Vilniaus universitetas
Fizikos fakultetas
Bendrosios fizikos ir spektroskopijos katedra

Anželika Pavlova

NEURONINIŲ TINKLŲ METODŲ TAIKYMO INTERNETO SRAUTO PROGNOZEI TYRIMAS

Pagrindinių studijų baigiamasis darbas
Studijų programa – moderniųjų technologijų fizika ir vadyba

Studentas	Anželika Pavlova
Darbo vadovas	Doc. Feliksas Kuliešius
Recenzentas	Dr. Kęstutis Aidas
Katedros vedėjas	Prof., Dr. (HP) Valdas Šablinskas

Vilnius
2017

Turinys

Įvadas

Neretai tam, kad kompiuterinis tinklas funkcionuotų efektyviai, reikia atlikti srauto prognozę. Vykdant tinklo stebėjimą, analizę bei prognozavimą, galima efektyviai aptikti anomalijas interneto sraute bei atitinkamai pakoreguoti tinklo veikimą arba sustabdyti kenkėjišką veiklą. Vienas iš didžiausią grėsmę kibernetiniam saugumui keliančių reiškinių yra atsisakymo aptarnauti (angl. *denial of service,* DoS) atakos. Iki šiol nėra efektyvaus metodo ankstyvam šios atakos aptikimui, o ankstesnis atakos aptikimas sąlygoja mažesnio masto padarinius. Efektyvi interneto srauto analizė ir prognozė leistų aptikti atakas vos joms prasidėjus – lyginant prognozuojamą ir realų srautą, galima aptikti plika akimi sunkiai pastebimus pokyčius sraute ir pradėti kovą su ataka.

Kursinio darbo [1] metu parodyta, kad netiesinės dinamikos prognozės algoritmas leidžia pasiekti pakankamai tikslią ir ilgalaikę sekos prognozę, jeigu fazinės erdvės rekonstrukcijos parametrai delsa τ ir rekonstrukcijos dimensija m yra parinkti tiksliai. Analizuojant interneto srauto rezultatus buvo pastebėta, jog tradiciniai autokoreliacijos, bendrosios informacijos bei laiko langa metodai nėra universalūs visų chaotinių laiko sekų delsos verčių radimui, tinkamų ilgalaikei bei tiksliai prognozei.

Šiame darbe yra nagrinėjamos įvairios dirbtinių neuroninių tinklų (angl. *artificial neural network*), kurie dar yra vadinami universaliais aproksimatoriais [2], pritaikymo galimybės. Visų pirma, neuroninis tinklas naudojamas chaotinės interneto srauto sekos analizei, siekiant nustatyti trajektorijos fazinėje erdvėje rekonstrukcijos parametrų vertes [3]. Tuomet, naudojant gautas rekonstrukcijos parametrų vertes, interneto srautas prognozuojamas netiesinės dinamikos metodu [4]. Kitas būdas, kuriuo prognozuojama interneto srauto bei chaotinių sekų eiga yra kuomet specifinės konstrukcijos neuroninis tinklas (taip vadinamas laiko delsos neuroninis tinklas (angl. *time delay neural network*) [5, 6] apmokomas taip, kad galėtų tiesiogiai vykdyti laiko sekos prognozę.

Šio darbo tikslas yra ištirti interneto srauto prognozės, naudojant netiesinės dinamikos bei neuroninių tinklų metodus, galimybes, ypatumus ir galimus taikymus. Šiam tikslui pasiekti buvo iškelti šie uždaviniai:

1) ištirti neuroninio tinklo metodo taikymo interneto srauto (chaotinės sekos) trajektorijos fazinėje erdvėje rekonstrukcijos parametrų nustatymui pritaikymo galimybes;
2) ištirti laiko delsos neuroninio tinklo (angl. *time-delay neural network*) panaudojimo galimybes tiesioginiam interneto srauto prognozavimui;
3) pritaikyti interneto srauto prognozės metodiką DDoS atakai detektuoti.

Dalis darbo rezultatų buvo pristatyta 60-oje tarptautinėje studentų fizikos ir gamtos mokslų konferencijoje „Open Readings 2017“.

1. Interneto srauto prognozės taikymo ir metodų apžvalga

1.1 Interneto srauto prognozės taikymo sritys

Adaptyviosios taikomosios programos

Pasaulinis tinklas yra labai heterogeninis kokybės prasme, t.y. skirtingų tinklo segmentų duomenų perdavimo kokybė gali ženkliai skirtis. Vienoje tinklo vietoje gali būti didesnė pralaida ir mažesnis vėlinimas, o kitoje – perdavimo sąlygos daug prastesnės. Adaptyviosios taikomosios programos analizuoja alternatyvius kelius ir parenka tuos, kuriuose perdavimas yra geresnės kokybės, bei pasinaudodamos jais atsisiunčia reikalingus duomenis. Suprantama, kad prieš užmegzdama komunikaciją su nuotoliniu įrenginiu, iš kurio bus atsisiunčiami duomenys, programa turi numatyti, kuris komunikacijos kanalas bus geresnės kokybės [7]. Efektyviai prognozuojant srautą tarp įrenginių porų ir parenkant geresnės kokybės, mažesnę apkrovą turinčius kanalus, galima efektyviau išnaudoti tinklo galimybes.

Srauto grūsčių reguliavimas

Kuomet viena tinklo linija arba viena beviele terpe naudojasi daug vartotojų, neretai dėl spartos ribotumo ir prasto srauto prioritetizavimo susidaro grūstys. Tokiu atveju, įvyksta paketų kolizijos, prarandama dalis srauto arba paketai retransliuojami iš naujo [8], priklausomai nuo to, kokiu protokolu (patikimu – naudojančiu paketo gavimo patvirtinimo sistemą, ar nepatikimu – siunčiančiu srautą be patvirtinimų), buvo siųsta informacija. Paketų retransliavimas yra nenaudingas energetiškai, be to, padidina tikimybę naujoms grūstims susidaryti.

Numatant tikėtinas grūsčių susidarymo vietas ir laiką, galima imtis tokių prevencinių priemonių, kaip dinaminis spartos priskyrimas (angl. *dynamic bandwidth allocation*) [9], jei numatoma, kad srautas netrukus sumažės - taikoma didesnė tolerancija paketų užlaikymams [10], arba vykdoma kitų kelių paieška ir kt.

Tinklo stebėsena bei kibernetinių atakų aptikimas

Paprastai įmonės, eksploatuojančios tinklo įrangą, nuolat stebi srautą realiu laiku bei, jei yra galimybė, kaupia statistinius duomenis. Tokia tinklo stebėsena yra naudojama siekiant aptikti sugedusią įrangą, kuomet srautas nutrūksta, analizuoti tinklo resursų vartojimo statistiką ir aptikti stambioms kibernetinėms atakoms. Tačiau, jei tinklas yra atakuojamas lėta, nedidelio arba lėtai augančio galingumo ataka, skirtumo tarp praeityje užfiksuoto normalaus srauto bei kenkėjiškos veiklos paveikto srauto pamatyti yra beveik neįmanoma. Galingas kibernetines atakas irgi sunku atskirti nuo staigaus legalaus srauto padidėjimo (angl. *flash crowds*).

Kova su kibernetinėmis atakomis susiveda į tris pagrindinius etapus: atakos aptikimą, apsaugą arba slopinimą bei šaltinio atsekimą [11]. Svarbiausias etapas, be kurio neįmanomas kitų

etapų vykdymas, yra aptikimas. Svarbu kuo anksčiau aptikti ataką, nes tuomet yra lengviau su ja susidoroti. Šiame darbe dėmesys ir skiriamas pirmajam etapui – atakos aptikimui.

Atakų aptikimo būdai bendrąja prasme yra skirstomi į taisyklėmis (šablonais) arba anomalijomis paremtus metodus [12, 13].

Taisyklėmis paremti metodai vadovaujasi jau anksčiau vykusių atakų analize: iš jų yra sukuriami kenkėjiško elgesio šablonai, kurie vėliau yra lyginami su įtartinu elgesiu tinkle. Šie metodai yra paprasti, nereikalaujantys daug išteklių ir yra efektyvūs, aptinkant žinomo algoritmo atakas [12]. Tačiau šios taisyklės yra labai griežtos, todėl esant bet kokiai kenkėjo elgesio modifikacijai, šio metodo taikyti nebegalima. Suvokdami tai, DDoS atakų skleidėjai nuolat modifikuoja puolimo algoritmus ir tokiu būdu lengvai apeina tokią apsaugos sistemą.

Anomalijomis paremtas aptikimas yra universalesnis, nei taisyklių metodas, nes iš pradžių yra analizuojami tam tikri normalaus srauto parametrai ir, statistiškai išanalizavus šiuos duomenis bei sukūrus srauto profilį, nukrypimai nuo jo atpažįstami kaip ataka [12, 13]. Anomalijų metodo privalumas yra tai, jog nereikia jokios išankstinės informacijos apie ataką [12], gali būti aptinkamos nežinomos iš anksčiau t.y. naujos atakos. Esant dideliems normalaus srauto pokyčiams apsaugos sistema juos identifikuoja kaip nuokrypį ir generuoja klaidingus pavojaus signalus [13].

Efektyvus įrankis kibernetinėms atakoms, pakeičiančioms srauto elgseną, aptikti yra interneto srauto numatymas. Kuomet fiksuojamas normalus srautas yra išanalizuojamas taip, kad būtų įmanoma numatyti, koks jis bus ateityje, lygindami prognozuojamą srautą su realiu, galime aptikti tinklo apkrovos anomalijas, taip pat ir kibernetines atakas.

1.1.1 DoS ir DDoS atakos

Dažniausios atakos, pakeičiančios tinklo apkrovą ir atitinkamai stebimą srautą, yra DoS bei DDoS atakos. Šių atakų užduotis yra sutrikdyti aukos prieigą prie įrenginių arba visiškai nutraukti jų darbą. Tai realizuojama siunčiant didelius atakos paketų srautus. Tuomet galimos dvi situacijos: prarandamas išorinis ryšys su įrenginiu arba pats įrenginys nustoja veikti, nes neužtenka jo procesoriaus resursų duomenų srautui apdoroti [14]. Paprastai atakos yra vykdomos siekiant išpirkos norint pakenkti konkurencinių įmonių paslaugų tiekimui arba siekiant užmaskuoti sudėtingesnes ir pavojingesnes atakas [12, 15, 16]. Pavyzdžiui, negalėdamas pasinaudoti vienos internetinės parduotuvės tinklalapiu, kuris tuo metu, paveiktas DoS atakos ir negalės būti pasiektas, klientas ieškos alternatyvos prekei įsigyti, praras pasitikėjimą ir nebegrįš į ją.

Vykdant interneto srauto stebėjimą, analizę bei prognozavimą, galima fiksuoti bet kokias anomalijas sraute, bei aptikti įvairias, net ir nežinomo tipo kibernetines atakas.

Yra manoma, jog pirmosios DDoS atakos atsirado 1998 metais [17]. Šios atakos buvo retos, nedidelės spartos, jos veikė auką tiesiogiai, todėl buvo nesunku atsekti šaltinį. Šiuo metu atakos vyksta itin dažnai: anot arbornetworks.com pateikiamos statistikos jos registruojamos

daugiau nei 2000 kartų per dieną, jų sparta siekia iki 500 Gbps [18]. Dabartinės atakos vykdomos labai sudėtingais būdais: pasitelkiami nesusiję su nusikaltėliu įrenginiai ar net jų grandinės tam, kad būtų neįmanoma atsekti tikrojo šaltinio. Tai vykdoma skleidžiant specialius kompiuterinius virusus: programinius kodus, nurodančius įrenginiams elgtis taip, kaip reikia kenkėjui.

DDoS atakų grėsmė yra vis didesnė, nes vis daugiau įrenginių turi prieigą prie interneto. Jei seniau tai buvo tik kompiuteriai ir serveriai, tai šiuo metu į pasaulinį tinklą yra sujungti telefonai, įvairūs išmanieji buities įrenginiai, tokie kaip televizoriai ar net šaldytuvai [19]. Kiekvienas šių įrenginių gali tapti potencialiu DDoS atakos taikiniu arba tarpininku. Pvz., 2016 metų spalio 21-ą dieną buvo įvykdyta didžiausia šiuo metu žinoma DDoS ataka [20]. Jos metu, viena stambiausių kompanijų Dyn, valdančių srities vardų struktūros (angl. *domain name system, DNS*) didžiąją dalį tapo DDoS atakos taikiniu. Atakos metu vartotojai negalėjo pasiekti daugelio svarbių interneto svetainių, tokių kaip Twitter, CNN, Netflix ir t.t. vien dėl to, kad vartotojų įvedami į interneto naršyklės paieškos langelį tinklalapių pavadinimai negalėjo būti susieti su kompiuteriams suprantamais IP adresais, nors patys puslapiai veikė. Toks atakos aukos pasirinkimas įgalina padidinti sukuriamos žalos mastelį, nes yra nukreiptas į daugelį objektų jungiantį tašką. Įdomu yra tai, jog pirminiu atakos šaltiniu tapo daiktų internete paskleistas virusas Mirai. Dyn ekspertų skaičiavimais, net 10000 įrenginių tapo kenkėjiško srauto šaltiniu ir maksimali šio srauto sparta buvo 1 Tbps, kuri yra net du kartus didesnė už prieš tai žinomą didžiausią DDoS ataką [20].

DDoS ataka iš esmės yra itin didelio interneto srauto sukūrimas ir nukreipimas. Jei ataka turi vieną šaltinį, ji yra vadinama DoS ataka. Jei kelis – DDoS [16]. Yra dvi silpnosios vietos, į kurias taiko DDoS atakų iniciatoriai: pilna tinklo pralaidos apkrova arba paties įrenginio procesoriaus resursų išnaudojimas. Pirmuoju atveju, įrenginys netenka prieigos prie interneto, nes neįmanoma išsiųsti arba gauti daugiau paketų, nei leidžia pralaida, ir visą šį resursą išnaudoja atakos srautas. Jei tokio tipo ataka buvo nukreipta į, pavyzdžiui, konkrečios interneto svetainės duomenis talpinantį serverį, vartotojai nebegalėtų pasiekti svetainės. Antruoju atveju, pats įrenginys nustoja veikęs, nes išnaudojamos visos jo procesoriaus ar atmintinės galimybės. Tuomet įrenginys persikrauna arba nustoja veikęs.

DDoS atakos algoritmas yra ganėtinai paprastas. Visų pirma yra surandami išoriniai įrenginiai, kurie turi labai silpną arba neturi jokios apsaugos sistemos. Paprastai tai vykdoma automatizuotai. Šie įrenginiai yra užvaldomi kenkėjo, kuris perduoda jiems programinį kodą, verčiantį įrenginį kurti ir skleisti kenkėjišką srautą [16]. Dar galimas variantas, kuomet kenkėjiškas kodas yra perduodamas įrenginiui tiesiogiai (dažnai per elektroninį paštą) ir vartotojas pats jį priima, nes šis atrodo kaip normalus objektas, pvz., nuotrauka ar tikra legalios programos kopija (tokios programos vadinamos Trojos arkliu). Dažnai vykdydami ataką šie įrenginiai įrašo

siunčiamuose paketuose netikrą šaltinio IP adresą, kuris padeda nuslėpti atakoje dalyvaujančius įrenginius.

1.1.2 Atakų tipai

DDoS atakos paprastai yra skirstomos į dvi plačias kategorijas: antplūdžio (angl. *flooding*) ir logines [21]. Pirmojo tipo atakų metu kenkėjas puola auką dideliu paketų srautu; antruoju atveju yra naudojamas mažesnis srautas, tačiau paketai yra specialiai modifikuojami taip, kad dėl kokių nors aukos programinės įrangos trūkumų, kenkėjiški paketai neleistų programoms normaliai veikti arba nutrauktų jų veiklą [17]. Šiame darbe nagrinėjamos tik antplūdžio atakos, nes logines atakas yra nesunku įveikti taisant programinės įrangos netobulumus arba didinant saugumą (pavyzdžiui pasitelkiant užkardas (angl. *firewall*)).

Antplūdžio tipo atakos yra trijų pagrindinių tipų:

a) *SYN* antplūdžio ataka. Ji išnaudoja TCP pažeidžiamumą. TCP yra vienas plačiausiai naudojamų protokolų vykdant informacijos perdavimą internete. Tokio perdavimo metu yra užmezgamas patikimas ryšys: prieš perduodant vartotojo sukurtą arba jam skirtą srautą, tinklo įrenginiai tarpusavyje užmezga sesiją. Tai yra įvykdoma pasikeičiant specialių paketų seka. Sesiją inicijuojantis kompiuteris siunčia kitam įrenginiui sinchronizacijos užklausą *SYN* (iš angl. *synchronization*). Tokio paketo gavėjas, grąžina tokią pačią *SYN* užklausą bei prie jos prideda iniciatoriaus siųstos *SYN* užklausos gavimo patvirtinimą, vadinamą *ACK* (angl. *acknowledgement*). Tuomet, normalios sesijos metu, kompiuteris turėtų patvirtinti *SYN*/*ACK* paketų gavimą, siųsdamas *ACK* paketą. Būtent *ACK* paketų siuntimas užtikrina patikimą ryšį tarp įrenginių, nes šie nuolat patvirtina gavę paketus. *SYN* antplūdžio atakos metu ryšį inicijuoja kenkėjo kompiuteris ir, gavęs normalų atsakymą iš aukos įrenginio, negrąžina jam *ACK* paketo, o auka tuomet laukia atsakymo, išnaudodama tam tikrą kiekį procesoriaus resursų [17]. Jei tokių ne iki galo užmegztų sesijų yra sukuriama daug, aukos kompiuteris nebepajėgia jų visų apdoroti ir nutrūksta jo veikla.

b) *ICMP* ataka. *ICMP* yra tinklo kontrolės protokolas, kurio vienas iš panaudojimo būdų yra tikrinimas, ar įrenginys yra prijungtas prie tinklo. Tai vykdoma siunčiant atsakymo užklausą (angl. *echo request*), kurios vienintelė paskirtis yra patikrinti ar įrenginys yra prijungtas prie tinklo. Į šią užklausą įrenginys atsako, siųsdamas specialų paketą (angl. *echo reply*), kuriame nėra jokios papildomos informacijos (gali būti laiko žyma), o svarbus tik tas faktas, kad šis paketas buvo išsiųstas. *ICMP* atakos metu, kenkėjas siunčia aukai labai didelį kiekį atsakymo užklausų, kurios,

kartu su atsakymų paketais, užpildo interneto pralaidą ir neleidžia aukai naudoti tinklo pagal paskirtį [17, 22].

c) *UDP* antplūdžio ataka. UDP yra TCP protokolo alternatyva duomenų perdavimui, tačiau skiriasi nuo jo tuo, kad nėra užmezgamas patikimas ryšys bei nėra tikrinamas paketų gavimas (nesiunčiami patvirtinimo paketai *ACK*). *UDP* atakos metu, kenkėjas siunčia *UDP* paketus į įvairius atsitiktinius aukos prievadus. Tuomet aukos kompiuteris ieško, kokia jo vidinė programa naudoją kiekvieną iš tų prievadų ir, kuomet tokių neranda, nes šios užklausos nėra susijusios su jokiais realiais procesais, atsako *ICMP* pranešimais apie tai, kad adresatas nepasiekiamas. Jei tokių netikrų *UDP* užklausų yra siunčiama labai daug, sistema nepajėgia jų visų aptarnauti ir nutraukia veiklą [17, 22].

1.1.3 Metodai DDoS atakoms aptikti

Anomalijų aptikimu paremti atakų aptikimo metodai gali būti skirstomi pagal tai, kokių parametrų anomalijos yra stebimos. Tokie parametrai gali būti paketų gyvavimo trukmė, srauto pasikartojimo dažnis, srauto entropija ir kt.

Kenkėjams labai svarbu, kad jie nebūtų identifikuojami, todėl dažnai DDoS atakų metu yra klastojamos kenkėjiško srauto paketų antraštėse esantys šaltinio IP adresai (angliškai tai vadinama *IP spoofing*). Tam, kad būtų apeinami tam tikri saugos mechanizmai, kurie nepraleidžia srauto iš nepatikimų šaltinių (t.y. iš neįrašytų į patikimų adresų sąrašą), atakos paketų šaltinio adresas pakeičiamas vienu iš patikimų adresų, pavyzdžiui vieno iš klientų adresu [11]. Siekiant apsisaugoti nuo srautų su suklastotais adresais, dažniausiai analizuojamas paketo TTL laukas. Paketas, perduodamas tinklu, turi antraštėje įrašytą žymę, vadinama gyvavimo trukme (angl. *time to live, TTL*), kuri mažėja kaskart paketui pereinant per tinklo įrenginį (t.y. atliekant taip vadinamą šuoliuką (angl. *hop*)). Atitinkamai, kuo iš toliau ateina paketas, tuo didesnis šuoliukų skaičius arba mažesnė likusi gyvavimo trukmė. Yra nustatyta, kad kenkėjui sunku falsifikuoti šią trukmę [23], todėl gavėjas, sudarydamas IP adresų ir gyvavimo trukmių lenteles [11], gali pastebėti, kad iš to paties adreso ateinantys paketai vienu metu pereina daugiau, o kitu mažiau šuoliukų, ir taip nufiltruoti srautą, kurio šaltinio adresas yra suklastotas.

Lėtoms DDoS atakoms (angl. *low rate DDoS attack*), kurios yra ypatingos tuo, kad kenkėjiškas srautas nėra itin stiprus ir yra siunčiamas periodiškai, yra siūloma naudoti spektrinę analizę [11, 24]. Jos metu yra vykdoma originalaus ir esamo srauto autokoreliacijos analizė, parodanti, ar egzistuoja periodiškumas signale. Tuomet, atlikus autokoreliacijos funkcijos Furjė transformaciją, pagal pasikartojimo dažnį atrenkami tam tikri periodiniai signalai, kurie žymi lėtosios DDoS atakos buvimą [11].

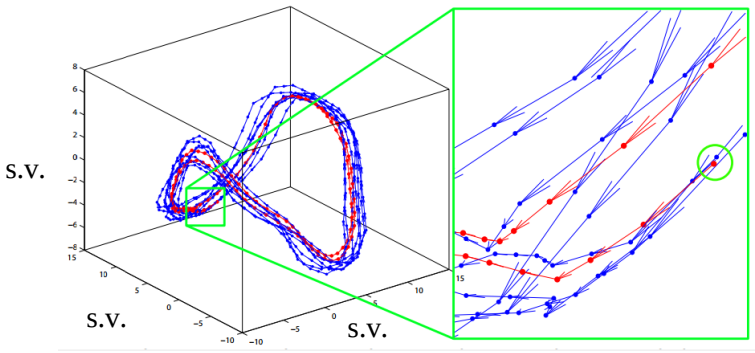
DDoS atakoms aptikti taip pat gali būti naudojama srauto entropijos analizė [11]. Jeigu mums yra žinomi keli galimi įvykiai, kurių tikimybės yra $P\{X = x_1\}, P\{X = x_2\}, \dots, P\{X = x_n\}$, ir tai yra vienintelė informacija, kurią turime apie sistemą, tuomet įvykio baigties „pasirinkimo“ dydžio arba įvykio baigties neapibrėžtumo matas yra vadinamas entropija [25]. Diskretaus kintamojo entropija, kuri gali būti matuojama bitais, yra [11, 25]:

$$H(X) = -\sum_i P\{X = x_i\} \log_2 P\{X = x_i\}, \quad (1)$$

Srauto entropiją (angl. *flow entropy*) galima apibūdinti kaip srauto apibrėžtumą. Kuomet srautas yra normalus (t.y. nevyksta DDoS ataka), srauto entropija yra stabili ir aukšta, tačiau atakos atveju, kuomet sraute dominuoja kenkėjiška dedamoji, srauto entropija ženkliai sumažėja [11]. Tokiu atveju galima detektuoti anomaliją sraute.

1.2 Netiesinės dinamikos metodai chaotinių sistemų elgesio prognozei

Chaotinės sistemos neretai gali būti aprašytos analizinėmis išraiškomis (pvz. šio darbo metu tirta Mackey Glass sistema (žr. 2.1 skyrelį)) ir gali būti vertinamos jas sprendžiant, tačiau chaotinės sistemos yra itin jautrios pradinėms sąlygoms – nedidelis pradinių sąlygų skirtumas ($10^{-7} - 10^{-14}$ eilės), sąlygoja drastiškus sistemos elgesio pokyčius. Dėl pradinių sąlygų nustatymo sudėtingumo ir dėl to, kad neretai analizinės išraiškos yra nežinomos ar net neįmanoma jų nustatyti, chaotinės sistemos dažnai yra nagrinėjamos naudojant statistinius laiko eilučių (angl. *time series*) metodus. Laiko eilutė yra sistemą nusakančių parametrų verčių laikinė priklausomybė (seka). Vienmatė laikinė eilutė yra išskleidžiama į trajektoriją m-matėje fazinėje erdvėje (dažnai vadinama faziniu portretu). Šiai rekonstrukcijai reikalingi du parametrai: delsa ir rekonstrukcijos dimensija. Kiekvienas taškas fazinėje erdvėje yra gaunamas sudarant delsos vektorius (žr. 1.2.1 skyrelį). Trajektorijos fazinėje erdvėje pavyzdys yra pateiktas 1 pav. Chaotinės sistemos atraktorius, skirtingai nuo deterministinės sistemos, yra tam tikras trajektorijų vidurkis, prie kurio artėja taškai, o deterministinės sistemos atveju tai yra griežtai apibrėžta kreivė.

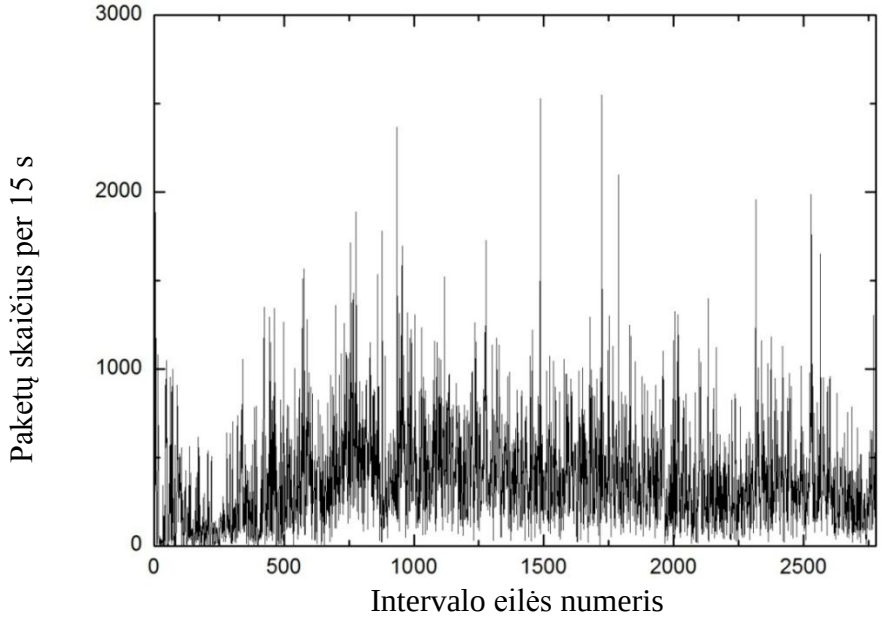


1 pav. Trajektorijos rekonstruotoje fazinėje erdvėje pavyzdys (adaptuota pagal [26]). Rekonstrukcija iš duomenų, užfiksuotų registruojant bėgančio žmogaus parametrus.

1.2.1 Sistemos analizės ir prognozės algoritmas

Šiuo metu yra aktyviai ieškoma efektyvaus būdo interneto srautui prognozuoti tam, kad žinant, koks numatomas normalus srautas, galima būtų detektuoti srauto anomalijas pagal skirtumus tarp prognozuojamo ir realaus srauto.

Interneto srauto neįmanoma aprašyti deterministiniu būdu – jis neturi aiškaus periodo, yra sunkiai nuspėjamas (2 pav.). Tačiau šis srautas nėra visiškai atsitiktinis. Dėl interneto srauto atsikartojamumo (savastiškumo (angl. *self-similarity*)) [27], galima traktuoti šį srautą kaip nestabilią dinaminę sistemą, vadinama chaotine. Pasitelkiant dalies srauto (lokalią), kaip chaotinės sistemos, analizę, galima išnagrinėti lokalų atraktorių (t.y. fazinės erdvės sritį, prie kurios periodiškai artėja kiti taškai), ir pagal jo savybes numatyti, kaip šis srautas keisis ateityje [4].



2 pav. Interneto srauto pavyzdys.

Iš vieno kintamojo laikinės priklausomybės $x(1), x(2), \dots, x(N)$, kurios kiekvienas taškas yra gaunamas fiksuojant stebimą dydį laiko intervalais Δt, galima sudaryti trajektoriją m-matėje fazinėje erdvėje, kuri atvaizduoja sistemos būsenas tam tikrais laiko momentais, aprašomą taip [28]:

$$\begin{matrix} x(1+(m-1)\tau) & \dots & x(i+(m-1)\tau) & \dots & x(N) \\ x(1+(m-2)\tau) & \dots & x\,(i+(m-2)\tau) & \dots & x(N-\tau) \\ \vdots & & \vdots & a & \vdots \\ x(1) & \cdots & x(i) & \cdots & x(N-(m-1)\tau) \end{matrix}, \qquad (2)$$

čia delsa yra $\tau = l\Delta t$, kur l yra sveikas skaičius, o Δt yra laiko intervalas tarp matavimų (diskretizacijos trukmė). Pirmosios problemos, su kuriomis susiduriama taikant šį chaotinės sistemos analizės metodą yra delsos bei rekonstrukcijos dimensijos radimas.

Tiesinės deterministinės sistemos atveju, signalas yra stabilus ir nuspėjamas laike. Skirtingai nuo jos, chaotinės sistemos atveju elgesys yra labai sudėtingas ir ilgainiui nenuspėjamas dėl didelio sistemos jautrumo pradinėms sąlygoms. Atvaizdavus laikinę eilutę kaip trajektoriją rekonstruotoje fazinėje erdvėje, galima stebėti labai artimą trajektorijos taško x_n, grįžimą prie prieš

tai buvusio taško x_n. Šį nedidelį atstumo skirtumą, tarp esamo ir prieš tai buvusio taško, $\Delta_0 = x_n - x_{n'}$ galima laikyti maža perturbacija, kuri laike auga eksponentiškai. Visoms kitoms taškų poroms galima rasti analogišką atstumą $\Delta_l = x_{n+l} - x_{n'+l}$. Jei stebima tendencija, kad $|\Delta_l| \approx \Delta_0 e^{\sigma l}$, tuomet σ – Liapunovo rodiklis, dažniausiai vadinamas tiesiog Liapunovo eksponente [29], išreiškia greitį, kuriuo trajektorijos taškai tolsta vienas nuo kito [27]. Apskaičiavus:

$$S = \langle \ln \left(\frac{1}{|U_n|} \sum_{x_{n'} \in U_n} |s_{n+t} - s_{n'+t}| \right) \rangle_n, \quad (3)$$

jei S kinta tiesiškai, tuomet šios tiesės polinkis yra Liapunovo eksponentė σ [29]. Kuomet $\sigma < 0$, kaimyniniai taškai susijungia į vieną tašką (tai parodo, kad taškai nejuda arba judėjimas yra periodinis). Jei $\sigma > 0$, taškai tolsta vienas nuo kito eksponentiškai, t.y. procesas yra chaotinis [27].

Rekonstruota fazinė erdvė yra sudaroma parenkant tinkamas rekonstrukcijos dimensijos m ir delsos τ vertes [4]. Tuomet kiekvienas erdvės taškas rekonstruotoje erdvėje yra vadinamas delsos vektoriumi ir yra išreiškiamas taip:

$$X(n) = [x(n), x(n-\tau), \dots, x(n-(m-1)\tau)]^T, \quad (4)$$

čia $n = (m-1)\tau + 1, (m-1)\tau + 2, \dots, N$.

Pirmasis prognozės žingsnis yra kaimyninių duotojo taško $X(n)$ taškų paieška rekonstruotoje erdvėje. Tuo tikslu yra surandami euklidiniai atstumai tarp esamojo delsos vektoriaus $X(n)$ ir visų iki jo buvusių $n-1$ delsos vektorių $X(i)$ (čia $i = 1, 2, \dots, n-1$):

$$d(i) = \|X(i) - X(n)\|. \quad (5)$$

Tuomet yra surandami k artimiausių kaimynų $X(n_i)$ (čia $i = 1, 2, \dots, k$). k vertė yra pasirenkama didesnė nei dimensijos vertė m. Trajektorijos rekonstrukcija m-matėje fazinėje erdvėje leidžia nustatyti prognozuojamas ateities taško koordinates X_{pr} taikant tiesinę m delsos vektoriaus elementų superpoziciją:

$$X_{pr}(n+1) = a_0 + \sum_{i=1}^{m} a_i x(n-(i-1)\tau) = \vec{A} Y(n), \quad (6)$$

čia $\vec{A} = [a_0, a_1, \dots, a_m]$, $Y(n) = [1, X(n)^T]^T$. Iš šios lygties akivaizdu, kad turime pasirinkti $k = m + 1$. Deterministiniai spėjimai yra paremti tuo, kad rekonstruojant trajektoriją, yra lyginami dabarties ir ateities taškai, t.y. jei būsena $X(t)$ laiko momentu t yra artima dabarties taške $X(n)$, vadinasi būsena $X(t+1)$ taip pat bus artima ateities taškui $X(n+1)$.

Randami koeficientų vektoriaus $\vec{A}$ nariai, kurie priklauso nuo dabarties taško $X(n)$ padėties bei nuo topologinių kaimynų $X(n_i)$. Vektorių $\vec{A}$ galima rasti iš šios lygybės:

$$\vec{A}\hat{B} = \vec{D}, \quad (7)$$

čia $\hat{B}$ yra $k \times k$ dydžio matrica, kurios i-tasis stulpelis yra sudarytas iš $Y(n_i)$, $\vec{A}$ yra $1 \times k$ dydžio vektorius, o $\vec{D}$ yra $1 \times k$ dydžio vektorius, sudarytas iš $x(n_i + 1)$, t.y.

$$Y(n_i) = [1, x(n_i), x(n_i - \tau), \dots, x(n_i - (m-1)\tau)]^T,$$

$$D = [x(n_1 + 1), x(n_2 + 1), \dots, x(n_k + 1)].$$

Tuomet vektorius $\vec{A}$ apskaičiuojamas taip:

$$\vec{A} = \vec{D}\hat{B}^{-1}. \quad (8)$$

Tada, įrašius gautą vektorių $\vec{A}$ į (6) lygtį, gaunama taško $X_{pr}(n+1)$, sekančio po esamojo $X(n)$ vertė. Vadovaujantis tuo pačiu algoritmu randama tolimesnio taško $X_{pr}(n+2)$, kuris seka paskui ką tik surastą $X_{pr}(n+1)$, vertė ir t.t.

Delsos ir dimensijos vertės, kurios yra tinkamos trajektorijai rekonstruotoje fazinėje erdvėje sudaryti bei artimiausiems kaimynams nustatyti, nebūtinai sutampa su analogiškais parametrais, tinkamais prognozės sudarymui [4].

1.2.2 Metodai delsai nustatyti

Nustatant delsą chaotinės sistemos analizės metu, yra ieškomas toks žingsnio dydis, per kurį paslinkus pradinę seką, gauta nauja ir pradinė sekos būtų silpnai, bet pakankamai koreliuotos.

Svarbu nepasirinkti delsos vertės pernelyg mažos, nes tuomet skirtumas tarp atskirų delsos vektorių yra labai mažas (dėl pertekliškumo ir stipraus sąryšio tarp koordinačių rekonstruota trajektorija būtų suspausta ties fazinės erdvės diagonale), tačiau taip pat svarbu ir nepasirinkti pernelyg didelės vertės, nes tuomet gali nelikti jokios koreliacijos tarp skirtingų koordinačių [4].

Galimi metodai delsai nustatyti yra autokoreliacijos, bendrosios informacijos ir laiko lango metodai [27, 30, 31].

1.2.2.1 Autokoreliacijos metodas

Vienas iš metodų, naudojamų delsos vertei nustatyti yra autokoreliacija. Koreliacijos funkcija (dar vadinama koreliacijos koeficientu) yra ryšio tarp dydžių stiprumo matas. Nustatant delsą, paslinkta per delsos dydžio žingsnį seka $x_{j+\tau}$ yra lyginama su pradine seka x_j (čia $j = 1, 2, \dots N$) ir apskaičiuojama koreliacijos funkcija:

$$R_\tau = \frac{\sum_{j=\tau+1}^{n}(x_j - \overline{x})(x_{j-\tau} - \overline{x})}{\sum_{j=1}^{n}(x_j - \overline{x})^2}, \quad (9)$$

čia $\overline{x}$ yra sekos verčių vidurkis [27]. Parenkama tokia delsos τ vertė, kuriai esant koreliacijos funkcijos vertė lygi arba mažesnė už $1/e$ [27].

1.2.2.2 Bendrosios informacijos metodas

Kitas metodas delsai nustatyti yra bendrosios informacijos (angl. *mutual information*) metodas, kuris, skirtingai nei koreliacijos koeficientas, įvertina ne tik tiesinę, bet ir aukštesnės eilės

koreliaciją tarp kintamųjų [32]. Jei yra du kintamieji, tai matas to, kiek informacijos gali būti nuspėta apie vieną seką, turint visą informaciją apie kitą seką, yra vadinama bendrąja informacija [30]. T.y. bendroji informacija parodo skirtumą tarp realių įvykių tikimybių ir tikimybės, kad šie įvykiai yra tarpusavyje nepriklausomi.

Bendroji informacija $I(K,L)$ tarp pirminės laiko sekos $K = \{x(t_1), x(t_2), \dots, x(t_n)\}$ ir paslinktosios per delsą τ sekos $L = \{x(t_1+\tau), x(t_2+\tau), \dots, x(t_n+\tau)\}$ nurodo, kiek vidutiniškai laiko seka K gali būti prognozuojama, žinant laiko seką L [33]. $I(K,L)$ yra išreiškiama taip:

$$I(K,L) = H(L) + H(K) - H(K,L), \tag{10}$$

čia $H(L)$, $H(K)$ yra atitinkamai L ir K sekų entropijos, $H(K,L)$ yra bendroji entropija tarp K ir L. Šios entropijos yra išreiškiamos taip:

$$H(L) = -\sum_{s=1}^{n} P(L_s)\log_2 P(L_s), \tag{11}$$

$$H(K) = -\sum_{s=1}^{n} P(K_s)\log_2 P(K_s), \tag{12}$$

$$H(L,K) = H(K,L) = -\sum_{i,j} P(K_i, L_j)\log_2 P(K_i, L_j), \tag{13}$$

čia $P(L_s)$ yra s -tojo L sekos nario marginalinė tikimybė, o $P(K_i, L_j)$ yra jungtinė narių kombinacijos tikimybė K_i ir L_j. Įrašius (11), (12) ir (13) į (10), gaunama tokia bendrosios informacijos išraiška:

$$I(K,L) = \sum_{i,j} P(K_i, L_i)\log_2 \frac{P(K_i, L_i)}{P(K_i)P(L_i)}. \tag{14}$$

Delsos vertė, atitinkanti pirmąjį bendrosios informacijos minimumą yra naudojama fazinei erdvei rekonstruoti [30, 33].

1.2.2.3 Laiko lango metodas

Nors autokoreliacijos arba bendrosios informacijos metodu rasta delsos vertė τ garantuoja tai, kad komponentai x_k bei $x_{k+\tau}$ ir komponentai $x_{k+\tau}$ bei $x_{k+2\tau}$ yra nekoreliuoti, tačiau nėra garantijos, kad komponentai x_k bei $x_{k+2\tau}$ taip pat bus nekoreliuoti [31]. Patikslintas metodas, naudojamas delsos vertei nustatyti yra laiko lango (angl. *time window*) metodas, kurio esminė idėja yra tai, kad laiko lango ilgis τ_w susieja rekonstrukcijos dimensiją m ir delsos vertę τ šiuo sąryšiu [31]:

$$\tau_w = m \cdot \tau \tag{15}$$

Laiko lango dydis τ_w iš esmės geometriškai yra asocijuojamas su vidutiniu orbitiniu periodu, t.y. vidutinis laiko tarpas tarp rekonstruotos trajektorijos fazinėje erdvėje taško grįžimo prie atraktoriaus. Paprastumo dėlei, mažos dimensijos chaotinėms sekoms laiko lango ilgis τ_w paprastai yra pasirenkamas kaip vidutinis atstumas tarp lokalių sekos pikų [31]. Atliekant tikslią skaitmeninę chaotinės sekos analizę, lokaliais pikais yra laikomi ir nežymūs ekstremumai, neatspindintys realaus sistemos grįžimo prie atraktoriaus, todėl juos galima laikyti triukšmu. Tam, kad būtų atrenkami tik didžiausi lokalūs pikai, atliekamas signalo filtravimas: pašalinamos mažo intensyvumo aukšto dažnio signalo harmonikos arba parenkamas apytiksliai vieno žymaus piko pločio minimalus atstumas tarp pikų.

1.2.3 Klaidingų artimiausių kaimynų metodas rekonstrukcijos dimensijai nustatyti

Rekonstrukcijos dimensijos vertė m yra randama, pasirenkant tam tikrą kriterijų, kurį turi atitikti atraktoriaus geometrija, bei lyginant, kokiai rekonstrukcijos dimensijos vertei m^* esant šis kriterijus yra išpildomas. Tuomet mažiausia dimensija m^*, atitinkanti šią sąlygą yra naudojama trajektorijos fazinėje erdvėje rekonstravimui [31].

Klaidingų artimiausių kaimynų metodas remiasi idėja, kad esant nedidelei atraktoriaus dimensijai, taškai, atstumas tarp kurių yra mažas, turi išlikti arti ir didėjant dimensijai, tik tuomet jie yra vadinami tikraisiais kaimynais. Ne visi taškai, buvę artimi, esant atraktoriaus geometrinės struktūros projekcijai į mažesnę dimensiją, liks tokie išskleidžiant atraktorių didesnėje dimensijoje [34]. Vadinasi kriterijus, pagal kurį parenkama rekonstrukcijos dimensija yra atstumas tarp kaimynų.

Esant mažai dimensijai d, tarkime, kad taško $x(n)$ r-tasis kaimynas yra $x^{(r)}(n)$, tuomet euklidinis atstumas tarp šių taškų apskaičiuojamas taip:

$$R_d^2(n,r) = \sum_{k=0}^{d-1} [x(n+k\tau) - x^{(r)}(n+k\tau)]^2, \tag{16}$$

čia τ yra delsa. Tuomet, didindami dimensiją tam tikru žingsniu, pavyzdžiui vienetu, kiekvieną kartą apskaičiuojame euklidinį atstumą tarp tų pačių taškų, tik esant didesnei atraktoriaus dimensijai. Jis bus lygus:

$$R_{d+1}^2(n,r) = R_d^2(n,r) + [x(n+k\tau) - x^{(r)}(n+k\tau)]^2. \tag{17}$$

Sąlyga tam, kad tašką laikytume klaidingu kaimynu išreiškiama taip:

$$\left(\frac{R_{d+1}^2(n,r) - R_d^2(n,r)}{R_d^2(n,r)}\right)^{\frac{1}{2}} > R_{sl}, \tag{18}$$

čia R_{sl} yra tam tikras slenkstinis atstumas. Esant tam tikrai dimensijos vertei, klaidingų kaimynų skaičius priartėja prie nulio. Ši dimensijos vertėyra laikoma tinkama fazinei erdvei rekonstruoti [32].

1.2.4 Neuroninių tinklų metodas rekonstrukcijos parametrams nustatyti

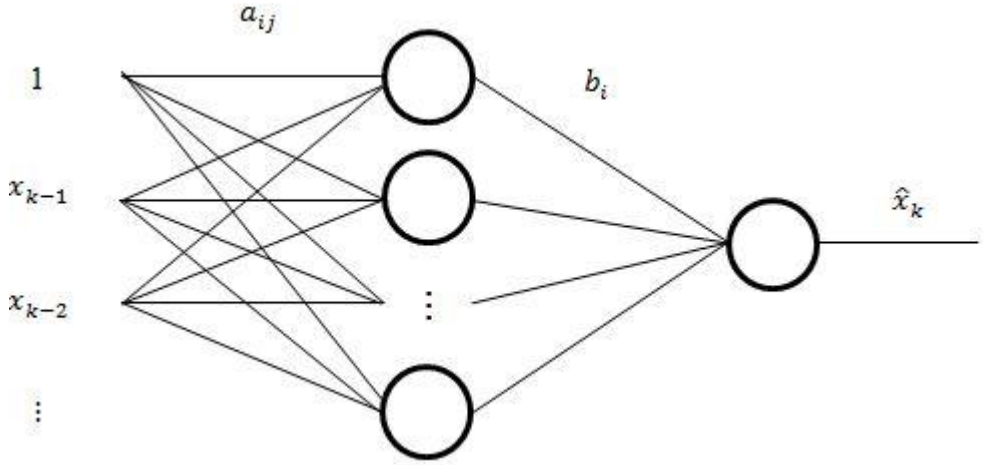
Rekonstrukcijos parametrų nustatymui naudojamas laiko lango metodas įgalina delsos ir rekonstrukcijos dimensijos sandaugos vertės – laiko lango τ_w – nustatymą. Norint naudoti šį metodą vienam iš parametrų rasti, reikia patikimo metodo kitam parametrui nustatyti. Tarkime, norint rasti delsos vertę, turime nustatyti rekonstrukcijos dimensiją. Klaidingų artimiausių kaimynų metodas neleidžia to padaryti patikimai, nes juo dimensija nustatoma jau žinant delsos vertę, kuri įeina į skaičiavimus kaip konstanta (žr. 1.2.3 skyrelį). Naudojant neuroninių tinklų metodą ir analizuojant įvairių delsos verčių indėlį [3], laiko sekos rekonstrukcijos dimensiją galima nustatyti patikimiau, nei naudojant pastovią delsos vertę, nuo kurios nustatymo patikimumo priklauso ir dimensijos vertė. Be to, neuroninis tinklas gali būti naudojamas ir abiems rekonstrukcijos parametrams rasti [35].

Neuroninis tinklas be grįžtamųjų ryšių [1] (angl. *feed-forward neural network*) yra apmokomas (naudojant laiko sekos dalį, paduodamą į tinklo neuronus per laiko delsos žingsnį (pvz. kiekviena vertė atitinka kiekvieną sekančią delsos vertę) tam, kad būtų prognozuojama tolimesnė sekos vertė, naudojant t delsos verčių, kur t yra parinktas pakankamai didelis, siekiant, kad jis atspindėtų sistemos dinamiką, tačiau daug mažesnis nei taškų skaičius sekoje s [3].

Vieno paslėpto sluoksnio neuroninis tinklas be grįžtamųjų ryšių (žr. 3 pav.), kurio paslėptame sluoksnyje yra h neuronų, yra naudojamas laiko sekos x_k sekančios vertės $\hat{x}_k$ radimui pagal formulę:

$$\hat{x}_k = \sum_{i=1}^{h} b_i \tanh\left(a_{i0} + \sum_{j=1}^{t} a_{ij} x_{k-j} \right), \tag{19}$$

kur a_{ij} yra $h \times (t+1)$ dydžio koeficientų matrica, atspindinti ryšių tarp neuroninio tinklo įvesčių ir neuronų svorius, o b_i yra h ilgio vektorius, atspindintis neuronų reikšmę neuroninio tinklo išvesčiai.



3 pav. Vieno paslėptojo sluoksnio neuroninio tinklo be grįžtamųjų ryšių schema.

[1] Plačiau apie neuroninio tinklo veikimą 1.3 skyriuje.

Apmokant tinklą, matricų a ir b narių vertės yra optimizuojamos taip, kad vieno laikinės eilutės žingsnio prognozės vidutinė kvadratinė paklaida, lyginant žinomą sekančią eilutės vertę su neuroninio tinklo gaunama verte, būtų minimali. Kuomet tinklas yra apmokytas, kiekvienos galimos rekonstrukcijos dimensijos vertės svoris yra nustatomas apskaičiuojant dalines neuroninio tinklo išvesties išvestines pagal delsą x_{k-j}:

$$\hat{S}(j) = \frac{1}{s-j} \sum_{k=j+1}^{s} \left| \frac{\partial \hat{x}_k}{\partial x_{k-j}} \right|. \tag{20}$$

(20) lygties išvestinė išreiškiama:

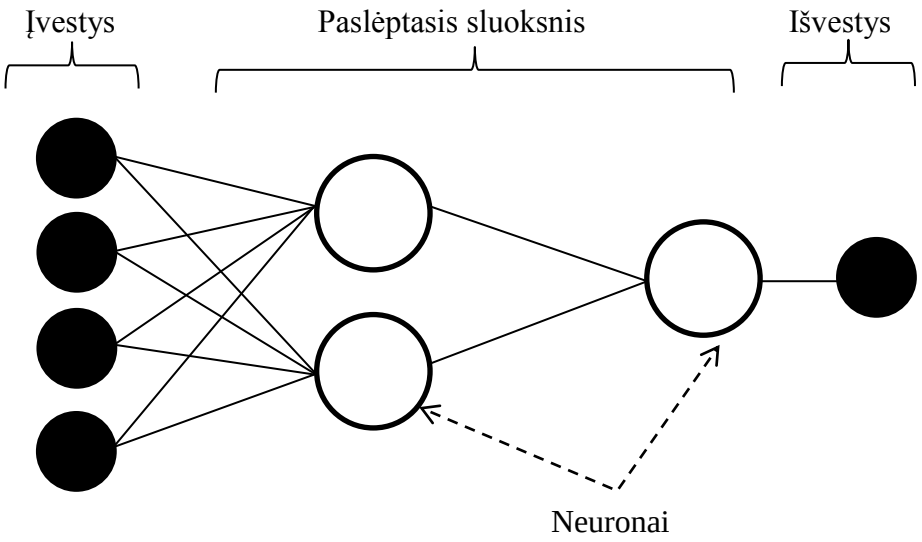
$$\frac{\partial \hat{x}_k}{\partial x_{k-j}} = \sum_{i=1}^{h} a_{ij} b_i \operatorname{sech}^2 \left(a_{i0} + \sum_{n=1}^{t} a_{in} x_{k-n} \right). \tag{21}$$

Optimali rekonstrukcijos dimensija yra laikoma didžiausia *j* vertė, kuriai esant $\hat{S}(j)$ vertė yra pakankamai reikšminga, panašiai kaip ir klaidingų artimiausių kaimynų metode, o individualios $\hat{S}(j)$ vertės (jų moduliai) atspindi kiekvieną delsos vertę [3].

1.3 Neuroninių tinklų metodai

Dirbtiniai neuroniniai tinklai (angl. *artificial neural network*) yra matematinių modelių, paremtų biologinių neuronų santvarka ir funkcijomis, visuma [36]. Dirbtinių neuroninių tinklų matematinės struktūros geba identifikuoti sudėtingus netiesinius ryšius tarp įvesčių ir išvesčių duomenų rinkinių ir yra ypač naudotinos tais atvejais, kai procesų charakteristikas sunku ar neįmanoma aprašyti analizinėmis išraiškomis [37]. Neuroniniai tinklai yra sudaryti iš paprastų elementų, veikiančių lygiagrečiai. Kaip ir žmogaus smegenyse, tinklo veikla stipriai priklauso nuo ryšių tarp šių elementų. Neuroninis tinklas gali būti apmokomas, naudojant tinklo įvestis (angl. *inputs*) ir tikslus (angl. *targets*), keičiant tarpusavio ryšių vertes (svorinius koeficientus, angl. *weigths*) tol, kol tinklo generuojamas rezultatas sutampa su tikslo verte [38].

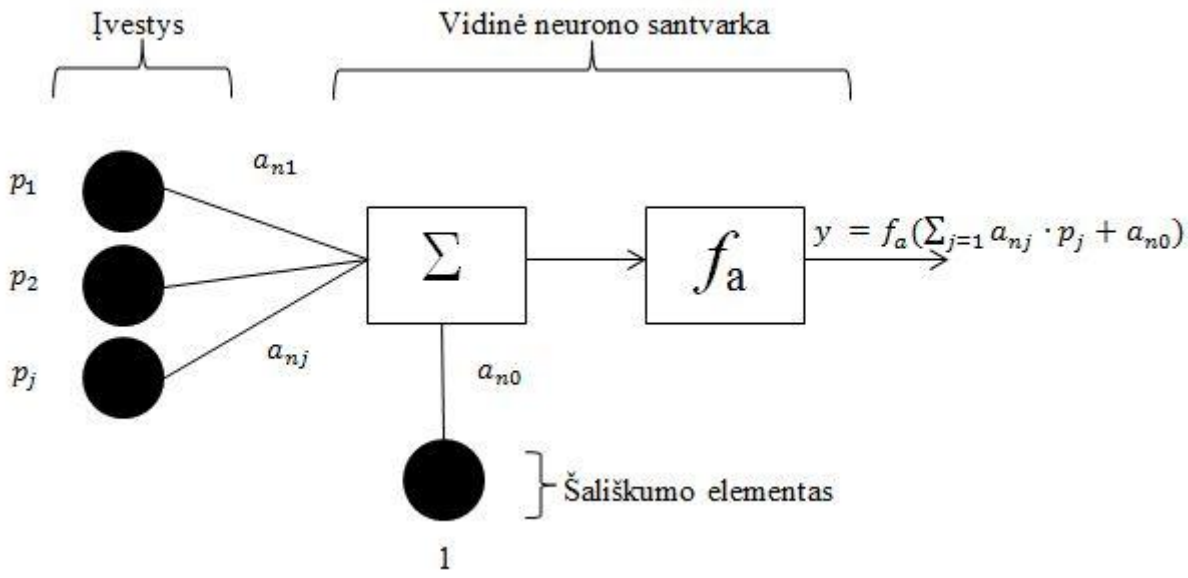
Dirbtinis neuroninis tinklas yra sudaromas iš trijų dalių (žr. 4 pav.): įvesčių dalis, išvesčių dalis ir viduryje tarp jų paslėptasis tinklas, kuriame yra neuronai, kurie gali būti išdėstomi vienu ar keliais sluoksniais. Kiekviena neuroninio tinklo dalis yra sujungiama su sekančiu daugybe jungčių. Tokia architektūra įgalina dirbtinį neuroninį tinklą mokytis sudėtingų sistemos elgsenos modelių [38]. Kiekviena jungtis yra susieta su taip vadinamu svoriniu koeficientu. Paslėptasis sluoksnis yra apmokomas teisingai interpretuoti kiekvieną įvestį. Kiekvieno neurono darbo rezultatas yra taip vadinama aktyvacijos funkcija paveikiama įvesčių, padaugintų iš atitinkamų svorinių koeficientų, siejančių įvestis su šiuo neuronu, suma [39]. Vieno neurono veikimo principas yra pavaizduotas 5 paveiksle.



4 pav. Principinė neuroninio tinklo schema.

Šališkumo elementas (angl. *bias*) yra pridedamas prie kiekvieno neurono įvesties metu (5 pav.). Jis yra reikalingas tam, kad neuroninis tinklas būtų lankstesnis: jei atsitiktų taip, kad visų įvesčių ir svorinių koeficientų sandaugos būtų lygios nuliui, šis neuronas negalėtų prisidėti prie tinklo išvesties rezultato. Siekiant, kad to būtų išvengta ir kiekvienas neuronas galėtų prisidėti prie tikslesnio rezultato kūrimo, yra įvedamas šališkumo elementas, kurio vertė yra lygi 1, o tinklo apmokymo metu kinta jo jungties su neuronu svorinis koeficientas ir taip nustatoma jo įtaka galutiniam neurono darbo rezultatui.

Įvesčių ir šališkumo elemento, sudaugintų su atitinkamais svoriniais koeficientais, suma yra paveikiama matematinės funkcijos, vadinamos aktyvacijos funkcija [38]. Ši funkcija yra reikalinga tam, kad neurono darbo rezultato skaitinė vertė tilptų į tam tikrus rėžius, todėl ji dar vadinama suspaudimo funkcija [40], t.y. kad nepriklausomai nuo įvesčių ir svorinių koeficientų dydžių, visų neuronų išvestys būtų palyginamos. Paprastai yra naudojamos tokios aktyvacijos funkcijos, kaip Hevisaido funkcija, tiesinė ($f_a(x) = x$), sigmoidinė ($f_a(x) = \frac{1}{1+e^{-x}}$), hiperbolinio tangento ($f_a(x) = \frac{2}{1+e^{-2x}} - 1$), Gauso ($f_a(x) = e^{\frac{-x^2}{2\sigma^2}}$). Dažniausiai naudojama hiperbolinio tangento aktyvacijos funkcija [40].



5 pav. Vieno neurono veikimo schema.

Neuroninio tinklo apmokymo algoritmai

Taigi, neuroninis tinklas veikia taip: kiekvienas įvesties elementas sudauginamas su svoriniu koeficientu, susijusiu su kiekvienu neuronu, šios sandaugos yra susumuojamos bei pridedamas šališkumo elementas kiekviename iš neuronų bei paveikiamos aktyvacijos funkcijos. Tuomet šie neuronų rezultatai yra sudauginami su į išvesties neuroną vedančiu svoriniu koeficientu bei yra sudedami ir taip gaunama neuroninio tinklo išvestis.

Tinklo apmokymas reikalingas tam, kad neuroninio tinklo gaunama išvestis būtų kuo artimesnė realioms vertėms [40]. Kadangi neuroninio tinklo įvestys yra atliktų matavimų rezultatai, kurių negalima keisti (ir tai neturėtų prasmės), paklaidai tarp realios ir tinklo gaunamos vertės mažinti yra keičiami svoriniai koeficientai.

Visų pirma, nustatoma kaip matuojama ši paklaida. Dažniausiai yra pasirenkamas mažiausias kvadratinis nuokrypis (angl. *least square error*) [38, 40]:

$$e = \frac{1}{2}\sum_i (y_i - \hat{y}_i)^2, \tag{22}$$

kur y yra reali vertė, o $\hat{y}$ yra neuroninio tinklo išvestis. Paprasčiausias būdas norint rasti tinkamą kiekvieno svorinio koeficiento vertę būtų tiesiog ieškoti, kuriai vertei esant paklaida yra mažiausia. Tačiau šis būdas turi rimtą trūkumą, nes net vienam svoriniam koeficientui įvertinti reikėtų patikrinti, tarkime, 1000 jo verčių, o šių koeficientų net ir paprasčiausiame neuroniniame tinkle yra ne vienas, o $p \cdot n + n$, kur p yra įvesčių skaičius, o n neuronų paslėptajame sluoksnyje skaičius. Taigi toks būdas reikalautų labai didelių kompiuterinių resursų, nes reikėtų patikrinti kiekvieną svorių kombinaciją.

Trajektorija paremti metodai

Norint išvengti sudėtingų skaičiavimų yra pasirenkamas metodas, įgalinantis, netikrinant kiekvienos svorinių koeficientų verčių kombinacijos, rasti paklaidos minimumą. Tokie metodai yra vadinami trajektorija paremtais metodais (angl. *trajectory*-based) [40]. Būtina sąlyga, reikalinga tokiems metodams įgyvendinti yra ta, kad neuronų naudojama aktyvacijos funkcija gali būti diferencijuojama [40]. Šie metodai yra paremti gradiento mažėjimo (angl. *gradient descent*) krypties radimu. Randant funkcijos išvestines, galima nustatyti, kuria kryptimi paklaidų erdvėje reikia judėti nuo esamų svorinių koeficientų verčių, siekiant, kad paklaida būtų mažiausia. Klaidos sklidimo atgal (angl. *backpropagation*), Levenberg–Marquardt, kvazi-Niutono (angl. *quasi-Newton's*) yra trajektorija paremti metodai [40].

Klaidos sklidimo atgal algoritmas yra svorinių koeficientų keitimas didžiausio mažėjimo gradiento kryptimi [41], t.y. ta kryptimi, kur išvestinė yra neigiama. Neuroninis tinklas sugeneruoja išvestį, patikrinamas jos atitikimas realiai vertei, t.y. išsiaiškinama paklaida, ir ji „grąžinama" į paslėptąjį sluoksnį tam, kad būtų pakeičiamos svorinių bei šališkumo koeficientų vertės.

Levenberg–Marquardt ir kvazi-Niutono metodai yra panašūs, tačiau pirmasis reikalauja mažiau skaičiavimo resursų, todėl naudojamas dažniausiai [38]. Levenberg–Marquardt metode naudojamas iteracinis skaičiavimas, kurio pagalba randamas lokalus paklaidos erdvės minimumas [38]. Šis algoritmas, kaip ir kvazi-Niutono metodas, pasitelkiamas greitiems skaičiavimams vykdyti, tačiau nenaudojant Hesiano (antros eilės išvestinių) (būtent ši savybė, kad skaičiuojamos ne antros eilės išvestinės, o tik pirmos ir lemia šio metodo spartą) matricų [38]. Hesiano matrica tuomet aproksimuojama taip:

$$H = \mathrm{J}^{\mathrm{T}}\mathrm{J}, \tag{23}$$

kur J yra Jakobiano matrica, susidedanti iš pirmųjų neuroninio tinklo paklaidų funkcijos išvestinių pagal svorinius ir šališkumo koeficientus [38], iš kurios gradientas gali būti apskaičiuojamas taip:

$$g_w = \mathrm{J}^{\mathrm{T}}e. \tag{24}$$

Naudojant Levenberg–Marquardt algoritmą tuomet apskaičiuojamas toks svorinio koeficiento w_k papildymas:

$$w_{k+1} = w_k - [\mathrm{J}^{\mathrm{T}}\mathrm{J} + \mu\mathrm{I}]^{-1}\mathrm{J}^{\mathrm{T}}e, \tag{25}$$

kur I yra vienetinė matrica, o μ yra konstanta. μ sumažėja kiekvienos sėkmingos paklaidos funkcijos mažinimo iteracijos metu ir didėja, jei iteracija blogina rezultatą [38]. Tokiu būdu (stebint μ kitimą) randamos svorinių koeficientų vertės, kurioms esant paklaida yra mažiausia.

1.3.1 Laiko delsos neuroninių tinklų metodas prognozei vykdyti

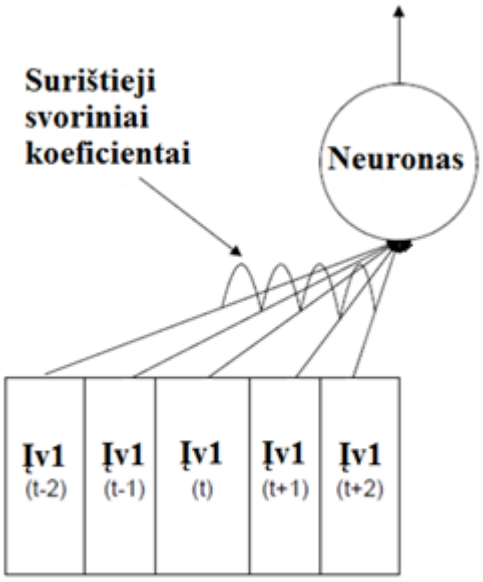
Neuroniniai tinklai gali būti įvairių konstrukcijų: gali skirtis įvesčių ir išvesčių skaičius, paslėptųjų sluoksnių bei neuronų juose skaičius ir kt. Nuo neuroninio tinko naudojimo tikslų gali skirtis jo vidinė sandara. Vienas iš perspektyvių įrankių prognozei vykdyti yra laiko delsos

neuroninis tinklas (angl. *time delay neural network*). Šis neuroninio tinklo tipas skiriasi nuo įprasto neuroninio tinklo be grįžtamųjų ryšių tuo, kad neuroniniam tinklui yra pateikiamos ne tik paskutinė žinoma įvesčių vertė, bet ir kelios aplinkinės vertės [5, 6]. 6 pav. yra parodyta, kaip atrodo tokio neuroninio tinklo įvestis, kuri šiuo atveju nėra vienas taškas, o tai yra laikinė seka.

Į paslėptojo sluoksnio neuroną yra įvedamas ne paskutinis įvesties laikinės sekos taškas, tačiau tam tikras intervalas, apimantis kelias vertes prieš ir po šio taško. Šio intervalo pusplotis (t.y. kiek verčių į vieną pusę nuo taško yra įtraukiama į intervalą) yra taip vadinama delsa. Būtent delsa skiria šį neuroninį tinklą nuo kitų.

Kiekviena uždelsta vertė yra paduodama į neuroninį tinklą per savo atskirą svorinį koeficientą ir atspindi praeities ir ateities taškų įtaką [5]. Taigi, jei turime laikinę seką (6 pav.), tai neuroninio tinklo paslėptojo sluoksnio neuronui pateikiama ne tik paskutinė įvesties sekos Įv(t) vertė, bet ir dvi ankstesnės (Įv(t-1) ir Įv(t-2) ir dvi vėlesnės vertės (Įv(t+1) ir Įv(t+2) (jei delsa lygi dviems).

Visos uždelstosios vertės yra paduodamus per atskirus svorinius koeficientus, tačiau jie yra apriboti tuo, kad vienos įvesties uždelstosios vertės visos turi turėti vienodą svorinį koeficientą, vadinamą surištuoju (angl. *constrained weigth*), kuris dažniausiai yra svorinių koeficientų, gaunamų laikant uždelstas vertes nepriklausomomis, vidurkis [5]. Tokia tinklo architektūra įgalina įvertinti ne tik dabartinės vertės poveikį ateities vertėms, bet ir įvertinti aplinkinių taškų įtaką, o tai leidžia išgauti tikslesnę informaciją iš įvesties verčių.



6 pav. Laiko delsos neuroninio tinklo dalis (adaptuota pagal [5]).

2. Tyrimo metodai

Pirmosios šio darbo dalies metu tipinė chaotinė seka (Mackey Glass) bei interneto srautas buvo modeliuoti naudojant aukščiau aprašytą netiesinės dinamikos metodiką (1.2 skyrius), nustatant rekonstrukcijos parametrus (rekonstrukcijos dimensiją m ir delsą τ) neuroninio tinklo metodu (1.2.4 skyrius). Šiam tikslui pasiekti buvo parašyta programa, pateikta I priede. Nustačius rekonstrukcijos parametrus, tyrimui naudota sekos prognozavimo programa (II priedas). Šio metodo rezultatai buvo palyginti su netiesinės dinamikos metodu vykdomos prognozės rezultatais, kai rekonstrukcijos parametrams nustatyti buvo naudojami bendrosios informacijos, laiko lango bei klaidingų artimiausių kaimynų metodai.

Antrosios šio darbo dalies metu chaotinių sekų bei interneto srauto analizė ir prognozė buvo vykdoma naudojant laiko delsos neuroninio tinklo metodą tiesiogiai. Šiam tyrimui vykdyti parašyta programa yra pateikta III priede.

Darbo metu sukurtos programos buvo parašytos MATLAB kalba, naudojant TISEAN paketo, turinčio įvairių netiesinės dinamikos funkcijų, bei neuroninių tinklų paketo Neural Network Toolbox moduliais.

Netiesinės dinamikos metodas

Naudojant netiesinės dinamikos metodą buvo išskirti šie tyrimo etapai:

1) nustatymas, ar sistema yra chaotinė, įvertinant Liapunov eksponentės vertę;
2) rekonstrukcijos parametrų nustatymas neuroninių tinklų metodu;
3) kiekvieno taško prognozė, pasinaudojant gautais parametrais:
    a) fazinės erdvės sudarymas;
    b) artimiausių k kaimynų fazinėje erdvėje paieška pagal euklidinius atstumus tarp taškų ir jų matricos $\hat{B}$ sudarymas;
    c) kiekvieno sekančio taško, einančio po artimiausių kaimynų vektoriaus $\vec{D}$ sudarymas;
    d) koeficientų vektoriaus $\vec{A}$ radimas.

Paruošus duomenų failą, kuriame yra apskaičiuotas paketų skaičius kiekviename laiko intervale, duomenys analizuojami netiesinės dinamikos metodu. Pirmiausia reikia įsitikinti, ar gauta seka yra chaotinė. Tai atliekama apskaičiuojant Liapunovo eksponentės vertę. Jei ši vertė yra teigiama, tuomet galima teigti, kad sistema yra chaotinė.

Patikrinus sistemos chaotiškumą, įvertinami parametrai, reikalingus tolimesniems skaičiavimams. Šie parametrai yra delsa (angl. *delay*) ir rekonstrukcijos dimensija (angl. *embedding dimension*).

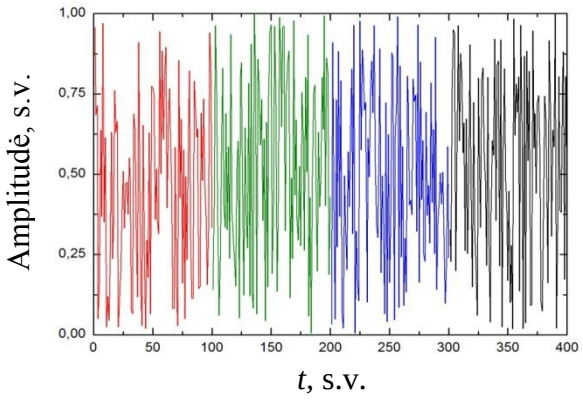
Rekonstrukcijos dimensija yra nustatoma naudojant neuroninių tinklų metodą, aprašytą 1.2.4 skyriuje. Programos (III priedas) pagalba sukonstruojamas toks neuroninis tinklas, kuris parodytas 3 pav., kur paslėptajame sluoksnyje yra 6 neuronai, kurių aktyvacijos funkcija yra hiperbolinis tangentas. Šiuo atveju, kiekvienas laiko eilutės taškas yra paduodamas į neuroninį tinklą kaip atskira įvestis. Tuomet neuroninis tinklas yra apmokomas, surandamos optimalios svorinių koeficientų vertės ir apskaičiuojamos $\hat{S}(j)$ vertės pagal (20) formulę. Pagal $\hat{S}(j)$ vertes nustatomi rekonstrukcijos parametrai – delsa ir rekonstrukcijos dimensija.

Apskaičiavus delsos ir rekonstrukcijos dimensijos vertes, prognozuojamam taškui iš rekonstruotos erdvės (angl. *reconstructed phase space*), kuri yra išreiškiama kaip stačiakampė matrica, kurios dydis yra $n-(m-1)\tau \times m,$ yra sudaromi delsos vektoriai . Minėta matrica yra užpildoma iš duomenų sekos ir atitinka (2) matricą. Sukūrus trajektoriją rekonstruotoje fazinėje erdvėje, kiekvieną tašką galima prognozuoti pagal algoritmą, aprašytą 1.2 skyriuje.

Neuroninių tinklų metodas

Paruošti duomenys suskirstomi į tris dalis. Pirmosios dvi dalys yra naudojamos neuroninio tinklo apmokymui. Šios dalys yra įvesčių seka, kurią neuroninis tinklas naudoja kaip sekos „istorinius“ duomenis, kurių sekančias vertes reikia prognozuoti, bei tikslų seka – po įvesčių sekos sekantys duomenys. Trečioji duomenų dalis yra seka, kuri yra naudojama kaip analogiški įvesčių sekai „istoriniai“ duomenys, pagal kuriuos bus vykdoma prognozė (7 pav.).

Pirmoji sekos dalis (7 pav. raudona kreivė) yra įvedama į neuroninį tinklą taip, kaip parodyta 6 pav. Tada tinklas yra apmokomas Levenberg–Marquardt metodu, t.y. prognozuojant sekančias po pirmosios sekos dalies laiko eilutės vertes ir nuolat keičiant svorinius koeficientus tam, kad būtų sumažinamas mažiausias kvadratinis nuokrypis tarp prognozuojamų ir tikslų sekos (antrosios sekos dalies, 7 pav. žalia kreivė) verčių. Tuomet prognozė yra vykdoma įvedant į apmokytą neuroninį tinklą (t.y. su nustatytomis svorinių koeficientų vertėmis) prognozės įvesčių seką (7 pav. mėlyna kreivė) ir iš jos gaunant tolimesnių taškų prognozę.

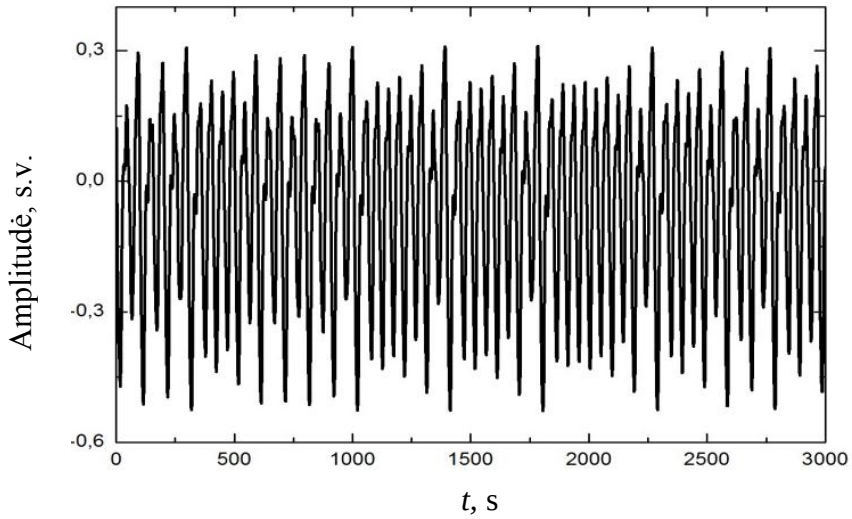


7 pav. Duomenų sekos padalinimas: raudona kreivė – neuroninio tinklo apmokymo įvesčių seka, žalia – apmokymo tikslų seka, mėlyna – prognozės įvesčių seka.

2.1 Tyrimui naudoti duomenys ir jų paruošimas

Siekiant įvertinti pateikiamų metodų patikimumą buvo naudojamos tipinės chaotinių funkcijų sekos.

Šiame darbe interneto srauto prognozei naudojamų metodų patikimumas buvo įvertintas naudojant plačiai žinomos chaotinės sistemos Mackey Glass duomenis. 8 pav. pateikta Mackey Glass chaotinė seka.



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

Mackey Glass chaotinė seka yra aprašoma diskretine lygtimi [42]:

$$x(t+1) = c_m \cdot x(t) \frac{a_m \cdot x(t-d_m)}{b_m + x^{h_m}(t-d_m)}. \quad (26)$$

Šiame darbe buvo naudojama seka, paskelbta Jacobs universiteto tyrėjų [33]. Tikslios (26) lygties parametrų vertės nėra žinomos.

Realaus tinklo srauto analizės įvertinimui buvo naudoti MIT pateikti interneto srauto duomenys [44]. Šie duomenys yra plačiai naudojami įvairiuose panašaus pobūdžio tyrimuose, todėl galima pasinaudoti srauto elementų priskyrimais, interpretacija bei palyginti gaunamus rezultatus.

Interneto srauto duomenų apdorojimas

Gauti interneto srauto duomenų failai buvo pertvarkyti taip, kad duomenys atspindėtų laikinį srauto intensyvumo (apkrovos) pasiskirstymą. Paprastai srautą galima stebėti tokių programų kaip „Wireshark“ pagalba. Iš jų srauto duomenis galima eksportuoti tekstinio failo formatu, kuriame kiekviena eilutė interpretuojama kaip atskiras įrašas ir atspindi individualų srauto paketą. Kiekvienas paketas yra aprašomas šiais parametrais:

a) eilės numeriu,

b) laiko žyme
c) šaltinio IP adresu
d) paskirties IP adresu
e) protokolu
f) paketo informacija ir kt.

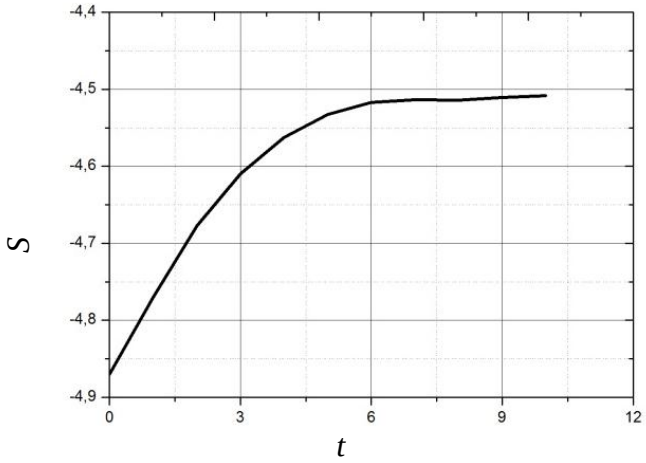
Iš šių parametrų, paliekame tik laiko stulpelį.

Norint apskaičiuoti tinklo apkrovą, t.y. kiek paketų per tam tikrą laiko tarpą fiksuojama, suskaičiuojama kiek įrašų atitinka apibrėžtą laiko tarpą. Programa šiai duomenų failo analizei yra pateikta IV priede (čia laiko intervalas yra pasirinktas 10 sekundžių (*m* žymi eilutės numerį, vektoriuje *M* yra tiek verčių, kiek yra laiko intervalų)).

3. Rezultatai ir jų aptarimas

Darbe pateikiamų metodų patikimumas buvo įvertintas naudojant plačiai žinomos chaotinės sistemos Mackey Glass duomenis.

Sekos chaotiškumas buvo patikrintas nustatant Liapunov rodiklį: jei jis yra daugiau už nulį, vadinasi seka yra chaotinė. Liapunov rodiklio vertė Mackey Glass sekos atveju yra 9 pav. pateiktos kreivės tiesinės dalies, atitinkančios (3) formulę, tangentas. Kadangi šis kampas $\theta \in (0{,}90]$, tai $\mathrm{tg}\theta > 0$, vadinasi seka yra chaotinė.

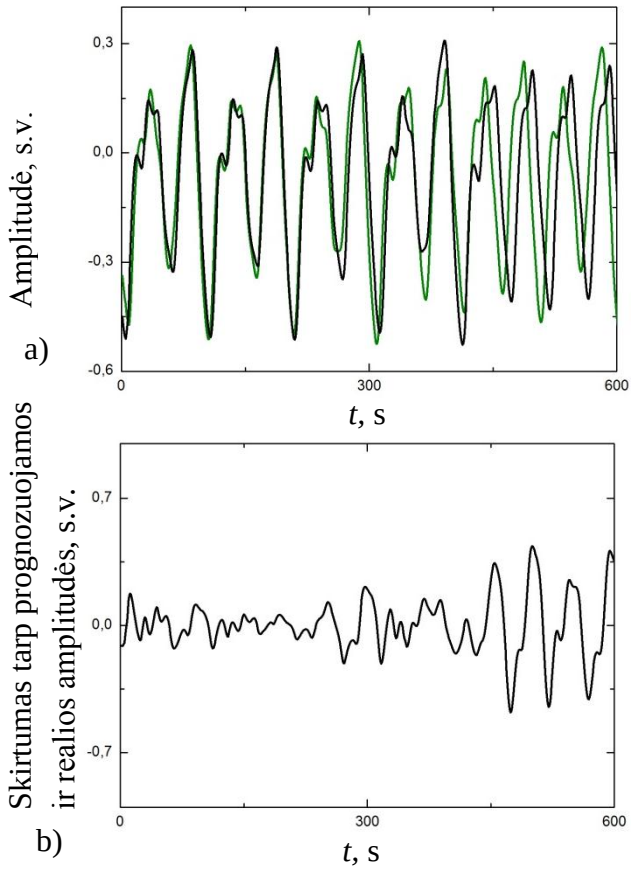


9 pav. Mackey Glass sekos S priklausomybė nuo t.

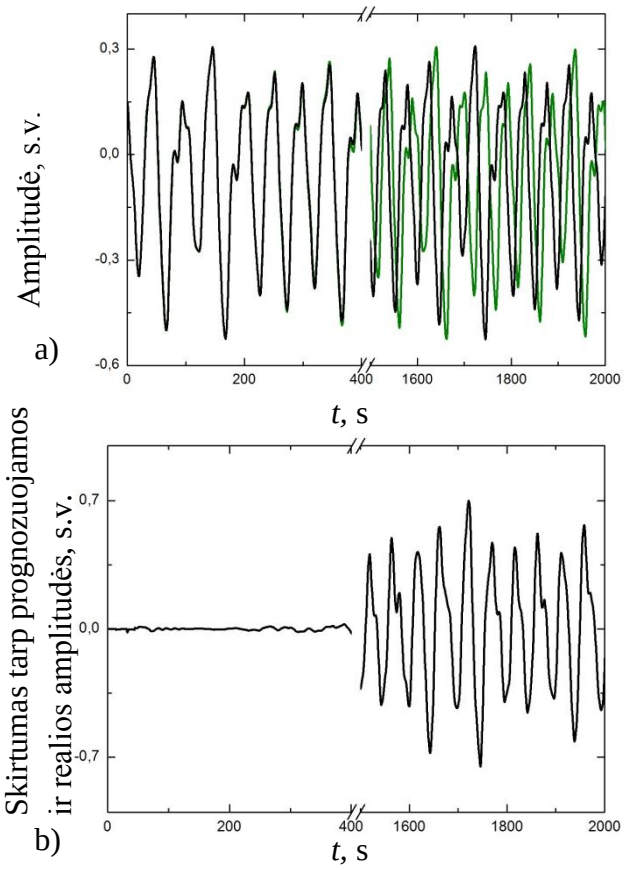
Netiesinės dinamikos metodas

Neuroninio tinklo (NT) metodu nustatyti Mackey Glass sekos rekonstrukcijos parametrai: delsa $\tau = 2$ ir rekonstrukcijos dimensija $m = 18$. Pasitelkiant tokius rekonstrukcijos parametrus buvo atlikta chaotinės sekos prognozė, pavaizduota 10 pav. a) dalyje, bei skirtumas tarp prognozuojamų ir realių verčių pavaizduotas 10 pav. b) dalyje. Prognozė yra pakankamai tiksli 600 taškų.

11 pav. a) yra pateikti Mackey Glass sekos prognozės rezultatai, gauti sekos rekonstrukcijos parametrus nustačius klaidingų artimiausių kaimynų (KAN) ir laiko lango (LL) metodais, kurių rezultatai $\tau = 17$, $m = 3$, 11 pav. b) yra pateiktas skirtumas tarp prognozuojamų ir realių verčių. Šiuo atveju gaunama pakankamai tiksli prognozė 2000 taškų.



10 pav. a) Mackey Glass sekos prognozė (juoda kreivė) ir realūs duomenys (žalia kreivė), (netiesinės dinamikos metodas, τ ir m gauti NT metodu) b) skirtumas tarp realių ir prognozuojamų duomenų.

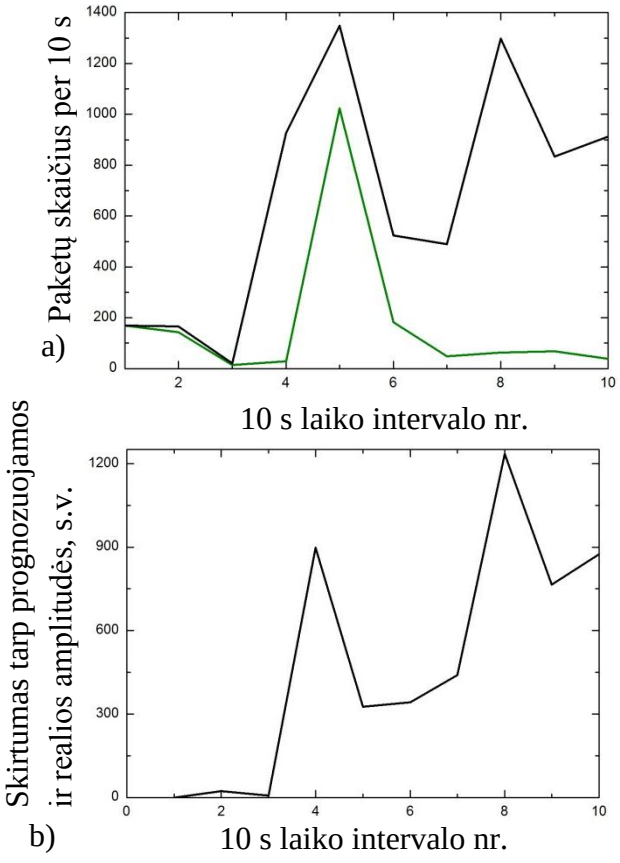


11 pav. a) Mackey Glass sekos prognozė (juoda kreivė) ir realūs duomenys (žalia kreivė), (netiesinės dinamikos metodas, τ ir m gauti LL ir KAN metodais) b) skirtumas tarp realių ir prognozuojamų duomenų.

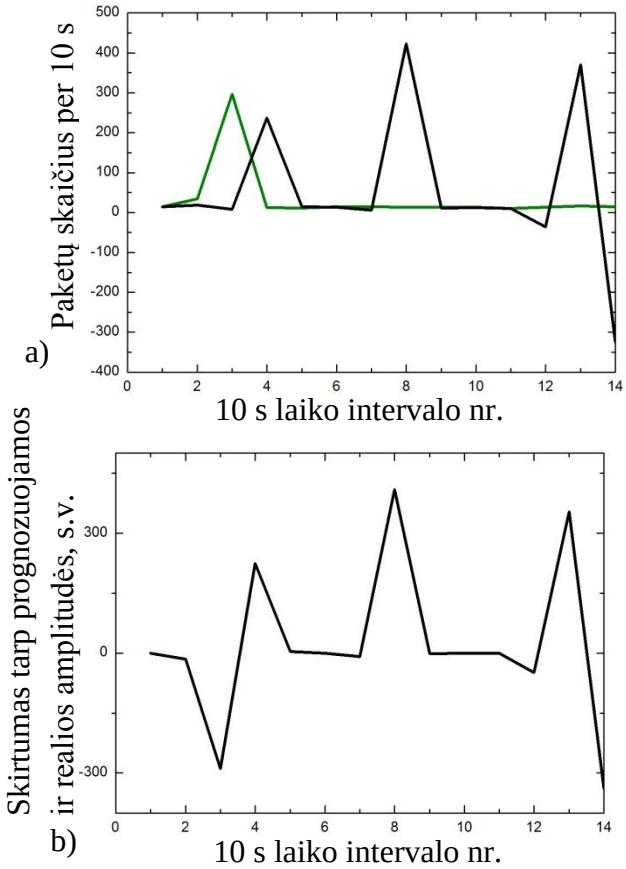
Nors prognozės rezultatai yra gana tikslūs, tačiau rekonstrukcijos parametrų, gautų panaudojant neuroninį tinklą, negalima logiškai paaiškinti.

Pritaikius neuroninio tinklo metodą interneto srauto analizei, gauti rekonstrukcijos parametrai – delsa $\tau = 2$ ir rekonstrukcijos dimensija $m = 3$. Prognozuojant interneto srauto seką, naudojant tokius rekonstrukcijos parametrus, prognozės rezultatai pateikti 12 pav. a) dalyje, o skirtumas tarp prognozuojamų ir realių verčių 12 pav. b) dalyje. Prognozė yra pakankamai tiksli 9 taškams.

13 pav. a) yra pavaizduota interneto srauto sekos prognozė, gauta kursinio darbo metu, kuri buvo gauta randant sekos rekonstrukcijos parametrus klaidingų artimiausių kaimynų (KAN) ir bendrosios informacijos (BI) metodais, kurių rezultatai $\tau = 4, m = 6$, 13 pav. b) yra pavaizduotas skirtumas tarp prognozuojamų ir realių verčių. Šiuo atveju gaunama pakankamai tiksli prognozė 13 taškų.



12 pav. a) Interneto srauto sekos prognozė (juoda kreivė) ir realūs duomenys (žalia kreivė), (netiesinės dinamikos metodas, τ ir m gauti NT metodu) b) skirtumas tarp realių ir prognozuojamų duomenų.



13 pav. a) Interneto srauto sekos prognozė (juoda kreivė) ir realūs duomenys (žalia kreivė), (netiesinės dinamikos metodas, τ ir m gauti BI ir KAN metodais) b) skirtumas tarp realių ir prognozuojamų duomenų.

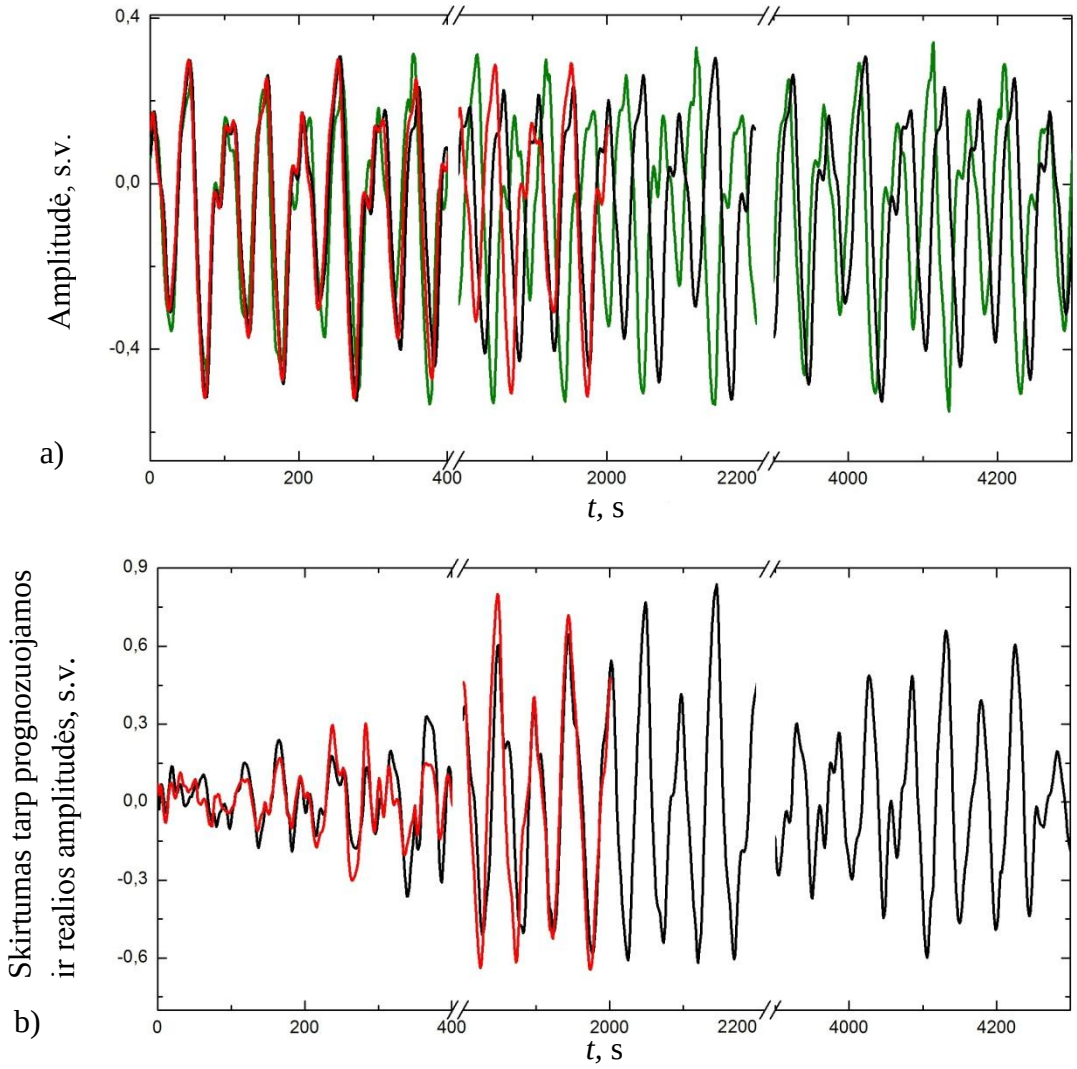
Taigi, neuroninio tinklo metodas chaotinių sekų rekonstrukcijos parametrams nustatyti nėra pakankamai veiksmingas, nes tiek Mackey Glass sekos, tiek interneto srauto atveju, galima prognozuoti mažiau taškų, nei su parametrais, gautais naudojant laiko lango ar bendrosios informacijos metodus delsai rasti bei klaidingų artimiausių kaimynų metodą rekonstrukcijos dimensijai rasti (10 pav. ir 11 pav. a) dalys bei 12 pav. ir 13 pav. a) dalys). Be to, interneto srauto atveju, neuroninio tinklo metodo atveju skirtumas tarp prognozuojamų verčių yra didesnis (12 pav. ir 13 pav. b) dalys).

Laiko delsos neuroninio tinklo prognozės metodas

Tiesioginės laiko delsos neuroninio tinklo prognozės atveju yra realizuojama 4 pav. pavaizduoto neuroninio tinklo tipo architektūra, kur įvestis yra laikinės sekos fragmentas. Įvestis neuronui yra paduodamos kaip tam tikro interalo apie tiriamą tašką duomenys, t.y. taip, kaip pavaizduota 6 pav. Delsos vertė yra parenkama pagal seką siekiant kuo tikslesnės prognozės, delsa apibrėžtas laiko sekos taškų intervalas turi apimti sistemos kitimą atspindinčią dinamiką. Neuronų

skaičius paslėptajame sluoksnyje yra parenkamas pakankamai didelis, tačiau toks, kad skaičiavimų trukmė būtų optimali.Tinklas yra apmokomas Levenberg–Marquardt metodu.

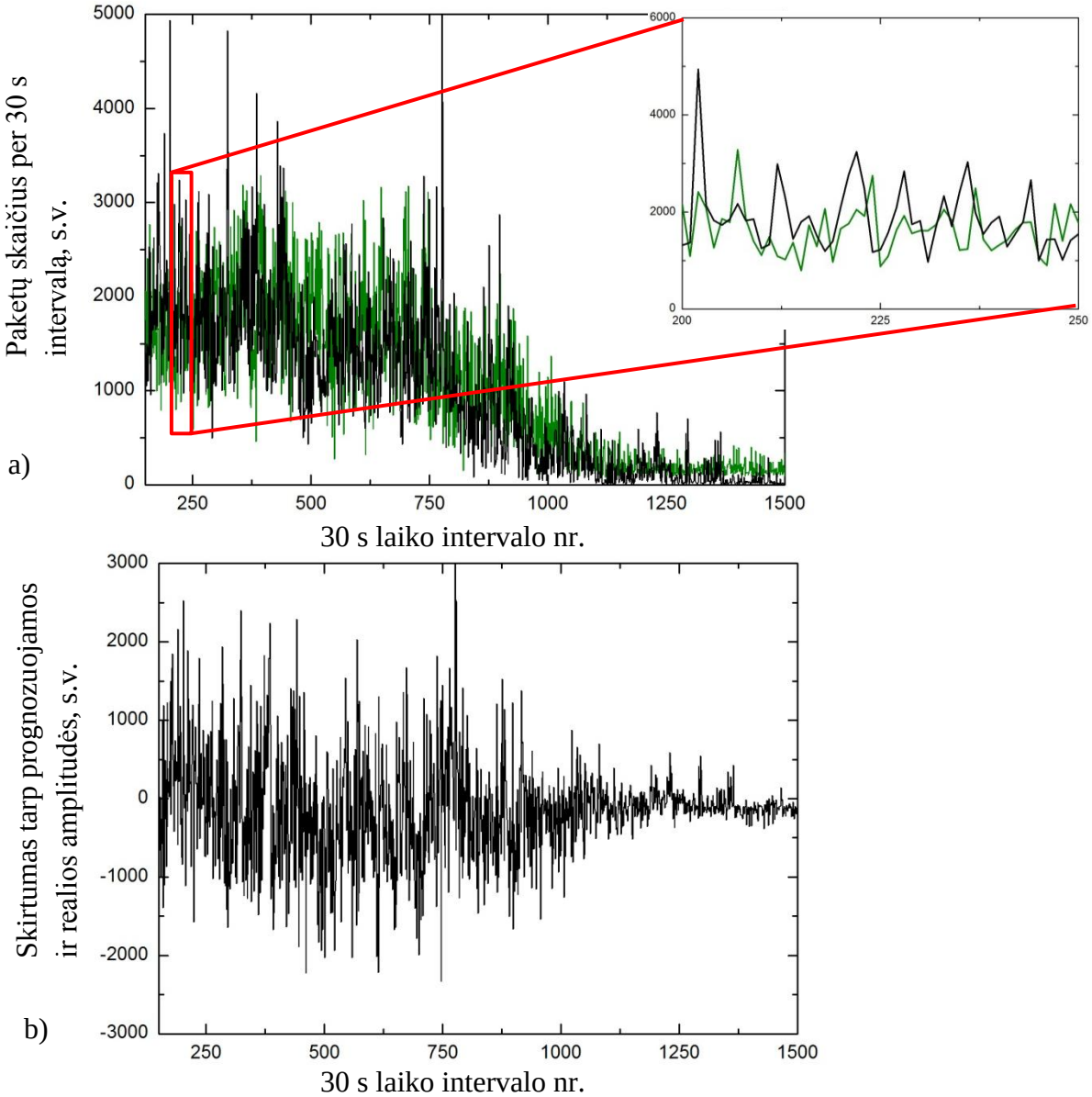
Mackey Glass sekos atveju laiko delsos neuroninis tinklas yra realizuotas naudojant 10 paslėptajame sluoksnyje esančių neuronų, delsos vertė buvo parinkta 100. Neuroninis tinklas yra apmokomas ir gaunama sekos prognozė. Rezultatai pateikti 14 pav. Nustatyta, kad prognozė yra tiksliausia prognozės pradžioje ([1,400] taškai), netiesinės dinamikos metodu gauta prognozė yra vaizduojama iki 2000 taško, nes tolimesnė prognozė praranda savo tikslumą, o laiko delsos neuroninio tinklo metodu gautos prognozės rezultatai yra pakankamai tikslūs iki 4300 taško.



14 pav. Mackey Glass sekos laiko delsos neuroninio tinklo prognozė (juoda kreivė), netiesinės dinamikos metodo prognozė (raudona kreivė) ir realūs duomenys (žalia kreivė) (a) ir skirtumas tarp prognozuojamų ir realių verčių: juoda kreivė laiko delsos neuroninio tinklo metodo atveju, raudona – netiesinės dinamikos metodo atveju (b).

Interneto srauto prognozės atveju laiko delsos neuroninis tinklas buvo sudaromas iš 10 neuronų paslėptajame sluoksnyje, delsos vertė yra 300. Prognozės rezultatas yra pateiktas 15 pav. Iš

viso yra prognozuojama 1500 taškų. Trumpalaikė prognozė nėra tiksli, tačiau atspindi kitimo tendencijas, o ilguoju laikotarpiu prognozės rezultatai pakankamai tiksliai atspindi srauto kitimą.



15 pav. Interneto srauto sekos prognozė (juoda kreivė) ir realūs duomenys (žalia kreivė), kai prognozė yra gauta naudojant laiko delsos neuroninio tinklo metodą (a), bei skirtumas tarp prognozuojamų ir realių verčių (b).

14 ir 15 pav. yra pateikti prognozės rezultatai be apmokymo duomenų.

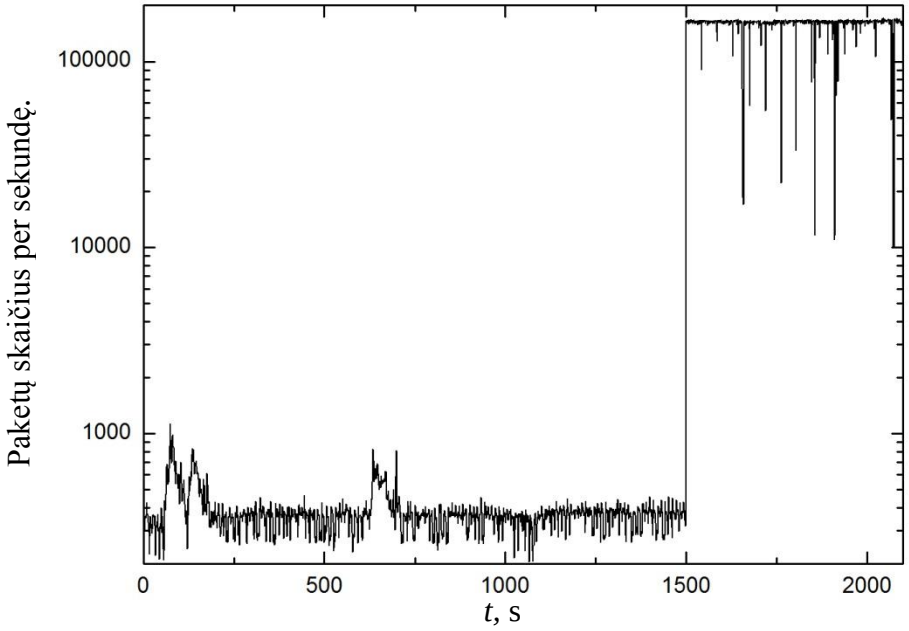
Palyginus šiuos rezultatus su prognoze, gauta netiesinės dinamikos metodu (10 ir 11 pav. lyginant su 14 pav. bei 12 ir 13 pav. lyginant su 15 pav.) nustatyta, kad laiko delsos neuroninio tinklo prognozė yra pakankamai tiksli daug platesniame laiko intervale nei netiesinės dinamikos metodo atveju. Mackey Glass sekos atveju paklaida yra tos pačios eilės kaip ir netiesinės dinamikos atveju, tačiau galima ilgalaikė prognozė. Interneto srauto atveju trumpalaikė prognozė nėra ypač

tiksli, tačiau praktiniams tikslams, tokiems kaip apytikslės tinklo apkrovos prognozė ar kibernetinės atakos aptikimas, gautas ilgojo laikotarpio prognozės tikslumas yra pakankamas [45].

DDoS atakos aptikimas, naudojant interneto srauto prognozės metodiką

Interneto srauto anomalijas, tokias kaip DDoS atakos, kurių metų staigiai ar palaipsniui išauga kompiuterinio tinklo apkrova, dažnai sunku detektuoti tiesiog analizuojant tinklo srauto statistinius duomenis. Efektyvi interneto srauto prognozė leidžia aptikti tokias srauto anomalijas – lyginant prognozuojamą srautą su realiu galima aptikti skirtumą. Netiesinės dinamikos metodas, naudojant įvairius rekonstrukcijos parametrų nustatymo būdus, neįgalino ilgalaikės interneto srauto prognozės, todėl šiame darbe DDoS ataka buvo bandoma aptikti naudojant laiko delsos neuroninio tinklo prognozės metodiką.

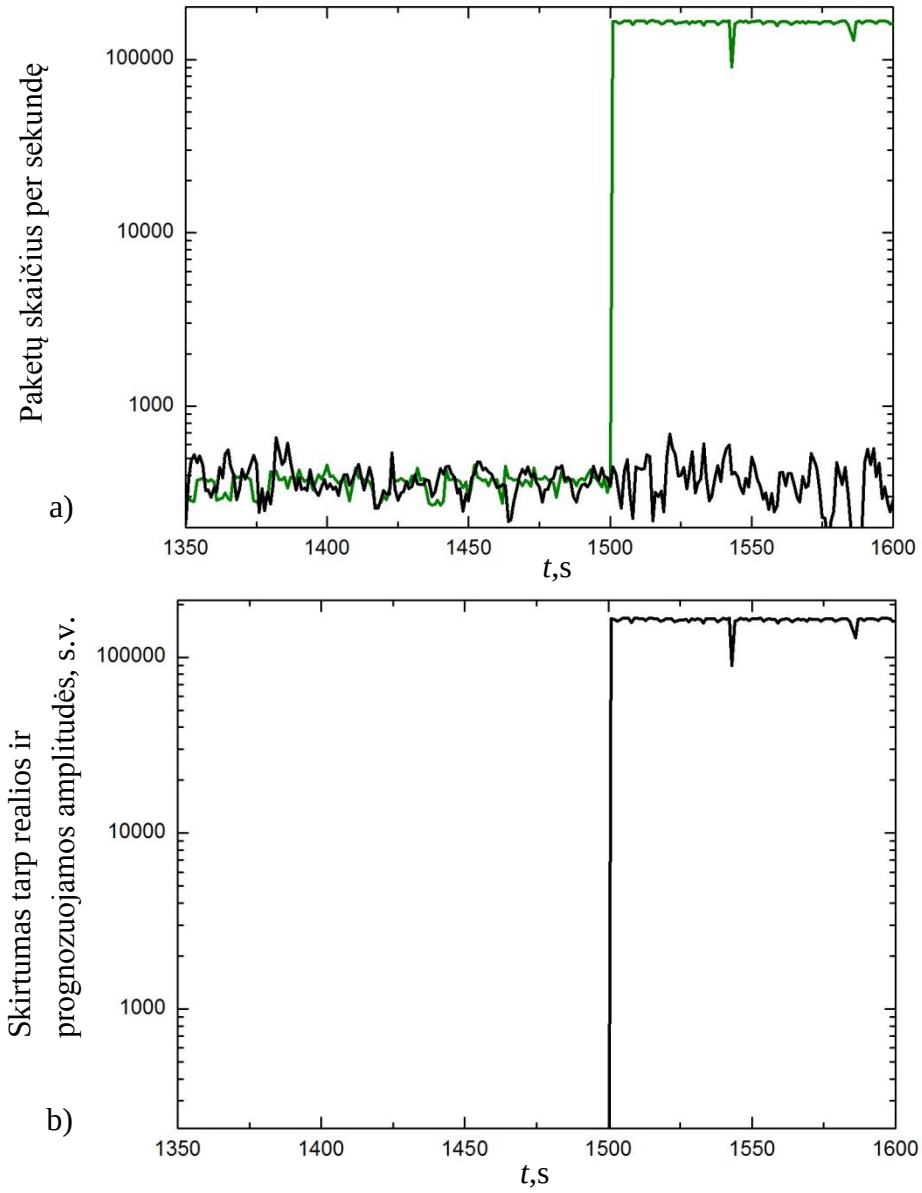
DDoS atakos aptikimo patikrinimui naudoti duomenys yra paskelbti CAIDA (*Center for Applied Internet Data Analysis*) [46]. Nagrinėta DDoS ataka buvo užregistruota 2007 metų rugpjūčio 4-ą dieną. Šiuos duomenis, pateiktus 16 pav., sudaro 1500 sekundžių normalaus interneto srauto (t.y. be atakos), o nuo 1500-osios sekundės iki 2100 yra matoma intensyvi DDoS ataka. Šios atakos tikslas buvo nutraukti prieigą prie duomenų serverio, apkraunant tiek paties serverio procesorių, tiek visą galimą tinklo pralaidą.



16 pav. CAIDA paskelbta DDoS ataka.

DDoS atakos aptikimo, pasinaudojant interneto srauto prognoze, algoritmas yra paremtas registruojamo normalaus srauto (paketų skaičius tam tikrame laiko intervale, pvz. per sekundę), tinkle panaudojimu vėlesnio srauto prognozavimui. Šiame darbe normalaus srauto duomenų dalis buvo panaudota neuroninio tinklo apmokymui tam, kad būtų gauta srauto prognozė. Gauta prognozė yra lyginama su registruojamu srautu. Analizuojant skirtumą tarp prognozės ir realaus srauto, galima pastebėti anomalijas sraute. Optimali delsa šiuo atveju yra 150. Prognozės rezultatai

(be apmokymo duomenų) yra pateikti 17 pav. Lyginant juos su realaus srauto duomenimis, akivaizdu, kad nors trumpalaikė prognozė nėra itin tiksli, kaip ir anksčiau, tačiau ilgalaikės prognozės tikslumas yra pakankamas DDoS atakai nustatyti. Nuo 1500-osios sekundės, prasidėjus DDoS atakai, matomas itin didelis skirtumas tarp realaus ir prognozuojamo srauto, t.y. gali būti registruojama srauto anomalijos pradžia.



17 pav. Interneto srauto su DDoS ataka sekos prognozė (juoda kreivė) ir realūs duomenys (žalia kreivė) (a) bei skirtumas tarp realių ir prognozuojamų verčių (b). Prognozė yra gauta naudojant laiko delsos neuroninio tinklo metodą.

Pagrindiniai rezultatai ir išvados

1. Darbo metu tiriant modelines chaotines sekas nustatyta, kad tradicinis netiesinės dinamikos metodas interneto srauto prognozavimui yra netinkamas, nes nėra universalaus, visoms chaotinėms sekoms tinkamo parametrų trajektorijos fazinėje erdvėje rekonstrukcijai nustatymo metodo.
2. Nustatyta, kad laiko delsos neuroninio tinklo metodo taikymas įgalina pakankamai tikslią ilgalaikę chaotinių sekų bei interneto srauto prognozę.
3. Įrodyta, kad naudojant interneto srauto prognozės rezultatus, gautus taikant laiko delsos neuroninio tinklo metodą, galima sėkmingai nustatyti DDoS atakas.

Literatūra

[1] A. Pavlova. Netiesinės dinamikos metodų taikymo interneto srauto prognozei tyrimas. Kursinis darbas. Vilnius (2017).

[2] K. Hornik, M. Stinchcombe, H. White. Universal Approximation Of An Unknown Mapping And Its Derivatives Using Multilayer Feedforward Networks, Neural Networks **3** (5), 551–560 (1990). DOI: 10.1016/0893-6080(90)90005-6

[3] A. Maus, J. C. Sprott. Neural Network Method For Determining Embedding Dimension Of A Time Series, Communications in Nonlinear Science and Numerical Simulation **16** (8), 3294–3302 (2011). DOI: 10.1016/j.cnsns.2010.10.030

[4] Q. Meng, Y. Peng. A New Local Linear Prediction Model For Chaotic Time Series. Physics Letters A **370**(5-6), 465-470 (2007). DOI: 10.1016/j.physleta.2007.06.010

[5] T. Colin. Speaker Adaptive Phoneme Recognition Using Time Delay Neura Networks, MSc Thesis, Computer Science, School of Computing, National University Of Singapore (2000).

[6] N.A. Charaniya, S.V. Dudul. Focused Time Delay Neural Network Model for Rainfall Prediction Using Indian Ocean Dipole Index, 2012 Fourth International Conference on Computational Intelligence and Communication Networks (CICN) (2012). DOI: 10.1109/CICN.2012.116

[7] M. Stemm, S. Seshan, R. H. Katz, A Network Measurement Architecture For Adaptive Applications, Proceedings of INFOCOM 2000 **1**, 285–294 (2000).

[8] G. W. Lee, S. Y. Lee, E. N. Huh, Congestion Prediction Modeling for Quality of Service Improvement in Wireless Sensor Networks, Sensors (Basel) **14** (5), 7857-7880 (2014). DOI: 10.3390/s140507857

[9] M. R. Joshi, T. H. Hadi. A Review of Network Traffic Analysis and Prediction Techniques, arXiv:1507.05722, 1-23 (2015). DOI: arXiv:1507.05722

[10] V. Jacobson, Congestion Avoidance And Control, Proceedings of the ACM SIGCOMM 1988 **18** (4), 314–329 (1988).

[11] 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

[12] O. Vasilecas, A. Čenys, S. Sosunovas, N. Goranin, Informacinių sistemų sauga: vadovėlis. Vilnius, (2008). ISBN: 978-9955-28-253-2

[13] M. Alenezi, M. J. Reed, Methodologies For Detecting Dos/Ddos Attacks Against Network Servers, The Seventh International Conference on Systems and Networks Communications (2012). ISBN: 978-1-61208-231-8

[14] R. R. Panko, Corporate Computer and Network Security, Second Edition. University of Hawaii, Prentice Hall (2010). ISBN 13: 978-0-13-185475-8

[15] R. J. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, 2nd Edition, Wiley Publishing, Inc., Indianopolis, Indiana (2008). ISBN: 978-0-470-06852-6

[16] J. Mirkovic, P. Reiher, A Taxonomy of DDoS Attack and DDoS Defense Mechanisms, ACM SIGCOMM Computer Communication Review **34** (2) (2004). DOI: 10.1145/997150.997156

[17] A. Srivastava, B.B. Gupta, A. Tyagi, A. Sharma, A. Mishra, A Recent Survey on DDoS Attacks and Defense Mechanisms, Springer Berlin Heidelberg (2011). DOI: 10.1007/978-3-642-24037-9_57, ISBN: 978-3-642-24037-9

[18] https://www.arbornetworks.com/threats/?utm_source=DAM&utm_medium=corp_website&utm_campaign=BRND_0916&utm_term=ALL&utm_content=web_page (žiūrėta 2016-11-22)

[19] H. Kopetz, Internet of Things, Springer US, 307-308 (2011). DOI: 10.1007/978-1-4419-8237-7_13, ISBN: 978-1-4419-8237-7

[20] N. Woolf, DDoS attack that disrupted internet was largest of its kind in history, experts say. The Guardian, San Francisco 2016-10-26.

https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet (žiūrėta 2016-11-15)

[21] B.B. Gupta, R.C. Joshi, M. Misra, Dynamic and Auto Responsive Solution for Distributed Denial-of-Service Attacks Detection in ISP Network, International Journal of Computer Theory and Engineering (IJCTE) **1**(1), 71–80 (2009). DOI: 10.7763/IJCTE.2009.V1.12. ISSN: 1793-821X

[22] B.B. Gupta, R.C. Joshi, M. Misra, Defending against Distributed Denial of Service Attacks: Issues and Challenges. Information Security Journal: A Global Perspective **18** (5), 224–247 (2009). DOI: 10.1080/19393550903317070. ISSN: 1939-3555

[23] H. Wang, C. Jin, K. G. Shin, DEFENSE AGAINST SPOOFED IP TRAFFIC USING HOP-COUNT FILTERING, IEEE/ACM Transactions on Networking **15**(1), 40–53 (2007). DOI: 10.1109/TNET.2006.890133

[24] Y. Chen, K. Hwang. Collaborative Detection And Filtering Of Shrew Ddos Attacks Using Spectral Analysis. Journal of Parallel Distributed Computing **66** (9), 1137–1151 (2006). DOI: 10.1016/j.jpdc.2006.04.007.

[25] К. Э. Шеннон. Работы по теории информации и кибернетике. Издательство иностранной литературы, Москва 1963.

[26] A. Basharat, M. Shah, Time Series Prediction by Chaotic Modeling of Nonlinear Dynamical Systems, IEEE 12th International Conference on Computer Vision (2009). DOI: 10.1109/ICCV.2009.5459429

[27] Z. L. Zhu, C. Fu, A Chaotic Dynamical Model for Internet Traffic. Natural Computation, ICNC '08, Fourth International Conference on Natural Computation (2008). DOI: 10.1109/ICNC.2008.812

[28] 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

[29] 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

[30] 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

[31] D. Kugiumtzis, State Space Reconstruction Parameters in the Analysis of Chaotic Time Series - the Role of the Time Window Length, Physica D: Nonlinear Phenomena **95** (1), 13-28 (1996) DOI:10.1016/0167-2789(96)00054-1

[32] T. K. Torku, Takens Theorem with Singular Spectrum Analysis Applied to Noisy Time Series. Electronic Theses and Dissertations, Paper 3013, 55 (2016) http://dc.etsu.edu/etd/3013 (žiūrėta 2016-10-22)

[33] A. H. Jiang, X. C. Huang, Z. H. Zhang, J. Li, Z. Y. Zhang, H. X. Hua, Mutual Information Algorithms, Mechanical Systems and Signal Processing **24** (8), 2947–2960 (2010). DOI: 10.1016/j.ymssp.2010.05.015

[34] 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).

[35] Y. Manabe, B. Chakraborty. A Novel Approach For Estimation Of Optimal Embedding Parameters Of Nonlinear Time Series By Structural Learning Of Neural Network, Neurocomputing **70** (7-9), p. 1360-1371 (2007). DOI: 10.1016/j.neucom.2006.06.005

[36] T. Hill, L. Marquez, M. O'Connor, W. Remus. Artificial Neural Network Models For Forecasting And Decision Making, International Journal of Forecasting **10** (1), 5-15 (1994). DOI: 10.1016/0169-2070(94)90045-0

[37] K. Hsu, H. V. Gupta, S. Sorooshian. Artificial Neural Network Modeling of the Rainfall-Runoff Process, Water Resources Research **31** (10), 2383-2635 (1995). DOI: 10.1029/95WR01955

[38] S. Chabaa, A. Zeroual, J. Antari. Identification and Prediction of Internet Traffic Using Artificial Neural Networks, Journal of Intelligent Learning Systems and Applications, **2** (3), 147-155 (2010). DOI: 10.4236/jilsa.2010.23018.

[39] 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). DOI: 10.1016/0140-3664(95)94479-U.

[40] S. J. Kwon. Artificial Neural Networks (New York: Nova Science Publishers, 2011), p. 185-201

[41] M.T.Hagan, M. B. Menhaj. Training Feedforward Networks with the Marquardt Algorithm, IEEE Transactions On Neural Networks **5** (6), 989-993 (1994).

[42] M. Farzad, H. Tahersima, H. Khaloozadeh. Predicting the Mackey Glass Chaotic Time Series Using Genetic Algorithm. Conference: SICE-ICASE, International Joint Conference (2006). DOI: 10.1109/SICE.2006.315603

[43] http://minds.jacobs-university.de/sites/default/files/uploads/mantas/code/MackeyGlass_t17.txt (žiūrėta 2017-01-13)

[44] https://www.ll.mit.edu/ideval/data/1999data.html (žiūrėta 2017-01-11)

[45] C. Fachkha, E. Bou-Harb, M. Debbabi. Towards a Forecasting Model for Distributed Denial of Service Activities, 2013 IEEE 12th International Symposium on Network Computing and Applications (2013). DOI 10.1109/NCA.2013.13

[46] The CAIDA UCSD "DDoS Attack 2007" Dataset http://www.caida.org/data/passive/ddos-20070804_dataset.xml (žiūrėta 2017-05-17)

Santrauka

Anželika Pavlova

NEURONINIŲ TINKLŲ METODŲ TAIKYMO INTERNETO SRAUTO PROGNOZEI TYRIMAS

Neretai tam, kad kompiuterinis tinklas funkcionuotų efektyviai, reikia atlikti srauto prognozę. Srauto prognozavimas yra nepakeičiamas įrankis norint optimizuoti tinklo veikimą bei vykdyti įvairių tinklo problemų bei grėsmių, tokių kaip kibernetinės atakos, prevenciją.

Šio darbo tikslas yra ištirti interneto srauto prognozės, naudojant netiesinės dinamikos bei neuroninių tinklų metodus, galimybes, ypatumus ir galimus taikymus.

Šiame darbe interneto srauto prognozė yra vykdoma nagrinėjant interneto srautą kaip chaotinę laiko seką netiesinės dinamikos metodu, lyginant prognozės rezultatus, gautus nustatant trajektorijos atvaizdavimui rekonstruotoje fazinėje erdvėje reikalingus parametrus (delsą τ ir rekonstrukcijos dimensiją m) bendrosios informacijos, klaidingų artimiausių kaimynų, laiko lango metodais bei panaudojant dirbtinį neuroninį tinklą. Be to, šiame darbe interneto srauto prognozė yra vykdoma panaudojant laiko delsos neuroninį tinklą. Abiem atvejais prognozės metodai yra testuojami, panaudojant plačiai žinomą chaotinę seką Mackey Glass.

Darbo metu parašytas programinis kodas MATLAB kalba, įgalinantis nustatyti, ar seka yra chaotinė, nustatant Liapunov rodiklį, rasti trajektorijos rekonstruotoje fazinėje erdvėje sudarymo parametrus (delsą ir rekonstrukcijos dimensiją) bei vykdyti sekos prognozę netiesinės dinamikos metodu. Taip pat buvo parašyta programa, skirta interneto srauto prognozei vykdyti, naudojant laiko delsos neuroninį tinklą (angl. *time-delay neural network*). Darbo metu buvo nagrinėjamos chaotinė laiko seko Mackey Glass, interneto srauto duomenys, paskelbti MIT (*Massachusetts Institute of Technology*) bei DDoS atakos duomenys, paskelbti CAIDA (*Center for Applied Internet Data Analysis*).

Gautų rezultatų analizė parodė, kad netiesinės dinamikos prognozės metodas leidžia pakankamai tikslią chaotinės sekos prognozę iki 2000 taškų, kuomet rekonstrukcijos parametrai nustatomi klaidingų artimiausių kaimynų ir laiko lango metodais, tačiau nustatant rekonstrukcijos parametrus dirbtinio neuroninio tinklo metodu, gaunami parametrai įgalina iki 600 taškų prognozę. Interneto srauto prognozė netiesinės dinamikos metodu yra labai netiksli bei itin trumpalaikė – iki 13 taškų. Laiko delsos neuroninio tinklo prognozės metodas įgalina ilgalaikę ir pakankamai tikslią ilguoju laikotarpiu chaotinės sekos (daugiau nei 4000 taškų) bei interneto srauto prognozę (apie 1500 taškų). Tokio ilgalaikės prognozės tikslumo pakanka aptikti DDoS atakai.

Summary

Anželika Pavlova

RESEARCH OF PREDICTION OF INTERNET TRAFFIC USING METHODS OF NEURAL NETWORKS

Aim of the work: investigation of adequacy of application of methods of nonlinear dynamics and artificial neural network for internet traffic prediction.

Main goals: 1) research on application of methods of nonlinear dynamics for prediction of internet traffic flow, using artificial neural network to determine the reconstruction parameters, 2) research on application of the method of time-delay neural network for prediction of internet traffic flow, 3) application of the internet traffic prediction to detect a DDoS attack.

It is very common that internet traffic prediction is required for the computer network to function effectively: traffic prediction is an irreplaceable tool for network optimization, prevention of malfunctions and attacks.

In this work, internet traffic flow is interpreted as a chaotic system and is predicted using method of nonlinear dynamics, comparing prediction results that were achieved using different methods of obtaining the reconstruction parameters (delay τ and embedding dimension m) required for the prediction. The methods are mutual information, false nearest neighbors, time-window and artificial neural network. Also, internet traffic flow prediction is achieved using the method of time-delay neural network. In all cases, the methods of prediction are tested using a well-known chaotic time series Mackey Glass.

MATLAB code programmed for the purpose of this work allows to determine whether the time series is chaotic by evaluating the Lyapunov coefficient, enables obtaining reconstruction parameters (time delay and embedding dimension) and predicting the series using method of nonlinear dynamics and time-delay neural network. In this work, chaotic time series (Mackey Glass) data and internet traffic flow published by MIT (Massachusetts Institute of Technology) are analyzed. The data of the DDoS attack were published by CAIDA (Center for Applied Internet Data Analysis).

Main results and conclusions: 1) the analysis of the results has shown that the method of nonlinear dynamics for internet traffic prediction is not applicable, since there is no single method of determining the reconstruction parameters suitable for all chaotic systems; 2) time-delay neural network method of prediction allows satisfactory long term prediction accuracy of chaotic time series and internet traffic flow; 3) DDoS attack detection, using internet traffic flow prediction by using time-delay neural network was successful.

I priedas. Neuroninio tinklo metodo rekonstrukcijos parametrams nustatyti programa

```
clear;
load Mackey.dat;
f = Mackey (:,1);

for u = 1:101
   intg(u) = f(u,1);
end

sumazinta = intg' - min(f(:));
norm = sumazinta ./ max(sumazinta(:));

T =20;
for p = 1:T
   input(p) = norm((0+p),1);
end
inputs = input';

for p = 1:1
   target(p) = norm((p+T),1);
end
targets = target';

hidden = 6;
net = newff(minmax(inputs),[hidden,1],{'tansig','purelin'},'trainlm');
net = init(net); %nustato pradinius svoriu parametrus
%    view(net)
net.trainParam.epochs = 500; %kiek epochu apmokys
net.trainParam.show = 50; %kiek parodys
net.trainParam.goal = 0; %error'o siekiama reiksme
net.trainParam.Ir = 0.01; %mokymo konstanta (learning rate)
xx = 0;

[net,tr] = train(net,inputs,targets(1)); %apmokymas
c = length(input);

%svoriniai tinklo koeficientai
weiin = net.IW{1};
weiout = net.LW{2};
biasin = net.b{1};
biasout = net.b{2};

%kaip gaunamas output
z1 = weiin*inputs+biasin;
h1 = tanh(z1);
xx = weiout*h1+biasout;

% tinklo output
A = sim(net,inputs);
```

```
lo = zeros(hidden);
for no = 1:hidden
   lo(no,no) = weiout(no);
end

% S radimas
for mm = 1:20
   h2 = power(sech(z1),2);
   sh = lo*weiin(:,mm);
   S(mm)= abs(sh'*h2);
end

[M,I] = max(S); %I zymes delay
plot(S)
```

II priedas. Netiesinės dinamikos prognozės metodo programa

Fazinės erdvės sudarymo paprogramė:

```
function [y]=fazine(midel, m, x) %rekonstruotos erdves sudarymas
N = length(x);
T = N-(m-1)*midel; %matricos ilgis
y = zeros(T,m); %sukuriam matrica reikiamo dydzio
for c=1:T
   for a=1:m
      y(c,a)=x((m-1)*midel-midel*(a-1)+c,1);
   end
end
end
```

Duomenų nuskaitymas, fazinės erdvės parametrų nustatymas bei prognozės atlikimas (pavyzdyje naudojama Lorenz seka)

```
clear;
tiseanPath = 'C:\TISEAN\';
load Lorenz.dat;
f = Lorenz(:,1);
l=length(f);
pst= 1000;
tsk = 700;
x(:,1)=f(1:end-pst,1);

%Liapunovo koeficiento ivertinimas

system([tiseanPath,'lyap_r -m',num2str(m),' -d',num2str(midel),' -o lyap.dat Lorenz.dat']);
Lya = dlmread('lyap.dat');
save ('Lorenz_Lyapunov.txt','Lya', '-ASCII');


%autokoreliacijos metodas delsai nustatyti
```

```
dr = 30; %kintamasis, zymintis iki kokios delsos vertes skaiciuoti
acf = autocorr(x,dr); %autokoreliacines funkcijos vertes
%ieskome, koks delay, kai autokoeliacija = 1/e
asd = abs(acf-0,368); %ieskoma artimiausia 1/e autokoreliacijos funkcijos vertes
adel = find(asd==min(asd))-1; %atmetama delsa = 0 verte

%bendrosios informacijos metodas delsos paieskai
system([tiseanPath,'mutual -D',(num2str(dr)),' -o mutualinfo.dat Lorenz.dat']);
muti = dlmread('mutualinfo.dat',' ',1,0); % pirmas stulpelis - delsa, antras – bendra informacija
save('Lorenz_MI.txt', 'muti', '-ASCII');

%pirmojo bendrosios informacijos minimumo paieska
siz = length(muti); %nustatom, kiek eiluciu muti vektoriuje
mival = muti(:,2); %sukuriam bendrosios informacijos verciu stulpeli
%ciklas minimuma atitinkanciai delsai rasti
for a = 2:siz
   if mival(a)>mival(a-1)
      midel = muti(a-1,1);
      break
   else
      a=a+1;
   end
end

%False nearest neighbors metodas rekonstrukcijos dimensijais rasti
maxdim = 5; %Iki kokios dimensijos vertes skaiciuoti
system([tiseanPath,'false_nearest -M',num2str(maxdim),' -d',num2str(midel),' -o false.dat
Lorenz.dat']);
fnn = dlmread('false.dat');
save('Lorenz_FNN.txt','fnn','-ASCII');

%ciklas rasti, ties kuria rekonstrukcijos dimensijos verte klaidingu artimiausiu kaimynu skaicius = 0
si = length(16);
for b = 1:si
   if fnn(b,2) < 0.001 %kur fraction nukrenta iki ~0, ten gera dimensija
      m = fnn(b,1);
      break
   else
      b=b+1;
   end
end

%laiko lango metodas delsai nustatyti
[pks, locs] = findpeaks(x);
vatp = mean(diff(locs));
tau = round(vatp/m);

%klaidingu artimiausiu kaimynu metodas rekonstrukcijos dimensijai patikslinti
maxdim = 5;
system([tiseanPath,'false_nearest -M',num2str(maxdim),' -d',num2str(tau),' -o false.dat
Lorenz.dat']);
```

```
fnn = dlmread('false.dat');
%save('Mackey_FNN2.txt','fnn','-ASCII');
si = length(16);
for b = 1:si
   if fnn(b,2) < 0.001 %kur fraction nukrenta iki ~0, ten gera dimensija
      m2 = fnn(b,1);
      break
   else
      b=b+1;
   end
end


%prognoze

for crt=1:tsk  %kiek tasku speja, tiek ir iteruoja
   [y]=fazine(tau, m2, x);
   yy=y(end,:);
   Y=y(1:end-1,:);
   k=m2+1;
   [IDX,Dr]=knnsearch(Y, yy, 'k', k, 'distance', 'euclidean');
   B(1:k,1:k)=1;
   B(:,2:end)=y(IDX(:),:);
   D(:,1)=x((m2-1)*tau+IDX(:)+1,1);
   if det(B) == 0
      C=transpose(D)*pinv(transpose(B));
      x(end+1,1)=C(1,1)+C(1,2:end)*transpose(y(end,:));
   else
      C=B\D;
      x(end+1,1)=C(1,1)+y(end,:)*C(2:end,1);
   end
end
prognoze = f(end-pst:end-pst+tsk,1);
original = x(end-tsk:end,1);

save('Lorenz_Prognoze.txt'prognoze,'-ASCII');
save('Lorenz_Originalas.txt','original','-ASCII');

plot(prognoze, 'r','linewidth',2)
hold on
plot(original, 'b','linewidth',2)
hold off
```

III priedas. Laiko delsos neuroninio tinklo prognozės metodo programa

```
clear;
load ddosIn.txt;
load ddosTa.txt;
load ddosIn2.txt;
load ddosTa2.txt
```

```
input = tonndata(ddosIn,false,false);
target = tonndata(ddosTa,false,false);
progin = tonndata(ddosIn2,false,false);
protar = tonndata(ddosTa2,false,false);

% Time Delay Neuroninio tinklo kurimas
maxdel = 150;
inputdel = 1:maxdel;
hidden =10;
net = timedelaynet(inputdel,hidden);

%Kiek tasku i prieki prognoze
%NN = length(w3d1datapo30)/2;

%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);
%[netc,inputStates3,layerStates3] = closeloop(net,inputStates2,layerStates2);

yPred = netc(predinput);
%[yPred,inputStates4,layerStates4] = netc(predinput,inputStates3,layerStates3);
```

```
prediction = cell2mat(yPred);
tar = cell2mat(predtarget);
plotpred = prediction(1,maxdel:end);
plottar = tar(1,maxdel:end);

pred = plotpred';
orig = plottar';
%view(netc);
semilogy (plotpred)
hold on
semilogy (plottar, 'r')
hold off

save('prognozeDDoS.txt', 'pred', '-ASCII');
 save('originalDDoS.txt', 'orig', '-ASCII');
```

IV priedas. Duomenų paruošimo programa

```
load seka.txt; %failo nuskaitymas
L = seka(:,:);
laikai=L(:,1);
T=10;%intervalas sekundemis

m = 1;
x = 1;
n=0;
irasai = ceil(L(end,1)/T);
M(irasai,1)=0; %sukuriamas duomenu vektorius
for m=1:length(L)   % m zymi eiluciu skaiciu
   if laikai(m,1)>=T*n && laikai(m,1)<T*(n+1)
      M(x,1)=M(x,1)+1;
   else
      n=n+1;
      x=x+1;
   end
end
save('data.txt', 'M', '-ASCII');
```