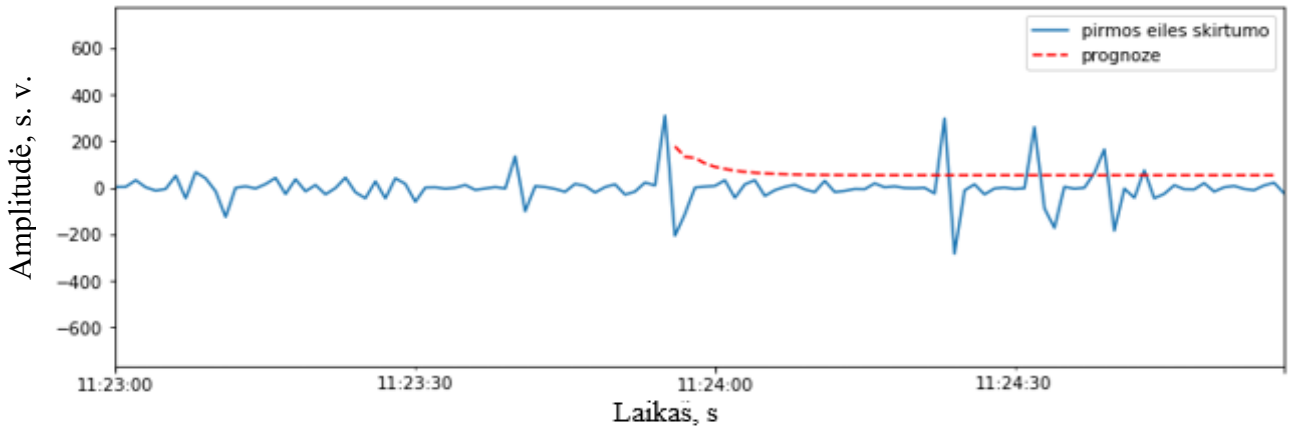
Palyginus šio darbo metu gautus rezultatus su darbe [13] pristatytais rezultatais, galima pasakyti, kad taikant neuroninį tinklą interneto srautui prognozuoti gauti rezultatai yra tikslesni ir galima prognozuoti didesnį ateities verčių skaičių.

Tiriamajame darbe [46] išanalizuotas tiesinės dinamikos ARIMA metodas, plačiai taikomas laiko sekų analizei (ekonomikoje, orų kaitoje ir energetikoje). Pagal [46] darbe pateiktus Mackey Glass sekos skaičiavimų rezultatus, taikant ARIMA metodą, galima teigti, kad pirmi 15 prognozuojamų taškų tiksliai atitinka realius duomenis (26 pav.). Tolimesnės prognozės rezultatai nukrypsta nuo realių duomenų, tačiau atkartoja duomenų trumpalaikes tendencijas ir artėja prie Mackey-Glass sekos vidurkio [46]. Prognozės parametrai buvo parinkti pagal Box-Jenkins straipsnyje [56] aprašomą metodiką.



26 pav. Mckey-Glass sekos prognozė atlikta taikant ARIMA metodą . Parametrų vertės $p = 10$, $i = 0$, $q = 0$ [46].

Darbe [46] taip pat pristatyta ARIMA metodo pritaikymo galimybių analizė interneto srautui prognozuoti. Interneto srauto duomenys buvo sugeneruoti MIT universitete 2000 metais. Kadangi ARIMA metodą galima taikyti tik stacionarumo sąlygą tenkinantiems duomenims, interneto srauto duomenys yra stacionarizuojami [46]. Stacionarizavimas atliktas skaičiuojant pirmos eilės skirtumą $y'_t = y_t - y_{t-1}$. Stacionarizuotų duomenų parametrai prognozei buvo parinkti remiantis Box-Jankins straipsnyje pristatyta metodika. Išanalizavus 27 pav. galima pasakyti, kad ARIMA metodas tinkamas tik trumpalaikei sekos tendencijai nusakyti.



27 pav. Interneto srauto duomenų prognozė atlikta taikant ARIMA metodą. Parametrų vertės $p = 3$, $i = 1$, $q = 0$ [46].

Apibendrinus darbe [46] pristatytus rezultatus, galima teigti, kad tiesinės dinamikos metodas ARIMA yra tinkamas tik laiko sekos tendencijai nusakyti. Todėl galima konstatuoti, kad neuroniniai tinklai yra tinkamesni laiko sekų prognozei.

Pagrindiniai rezultatai ir išvados

1. Darbo metu parašytas programinis kodas skirtas neuroninių tinklų pritaikymo galimybės laikinių sekų prognozės analizei.
2. Išanalizuotos neuroninių tinklų taikymo galimybės chaotinėmis savybėmis pasižyminčioms sekoms ir interneto srauto duomenims.
3. Ištyrus interneto srauto duomenis CICIDS2017 nustatyta, kad parinkus paslėptų neuronų skaičių 10 ir maksimalią laiko delsą 1000 galima apytiksliai prognozuoti daugiau kaip 700 laikinės sekos verčių
4. Įvertinus interneto srauto duomenų CICIDS2017 Hurst eksponentę, kurios vertė yra $\sim 0{,}016$, nustatyta, kad duomenų seka nepasižymi ilgalaike atmintimi, todėl sekos prognozės vertės neatspindi staigių, bet trumpų verčių padidėjimų.
5. Palyginus skaičiavimų rezultatus gautus taikant neuroninius tinklus su kitų dažniausiai taikomų metodų interneto srautui prognozuoti, rezultatais, nustatyta, kad taikant neuroninius tinklus galima apytiksliai prognozuoti iki 700 ir daugiau verčių, o alternatyviais metodais metodais iki 7 verčių.

Literatūra

[1] http://www.internetlivestats.com/internet-users/, (Žiūrėta 2018-01-17).

[2] S. Yu, Distributed Denial of Service Attack and Defence. Springer-Verlag, New York (2013). DOI: 10.1007/978-1-4614-9491-1. eISBN: 978-1-4614-9491-1

[3] T. Thapngam, S. Yu, W. Zhou and G. Beliakov, Discriminating DDoS Attack Traffic from Flash Crowd through Packet Arrival Patterns,Computer Communications Workshops (INFOCOM WKSHPS) (2011). DOI: 10.1109/INFCOMW.2011.5928950

[4] Imperva, https://www.incapsula.com/ddos/denial-of-service.html, (Žiūrėta 2018-04-26).

[5] S. Simkin, https://www.paloaltonetworks.com/cyberpedia/ what-is-botnet, (Žiūrėta 2018-04-11).

[6] T. Fox-Brewster, How Hacked Cameras Are Helping Launch The Biggest Attacks The Internet Has Ever Seen, https://www.forbes.com/sites/thomasbrewster/2016/09/25/brian-krebs-overwatch-ovh-smashed-by-largest-ddos-attacks-ever/#2fafa6685899, (Žiūrėta 2018- 03-08).

[7] F. Soldo, A. Markopoulou, and K. J. Argyraki, Optimal filtering of source address prefixes:Models and algorithms, IEEE INFOCOM, 2446–2454, (2009). DOI: 10.1109/INFCOM.2009.5062172

[8] M. Alenezi and M. J. Reed, Methodologies for detecting DoS/DDoS attacks against network servers, ICSNC: The Seventh International Conference on Systems and Networks Communications, (2012). ISBN: 978-1-61208-231-8

[9] A. Lengvinas, Netiesinės dinamikos metodų taikymas DDoS atakų aptikimui, Bakalaurinis darbas, Vilnius (2016).

[10]. H. Aljifri, Ip traceback: A new denial-of-service deterrent?, IEEE Security and Privacy, **99** (3), 24–31 (2003). DOI: 10.1109/MSECP.2003.1203219

[11] A. Yaar, A. Perrig, and D. X. Song, Fit: fast internet traceback, 24th Annual Joint Conference of the IEEE Computer and Communications Societies, 1395–1406, (2005).

[12] https://securelist.com/analysis/quarterly-malware-reports/72560/kaspersky-ddos-intelligence-report-q3-2015/, (Žiūrėta 2018-03-15).

[13] A. Lengvinas, Duomenų srauto anomalijų analizė taikant netiesinės dinamikos metodus, Kursinis darbas, Vilnius (2017).

[14] Q. F. Meng, Y. H. Peng, A new local linear prediction model for chaotic time series, Physics Letters A, **370** (5-6), 465–470, (2007). https://doi.org/10.1016/j.physleta.2007.06.010

[15] H. Kantz, T. Schreiber, Nonlinear Time Series Analysis, second edition, Cambridge Univ. Press, Cambridge, (2003).

[16]. R. Muñoz, O. Castillo and P. Melin, Optimization of Fuzzy Response Integrators in Modular Neural Networks with Hierarchical Genetic Algorithms: The Case of Face, Fingerprint and Voice Recognition, Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control, **257**, 111-129 (2009).

[17] W. S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, The bulletin of mathematical biophysics, **5** (4), 115–133 (1943).

[18] W. M. Moh, M.-J. Chen, N.-M. Chu and C.-D. Liao, Traffic Prediction and Dynamic Bandwidth Allocation over ATM: A Neural Network Approach, Computer Communications, **18** (8), 563-571 (1995). https://doi.org/10.1016/0140-3664(95)94479-U

[19] C. Looney, Pattern Recognition Using Neural Networks, Oxford Press, Oxford, 1997. ISBN:0-19-507920-5

[20] V. Vemuri and R. Rogers, Artificial Neural Networks: Forecasting Time Series, The IEEE Computer Society Press, Los Alamitos, (1994). ISBN:978-0-8186-5121-2

[21] D. C. Park, M. A. El-Sharkawi and R. J. Marks II, Adaptively Trained Neural Network," IEEE Transac-tions on Neural Networks, **2** (3), 34-345 (1991). DOI:10.1109/72.97910

[22] D. C. Park and T. K. Jeong, Complex Bilinear Recurrent Neural Network for Equalization of a Satellite Channel, IEEE Transactions on Neural Networks, **13** (3), 711-725 (2002). DOI: 10.1109/TNN.2002.1000135

[23] D. C. Park, M. A. El-Sharkawi, R. J. Marks II, L. E. Atlas and M. J. Damborg, Electric Load Forecasting Using an Artificial Neural Network, IEEE Transactions on Power Systems, **6** (2), 442-449 (1991). DOI: 10.1109/59.76685

[24] D. C. Park, Structure Optimization of Bi-Linear Recur-rent Neural Networks and its Application to Ethernet Network Traffic Prediction, Information Sciences, **237**, 18-29 (2013) https://doi.org/10.1016/j.ins.2009.10.005

[25] E. I. Vlahogianni, M. G. Karlaftis and J. C. Golias, Opti-mized and Meta-Optimized Neural Networks for Short- Term Traffic Flow Prediction: A Genetic Approach, Transportation Research Part C, **13** (3), 211-234 (2005). DOI:10.1016/j.trc.2005.04.007

[26] T, Hill, L, Marquezb, M. O'Connor", W. Remusa, Artificial neural network models for forecasting and decision making, International Journal of Forecasting **10** (1), 5-15 (1994). https://doi.org/10.1016/0169-2070(94)90045-0

[27] https://se.mathworks.com/help/nnet/ref/logsig.html, (Žiūrėta 2018-03-20).

[28]. K. Hornik, M. Stinchcombe and H. White, Multilayer feedforward networks are universal approximators. Neural Networks, **2** (5), 359-366. (1989). https://doi.org/10.1016/0893-6080(89)90020-8

[29] T. M.Nabhan and A. Y.Zomaya, Toward generating neural network structures for function approximation, Neural Networks, **7** (1), 89-99 (1994).

[30] T. S. Chang, and K. A. S. Abdel-Ghaffer, A universal neural net with guaranteed convergence to zero system error, IEEE Transactions on Signal Processing, **40** (12), 3022-3031 (1992). DOI: 10.1109/78.175745

[31] T. Masters, Practical Neural Network Recipes in C++, 173-199, Academic Press Profesional, Inc, San Diego, CA, USA, 1993). ISBN:0-12-479040-2

[32] A. Eswaradass, X. H. Sun and M. Wu, "A Neural Network Based Predictive Mechanism for Available Bandwidth," *19th IEEE International Parallel and Distributed Processing Symposium*, 33a-33a (2005) DOI: 10.1109/IPDPS.2005.51

[33] N. Stamatis, D. Parthimos and T. M. Griffith, Forecast-ing Chaotic Cardiovascular Time Series with an Adaptive Slope Multilayer Perceptron Neural Network, IEEE Transactions on Biomedical Engineering, **46** (12), 1441-1453 (1999).

[34] H. Yousefi'zadeh, E. A. Jonckheere and J. A. Silvester, Utilizing Neural Networks to Reduce Packet Loss in Self-Similar Tele Traffic, Proceeding of IEEE Interna-tional. Conference on Communications, **3**, 1942-1946 (2003). DOI: 10.1109/ICC.2003.1203937

[35] C. L. Zhang, Generalized Correlation of Refrigerant Mass Flow Rate Through Adiabatic Capillary Tubes Us-ing Artificial Neural Network, International Journal of Reference, **28** (4), 506-514 (2005). https://doi.org/10.1016/j.ijrefrig.2004.11.004

[36] A. Sencan and S. A. Kalogirou, A New Approach Using Artificial Neural Networks for Determination of the Thermodynamic Properties of Fluid Couples, Energy Conversion and Management, **46** (15-16), 2405-2418 (2005). http://dx.doi.org/10.1016/j.enconman.2004.11.007

[37] H. Esen, M. Inalli, A. Sengur and M. Esen, Artificial Neural Networks and Adaptive Neuro-Fuzzy Assessments for Ground-Coupled Heat Pump System, Energy and Buildings, **40** (6), 1074-1083 (2008). https://doi.org/10.1016/j.enbuild.2007.10.002

[38] D. E. Rumelhart, E. Hinton, and J. Williams, Learning internal representation by error propagation, Parallel Distributed Processing, **1**, 318-362 (1986).

[39] M. B. Kennel, R. Brown, H. D. I. Abarbanel, Determining embedding dimension for phase-space reconstruction using a geometrical construction, Physical Review A, **45** (6), 3403–3411, (1992).

[40] L. Cao, Practical method for determining the minimum embedding dimension of a scalar time series, Physics D: Nonlinear Phenomena, **110** (1 – 2), 43–50, (1997). https://doi.org/10.1016/S0167-2789(97)00118-8

[41] F. Liu, C. Quek and G. S. Ng, Neural Network Model for Time Series Prediction by Reinforcement Learning, Proceedings. IEEE International Joint Conference on Neural Networks, **2**, 809 – 814 (2005). doi: 10.1109/IJCNN.2005.1555956

[42] J. Xue, Z. Shi, Short-Time Traffic Flow Prediction Based On Chaos Time Series Theory, Journal of Transportation Systems Engineering and Information Technology, **8** (5), 68-72 (2008). DOI:10.1016/S1570-6672(08)60040-9

[43] R. Hegger, H. Kantz, T. Schreiber, Practical Implementation Of Nonlinear Time Series Methods: The TISEAN Package, Chaos (Woodbury, N.Y.) **9** (2), 413-435 (1999). DOI: 10.1063/1.166424

[44] B. Henry, N. Lovell, F. Camacho, Nonlinear Dynamics Time Series Analysis, Nonlinear Biomedical Signal Processing: Dynamic Analysis and Modeling **2**, Wiley-IEEE Press, (2010). ISBN: 978-0-7803-6012-9

[45] A. Pavlova, Neuroninių tinklų metodų taikymo interneto srauto prognozei tyrimas, Bakalaurinis darbas, Vilnius (2017).

[46] A. Lengvinas, Regresijos metodų taikymo galimybių interneto srauto anomalijoms tyrimas, Tiriamoji praktika, Vilnius 2018.

[47] http://mathworld.wolfram.com/Levenberg-MarquardtMethod.html, (Žiūrėta 2018-04-28).

[48] https://www.ll.mit.edu/ideval/data/1999data.html, (Žiūrėta 2018-03-11).

[49]. I. Sharafaldin, A.H. Lashkari, and A. A. Ghorbani, Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization, In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), 108-116 (2018). DOI:10.5220/0006639801080116

[50] http://www.unb.ca/cic/datasets/IDS2017.html, (Žiūrėta 2018-03-19).

[51] S. Mansukhani, The Hurst Exponent: Predictability of Time Series, Analytics magazine org, 29 – 31 (2012).

[52] L. Song and P. Bandon, A local stationary long-memory model for internet traffic, 17th European Signal Processing Conference, (2009).

[53] A. Erramilli, O. Narayan, and W. Willinger, Experimental queueing analysis with long-range dependent packet traffic, ACM/IEEE transactions on Networking, **4** (2), 209–223 (1996). DOI: 10.1109/90.491008

[54] P. Joshl, What Is Long Memory In Time Series Analysis, Perpetual enigma perennial fascination with all things tech., (2016), https://prateekvjoshi.com/2016/08/27/what-is-long-memory-in-time-series-analysis/, (2018-02-05).

[55] http://www.caida.org/data/passive/passive_2016_dataset.xml, (2018-05-03).

[56] T. H. D. Ngo, The Box-Jenkins Methodology for Time Series Models, SAS Global Forum Statistics and Data Analysis, 454-466 (2013).

Summary

Andrius Lengvinas

APPLICATION OF NEURAL NETWORKS TO THE ANALYSIS OF INTERNET TRAFFIC ANOMALIES

The Internet is increasingly used in everyday life virtually in all areas: emails, videoconferencing, films, etc. Also the Internet is used in virtual systems ("cloud"). Moreover in past few years growing up “fog” network, which consists of smart home appliances that can be managed online. The Internet gives us "unlimited" capabilities, but there are plenty of malicious files, deceptive websites and cyber-attacks, such as DDoS (Distributed Denial of Service) attacks [2]. The purpose of these attacks is to disrupt the victim's activities by overwhelming targets equipment with a massive packet stream [3]. This kind of attack can be detected by analyzing anomalies in the network stream. In this work, possibility of using ANN (artificial neural networks) to analyze internet traffic anomalies is investigated. NAR (Nonlinear autoregressive neural network) structure NN (neural network) method is tested with the Mackey-Glass data series, which has chaotic features. After analyzing the results of the CICIDS2017 internet traffic data forecast, it was found that the forecast corresponds to most of the real values and represents long term changes. However traffic prediction does not represents traffic spikes. In the course of the study, long-term memory is analyzed by calculating Hurst exponent of internet traffic. Long-term memory describes whether the data sequence values contain encoded information about the previous values. In course of work ANN was compared with other popular methods like ARIMA (autoregressive integrative moving average) and nonlinear dynamics for time series analyze. After evaluating prediction results of different methods was determined that using ANN prediction results was most accurate compared with original data. and ANN predicts more future values accurately

Santrauka

Andrius Lengvinas

INTERNETO SRAUTO ANOMALIJŲ TYRIMAS TAIKANT NEURONINIUS TINKLUS

Internetas vis plačiau naudojamas kasdieniniame gyvenime praktiškai visose srityse: laiškai, vaizdo konferencijos, filmai ir t. t. Taip pat internetas naudojamas ir virtualiose sistemose („cloud“). Reikėtų dar paminėti ir keletą pastarųjų metų populiarėjantį rūko („fog“) tinklą, kurį sudaro mūsų namų buities išmanieji įrenginiai, kuriuos galima valdyti internetu. Internetas mums suteikia „neribotas“ galimybes, tačiau internetinėje erdvėje gausu kenksmingų failų, apgaulingų svetainių ir kibernetinių atakų, pvz.: atsisakymo aptarnauti – DDoS (*Distributed Denial of Service*) atakos [2]. Šiomis atakomis siekiama sutrikdyti aukos veiklą apkraunant jos įrangą didžiuliu paketų srautu [3]. Tokio tipo atakas galima aptikti analizuojat anomalijas interneto sraute. Šiame darbe ištirtos dirbtinių neuroninių tinklų *(artificial neural networks*) taikymo galimybės interneto srauto anomalijoms analizuoti. NAR (Nonlinear autoregressive neural network) struktūros neuroninis tinklas ištestuotas su Mackey-Glass duomenų seka, kuri pasižymi chaotinėmis savybėmis. Išanalizavus CICIDS2017 interneto srauto duomenų prognozės rezultatus, nustatyta, kad prognozė atitinka realius srauto duomenis ar atspindi sekos ilgalaikes tendencijas. Tačiau srauto prognozė neatkartoja trumpų, bet staigų srauto padidėjimų. Tęsiant tyrimą apskaičiuota Hurst eksponentė siekiant išanalizuota ar duomenų sekos vertėse yra užkoduota informacija apie ankstesnes vertes žinoma kaip ilgalaikė atmintis („long-term memory). Darbo metu palygintos neuroninių tinklų pritaikymo galimybes su dažniausiai taikomais ARIMA (autoregressive integrative moving average) ir netiesinės dinamikos metodais, interneto srautui analizuoti. Palyginus skirtingų metodų skaičiavimų rezultatus nustatyta, kad taikant neuroninius tinklus galima prognozuoti daugiau sekos ateities verčių ir gauti prognozės rezultatai yra tiksliausi.

1 priedas „timedelay“ neuroninio tinklo programinis kodas

```
Clear all close all;
y =load('<kelias iki duomenu failo>');

%% data settings
% duomenu padalinas i 4 dalis
y = y(2:end, 2);
N  = length(y);
Nu = abs(N/4);
% prepare training data
input = con2seq(y(1:Nu)');
% prepare validation data
target = con2seq(y(Nu+1:Nu*2)');
% prepare training data
progin = con2seq(y(Nu*2+1:Nu*3)');
% prepare validation data
protar = con2seq(y(Nu*3+1:end)');

%%  Time Delay Neuroninio tinklo kurimas
maxdel = 150;
inputdel = 1:maxdel;
hidden = 10;
net = timedelaynet(inputdel,hidden);
view(net)
%Seka apmokymui
trainput = input;
tratarget = target;
%Seka prognozei
predinput = progin;
predtarget = protar;
%Tinklo paruosimas
[inputs,inputStates,layerStates,targets] = preparets(net,trainput,tratarget);
% Duomenu isdalinimas: Training, Validation, Testing
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
% Tinklo apmokymas
[net,tr] = train(net,inputs,targets,inputStates,layerStates);

% Testavimas
outputs = net(inputs,inputStates,layerStates);
errors = gsubtract(targets,outputs);
performance = perform(net,targets,outputs)

% Perziureti tinklo architektura
view(net)
```

```
Y = net(inputs,inputStates,layerStates);
nets = removedelay(net);
[inputs1,inputStates1,layerStates1] = preparets(net,trainput(1:end-maxdel),tratarget(1:end-maxdel));
[Y1,inputStates2,layerStates2] = net(inputs1,inputStates1,layerStates1);
netc = closeloop(net);
yPred = netc(predinput);
view(netc)
prediction = cell2mat(yPred);
tar = cell2mat(predtarget);
plotpred = prediction(1,maxdel:end);
plottar = tar(1,maxdel:end);
pred = plotpred';
orig = plottar';
view(netc);
figure(1)
plot(pred, 'b')
hold on
plot(orig, 'r')
figure(2)
err = gsubtract(orig,pred);
plot(err, 'g')
```

2 priedas NAR neuroninio tinklo programinis kodas

```
close all, clear all, clc, format compact
%% data path
y =load('<kelias iki duomenu failo>');
% duomenu padalinas i 4 dalis

y = y(2:end, 2);
N  = length(y);
Nu = abs(N/4);
% grafiko prauosimas
plot(y,'r-')
grid on, hold on
plot(y(1:Nu),'k')
set(gcf,'position',[1 60 800 400])
%% duomenu konvertavimas I series tipa
yt = con2seq(y(1:Nu)'); % apmokymo
yv = con2seq(y(Nu+1:Nu*2)'); % patikrinimo
%% zymos
mark1 = Nu;
mark2 = Nu*2;
mark3 = Nu*3;
mark4 = Nu*4;
%% ---------- tinko parametrai ----------
```

```
inputDelays = 1:50;
hiddenSizes = [10];   % network structure (number of neurons)
%% ---------------------------------
% NARNET nonlinear autoregressive neural network
net = narnet(inputDelays, hiddenSizes);
[Xs,Xi,Ai,Ts] = preparets(net,{},{},yt);
net = train(net,Xs,Ts,Xi,Ai);
%% watch learn reults
Yv = cell2mat(yv);
%% Tinklo transformacija I uzdaro tipo
net = closeloop(net);
% uzdaro tipo tinkle atvaizdavimas
view(net);
%% duomenu paruosimas prognozei
yini2 = yt(end-max(inputDelays)+1:end);
[Xs2,Xi2,Ai2] = preparets(net,{},{},[yini2 yv]);
% prognoze
predict2 = net(Xs2,Xi2,Ai2);
% prognozes verciu konvertamas is cell I mat
Yp2 = cell2mat(predict2);
% paklaidos apskaiciavimas
e2 = Yv - Yp2;
%% integravimas
% kintamo vidurkio
windowk = 5;
Yvma = movmean(Yv, windowk);
Ypma = movmean(Yp, windowk);
% comulative average
Yvca = cumsum(Yv)./((1:length(Yv)));
Ypca = cumsum(Yp)./((1:length(Yp)));
%% grafikia
figure(1)
% plot training results
% plot(Nu+1:mark2,Yp,'g') %training rezultatai
plot(Nu+1:mark2,Yp2,'b-o') %prognozes rezultatai
plot(Nu+1:mark2,e2,'g')  %paklaidu grafikas zalia kreive

legend('patikrinimo duomenys','apmokymo duomenys',...
  'prognoze','paklaida','location','southwest')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2)
grid on, hold on
plot(Nu+1:mark2,Yv,'b')
plot(Nu+1:mark2,Yp2,'r')
plot(Nu+1:mark2,Yvma,'k')
plot(Nu+1:mark2,Ypma,'-.k')
plot(Nu+1:mark2,Yvca,'c')
plot(Nu+1:mark2,Ypca,'c-.')
```

3 priedas Hurst eksponentės apskaičiavimo programinis kodas

Kodas pateiktas Python kalba.

```
import pandas as pd
from numpy import cumsum, log, polyfit, sqrt, std, subtract
from numpy.random import randn
from pandas import read_csv
def hurst(ts):
  """Returns the Hurst Exponent of the time series vector ts"""
  # Create the range of lag values
  lags = range(2, 1000)
  # Calculate the array of the variances of the lagged differences
  tau = [sqrt(std(subtract(ts[lag:], ts[:-lag]))) for lag in lags]
  # Use a linear fit to estimate the Hurst Exponent
  poly = polyfit(log(lags), log(tau), 1)
  # Return the Hurst exponent from the polyfit output
  return poly[0]*2.0
data5 = "<kelias iki failo>"
columns = ['time', 'data']
data2 = read_csv(data5, names=columns, delimiter=r"\s+")
#print(data2)
print ("Hurst(data):  %s" % hurst(data2['data']))
```