Određivanje dinamike fizikalnog sustava pomoću interpretabilnih algoritama dubokog učenja

Pavlović, Bartol

Master's thesis / Diplomski rad

2025

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:479192

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-24**

Repository / Repozitorij:

Repository of the Faculty of Science - University of Zagreb

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

Bartol Pavlović

Određivanje dinamike fizikalnog sustava pomoću interpretabilnih algoritama dubokog učenja

Diplomski rad

Zagreb, 2025.

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

INTEGRIRANI PREDDIPLOMSKI I DIPLOMSKI SVEUČILIŠNI STUDIJ
FIZIKA; SMJER ISTRAŽIVAČKI

Bartol Pavlović

Diplomski rad

Određivanje dinamike fizikalnog sustava pomoću interpretabilnih algoritama dubokog učenja

Voditelj diplomskog rada: prof. dr. sc. Davor Horvatić

Ocjena diplomskog rada: _____

Povjerenstvo: 1. _____

2. _____

3. _____

Datum polaganja: _____
Zagreb, 2025.

Zahvaljujem mentoru, prof. dr. sc. Davoru Horvatiću, na izvrsnom vodstvu i podršci tijekom naše suradnje. Također, zahvaljujem Domjanu Bariću, mag. phys. na pruženoj pomoći i nesebičnom dijeljenju znanja.
Posebnu zahvalnost izražavam svojim bližnjima na neprekidnoj podršci i vjeri u mene tijekom studija.

Sažetak

Ovaj diplomski rad istražuje primjenu interpretabilnih algoritama dubokog učenja za modeliranje dinamike populacije neurona koristeći podatke vremenskih nizova. Neuronske simulacije generirane Brian 2 simulatorom služe kao primarni skup podataka, replicirajući ponašanje međusobno povezane mreže neurona u kontroliranim uvjetima. U istraživanju se primjenjuje novi duboki konvolucijski okvir osmišljen za multivarijatne vremenske nizove, s naglaskom na interpretabilnost bez gubitka točnosti predikcije. To se postiže arhitekturom sposobnom za hvatanje aktivnosti pojedinačnih neurona i kolektivne dinamike, kao i pružanje uvida u temeljne mehanizme koji upravljaju tim ponašanjima. Prepoznavanjem najbitnijih značajki filtriranjem signala od šuma i aproksimacijom interakcija zadanim funkcijama, model spaja fleksibilnost dubokog učenja i interpretabilnost klasičnih regresija. Fokusirajući se na Izhikievichev model neurona, rad istražuje sposobnost modeliranja sustava vremenskih nizova i interpretabilnost modela. Rezultati pokazuju da ovakav interpretabilni pristup dubokog učenja može učinkovito analizirati i predvidjeti neuronsku dinamiku, povezujući područja strojnog učenja, neuroznanosti i fizike. Također, razvijene su nove ideje za prilagođavanje izlaza modela promatranom sustavu.

Ključne riječi: dinamika neurona, Izhikievichev model, predviđanje multivarijatnih vremenskih nizova, interpretabilno duboko učenje, interpretabilnost modela, mehanistička interpretabilnost

Determining the dynamics of a physical system using interpretable deep learning algorithms

Abstract

This thesis explores the application of interpretable deep learning algorithms for modeling the dynamics of neuron populations using time-series data. Neural simulations generated with the Brian 2 simulator serve as the primary dataset, replicating the behavior of interconnected neural networks under controlled conditions. The study employs a novel deep convolutional framework for multivariate time series, emphasizing interpretability without sacrificing predictive accuracy. This is achieved through an architecture capable of capturing the activity of individual neurons and collective dynamics while providing insights into the fundamental mechanisms governing these behaviors. By identifying key features, filtering signals from noise, and approximating interactions with predefined functions, the model combines the flexibility of deep learning with the interpretability of classical regression techniques. Focusing on the Izhikevich neuron model, the work examines the capacity of time-series systems to model and interpret the dynamics. The results demonstrate that such an interpretable deep learning approach can effectively analyze and predict neural dynamics, bridging the fields of machine learning, neuroscience, and physics. Additionally, new ideas were developed for adapting the model's outputs to the observed system.

Keywords: neuron dynamics, Izhikevich model, multivariate time series forecasting, interpretable deep learning, model interpretability, mechanistic interpretability

Sadržaj

1 Uvod **1**

2 Modeli neurona i simulacije **3**

2.1 Građa neurona 3

2.2 Membranski potencijal 4

2.3 Generiranje akcijskog potencijala 5

2.3.1 Minimalni mehanizam 5

2.3.2 Hodgkin-Huxley jednadžbe 6

2.4 Integriraj-i-pali model 8

2.5 Izhikievichev model 10

2.6 Dodavanje šuma 11

2.7 Simuliranje mreže neurona - Brian 2 13

3 Osnove dubokog učenja **14**

3.1 Plitke neuronske mreže 14

3.2 Duboke neuronske mreže 15

3.3 Konvolucija 18

4 Vremenski nizovi **21**

4.1 Autoregresivni model 22

4.2 Modeli zasnovani na strojnom učenju 23

5 Duboki konvolucijski interpretator vremenskih nizova **26**

5.1 Dizajn modela 27

5.2 Fokuser i Modelator 29

5.3 Interpretabilnost 33

5.4 Aproksimacije višeg reda 34

6 Određivanje dinamike populacije neurona **37**

6.1 Generiranje i priprema podataka 37

6.2 Pristup modeliranju podataka 39

6.3 Rezultati i interpretabilnost 43

7 Zaključak **46**

Dodaci **48**

A Kod za generiranje podataka **48**

1 Uvod

Ovaj diplomski rad bavi se modeliranjem vremenskih nizova algoritmima dubokog učenja. Proučavani vremenski nizovi su stanja neurona u simuliranoj mreži neurona. Bitno je naglasiti na početku, kako ne bi došlo do zabune, da sljedeće poglavlje nema veze s pojmom neuronske mreže iz dubokog učenja te da je sličnost između proučavanog sustava (simulirana mreža neurona) i korištenog modela za analizu (neuronska mreža) u kontekstu ovog diplomskog rada slučajna.

Neuron ili živčana stanica, osnovna je gradivna jedinica živčanog sustava i ključni element u procesiranju informacija unutar organizma. Njegova složena struktura i funkcionalnost omogućuju integraciju ulaznih signala iz okoline i drugih neurona, njihovu obradu, te prijenos rezultirajućih impulsa prema sljedećim neuronskim jedinicama. Ovaj proces temelji se na koordiniranom djelovanju fizikalnih i kemijskih mehanizama, čime neuroni omogućuju stvaranje složenih mreža koje upravljaju ponašanjem, senzornim percepcijama i kognitivnim funkcijama.

Razumijevanje načina na koji neuroni funkcioniraju zahtijeva proučavanje kako njihove strukture, tako i njihovih dinamičkih svojstava. Strukturalno, neuron se sastoji od tri dijela putem kojih skuplja i prosljeđuje informacije. Funkcionalno, prijenos informacija unutar neurona temelji se na električnim signalima, poznatim kao akcijski potencijali, dok se komunikacija između neurona ostvaruje putem kemijskih spojeva koji se prenose preko sinapsi. Kako bi se olakšalo proučavanje ovih složenih procesa, razvijeni su matematički modeli i simulacije koji opisuju ponašanje neurona. Ovi modeli pružaju uvid u ključne aspekte neuronskih funkcija, poput generiranja akcijskog potencijala, prijenosa signala kroz sinapse i interakcija u neuronskim mrežama.

Umjetna inteligencija i duboko učenje u posljednjih nekoliko godina bilježe iznimno brz razvoj, potaknut napretkom računalne moći, osobito procesora i grafičkih kartica, ali i inovativnim idejama te novim modelima. Ključnu ulogu u ovom napretku imaju velike tehnološke tvrtke poput Nvidije, Googlea (Alphabet Inc.), OpenAI-a i Microsofta. Svaka od ovih organizacija ulaže znatne resurse u razvoj umjetne inteligencije, pri čemu donose vlastite specifične pristupe i vizije budućnosti tehnologije. No, umjetna inteligencija nije ograničena samo na poznate aplikacije poput chatbotova, generatora slika ili autonomnih vozila. Njena primjena značajno je proširena

na područja koja uključuju medicinu, kemiju, biologiju, fiziku čvrstog stanja, ekonomiju, agrikulturu i logistiku. Zahvaljujući modelima dubokog učenja, otvaraju se nove mogućnosti za rješavanje složenih problema u ovim disciplinama.

Primjerice, u kemiji je model AlphaFold [1] revolucionarizirao istraživanja proteina omogućivši predviđanje njihovih 3D struktura s velikom točnošću, čime je ubrzan proces razumijevanja ključnih bioloških mehanizama. U fizici čvrstog stanja, modeli dubokog učenja omogućuju predviđanje svojstava materijala i simulaciju kompleksnih sustava višestruko brže nego tradicionalne simulacije temeljene na prvim principima [2]. Ovakvi primjeri ilustriraju kako umjetna inteligencija ne samo da doprinosi ubrzanju istraživanja, već i širenju granica znanja.

Unatoč izvanrednim uspjesima, modeli dubokog učenja suočavaju se s izazovima interpretabilnosti. Oni su poznati kao "crne kutije", jer način na koji donose odluke ostaje često netransparentan. Iako razumijemo matematičku strukturu i principe rada neuronskih mreža, teško je objasniti kako točno dolazi do specifičnih predviđanja. Ova nejasnoća otežava njihovu upotrebu u područjima gdje je razumijevanje procesa donošenja odluka ključno, poput medicine ili znanosti.

Kako bi se riješio problem netransparentnosti, sve veći fokus stavlja se na razvoj interpretabilnih modela i metoda koje omogućuju bolje razumijevanje unutarnjih procesa neuronskih mreža. Te tehnike ne samo da povećavaju povjerenje u ove modele, već otvaraju mogućnost za dublje uvide u podatke koje obrađuju, što dodatno proširuje njihov potencijal za primjenu u znanosti i industriji.

Stoga će u sljedećem poglavlju biti opisani osnovni principi funkcioniranja neurona te modeli koji se koriste za njihovo matematičko i računalno opisivanje. Ovo pruža teorijsku osnovu potrebnu za razumijevanje simuliranih podataka i kasnijih analiza. Treće poglavlje bavi se osnovnim modelima dubokog učenja. Sastavne cjeline i alati slični su u svim modelima neuronskih mreža, međutim redoslijed i ideja iza njihove uporabe je ono što razlikuje modele i njihove performanse. Četvrto poglavlje daje pregled i opis vrste podataka korištenih u ovom radu – vremenski nizovi, koji predstavljaju tip podataka u kojem je bitan redoslijed mjerenja. U petom poglavlju predstavljen model koji je korišten za analizu podataka. Ovaj je model dizajniran da njegovi rezultati budu interpretabilni i posebno je napravljen za analizu multivarijatnih vremenskih nizova. Rezultati analize i određivanje dinamike populacije neurona predstavljeno je u šestom poglavlju, a zaključak u sedmom poglavlju.

2 Modeli neurona i simulacije

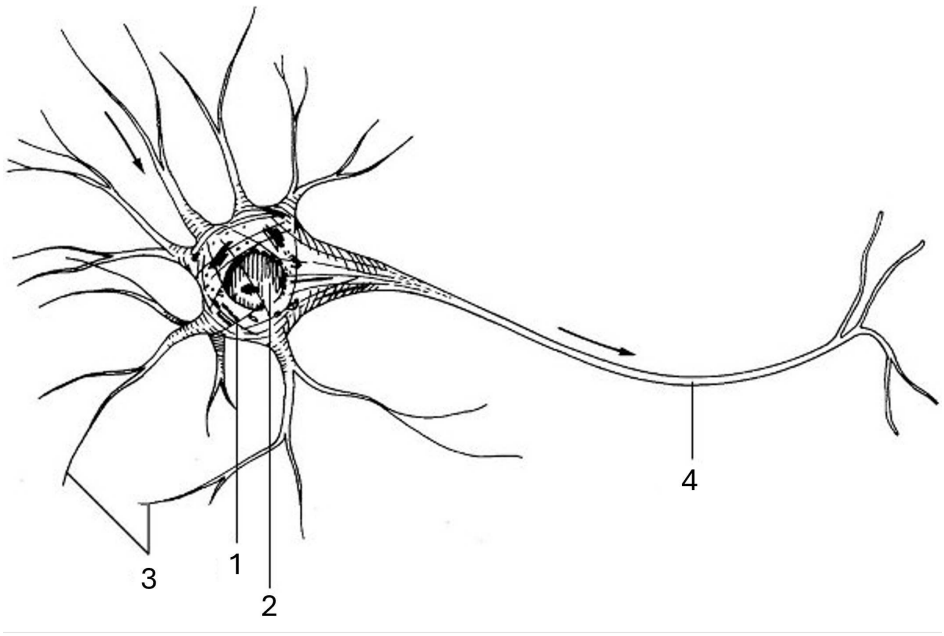
U ovom poglavlju pokriveni su ključni aspekti modeliranja neurona i njihovih mreža. Počevši pregledom građe neurona i membranskog potencijala, zatim opisom generiranja akcijskog potencijala, uključujući minimalne mehanizme i Hodgkin-Huxleyjeve jednadžbe. Nadalje, obradit će se pojednostavljeni modeli poput integriraj-i-pali modela i Izhikievichevog modela, kao i ulogu šuma u simulacijama. Poglavlje završava prikazom simulacija neuronskih mreža pomoću alata Brian 2, pružajući osnovu za generiranje i razumijevanje simuliranih podataka.

2.1 Građa neurona

Postoji mnogo vrsta neurona čiji je oblik, veličina i fiziološka uloga specijalizirana za njihovo mjesto u živčanom sustavu, no generalno se neuron može podijeliti na tri najbitnija dijela:

- Tijelo (soma): centralni dio neurona koji sadrži jezgru i organele. Odgovorna je za metaboličke procese ćelije kao što su proizvodnja energije i sinteza proteina.
- Dendriti: kraći produžeci iz tijela koji dovode signal na tijelo stanice. Završeci dendrita su povezani s drugim neuronima sinapsom; vezom između dva neurona za prenošenje signala.
- Akson: duži produžetak neurona koji se na kraju razgrana i prenosi signal s tijela stanice na druge neurone. Akson je često obavijen mijelinskom ovojnicom koja služi za električnu izolaciju i brže provođenje signala. Na kraju aksona nalaze se vrećice s neurotransmiterima i zajedno s dendritima drugih neurona čine sinapse.

Na slici 2.1 prikazana je građa neurona koja omogućava njegovu komunikaciju s velikim brojem drugih neurona (tipičan red veličine je 10^4 neurona), čime se postiže obrada informacija. Specijalizirana mjesta za prijenos signala s jednog neurona na drugi zovu se sinapse. Preko njih neuron dobiva signal na dendritima ili na tijelo. Postoji nekoliko mehanizama za prijenos signala, električnog impulsa, između presinaptičkog neurona (neuron koji šalje signal) i postsinaptičkog neurona (neuron koji



Slika 2.1: Shema živčane stanice ili neurona. Označeni su osnovni dijelovi stanice: (1) tijelo ili soma, (2) jezgra stanice, (3) dendriti i (4) akson. Na krajevima aksona i dendrita nalaze se sinaptičke veze s drugim neuronima. Strelicama je označen smjer prijenosa signala. Preuzeto iz [3].

dobiva signal). Taj električni impuls, tzv. akcijski potencijal, nastaje na početku aksona te putuje niz njega. Svaki neuron ima jedan akson koji raste iz tijela, no može se i razgranati te slati signal na više neurona, čak i onima koji nisu u neposrednoj blizini.

2.2 Membranski potencijal

Sposobnost neurona da promjeni vlastiti električni potencijal membrane omogućuje slanje signala niz akson. Potencijal membrane V_m je definiran kao razlika potencijala između unutrašnjosti stanice i okoline, a potječe od neravnoteže koncentracije iona izvan i unutar stanice. Na primjer, koncentracija kalija (K^+) je 20 puta veća unutar neurona, stoga se neravnoteža želi smanjiti procesom difuzije. Međutim, neuronska membrana nije propusna na anione koji se obično vežu na kalij, tako da proces difuzije ostavlja višak negativnog naboja unutar neurona i višak pozitivnog naboja u okolnoj tekućini neurona. Tako nastala razlika potencijala suprotstavlja se sili koju stvara razlika koncentracije, čime je sustav u ravnoteži.

Jednostavan fizikalni opis ovog potencijala može se zapisati kao iščezavanje struje iona koja potječu od razlike električnog potencijala i razlike u koncentraciji:

$$J_V + J_D = -q\mu_{\mathrm{X}} c_{\mathrm{X}} \frac{dV}{dx} - qD_{\mathrm{X}} \frac{dc_{\mathrm{X}}}{dx} = 0, \tag{2.1}$$

gdje je c_{X} koncentracija iona, μ_{X} električna pokretljivost iona i D_{X} konstanta difuzije iona. Integracijom se dobije veza između razlike potencijala i koncentracija unutar i izvan stanice:

$$\Delta V = -\frac{kT}{q} \ln \left(\frac{c_{\mathrm{X}}^{(1)}}{c_{\mathrm{X}}^{(2)}} \right). \tag{2.2}$$

Dodatno je uvršteno $D_{\mathrm{X}} = kT\mu_{\mathrm{X}}/q$, pri čemu je k Boltzmannova konstanta, a T temperatura. U tipičnom neuronu ravnotežni potencijal za ione kalija je -80 mV, što se inače naziva povratni potencijal neurona. Za ione natrija taj potencijal iznosi -65 mV i to je potencijal mirovanja neurona.

Potencijal membrane ovisi o koncentracijama iona unutar stanice i okolne tekućine i može se mijenjati raznim mehanizmima, ovisno o ulozi za koju je neuron specijaliziran. Osjetilni neuroni pretvaraju vanjski podražaj kao što su svjetlo ili pritisak u električni signal. Prijenos informacija između neurona najčešće se odvija preko kemijskih sinapsi, ili ponekad električnih sinapsi, gdje otpuštanjem neurotransmitera, ili direktno električnim potencijalom, neuroni utječu na potencijal membrane sljedećeg neurona.

2.3 Generiranje akcijskog potencijala

Ovaj odjeljak opisuje kako promjena membranskog potencijala može u konačnici biti okidač za generiranje akcijskog potencijala u dijelu neurona s ionskim kanalima osjetljivim na napon. Akcijski potencijal karakteriziraju naglo povećanje (depolarizacija) potencijala membrane, padanje potencijala ispod potencijala mirovanja (hiperpolarizacija) i vraćanje potencijala na stanje mirovanja. Hodgkin i Huxley su kvantitativno objasnili akcijski potencijal s četiri diferencijalne jednadžbe koje opisuju dinamiku ionskih struja i promjene membranskog potencijala tijekom akcijskog potencijala [4].

2.3.1 Minimalni mehanizam

Za generiranje akcijskog potencijala potrebna su barem dva ionska kanala ovisna o potencijalu i jedan statički ionski kanal. Propusnost iona u kanalima ovisnim o

potencijalu može se opisati jednadžbom, a u Hodgkin-Huxley modelu to su kanali natrija i kalija. Kada impulsi ostalih neurona preko dendrita dovoljno depolariziraju neuron, natrijev kanal se otvori, zbog razlike u koncentracijama i potencijala natrij iz okoline ulazi u stanicu i dolazi do nagle depolarizacije membrane od oko 65 mV. 1 ms nakon, tok natrija se zatvara i otvara se kalijev kanal. Također, zbog razlike u koncentracijama i potencijalu, kalij izlazi iz stanice u okolnu tekućinu i dolazi do naglog pada potencijala do -80 mV, ispod potencijala mirovanja. Zbog hiperpolarizacije kanal se zatvara i potencijal membrane se nakon nekoliko milisekundi vrati na potencijal mirovanja.

Generiranje akcijskih potencijala smanjuje razliku koncentracije iona unutar i izvan stanice koja je potrebna za stvaranje akcijskih potencijala. Kako bi se natrij vratio iz okoline u stanicu, a kalij iz stanice u okolinu, neuroni imaju ionske pumpe koje neovisno o potencijalu membrane premještaju ione iz nižih koncentracija u više vraćajući potencijal u stanje mirovanja. Cijena ionskih pumpi je visoka potrošnja energije potrebna za ovaj proces.

2.3.2 Hodgkin-Huxley jednadžbe

Hodgkin i Huxley su temeljem mjerenja na velikom aksonu lignje formulirali četiri vezane diferencijalne jednadžbe koje kvantificiraju generiranje akcijskog potencijala, a ključnu ulogu u tome imaju ionske struje. Broj i propusnost otvorenih ionskih kanala određuje električnu provodljivost kanala σ_{ion} koja ovisi o nekoliko parametara. Ohmov zakon povezuje struju ionskog kanala, provodljivost i napon:

$$I_{\text{ion}} = \sigma_{\text{ion}} \left(V - E_{\text{ion}}\right) \tag{2.3}$$

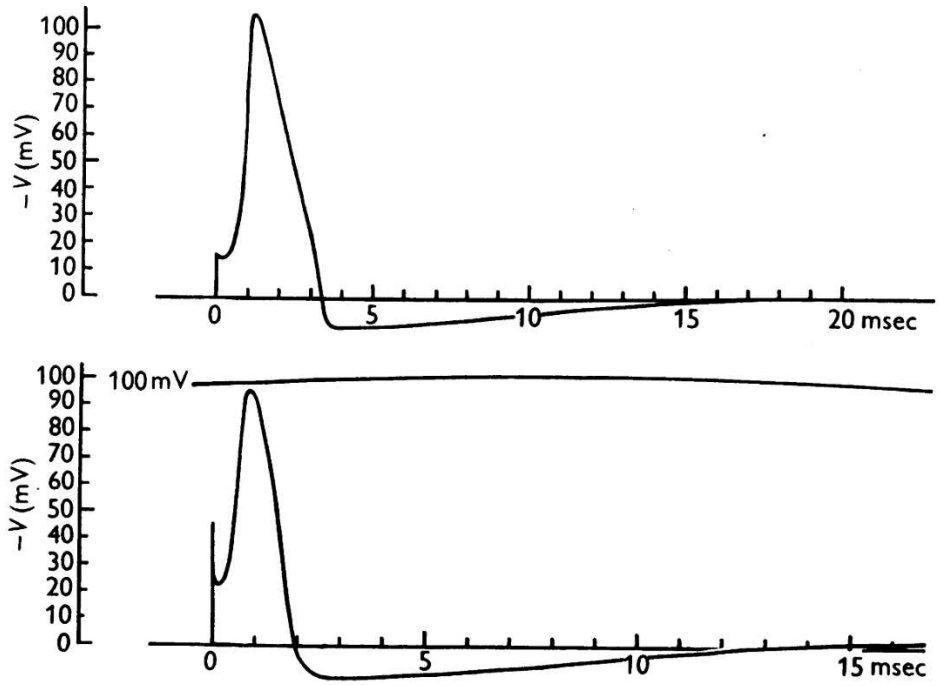
gdje je E_{ion} ravnotežni potencijal ionskog kanala kada se u obzir uzme razlika koncentracija iona unutar i izvan stanice.

Ovisnost Na^+ i K^+ kanala o potencijalu pokreće mehanizam stvaranja akcijskog potencijala. Hodgkin i Huxley su empirijski uveli dinamičke varijable n, m i h kako bi opisali dinamiku i ovisnost ionskih kanala o potencijalu te što bolje aproksimirali izmjerene podatke. Pretpostavili su vodljivost ionskih kanala o potencijalu i vremenu na sljedeći način:

$$\sigma_{\mathrm{K}}(V,t) = \sigma_{\mathrm{K}} n^4 \tag{2.4}$$

$$\sigma_{\mathrm{Na}}(V,t) = \sigma_{\mathrm{Na}} m^3 h, \tag{2.5}$$

Varijabla n opisuje aktivaciju K^+ kanala, a m i h opisuju aktivaciju i deaktivaciju Na^+ kanala respektivno.



Slika 2.2: Gore: Rješenje Hodgkin-Huxley jednadžbi za početnu depolarizaciju od -15 mV pri 6 °C. Dolje: mjerenja akcijskog potencijala pri 9.1 °C. Razlika u skali vremenske osi odgovara razlici u temperaturi. Preuzeto iz [4].

Zadnja činjenica koja se mora iskoristiti jest da se neuroni u ovom sustavu ponašaju kao kondenzatori jer pohranjuju ione. Hodgkin-Huxley model može se prikazati uzemljenim strujnim krugom s paralelno spojenim kondenzatorom i tri otpornika. Poštujući Kirchhoffov zakon, struje u sustavu možemo raspisati jednadžbom:

$$I(t) = C\frac{dV}{dt} + \sum_{\mathrm{ion}} I_{\mathrm{ion}}. \tag{2.6}$$

$I(t)$ je ulazna struja koja potječe od presinaptičkih neurona utjecajem, na primjer, neurotransmitera. Pod I_{ion} spadaju struje Na^+ i K^+ ionskih kanala, koje imaju promjenjivu propusnost, i struja curenja iona koja ima konstantnu propusnost jer membrana nije savršeno nepropusna. Objedinjenjem svih jednadžbi dobije se Hodgkin-Huxley

model akcijskog potencijala:

$$C\frac{dV}{dt} = -\sigma_{\mathrm{K}} n^4 \left(V - E_{\mathrm{K}}\right) - \sigma_{\mathrm{Na}} m^3 h \left(V - E_{\mathrm{Na}}\right) - \sigma_{\mathrm{L}} \left(V - E_{\mathrm{L}}\right) + I(t) \tag{2.7}$$

$$\tau_n(V)\frac{dn}{dt} = -\left[n - n_0(V)\right] \tag{2.8}$$

$$\tau_n m(V)\frac{dm}{dt} = -\left[m - m_0(V)\right] \tag{2.9}$$

$$\tau_h(V)\frac{dh}{dt} = -\left[h - h_0(V)\right]. \tag{2.10}$$

Subskript L označava struju curenja, a oblik funkcija τ_x i x_0 ovisnih o V su Hodgkin i Huxley odabrali kako bi što bolje opisali eksperimentalna mjerenja.

Ovaj model uključuje struje dva ionska kanala i dobra je aproksimacija stvarnog mehanizma generiranja akcijskog potencijala za specifičan tip aksona koji su Hodgkin i Huxley proučavali. Na primjer, Wilsonov model [5] daje kompleksnije mogućnosti ponašanja akcijskog potencijala uključivanjem dva druga kanala i interakcijom nekih kanala. U stvarnosti postoji mnogo više ionskih kanala koji interagiraju međusobno, a ovaj model opisuje prosječnu dinamiku ionskih kanala te njihovu vremensku ovisnost funkcijama n, m i h. Hodgkin-Huxley model je značajan jer je prvi jednostavne, kvalitativne opise kako stvari možda funkcioniraju zamijenio kvantitativnim modelom koji može pouzdano predviđati ponašanje ovog sustava kao na slici 2.2.

2.4 Integriraj-i-pali model

Kako bi se lakše proučavali sustavi neurona, često se opis potencijala u neuronu pojednostavi. Jedan od razloga je što to omogućuje računanje interakcija za velik broj neurona i jednostavniju analizu sustava. Također, korisno je istražiti koliko se model može pojednostaviti, a da i dalje zadrži bitne fenomene koje se žele proučiti.

Generalno oblik akcijskog potencijala nema utjecaj na prijenos ili obradu informacija, međutim samo generiranje impulsa i trenutak kada nastane imaju utjecaj. Stoga, u ovom slučaju bitno je samo opisati proces stvaranja impulsa i integriranje signala s drugih neurona, a detalji poput oblika impulsa i dinamike ionskih kanala se mogu ostaviti po strani.

Budući da se sada promatra samo slučaj kada je potencijal ispod praga stvaranja impulsa, jednadžba koja opisuje vremensku ovisnost potencijala je:

$$\tau_m \frac{dv(t)}{dt} = -\left(v(t) - E_{res}\right) + RI(t), \tag{2.11}$$

gdje je E_{res} potencijal mirovanja, a $I(t)$ ulazni signal na sinapsama, ili umjetno stvoren u eksperimentu. U konstantu τ_m pohranjen je kapacitet membrane i prosječna provodljivost ionskih kanala.

Umjesto korištenja Hodgkin–Huxley jednadžbi za opis akcijskog potencijala, stvaranje tog impulsa u ovom modelu je maksimalno pojednostavljeno. U trenutku t^f kada potencijal naraste preko praga za stvaranje impulsa θ, potencijal se vraća na vrijednost potencijala mirovanja:

$$v(t^f) = \vartheta \tag{2.12}$$

$$\lim_{\delta \longrightarrow 0} v(t^f + \delta) = v_{res}. \tag{2.13}$$

Iako je Integriraj-i-pali model [6] prilično općenit, ovaj model neurona se često koristi u kombinaciji s odabranim ulaznim strujama za specifična istraživanja. Te struje mogu biti neki vanjski izvor u eksperimentu ili se može opisati struja koja dolazi iz drugih stanica. U mnogim simulacijama mreža neurona, uglavnom nas interesira zbroj sinaptičkih struja koje se generiraju signalima presinaptičkih stanica. Taj zbroj sinaptičkih struja ovisi o učinkovitosti pojedinačnih sinapsi, koje su opisane jačinom sinapse w_j, pri čemu je j indeks presinaptičkog neurona. Vrijednosti w_j su težine u linearnoj kombinaciji ulaznih struja uz pretpostavku da sinapse ne interagiraju međusobno. Ukupna ulazna struja može se zapisati kao linearna kombinacija sinaptičkih struja:

$$I(t) = \sum_j \sum_{t_j^f} w_j \alpha \left(t - t_j^f\right). \tag{2.14}$$

Svaka postsinaptička reakcija na impuls, koja se opisuje funkcijom α, pomnožena je s odgovarajućom težinom. Vrijeme t^f označava trenutak generiranja impulsa presinaptičkog neurona.

Integriraj-i-pali model svojevremeno je pokazao da je značajan napredak moguć, ako je neki fenomen adekvatno opisan, čak i ako se njegov biofizički proces ne može modelirati. Vrlo je pogodan za numeričke izračune zbog svoje jednostavnosti,

računalne učinkovitosti i laganog skaliranja. Matematički je jednostavan, što smanjuje računalnu složenost i omogućava simulaciju velikih neuronskih mreža s velikim brojem neurona. Također je koristan jer dopušta jednostavno modificiranje oblika ulaznih struja ili vrsta i jačina sinapsa, što omogućuje reproduciranje raznih uzoraka impulsa.

2.5 Izhikievichev model

Izhikievichev model neurona [7] kompaktna je matematička aproksimacija složenih bioloških ponašanja neurona, osmišljen kako bi se zadržala dinamička realističnost Hodgkin-Huxleyjevog modela, ali uz značajno manju računalnu zahtjevnost. Model koristi sustav dvije nelinearne diferencijalne jednadžbe prvog reda za opisivanje električnih svojstava neurona, uključujući generiranje akcijskih potencijala, oscilacije membrane i postizanje stabilnih stanja.

Model je zadan jednadžbama:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I \tag{2.15}$$

$$\frac{du}{dt} = a(bv - u), \tag{2.16}$$

uz pomoćno resetiranje nakon generiranja impulsa:

$$\text{ako je } v \geq v_{\text{prag}}, \text{ tada: } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases}, \tag{2.17}$$

gdje je v membranski potencijal neurona, a u predstavlja varijablu oporavka membrane koja uzima u obzir aktivaciju K^+ ionskih struja i inaktivaciju Na^+ ionskih struja te pruža negativnu povratnu vezu na v. I je vanjska ulazna struja, koja kod simulacije mreže neurona predstavlja sinaptičke veze s drugim neuronima, i a, b, c, d su parametri modela koji određuju dinamiku različitih vrsta neurona.

Za razliku od Hodgkin-Huxleyjevog modela, koji uključuje višedimenzionalnu dinamiku ionskih kanala, Izhikievichev model je jednostavniji za numeričku integraciju uključivanjem svih efekata ionskih struja u jednu jednadžbu, čineći ga pogodnim za simulaciju velikih mreža neurona. Model nije izravno fiziološki (ne simulira pojedinačne ionske kanale), ali dovoljno dobro aproksimira biološka svojstva da bude

koristan u simulacijama mreža i u istraživanjima neuronske dinamike na sustavnoj razini.

U odnosu na integriraj-i-pali model, Izhikievichev model ima i nelinearni doprinos, uključuje efekte ionskih struja, može reproducirati širok spektar bioloških ponašanja neurona i ima bogatiji fazni prostor bez velikog povećanja računalne zahtjevnosti.

2.6 Dodavanje šuma

U ovim modelima neuroni će predvidljivo, točno u trenutku kada potencijal dođe do praga generirati impuls i započeti proces ispočetka, ako je ulazna struja konstantna. U stvarnosti, neuroni u organizmima ne stvaraju impulse redovito, nego naizgled nasumično. Vrijeme između dva impulsa obično ima neku distribuciju, no u Integriraj-i-pali modelu je ono točno određeno.

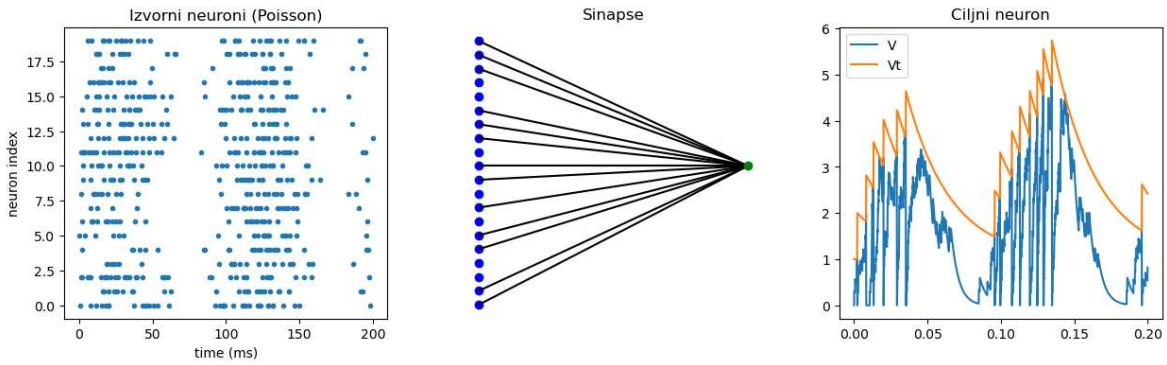
Šum u neuronima može potjecati iz različitih izvora. S jedne strane, varijacija ulaznih podražaja iz okoline može uzrokovati fluktuacije u neuronskoj aktivnosti. Osjetilni input nije konstantan, već varira, što doprinosi nepravilnom ponašanju neurona. S druge strane, šum proizlazi i iz unutarnjih faktora zbog statističke prirode sinapsi i strukturnih nepravilnosti u građi neurona. Difuzno širenje neurotransmitera preko sinaptičke veze, otvaranje i zatvaranje ionskih kanala te širenje membranskog potencijala duž dendrita s različitim geometrijama doprinosi nepravilnostima. Također, probabilistička priroda oslobađanja neurotransmitera uslijed aksijalnih impulsa dodatno doprinosi šumu.

Ovi kombinirani faktori uzrokuju nepravilnosti i varijacije u neuronskoj aktivnosti, što se često modelira uključivanjem šuma u simulacije neuronskih mreža. Tri načina kojima se obično uključuje šum u modele su:

- Stohastički prag: potencijal potreban da neuron generira impuls je nasumičan

$$\vartheta \to \vartheta + \eta(t). \tag{2.18}$$

- Nasumično resetiranje: nakon generiranja impulsa neuron se resetira na nasumičnu vrijednost potencijala



Slika 2.3: Jednostavan primjer simulacije mreže neurona Brian 2 paketom. Lijevo: točkasti dijagram trenutaka generiranja impulsa za svaki neuron. Sredina: shema sinapsi mreže neurona. Desno: graf ovisnosti potencijala ciljnog neurona o vremenu. Preuzeto iz [8].

$$v^{res} \to v^{res} + \eta(t). \tag{2.19}$$

- Stohastička integracija: mehanizam integracije signala u neuronima može sadržavati član nasumičnog šuma i moguće je opisati ga stohastičkom diferencijalnom jednadžbom

$$\tau_m \frac{dv}{dt} = -v + v_0 + \eta(t). \tag{2.20}$$

Odabirom odgovarajuće distribucijske funkcije za slučajnu varijablu η mogu se dobiti ekvivalentni rezultati stohastičkih procesa neurona. Iako ista distribucija vjerojatnosti za nasumičnu varijablu može proizvesti različite rezultate za svaki model šuma, u praksi želimo odabrati distribucije koje su prikladne za opisivanje eksperimentalnih podataka, stoga izbor modela šuma više je pitanje pogodnosti. Numerička istraživanja često koriste ulazne signale sa šumom za modeliranje stohastičkih procesa u mozgu. Ovaj model ima jednostavnu interpretaciju nasumičnog sinaptičkog prijenosa koji se može opaziti i u stvarnim neuronskim sustavima. Iako je analitičko proučavanje ovog modela zahtjevno, u numeričkim istraživanjima je jednostavno integrirati šum na ovaj način.

2.7 Simuliranje mreže neurona - Brian 2

Simuliranje neurona je ključno za istraživanje i razumijevanje neuronskih funkcija jer omogućuje proučavanje složenih procesa koji se odvijaju unutar i između neurona u kontroliranim uvjetima. Simulacije omogućuju istraživanje različitih modela neuronske dinamike, od jednostavnih integriranih modela poput Integriraj-i-pali, složenijih biološki realističnih modela poput navedenog Hodgkin-Huxley modela i modela koji uključuju brojne ionske kanale za opis generiranja akcijskog potencijala.

Simuliranje podataka u kontroliranim uvjetima izuzetno je korisno u istraživanju modela jer omogućuje proučavanje specifičnih fenomena bez ometajućih faktora iz stvarnog svijeta. U kontroliranim uvjetima mogu se izolirati i varirati pojedinačni parametri, što omogućuje razumijevanje njihovog utjecaja na model. Ovo je posebno važno za testiranje hipoteza i predviđanje ponašanja složenih sustava.

Alat za simulaciju neuronskih mreža korišten u sklopu ovog diplomskog rada je paket Brian 2 [9], simulator za mreže neurona napisan u programskom jeziku Python. Paket Brian 2 predstavlja vrlo moćan i fleksibilan alat za simulaciju mreža neurona, posebno zbog svoje jednostavnosti korištenja i visoke računalne učinkovitosti, što ga čini popularnim izborom u neuroznanstvenim istraživanjima. U sklopu ovog diplomskog rada, Brian 2 omogućuje jednostavno modeliranje i simuliranje neuronskih mreža, uključujući razne vrste neurona (od jednostavnih do biološki realističnih modela) i sinaptičke veze među njima.

U paketu Brian 2, svaki neuron može se definirati s vlastitim skupom parametara koji određuju njegovu dinamiku: membranskog potencijala, pragova za generiranje impulsa i mehanizme resetiranja nakon generiranja akcijskog potencijala. Također je moguće implementirati različite vrste neuronskih modela (poput Integriraj-i-pali modela, Izhikievichevog modela i drugih biološki realističnih modela). Paket omogućuje kontrolu nad sinaptičkim vezama između neurona. To uključuje odabir vrste sinapsi (pobuđujuće, inhibirajuće), jačinu sinaptičkih veza, vjerojatnost prijenosa impulsa (spontana aktivacija), kao i vremensku dinamiku sinaptičkog prijenosa. Glavni razlog korištenja Brian 2 paketa u ovom diplomskom radu je generiranje podataka za testiranje i treniranje modela. Nakon simuliranja mreže neurona, prikupljeni su podaci u obliku vremenskih serija, uključujući potencijale pojedinih neurona i vrijeme generiranih impulsa. Primjer generiranih podataka pomoću Brian 2 prikazan je na slici 2.3

3 Osnove dubokog učenja

Umjetna inteligencija bavi se izgradnjom sustava koji pokušavaju simulirati inteligentno ponašanje. Obuhvaća širok raspon metoda, uključujući one koji se temelje na logici, pretraživanju i probabilističkom zaključivanju. Strojno učenje je podskup umjetne inteligencije koji uči donositi odluke prilagođavanjem matematičkih modela promatranim podacima. Duboka neuronska mreža je vrsta modela strojnog učenja, a proces prilagođavanja modela na podatke se naziva duboko učenje. Duboke neuronske mreže pronalaze široku primjenu u obradi prirodnog jezika, računalnom vidu, modelima i simulacijama u znanosti, generativnim modelima za slike i videe, prepoznavanju govora i još mnogo drugdje.

Neuronske mreže su vrlo fleksibilne i mogu se prilagoditi na podatke bez potrebe da poznajemo ili pretpostavimo kakva je funkcijska ovisnost podataka. Ta sposobnost omogućava njihovu široku primjenu, no nosi sa sobom veliku manu. Za razliku od prilagođavanja funkcije na podatke i ostalih metoda u strojnom učenju, model dobiven korištenjem neuronskih mreža teško je interpretirati. Nemamo informaciju kako analitički opisati podatke, niti, na primjer, o kojoj potenciji ovisi neki ulazni parametar ili koeficijente nagiba. Dobiveni model može vrlo dobro opisivati podatke, u nekim slučajevima bolje od drugih metoda strojnog učenja, ali to je sve što nam model omogućuje - generaliziranje podataka.

Ovo poglavlje bavi se opisom osnovnih pojmova i gradivnih jedinica modela dubokog učenja korištenog u ovom diplomskom radu: plitka neuronska mreža, duboka neuronska mreža i konvolucija.

3.1 Plitke neuronske mreže

Neuronska mreža sastoji se od ulaznog sloja, skrivenih slojeva i izlaznog sloja. Ulazni sloj su podaci koji imaju unaprijed određene dimenzije. U skrivenom sloju računa se s podacima i rezultati se prosljeđuju u sljedeći skriveni sloj, a nazivaju se tako jer su smješteni između ulaznog i izlaznog sloja, nisu izravno dostupni kao dio ulaznih ili izlaznih podataka te nemamo izravni pristup njima, već su unutarnji mehanizam za obradu i transformaciju informacija unutar mreže. Jedan skriveni sloj sastoji se od težina, pristranosti i aktivacijske funkcije. Izlazni sloj je konačni rezultat neuronske mreže koji također ima unaprijed određene dimenzije.

Plitke neuronske mreže sastoje se od samo jednog skrivenog sloja, uz ulazni i izlazni sloj. Skriveni sloj može imati proizvoljan broj jedinica za računanje, a svaka jedinica ima svoj parametar pristranosti. U potpuno povezanom sloju, svaka jedinica prethodnog sloja povezana je odgovarajućom težinom s jedinicom sljedećeg skrivenog sloja. Neuronske mreže su funkcije $\mathbf{y} = \mathbf{f}[\mathbf{x}, \Phi]$ s parametrima Φ koje preslikavaju višedimenzionalni ulaz $\mathbf{x} \in \mathbb{R}^{D_i}$ u višedimenzionalni izlaz $\mathbf{y} \in \mathbb{R}^{D_o}$ koristeći skrivene jedinice $\mathbf{h} \in \mathbb{R}^{D_1}$. Svaka skrivena jedinica je funkcija oblika:

$$h_i = \mathrm{a}\left(\beta_i + \sum_{j=1}^{D_i} \omega_{ij} x_j\right), \tag{3.1}$$

i kombiniranjem svih skrivenih jedinica dobiva se izlaz:

$$y_i = \phi_{j0} + \sum_{d=1}^{D} \phi_{jd} h_d, \tag{3.2}$$

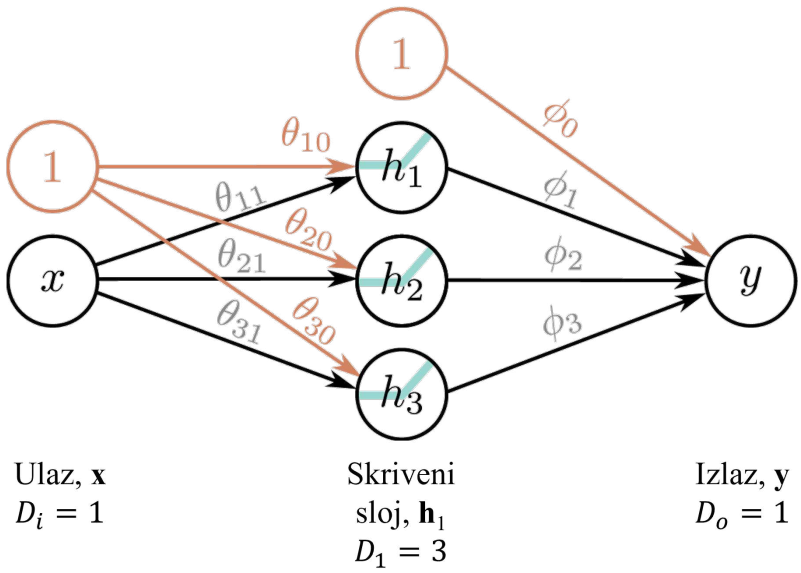
gdje je $\mathbf{a}(\bullet)$ nelinearna aktivacijska funkcija, a $\Phi = \{\theta_{\bullet\bullet}, \phi_{\bullet}\}$ su parametri modela. Aktivacijska funkcija omogućava modelu da opiše kompleksne odnose između ulaza i izlaza, inače bi neuronska mreža bila ograničena samo na linearna preslikavanja. Najčešće se koristi ReLU (*Rectified Linear Unit*) kao aktivacijska funkcija zbog svoje jednostavnosti:

$$\mathrm{ReLU}(x) = \begin{cases} 0 & \text{za } x \leq 0 \\ x & \text{za } x > 0. \end{cases} \tag{3.3}$$

S ovakvom aktivacijskom funkcijom skriveni sloj dijeli ulazni prostor na potprostore, a svaki potprostor je vlastitim linearnim preslikavanjem povezan s izlazom. Povećavanjem broja skrivenih jedinica povećava se mogući broj potprostora i model može detaljnije opisati podatke. Slično vrijedi i za druge aktivacijske funkcije jer su sve oblikom nalik na ReLU, no zbog drugačijeg oblika ulazni potprostori će biti nelinearno preslikani na izlaz i dijeljenje na potprostore nije jasno definirano.

3.2 Duboke neuronske mreže

Univerzalni aproksimacijski teorem tvrdi da postoji plitka neuronska mreža s konačnim brojem skrivenih jedinica koja može aproksimirati bilo koju kontinuiranu funkciju na kompaktnom potprostoru iz $\mathbb{R}^n$ proizvoljno precizno [11, 12]. Međutim, u praksi



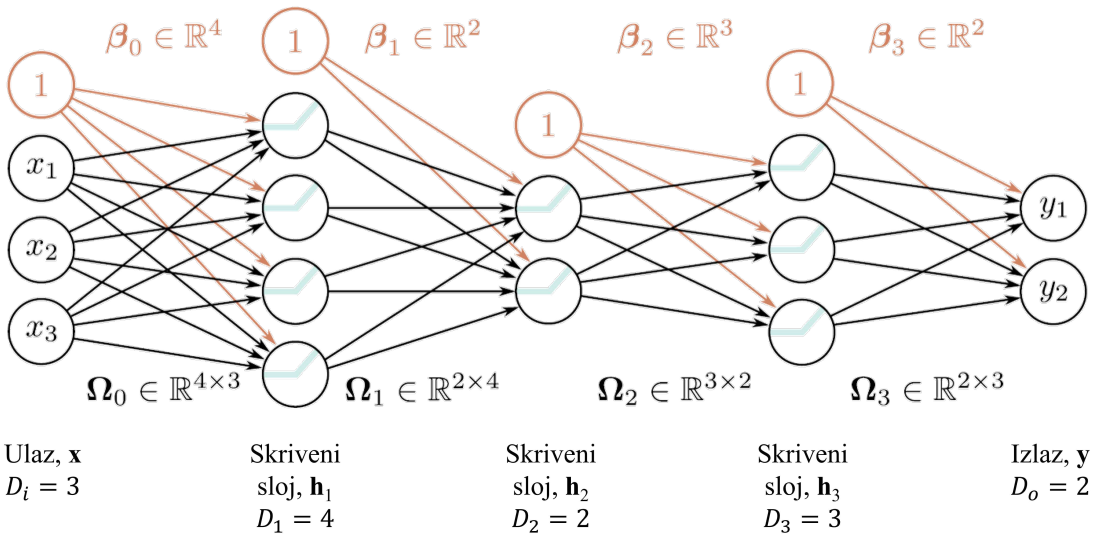
Slika 3.1: Primjer potpuno povezane plitke neuronske mreže jednom ulaznom varijablom, jedan skriveni sloj s tri skrivene jedinice i jedan izlaz. Ukupno ima deset parametara: šest težina prikazane crnim strelicama, i četiri pristranosti prikazane narančastim strelicama. Svaka jedinica u skrivenom sloju ima i aktivacijsku funkciju prikazana plavom bojom. Preuzeto iz [10].

to nije izvedivo jer je u većini slučajeva potreban broj skrivenih jedinica nepraktičan za računati. Tu u priču dolaze duboke neuronske mreže, jednostavna ideja koja je značajno pokrenula razvoj neuronskih mreža.

Duboke neuronske mreže, za razliku od plitkih, imaju nekoliko skrivenih slojeva koji su povezani težinama s prethodnim slojem. Analogijom dijeljenja potprostora, dodavanje skrivenih slojeva efikasnije povećava broj potprostora, za razliku od dodavanja više skrivenih jedinica u plitkoj neuronskoj mreži. Stoga, ovakve neuronske mreže mogu bolje aproksimirati veći podskup funkcija i prilagoditi se podatcima.

Budući da sada neuronska mreža može imati više skrivenih slojeva, mogu se definirati hiperparametri koji opisuju arhitekturu neuronske mreže. Oni se mogu optimizirati za dane podatke poput parametara, ali za vrijeme treniranja modela oni su fiksni. Moderni modeli dubokih neuronskih mreža mogu imati preko stotinu skrivenih slojeva i tisuće skrivenih jedinica u svakom sloju. Broj skrivenih slojeva obično se naziva dubina, broj skrivenih jedinica u sloju se naziva širina sloja i ukupan broj skrivenih jedinica je mjera kapaciteta neuronske mreže.

Neka D_i označava broj skrivenih jedinica po sloju, a K broj skrivenih slojeva. Skriveni slojevi u dubokoj neuronskoj mreži mogu se raspisati kao



Slika 3.2: Duboka neuronska mreža s $D_i = 3$-dimenzionalnim ulazom $\mathbf{x}$, $K = 3$ skrivena sloja $\mathbf{h}_1$, $\mathbf{h}_2$ i $\mathbf{h}_3$, $D_o = 2$-dimenzionalnim izlazom $\mathbf{y}$ i ukupno ima 43 parametra. Preuzeto iz [10].

$$
\begin{aligned}
\mathbf{h}_1 &= \mathbf{a}\left(\boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}\right) \\
\mathbf{h}_2 &= \mathbf{a}\left(\boldsymbol{\beta}_1 + \boldsymbol{\Omega}_1 \mathbf{h}_1\right) \\
&\vdots \\
\mathbf{h}_K &= \mathbf{a}\left(\boldsymbol{\beta}_{K-1} + \boldsymbol{\Omega}_{K-1} \mathbf{h}_{K-1}\right),
\end{aligned} \tag{3.4}
$$

gdje su parametri modela $\boldsymbol{\beta}_{i-1}$ vektori pristranosti dimenzije D_i te $\boldsymbol{\Omega}_{i-1}$ matrice težina dimenzija $D_i \times D_{i-1}$ za računanje i-tog skrivenog sloja. Konačno, izlaz neuronske mreže je vektor

$$
\mathbf{y} = \boldsymbol{\beta}_K + \boldsymbol{\Omega}_K \mathbf{h}_K. \tag{3.5}
$$

Ekvivalentno se neuronska mreža može zapisati jednom jednadžbom koja nalikuje rekurziji:

$$
\mathbf{y} = \boldsymbol{\beta}_K + \boldsymbol{\Omega}_K \mathbf{a}\left(\boldsymbol{\beta}_{K-1} + \boldsymbol{\Omega}_{K-1} \mathbf{a}\left(\ldots \mathbf{a}\left(\boldsymbol{\beta}_1 + \boldsymbol{\Omega}_1 \mathbf{a}\left(\boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}\right)\right) \ldots\right)\right). \tag{3.6}
$$

3.3 Konvolucija

Konvolucija je matematička operacija koja se koristi za kombiniranje dva skupa podataka, često signala ili slika, kako bi se stvorio treći skup podataka. Ona izražava način na koji oblik jedne funkcije mijenja oblik druge funkcije. U kontinuiranom slučaju, konvolucija dvije funkcije $f(t)$ i $g(t)$ definirana je kao:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\, d\tau \tag{3.7}$$

Diskretna konvolucija je specifičan slučaj konvolucije koji se koristi u digitalnoj obradi signala i slikama gdje su ulazni podaci diskretni. Za dvije diskretne sekvence $f[m]$ i $g[m]$ gdje je $m \in S$, diskretna konvolucija $(f * g)[m]$ definira se kao:

$$(y * h)[n] = \sum_{m \in S} x[m]\, h[n-m] \tag{3.8}$$

Diskretna konvolucija je bitna u računalnom vidu, obradi signala i digitalnoj obradi slika, gdje se koristi za primjenu filtera, izoštravanje slika, detekciju rubova i druge obradbe podataka.

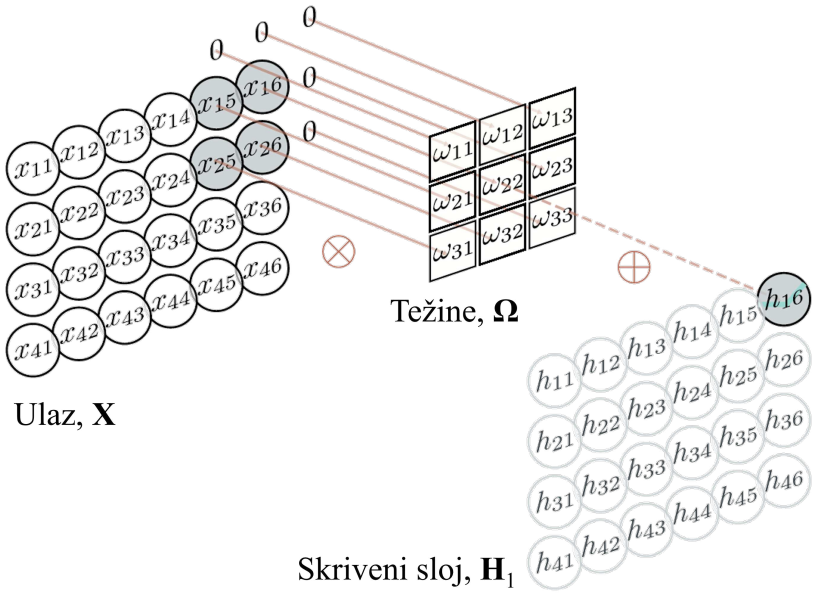
Konvolucijske neuronske mreže (CNN) koriste operaciju konvolucije za ekstrakciju značajki iz ulaznih podataka, obično slika, grafova ili vremenskih serija. U CNN-ovima, konvolucijski slojevi primjenjuju filtere (ili jezgre) preko ulaznih podataka kako bi se informacije sa susjednih piksela (varijabli) izmijenile. Imaju manje parametara u usporedbi s potpuno povezanim slojevima jer koriste iste težine i pristranosti za cijelu sliku, iskorištavaju prostorne odnose između susjednih piksela i ne moraju ponovno učiti interpretaciju piksela na svakoj poziciji.

U 1D slučaju (npr. vremenska serija, zvuk, tekst), ako filter ima dimenzije 1×3 i primjenjujemo ga na ulazni vektor, i-ta skrivena jedinica računa se kao

$$h_i = a\left(\beta + \sum_{j=1}^{3} \omega_j x_{i+j-2}\right), \tag{3.9}$$

gdje su pristranost β i težine filtera ω_1, ω_2, ω_3 parametri konvolucijskog sloja. Vrijednosti ulaza x su 0 (zero padding), kada je njegov indeks izvan važećeg raspona (npr. x_0).

Kada bi se ista skrivena jedinica računala potpuno povezanim slojem, kao u prijašnjim poglavljima, bilo bi potrebno je puno više parametara. Na primjer, za



Slika 3.3: 2D konvolucija s 3×3 filterom $\mathbf{\Omega}$. Kada filter pređe zadani raspon ulaznih varijabli, ulazna vrijednost je nula (zero padding). Ova konvolucija ima deset parametara - devet težina i jednu pristranost. Preuzeto iz [10].

D ulaznih varijabli i D skrivenih jedinica potpuno povezani sloj bi imao D^2 težina i D pristranosti, a navedeni konvolucijski sloj samo tri težine i jednu pristranost. To je ekvivalentno potpuno povezanom sloju u kojem je većina težina jednaka nuli, a ostale težine su ograničene na tri različite vrijednosti. Međutim, ne znači da će jedna konvolucija biti jednako efektivna kao potpuno povezani sloj, no često se može primijeniti nekoliko konvolucija na isti ulaz (kanali) i "uštedjeti" puno parametara u odnosu na potpuno povezan sloj.

Kod 2D slučaja (slike ili podaci strukturirani u tablice), filter je sada matrica. Primjenjivanjem 3×3 filtera $\mathbf{\Omega} \in \mathbb{R}^{3\times3}$ na ulazne podatke x_{ij}, račun za skrivenu jedinicu h_{ij} izgleda kao:

$$h_{ij} = a\left(\beta + \sum_{m=1}^{3}\sum_{n=1}^{3} \omega_{mn}\, x_{i+m-2,\, j+n-2}\right), \tag{3.10}$$

gdje su ω_{ij} matrični elementi filtera, to jest težine. Kao i u 1D slučaju, konvolucija je težinska suma vrijednosti koje filter obuhvaća, a sve skrivene jedinice računaju se pomicanjem filtera horizontalno i okomito preko ulaznih podataka.

Primjenjivanjem iste konvolucije, vrijednosti susjednih varijabli se težinski uprosječuju, aktivacijska funkcija vraća samo nenegativne vrijednosti i svakim korakom gube se detaljne informacije. Zbog toga, obično se primjenjuje više konvolucija para-

lelno, svaka sa svojim težinama što stvara različite kanale kojima informacije mogu teći.

Prednosti korištenja konvolucije u određenim slučajevima su značajno smanjenje broja parametara i bolje performanse modela što je vrlo bitno kada ima puno ulaznih varijabli te je potrebna velika neuronska mreža za dobar model. Jedan od razloga zašto su konvolucije tako efikasne, kod npr. računalnog vida ili u analizi vremenskih serija, je što je korištenje konvolucije snažno ograničenje za model i indirektna informacija o podatcima koje se želi opisati (susjedne varijable su u korelaciji, translacijska inverzija...)

4 Vremenski nizovi

Vremenski nizovi su su nizovi podataka kod kojih je bitan redoslijed, obično u vremenu. U fizici većina mjerenja su upravo vremenski nizovi i opažanja je moguće opisati pomoću diferencijalnih jednadžbi. Opis nekog dinamičkog sustava temeljni je alat za analiziranje i razumijevanje sustava te omogućuje predviđanja budućih stanja. Primjeri vremenskih nizova uključuju mjerenja temperature i tlaka, položaja, brzine i akceleracije tijela, potencijala, struje... Izvan fizike, vremenski nizovi mogu biti i cijene dionice, videoprijenos kamere na samovozećem automobilu ili robotu i slično. Formalno, vremenski nizovi su nizovi podataka koji predstavljaju sukcesivne zabilježene vrijednosti jedne ili više varijabli kroz vrijeme i mogu se zapisati kao $\{x_t\}$, $t \in T$.

Kod klasične statistike primarni fokus je analiza vremenskih nizova. Analiza vremenskih nizova uključuje izradu modela koji najučinkovitije objašnjavaju ili opisuju promatranu vremensku seriju kako bi se razumjeli procesi koji stoje iza nje. Ova grana istraživanja usredotočena je na traženje razloga koji objašnjavaju podatke u vremenskom nizu. Često se pritom postavljaju pretpostavke o strukturi podataka te se vremenski niz razlaže na njezine sastavne dijelove. Kvaliteta deskriptivnog modela ocjenjuje se prema tome koliko dobro opisuje sve dostupne podatke i kakvo tumačenje pruža za bolje razumijevanje analiziranog problema.

Analiza vremenskih nizova pruža niz tehnika za bolje razumijevanje skupa podataka. Obično se kreće od dekompozicije vremenskog niza na 4 sastavna dijela:

- Razina: Osnovna vrijednost niza, koja bi bila konstantna ako bi niz bio prikazan kao ravna linija. Ovo predstavlja temeljnu referentnu točku podataka.

- Trend: Opcionalna komponenta koja odražava dugoročno povećanje ili smanjenje vrijednosti kroz vrijeme. Trend može biti linearan, nelinearan ili čak promjenjiv tijekom različitih vremenskih razdoblja, ovisno o podacima.

- Sezonalnost: Također opcionalna komponenta koja predstavlja ponavljajuće uzorke ili cikluse u ponašanju tijekom vremena. Sezonalni obrasci često su povezani s godišnjim dobima, praznicima, radnim danima ili drugim periodičnim faktorima.

- Šum: Neobjašnjiva varijabilnost u podacima koja ne može biti pripisana ni jed-

noj od gore navedenih komponenti. Šum predstavlja nasumične fluktuacije ili pogreške mjerenja koje ne slijede prepoznatljive obrasce i ne mogu se opisati.

vremenski nizovi mogu se podijeliti na univarijatne i multivarijatne. Univarijatni vremenski niz sastoji se od samo jedne varijable koja se prati kroz vrijeme. Ova vrsta niza fokusira se na analizu i modeliranje ponašanja jedne vremenski ovisne varijable. Multivarijatni vremenski nizovi uključuju dvije ili više vremenski ovisnih varijabli koje se istodobno prate tijekom vremena. Ova vrsta niza omogućuje proučavanje međusobnih odnosa i utjecaja između varijabli. Multivarijatni vremenski nizovi zahtjevniji su za analizu ne samo zbog mogućih kompleksnih interakcija između različitih varijabli, nego i potrebe za korištenjem drugih metoda poput vektorske autoregresije ili strojnog učenja.

4.1 Autoregresivni model

Autoregresivni model je alat za analizu i predviđanje vremenskih nizova koji koristi opažanja iz prethodnih vremenskih koraka kao ulaz u regresijsku jednadžbu kako bi predvidio vrijednost u sljedećem vremenskom koraku. Drugim riječima, model radi predviđanja za sljedeće vremenske korake koristeći informacije iz prošlosti istog niza. Ovaj model je odličan početak analize vremenskih nizova, ne samo zbog svoje jednostavnosti, nego i sposobnosti da točno opiše različite probleme vremenskih nizova. Autoregresivni model može se zapisati jednadžbom

$$x_t = c + \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \cdots + \alpha_l x_{t-l} + \epsilon_t, \tag{4.1}$$

gdje je x_t trenutačna vrijednost varijable, c je razina ili pristranost vremenskog niza, α_l su korelacijski koeficijenti koji pokazuju koliko prošle vrijednosti doprinose trenutačnoj i ϵ_t je šum ili rezidual koji predstavlja nepredvidive doprinose (stohastičke smetnje).

Da bi autoregresivni model bio koristan i točan, vremenski niz mora biti stacionaran - u suprotnom, vremenski niz se može diferencirati dok ne postane stacionaran. Također je bitno odabrati dobar red modela l kako bi svi bitni doprinosi bili uključeni. Metode za odabir optimalnog reda l može se odabrati korištenjem kriterija kao što su Akaikeov informacijski kriterij, Bayesov informacijski kriterij te analizama korelacija i parcijalnih korelacija. Ograničenja ovakvog modela su lošije opisivanje složenijih

ovisnosti u podacima poput nelinearnosti.

Postoje i proširenja ovog modela poput ARIMA modela [13] koji dodatno opisuju vremensku seriju i prošlim vrijednostima šuma. Vektorska autoregresija proširenje je ovog modela za analizu multivarijatnih vremenskih nizova. Međutim, svako proširenje i dodatna kompleksnost modela otežava korištenje tih modela za analizu vremenskih nizova te se često odabir najboljeg modela svodi na isprobavanje različitih modela i odabir onoga koji daje najbolje rezultate.

4.2 Modeli zasnovani na strojnom učenju

Analiza vremenskih nizova metodama strojnoga učenja omogućuje rješavanje složenih problema koji premašuju mogućnosti klasičnih statističkih modela. Dok tradicionalni modeli poput autoregresije ili ARIMA modeliraju linearne odnose i zahtijevaju stacionarnost niza, algoritmi strojnog učenja pružaju veću fleksibilnost i sposobnost hvatanja nelinearnih obrazaca i složenih međusobnih veza u podacima. Metodama strojnoga učenja mogu se također raditi regresijski i klasifikacijski modeli.

Jačanje gradijentom (gradient boosting) je tehnika strojnog učenja koja se koristi za rješavanje problema regresije, klasifikacije i drugih zadataka. Temelji se na principu izgradnje snažnog prediktivnog modela kombiniranjem velikog broja jednostavnih modela, poznatih kao slabi učenici (weak learners, najčešće plitka stabla odluke):

$$F(\boldsymbol{x}; \{\beta_m, \boldsymbol{\alpha}_m\}_1^M) = \sum_m^M \beta_m f(\boldsymbol{x}, \boldsymbol{\alpha_m}). \tag{4.2}$$

Glavna ideja gradient boostinga je iterativno dodavanje novih modela koji ispravljaju greške prethodnih smanjujući ukupnu grešku predviđanja.

Nadalje, rekurentne neuronske mreže (RNN) razvijene su kao prirodni nastavak klasičnih neuronskih mreža kako bi omogućile modeliranje sekvencijalnih podataka. Ideja RNN-a temelji se na potrebi da modeli ne samo obrađuju ulazne podatke, već i ”pamte” kontekst iz prošlih ulaza. Obične neuronske mreže tretiraju ulazne podatke kao neovisne jedne o drugima, dok RNN-ovi imaju povratne veze koje omogućuju modelu da zadrži informacije iz prethodnih vremenskih koraka, što je ključno za zadatke kao što su analiza vremenskih nizova, prepoznavanje govora i obrada jezika.

Ključna karakteristika RNN-a je interna petlja unutar arhitekture, koja omogućuje propagaciju informacija kroz vremenske korake. To znači da su izlazi iz prethodnog

vremenskog koraka povezani s ulazima trenutačnog vremenskog koraka. RNN koristi skrivene slojeve (h_t) za očuvanje stanja kroz vrijeme:

$$h_t = f(Wx_t + Uh_{t-1} + b), \tag{4.3}$$

gdje je h_t stanje u trenutačnom vremenskom koraku, x_t ulaz u trenutačnom vremenskom koraku, h_{t-1} s tanje u prethodnom vremenskom koraku, W, U, b matrice težina i pristranost, i f aktivacijska funkcija.

Iako su RNN dobar alat za obradu sekvencijalnih podataka, postoje značajna ograničenja koja otežavaju njihovu primjenu u složenijim zadacima. Jedan od glavnih problema je poteškoća u pamćenju dugoročnih ovisnosti unutar podataka. Iako su RNN osmišljene kako bi koristile informacije iz prošlih vremenskih koraka, njihov kapacitet za pamćenje tih informacija značajno opada s duljinom sekvenci. Tijekom propagacije greške unatrag kroz mrežu, gradijenti težina se mogu eksponencijalno smanjivati ili povećavati, što dovodi do problema nestajanja gradijenta (vanishing gradient problem) ili eksplodirajućeg gradijenta (exploding gradient problem). Ovi problemi čine RNN neučinkovitima za učenje dugoročnih odnosa u podacima, gdje udaljeni elementi sekvenci imaju ključnu ulogu. Osim toga, RNN se procesiraju sekvencijalno, što znači da svaki korak obrade ovisi o prethodnom. Ova ograničenja arhitekture otežavaju paralelizaciju izračuna, što rezultira sporijim treniranjem u usporedbi s drugim arhitekturama poput konvolucijskih neuronskih mreža (CNN) ili transformera. Posebno kod velikih sekvenci treniranje može postati vrlo sporo i računalno intenzivno.

RNN su s vremenom postale manje popularne u korist novijih arhitektura kao što su Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU) i transformeri, koji bolje rješavaju mnoge od navedenih problema. Te novije metode omogućuju bolju obradu dugoročnih ovisnosti, veću paralelizaciju te preciznije rezultate. Unatoč tome, RNN i dalje nalaze svoju primjenu u jednostavnijim zadacima i situacijama gdje sekvence nisu preduge, a modeli ne zahtijevaju složene strukture.

Konvolucijske neuronske mreže (CNN) tradicionalno su razvijene za obradu slikovnih podataka, ali su se pokazale izuzetno korisnima i u analizi vremenskih nizova. Njihova sposobnost prepoznavanja uzoraka kroz lokalne receptivne prostore može se učinkovito primijeniti na sekvencijalne podatke, gdje vremenski nizovi predstavljaju

jednodimenzionalne ili višedimenzionalne ulazne podatke.

CNN u analizi vremenskih nizova koriste konvolucijske slojeve za automatsko izdvajanje značajki koje predstavljaju lokalne obrasce ili trendove unutar vremenskih okvira. Svaki konvolucijski filter djeluje kao alat za detekciju specifičnih značajki koji se nauče tijekom treninga. Filteri klize kroz vremenski niz i omogućuju mreži da identificira uzorke bez obzira na njihovu točnu poziciju u vremenskom okviru. To pruža određenu razinu translacijske invarijantnosti, što znači da obrasci mogu biti prepoznati neovisno o njihovom mjestu u vremenskoj domeni. Jedna od prednosti CNN-a u analizi vremenskih nizova je njihova sposobnost da istovremeno procesiraju podatke iz više ulaznih kanala. Na primjer, u multivarijatnim vremenskim nizovima, svaki kanal može predstavljati različitu varijablu, a mreža može učiti međusobne odnose između tih varijabli. Ovo omogućuje bolju obradu složenih problema gdje je potrebno uzeti u obzir interakcije između više vremenski zavisnih varijabli. Osim toga, CNN-ovi su učinkovitiji u računalnom smislu u usporedbi s rekurentnim neuronskim mrežama. Dok RNN procesira podatke sekvencijalno, CNN obrađuje podatke paralelno, što omogućuje brže treniranje i primjenu na velikim skupovima podataka. Njihova arhitektura također smanjuje rizik od problema s gradijentima, koji su česti u RNN-ovima tijekom propagacije pogreške kroz vrijeme.

Unatoč ovim prednostima, CNN ima određena ograničenja, posebno u kontekstu obrade dugoročnih ovisnosti u podacima. Budući da se filtri fokusiraju na lokalne obrasce, mreža može imati poteškoća u prepoznavanju veza između udaljenih vremenskih točaka. Zbog toga se CNN često kombinira s drugim arhitekturama, poput rekurentnih mreža ili transformera, kako bi se povećala njihova sposobnost obrade složenih sekvencijalnih odnosa.

5 Duboki konvolucijski interpretator vremenskih nizova

Predviđanje multivarijatnih vremenskih nizova i razumijevanje njihove složene dinamike predstavlja važan i zahtjevan problem koji seže daleko izvan područja fizike, obuhvaćajući širok spektar disciplina poput financija, zdravstva i zaštite okoliša. U ovim područjima, gdje preciznost predikcija igra ključnu ulogu, čak i najmanje greške u predviđanjima mogu imati značajne posljedice. Pogreške u procjenama često se kumuliraju, što može uzrokovati dugoročne negativne učinke, bilo da je riječ o gubicima na financijskim tržištima, komplikacijama u kliničkoj njezi ili poteškoćama u upravljanju resursima okoliša.

S obzirom na važnost ovog izazova, u posljednjih nekoliko godina sve su izraženiji napori da se razviju modeli predikcije temeljeni na metodama dubokog učenja. Ovi modeli često nadmašuju tradicionalne statističke metode u točnosti predikcija zahvaljujući svojoj sposobnosti učenja složenih uzoraka u podacima. Međutim, visoka učinkovitost dolazi uz cijenu — veća složenost i netransparentnost dubokih modela otežavaju razumijevanje njihovih mehanizama donošenja odluka. Ova "crna kutija" ograničava njihovu primjenjivost u područjima u kojima je transparentnost ključna, primjerice u zdravstvu, gdje sigurnost pacijenata zahtijeva da liječnici i drugi stručnjaci mogu razumjeti logiku iza predikcija kako bi se osigurala najviša razina skrbi.

Kako bi se smanjila ta prepreka i povećalo povjerenje u modele, istraživači se počinju više usmjeravati prema razvoju objašnjivih modela dubokog učenja koji ne samo da postižu visoku točnost, nego i omogućuju bolji uvid u to kako se dolazi do predikcija. Ovi napori rezultirali su razvojem modela koji nastoje identificirati i naglasiti ključne značajke unutar ulaznih podataka, čime omogućuju korisnicima uvid u utjecaj pojedinih značajki na konačni ishod.

U zadnjih nekoliko godina, napredak u metodama objašnjivog dubokog učenja posebno je prepoznat kroz implementaciju mehanizama pažnje [14]. Mehanizmi pažnje pridonose interpretabilnosti isticanjem najrelevantnijih dijelova ulaznog vremenskog niza, što omogućava jasnije razumijevanje predikcija. Ovi mehanizmi sada su sastavni dio različitih modela dubokog učenja, od rekurentnih i konvolucijskih neuronskih mreža do transformera, te doprinose sve većoj interpretabilnosti i pri-

mjenjivosti u analizi i predikciji multivarijatnih vremenskih nizova. Također, postoje pristupi koji daju uvid u doprinos pojedinih značajki modela konačnoj predikciji poput SHAP metode [15].

U nastavku je opisan model dubokog učenja korišten u ovom diplomskom radu. Model je dizajniran specifično za predviđanje multivarijatnih vremenskih nizova uz mogućnost interpretacije rezultata. Također su predstavljeni algoritmi za automatsku optimizaciju modela koji smanjuju potrebu za ručno nadziranje modela za vrijeme treninga.

5.1 Dizajn modela

Multivarijatni vremenski nizovi (MVN) sastoje se od N vremenskih nizova dužine M te su kolektivno predstavljene s $\boldsymbol{X} = \boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_N$. Oznaka za i-ti vremenski niz sistema je $\boldsymbol{X}_i = (X_{i,1}, X_{i,2}, ..., X_{i,M})$, gdje $i = 1, 2, ..., N \in \mathbb{N}$ indeks vremenskog niza, $t = 1, 2, ..., M \in \mathbb{N}$ je indeks trenutka u vremenu za svaki vremenski niz, i $X_{i,t} \in \mathbb{R}$ označava vrijednost i-tog vremenskog niza u trenutku t.

Cilj je analizirati i dešifrirati interakcije između različitih vremenskih nizova i njihovih vremenskih zaostataka. Kako bi se to postiglo, razvijen je duboki konvolucijski interpretator za vremenske nizove (DKIvn) [16]. Ova arhitektura je usmjerena na aproksimaciju prijelaznog operatora $\boldsymbol{\alpha}$, koji opisuje evoluciju vremenskog niza između uzastopnih vremenskih točaka, koristeći L prethodnih vrijednosti u prozoru multivarijatnog vremenskog niza $\boldsymbol{Q}_{t-1}$:

$$\boldsymbol{X}_t = \boldsymbol{\alpha}\boldsymbol{Q}_{t-1}. \tag{5.1}$$

Prvi korak u primjeni DKIvn na MVN uključuje segmentaciju vremenskih nizova pomoću vremenskog prozora fiksne širine L, pri čemu je $L < M$. Svaki vremenski niz dijeli se na preklapajuće segmente tako da se za svaki vremenski niz $\boldsymbol{X}_i$ konstruira niz prozora. Niz prozora označeni s $\boldsymbol{Q}_{t-1}$ definirani su kao:

$$\boldsymbol{Q}_{t-1} = (X_{t-1}, X_{t-2}, ..., X_{t-L+1}), \quad \text{za} \quad t = L, L+1, ..., M. \tag{5.2}$$

U terminima strojnog učenja, skup podataka je predstavljen kao $\boldsymbol{Q} \in \mathbb{R}^{(M-L+1)\times N\times L}$, gdje prva dimenzija označava broj uzoraka unutar skupa podataka, a $\boldsymbol{Q}_{t-1}$ označava

pojedinačni uzorak. Cilj DKIvn je aproksimirati prijelazni operator $\boldsymbol{\alpha}$ za svaki uzorak $\boldsymbol{Q}_{t-1}$. U tu svrhu, DKIvn koristi tenzor $\boldsymbol{\alpha} \in \mathbb{R}^{N \times N \times L}$. Dimenzije tenzora $\boldsymbol{\alpha}$ usklađene su s onima pojedinačnog uzorka $\boldsymbol{Q}_{t-1}$, gdje prve dvije dimenzije predstavljaju interakciju između vremenskih nizova, a treća dimenzija odgovara prijašnjim vremenskim koracima unutar prozora. Dakle, α_{ijl} objašnjava utjecaj j-tog vremenskog niza u trenutku l na buduće stanje i-tog vremenskog niza. Jednadžba 5.1 zapisana ovim tenzorima je:

$$X_{i,t} = \sum_{j,l}^{N,L} (\boldsymbol{\alpha} \diamondsuit \boldsymbol{Q}_{t-1})_{ijl} \quad , \tag{5.3}$$

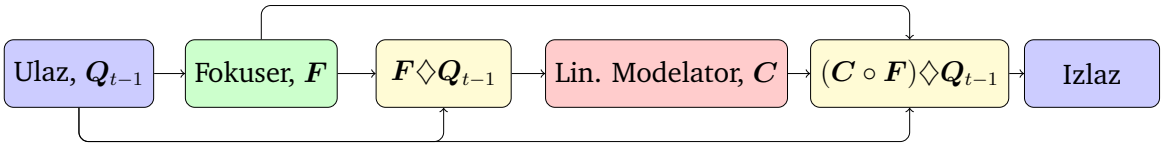
gdje operator $\diamondsuit$ označava Hadamardov umnožak s proširenjem:

$$(\boldsymbol{\alpha} \diamondsuit \boldsymbol{Q}_{t-1})_{ijk} = \alpha_{ijk} \cdot Q_{j,t-k} \, . \tag{5.4}$$

Proširenje znači da je $\boldsymbol{Q}_{t-1}$ kopiran po prvoj dimenziji tenzora $\boldsymbol{\alpha}$ što omogućuje Hadamardov umnožak, to jest elementni umnožak, te je $\boldsymbol{\alpha} \diamondsuit \boldsymbol{Q}_{t-1} \in \mathbb{R}^{N \times N \times L}$ rezultantni tenzor po kojem se sumira.

Kao što je bilo rečeno, tenzor $\boldsymbol{\alpha}$ daje uvid u dinamiku MVN sistema, odnosno omogućuje dešifriranje jednadžbi sistema koje generiraju MVN. Kako bi se ostvario taj cilj, DKIvn ima dva koraka za modeliranje predikcija. Prva faza, koja se naziva ”Fokuser”, odgovorna je za identificiranje relevantnih vremenskih nizova i značajnih vremenskih trenutaka za dinamiku sistema, odnosno filtrira signal od šuma u ulaznim podacima. Ova faza je bitna za izoliranje najutjecajnijih značajki u podacima, potrebno za predikcije i interpretabilnost modela. Nakon Fokusera, faza ”Modelator” koristi uvide stečene tijekom prve faze, zajedno s originalnim ulaznim podacima, za izračunavanje konačnih koeficijenata tenzora prijelaza $\boldsymbol{D}$. Rezultati se zatim određuju kombiniranjem ulaznih podataka s izlazima iz obje faze. Generalna shema modela prikazana je na slici 5.1.

Razni modeli koji teže interpretabilnosti u prognoziranju vremenskih nizova koriste dvofaznu arhitekturu [17, 18] i često se usredotočuju na razdvajanje autokorelacije (korelacija vremenskog niza s njegovim prijašnjim točkama) i kros-korelacije (korelacija između različitih vremenskih nizova). Ovaj pristup pruža jasnoću u razumijevanju pojedinačnih komponenti, ali ograničava sposobnost modela da prepozna složenije ovisnosti koje mogu postojati unutar podataka. DKIvn također koristi dvo-



Slika 5.1: Shema arhitekture DKIvn modela koja se sastoji od dvije faze: Fokusera i linearnog Modelatora. Fokuser u ulazu određuje najbitnije dijelove podataka - oni koji najviše utječu na izlazni rezultat. Modelator računa linearne kombinacije vremenskih nizova i prošlih trenutaka koje se koriste za računanje sljedećeg trenutka vremenskih nizova. Konačni rezultat $\boldsymbol{X}$ računa se kao produkt ulaza, Fokusera i Modelatora. Preuzeto iz [16].

faznu strukturu, ali izbjegava razdvajanje autokorelacije od kros-korelacije. Time model zadržava sposobnost istovremenog obuhvaćanja dinamike unutar pojedinačnih nizova te osigurava da model može iskoristiti takve složene interakcije bez žrtvovanja interpretabilnosti.

5.2 Fokuser i Modelator

Fokuser i Modelator koriste istu osnovnu arhitekturu prikazanu na slici 5.2, a jedina razlika je u završnom sloju.

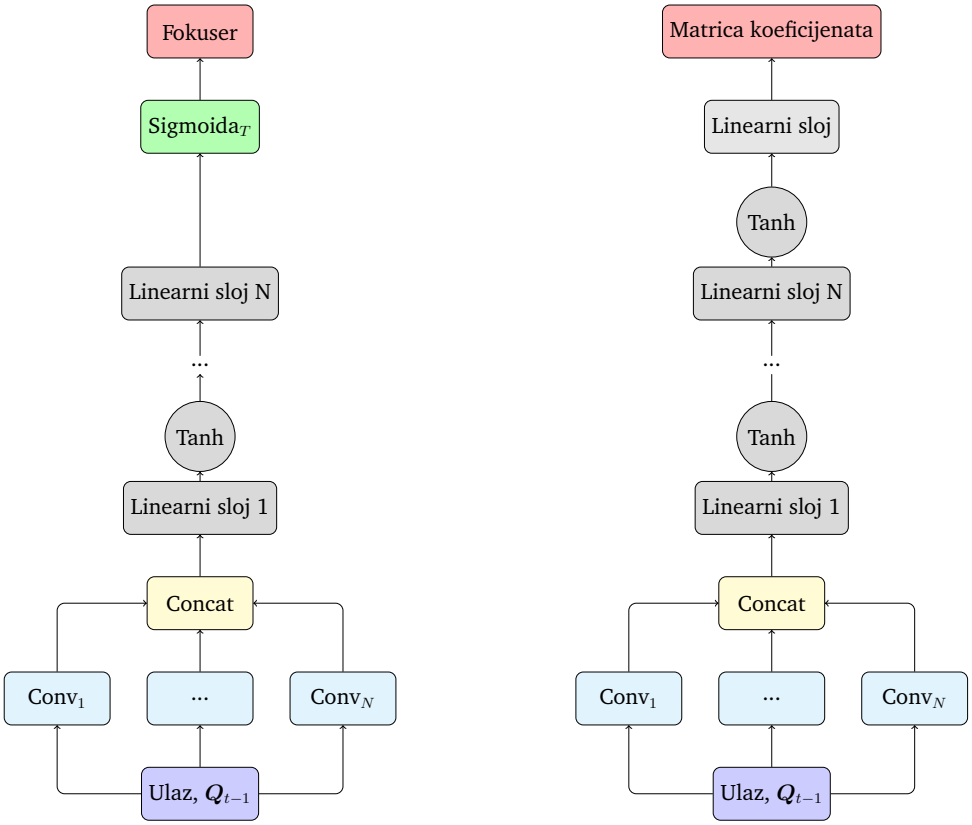
U Fokuseru i Modelatoru prvo se primjenjuje niz konvolucijskih jezgri (Conv) različitih oblika. Konvolucijama se izmjenjuju informacije između susjednih trenutaka i vremenskih nizova, i smanjuje se količina podataka koja se prosljeđuje sljedećim slojevima. Nakon primjene konvolucijskih slojeva, rezultati se spajaju (Concat). Dobiveni tenzor zatim prolazi kroz tri potpuno povezana sloja, aktivirana $\tanh$ funkcijama, što daje izlaz $\boldsymbol{H} \in \mathbb{R}^{N\times N\times L}$. To znači da svaki vremenski niz u sustavu generira matricu koja odražava oblik ulaza. Razlog za ovakvu strukturu izlaza je stvaranje prijelaznog operatora za svaki vremenski niz, gdje prva dimenzija izlaza omogućuje iteraciju kroz ulazne vremenske nizove. To se može sažeti na ovaj način:

$$\boldsymbol{K}_m = \mathrm{Conv}(\boldsymbol{Q}_t), \quad m = 1, 2, \ldots, C_N, \tag{5.5}$$

$$\boldsymbol{K} = \mathrm{Concat}([\boldsymbol{K}_1, \boldsymbol{K}_2, \ldots, \boldsymbol{K}_{C_N}]), \tag{5.6}$$

$$\boldsymbol{H} = \tanh(\boldsymbol{W}_{H_N} \tanh(...\boldsymbol{W}_2 \tanh(\boldsymbol{W}_1\boldsymbol{K} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_{H_N}). \tag{5.7}$$

Završni sloj je linearni sloj koji se koristi za izračunavanje tenzora Modelatora $\boldsymbol{C} \in$



Slika 5.2: Lijeva slika prikazuje arhitekturu Fokusera, a desna slika prikazuje Modelator. Može se primijetiti značajna sličnost u osnovnom dizajnu. Oboje počinju s nizom konvolucijskih slojeva, čiji se izlazi spajaju i ulaze u slijed potpuno povezanih linearnih slojeva s $\tanh$ aktivacijskim funkcijama. Međutim, razlika je prisutna u završnom sloju: Međutim, postoji razlika u završnom sloju: Fokuser koristi sigmoidnu funkciju kako bi eliminirao ulaze manje važnosti, učinkovito razlikujući šum od signala. S druge strane, Modeler nema nikakvu aktivacijsku funkciju, jer mu je cilj izračunati koeficijente za svaku ulaznu vrijednost. Ovaj aspekt Modelatora ga približava principima linearne regresije, tretirajući ulaze kao varijable unutar tog okvira. Odsutnost aktivacijske funkcije je namjerna, osiguravajući da modeliranje koeficijenata ostane neograničeno. To je ključno, primjerice, pri modeliranju anti-korelacije, što ne bi bilo moguće s primjenom ReLU ili sličnih aktivacijskih funkcija. Preuzeto iz [16].

$\mathbb{R}^{N \times N \times L}$:

$$\boldsymbol{C} = \boldsymbol{W}_H \boldsymbol{H} + \boldsymbol{b}_H. \tag{5.8}$$

Tenzor $\boldsymbol{C}$ omogućuje dobivanje amplitude i predznaka elemenata $\boldsymbol{\alpha}$. Tenzor $\boldsymbol{C}$ može sadržavati negativne vrijednosti, što omogućuje modeliranje antikorelacija koje postoje u sustavu. Osim toga, veće vrijednosti (u apsolutnom smislu) ukazuju na snažniji utjecaj na izlaz.

Broj konvolucijskih slojeva C_N i broj potpuno povezanih slojeva H_N može varirati i prilagoditi se specifičnim sustavima MVN. Konvolucijski slojevi korišteni u prikazanom modelu, svaki karakteriziran različitim dimenzijama koje odgovaraju različitim aspektima podataka vremenskih nizova, su:

- Globalni konvolucijski sloj: Dimenzija $N \times L$ - obrada preko svih vremenskih nizova s duljinom prozora W.
- Vremenski konvolucijski sloj: Dimenzija $1 \times L$ - fokusira se na vremenske uzorke unutar jednog prozora preko svih nizova.
- Konvolucijski sloj vremenskih nizova: Dimenzija $N \times 1$ - hvata značajke preko svih vremenskih nizova u jednom trenutku.
- Sloj prvog susjeda: Dimenzija $N \times 3$ - analizira svaki vremenski niz i njene neposredne susjede.
- Sloj drugog susjeda: Dimenzija $N \times 5$ - proširuje analizu uključujući susjede drugog reda za svaki niz.
- Sloj prvog susjeda jednog vremenskog niza: Dimenzija 1×3 - izolira jedan vremenski niz i analizira ga s njegovim neposrednim susjedima.
- Sloj drugog susjeda jednog vremenskog niza: Dimenzija 1×5 - dalje izolira jedan niz za analizu sa susjedima drugog reda.

Kako bi se dobio Fokuser $\boldsymbol{F}$, potrebno je samo primijeniti sigmoidalnu funkciju na svaki element rezultata linearnog sloja:

$$\boldsymbol{F} = \sigma_T(\boldsymbol{H}), \tag{5.9}$$

gdje $\sigma_T(x)$ sigmoidalna funkcija:

$$\sigma_T(x) = \frac{1}{1 - e^{-x}}. \tag{5.10}$$

Elementi tenzora $\boldsymbol{F}$ kreću se između 0 i 1, označeni kao $F_{ijk} \in (0, 1)$. Svrha tenzora $\boldsymbol{F}$ je razlikovanje između šuma i signala, pomažući modelu da identificira koje segmente $\boldsymbol{Q}_{t-1}$ treba istaknuti.

Postoje dvije razlike između Modelatora i Fokusera. Prvo, ulaz u Modelatoru nije $\boldsymbol{Q}_{t-1}$, već $\boldsymbol{F} \diamondsuit \boldsymbol{Q}_{t-1}$. Drugo, ne koristi sigmoidalnu funkciju kao konačnu aktivaciju, već koristi drugi linearni sloj za izračunavanje tenzora $\boldsymbol{C}$. Konačno, kako bismo dobili izlaz modela, trebamo pomnožiti ulaz s oba tenzora:

$$X_{i,t} = \sum_{j,l}^{N,L} ((\boldsymbol{C} \circ \boldsymbol{F}) \diamondsuit \boldsymbol{Q}_{t-1})_{ijl}\,, \tag{5.11}$$

gdje je $\circ$ Hadamardov produkt generaliziran na tenzore višeg reda:

$$(\boldsymbol{C} \circ \boldsymbol{F})_{ijl\ldots} = C_{ijl,\ldots} \cdot F_{ijl\ldots}. \tag{5.12}$$

Uspoređujući jednadžbe 5.1 i 5.11 vidi se da je

$$\boldsymbol{\alpha} = \boldsymbol{C} \circ \boldsymbol{F}. \tag{5.13}$$

Tenzor prijelaza je Hadamardov produkt izlaza Fokusera $\boldsymbol{F}$ i izlaza Modelatora $\boldsymbol{C}$, pri čemu je interpretabilnost već ugrađena u DKIvn. Model ne može davati predikcije bez pružanja naučene interpretabilnosti i s time je postignuta transparentnost modela. Nadalje, tenzori $\boldsymbol{F}$ i $\boldsymbol{C}$ se generiraju za svaki primjer tijekom faze treninga i predikcije (testiranja) i moguće je lokalno razumjeti zašto je model donio svaku predikciju. Interpretabilnost koju pruža model nije prosječna za sve primjere, što bi moglo izgubiti neke aspekte dinamike, već se zadržava lokalna interpretacija za svaki pojedini uzorak, to jest interpretabilnost je lokalna.

5.3 Interpretabilnost

Budući da ima $N - L + 1$ uzoraka, postoji i jednak broj $\boldsymbol{\alpha}$ koeficijenata, pri čemu svaki odgovara određenom uzorku označenom kao $\boldsymbol{\alpha}_t$. Tijekom ostatka ovog rada, osim ako nije drugačije specificirano, svako spominjanje $\boldsymbol{\alpha}$ koeficijenata ili izvedenih veličina odnosi se na koeficijente pojedinog uzorka $\boldsymbol{\alpha}_t \in \mathbb{R}^{N \times N \times L}$.

$\boldsymbol{\alpha}$ koeficijenti modela kvantificiraju utjecaj svih vremenskih nizova, uključujući i samoutjecaj, na konačni ishod u svakoj vremenskoj točki t unutar prozora, odnosno modelirajući sljedeći korak u vremenskom nizu. Pojednostavljeno rečeno, $\alpha_{i,j,l}$ predstavlja utjecaj j-tog vremenskog niza s odmakom l na i-ti vremenski niz. Postupak dobivanja interpretacije opisan je u algoritmu 1. Kako bi se poboljšala izvedba interpretabilnosti, općenito je preporučljivo nasumično prerasporediti skup za treniranje.

Kako bi se usporedili isključivo međusobni utjecaji između vremenskih nizova, stavljajući po strani vremenski aspekt njihovih interakcija, uvode se $\tilde{\boldsymbol{\beta}}$ i $\boldsymbol{\beta}$ koeficijenti. $\tilde{\boldsymbol{\beta}}$ i $\boldsymbol{\beta}$ imaju dimenzije $\mathbb{R}^{N \times N}$, i omogućuju analizu relativnog utjecaja između vremenskih nizova bez izravnog uzimanja u obzir vremenskog odmaka tih interakcija:

$$\tilde{\beta}_{ij} = \sum_{l=1}^{L} |\alpha_{ijl}|, \tag{5.14}$$

$$\beta_{ij} = \frac{\tilde{\beta}_{ij}}{\sum_{k=1}^{N} \tilde{\beta}_{ik}}. \tag{5.15}$$

Ključna razlika između $\tilde{\boldsymbol{\beta}}$ i $\boldsymbol{\beta}$ leži u normalizaciji koja se primjenjuje na potonju. $\boldsymbol{\beta}$ je normalizirana po vremenskom nizu, što znači da $\beta_{i,j}$ predstavlja proporcionalan utjecaj vremenskog niza j na vremenski niz i. To učinkovito kvantificira relativan utjecaj između vremenskih nizova. S druge strane, $\tilde{\boldsymbol{\beta}}$ je jednostavno zbroj $\boldsymbol{\alpha}$ koeficijenata kroz sve vremenske odmake i ne daje uvid u to kako je utjecaj raspodijeljen među različitim vremenskim nizovima.

Koristeći $\boldsymbol{\beta}$ koeficijente, može se odrediti koje su vremenski nizovi ključni pokretači svakog niza u sustavu. Iako apsolutna vrijednost u jednadžbi uzrokuje gubitak predznaka interakcije, ostaje veličina utjecaja svakog vremenskog niza na drugog. Nadalje, $\boldsymbol{\beta}$ vrijednosti su intuitivne - veće $\boldsymbol{\beta}$ vrijednosti ukazuju na jači utjecaj, dok manje $\boldsymbol{\beta}$ vrijednosti sugeriraju slabiji ili zanemariv utjecaj što ih čini pristupačnim za razumijevanje i široj publici.

Ako je cilj razumjeti kako su podaci generirani, tj. otkriti temeljne jednadžbe

Algoritam 1 Dobivanje interpretacije za multivarijatne vremenske nizove

Zahtjev: Skup uzoraka sa prozorima $\boldsymbol{Q}$
Zahtjev: Funkcija gubitka $\mathcal{L}$
Podijeli skup uzoraka $\boldsymbol{Q}$ na skup za treniranje $\boldsymbol{Q}_{\text{train}}$, skup za validaciju $\boldsymbol{Q}_{\textbf{valid}}$ i skup za testiranje $\boldsymbol{Q}_{\text{test}}$
Promiješaj skup za treniranje $\boldsymbol{Q}_{\text{train}}$
Treniraj model M_{DKIvn} na $\boldsymbol{Q}_{\text{train}}$
Evaluiraj model na $\boldsymbol{Q}_{\text{valid}}$ sa $\mathcal{L}$
ako se validacijski gubitak ne smanji nakon određenog broja epoha, **onda:**
    Zaustavi treniranje
za svaki $\boldsymbol{Q}_{t,\text{test}}$ u $\boldsymbol{Q}_{\text{test}}$ **:**
    Testiraj trenirani model M_{DKIvn} na $\boldsymbol{Q}_{t,\text{test}}$
    Izračunaj matrice $\boldsymbol{F}_{\text{test}}$ i $\boldsymbol{C}_{\text{test}}$ za $\boldsymbol{Q}_{t,\text{test}}$
    Izračunaj prijelazni tenzor $\boldsymbol{\alpha}_{t,\text{test}}$ kao $\boldsymbol{\alpha}_{t,\text{test}} = \boldsymbol{F}_{\text{test}} \circ \boldsymbol{C}_{\text{test}}$

koje upravljaju sustavom MVN , možemo se vratiti na $\boldsymbol{\alpha}$ koeficijente. Ovi koeficijenti pružaju izravan način rekonstrukcije jednadžbi koje generiraju podatke. Na kraju, DKIvn se ne koristi samo za predviđanje sljedećeg koraka u vremenskom nizu, već i za dešifriranje jednadžbi koje upravljaju podacima. Analizom $\boldsymbol{\alpha}$ i $\boldsymbol{\beta}$ koeficijenata, koji se dobiju algoritmom 1, može se steći dublje razumijevanje dinamike i strukture sustava.

Model se trenira sa svrhom predviđanja jer je cilj da model učinkovito uči obrasce u podacima i stvara precizne predikcije. interpretabilnost je intrinzična karakteristika ovog modela, što znači da ga nije potrebno posebno prilagođavati za razumijevanje podataka. Time model prirodno uči interpretirati obrasce i odnose u podacima kroz sam proces treniranja, bez nametanja dodatnih ograničenja ili struktura za postizanje interpretabilnosti. Osim toga, ovaj pristup osigurava i da aproksimacija vjerno odražava temeljne jednadžbe koje generiraju podatke. Ta je usklađenost ključna jer omogućuje modelu da istovremeno s predviđanjem osigura interpretabilnost. Drugim riječima, model može prepoznati različite jednadžbe koje upravljaju podacima ovisno o specifičnim ulaznim vrijednostima.

5.4 Aproksimacije višeg reda

Do sada je bio razmatran model s linearnim članom, tj. konačni izlaz bio je linearna aproksimacija dinamike. U ovom dijelu proširujemo DKIvn model kako bi uključili pristranost (bias) i članove višeg reda.

U ovom poglavlju, za preglednost, $\boldsymbol{Q}_{t-1}$ je označeno kao $\boldsymbol{q} = \boldsymbol{Q}_{t-1}$. Kako bismo

uključili članove višeg reda i pristranost, definira se operacija produkta na sekvenci tenzora $\boldsymbol{q}^{(k)}$, što omogućuje generalizaciju umnoška po elementima. Konkretno, definira se:

$$\bigcirc_{k=0}^{p} \boldsymbol{q}^{(k)} = \begin{cases} \boldsymbol{I}, & \text{ako } p = 0, \\ \boldsymbol{q}^{(1)} \circ \boldsymbol{q}^{(2)} \circ \cdots \circ \boldsymbol{q}^{(p)}, & \text{ako } p \geq 1, \end{cases} \tag{5.16}$$

pri čemu je $\boldsymbol{I}$ $n \times n$ matrica identiteta, koja predstavlja pristranost kada je $p = 0$. Ovakav pristup omogućuje uključivanje člana pristranosti u isti okvir kao i interakcije višeg reda. Jednadžba prijelaza s ovom formulacijom dana je sljedećim izrazom:

$$X_{i,t} = \sum_{p=0}^{P} \sum_{i=1}^{N} \sum_{j=1}^{L} \left((\boldsymbol{C}_p \circ \boldsymbol{F}_p) \diamondsuit \bigcirc_{k=0}^{p} \boldsymbol{q}^{(k)} \right)_{ijl}, \tag{5.17}$$

gdje je:

- Za $p = 0$ produkt $\bigcirc_{k=0}^{p} \boldsymbol{q}^{(k)} = \boldsymbol{I}$ uvodi član pristranosti b_i, označeno s $(\boldsymbol{C}_0 \circ \boldsymbol{F}_0)$.
- Za $p \geq 1$ član $\bigcirc_{k=0}^{p} \boldsymbol{q}^{(k)}$ predstavlja interakcije vremenskih nizova sa prošlim vrijednostima $\boldsymbol{q}^{(1)}, \boldsymbol{q}^{(2)}, \ldots, \boldsymbol{q}^{(p)}$.

Ova formulacija omogućuje DKIvn modelu da aproksimira interakcije između vremenskih nizova pri različitim vremenskim kašnjenjima i konstantni član pristranosti za svaku seriju unutar jedinstvenog, interpretabilnog okvira.

Oznake za koeficijente $\boldsymbol{\alpha}$ i $\boldsymbol{\beta}$ prilagođene su u $\boldsymbol{\alpha}^{(p)}$ i $\boldsymbol{\beta}^{(p)}$ radi bolje interpretabilnosti za specifične članove višeg reda. Općenito, $\boldsymbol{\alpha}^{(p)}$ za p-ti član izražava se kao:

$$\boldsymbol{\alpha}^{(p)} = \boldsymbol{C}^{(p)} \circ \boldsymbol{F}^{(\boldsymbol{p})}. \tag{5.18}$$

Općenito, bilo koji član se može dodati u model, a model će predvidjeti koeficijente uz te članove, to jest funkcije. Kako bismo dodali neki član izvan Taylorovog reda, što je opisano iznad, moramo modelu kao ulaz proslijediti vrijednosti vremenskih nizova transformirane tom funkcijom. Na primjer, ako želimo u model dodati član ovisan o $\sin(\boldsymbol{q})$, za taj član ulaz će biti $\sin(\boldsymbol{q})$, a $\boldsymbol{C}^{(sin)} \circ \boldsymbol{F}^{(sin)}$ će dati koeficijente uz elemente vremenskih nizova. Matematički, doprinos ovog člana u modelu se može zapisati kao:

$$\Delta \boldsymbol{X}_{i,t}^{(sin)} = \sum_{jl}^{NM} \left((\boldsymbol{C}^{(sin)} \circ \boldsymbol{F}^{(sin)}) \diamondsuit \boldsymbol{q}^{(sin)} \right)_{jl} \quad , \tag{5.19}$$

pri čemu je $\boldsymbol{q}^{(sin)} = \sin(\boldsymbol{q})$. Dodavanje dodatnih funkcija u ovaj model uklapa se u okvir zadan jednadžbom (5.17) i govori o potencijalu koji DKIvn model ima glede interpretabilnosti.

6 Određivanje dinamike populacije neurona

6.1 Generiranje i priprema podataka

U okviru ovog diplomskog rada DKIvn model je testiran na simuliranim podacima mreže neurona. Snimljeni potencijal pojedinih neurona u mreži neurona i predviđanje sljedeće vrijednosti potencijala su tipičan primjer MVN podataka i primjene autoregresivnih modela. Cilj ovog istraživanja je ispitati točnost predikcija DKIvn modela te, još važnije, njegovu sposobnost prepoznavanja dinamike i pravila na temelju kojih su podaci generirani.

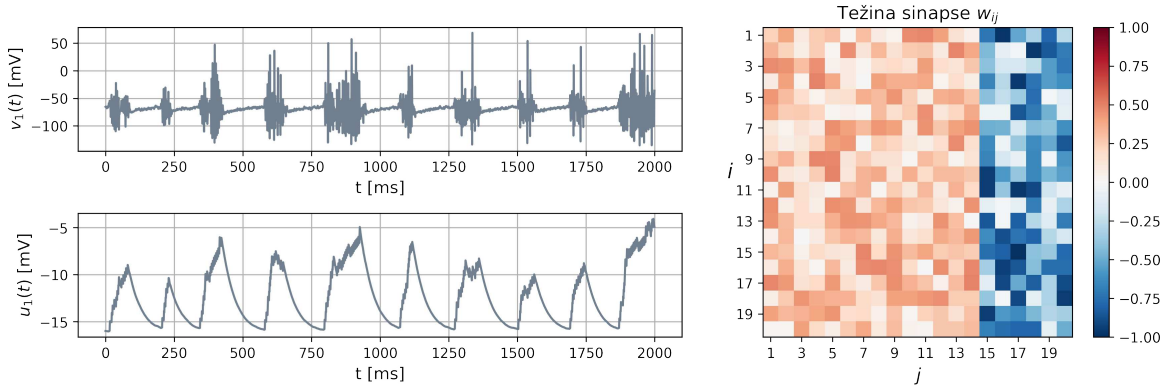
Za generiranje podataka korišten je paket Brian 2 u programskom jeziku Python. Simulacija mreže neurona izvedena je korištenjem javno dostupnog koda [19] temeljenog na Izhikievichevom modelu [7] s parametrima jednadžbi, prikazani u tablici 6.1, koji približno odgovaraju LTS neuronima (neuroni s niskim pragom pobude). Takvi parametri su odabrani jer simulirani neuroni tada pokazuju razdoblja evolucije stanja prema Izhikievichevom modelu, kao i razdoblja u kojima kontinuirano generiraju impulse, pri čemu sinapse između neurona imaju ključnu ulogu u evoluciji njihovih stanja.

v^{prag}	a	b	c	d	σ
-55	0.015	0.24	-65	1	0.05

Tablica 6.1: Parametri korišteni za generiranje podataka Izhikievichevim modelom. Parametrima je dodana nasumičnost iz normalne distribucije reda veličine 5% iznosa parametra kako ne bi bili jednaki za sve neurone.

20 neurona podijeljeno je na 14 pobuđujućih i 6 inhibirajućih neurona. Pobuđujući neuroni doprinose povećanju potencijala na povezanim neuronima, a inhibicijski ga smanjuju. Doprinos sinaptičkih veza predstavlja član I u jednadžbi (2.15), stoga pobuđujući neuroni imaju pozitivne sinapse, a inhibicijski negativne. Jačina sinapsi između neurona određena je matricom težina w, gdje su težine iz uniformne distribucije unutar intervala [0, 0.5$\rangle$ za pobuđujuće i $\langle$0, -1] za inhibirajuće.

U ovoj simulaciji modela, sinaptičke veze definirane su sljedećim redoslijedom događaja kada potencijal nekog neurona pređe prag (generira impuls): vraćanje potencijala na početnu vrijednost c, povećanje vrijednosti u za d, dodavanje konstant-



Slika 6.1: Primjer generiranih podataka. Lijevo gore: graf prvih 2000 vremenskih točaka potencijala v prvog neurona. Jasno se vide periodi generiranja impulsa i periodi neometane evolucije potencijala u vremenu. Lijevo dolje: graf prvih 2000 vremenskih točaka vremenskog niza u prvog neurona. Može se primijetiti da generiranje impulsa povećava vrijednost u, a inače vrijednost opada otprilike eksponencijalno. Desno: nasumične težine sinapsi w za 14 pobuđujućih i 6 inhibirajućih neurona. Red označava neuron na kojeg utječe sinapsa, a stupac označava neuron s kojeg dolazi impuls, to jest jačina utjecaja neurona s x-osi na neuron s y-osi.

nog člana I u jednadžbu Izhikievichevog modela sa zakašnjenjem od t_{on} ms, i gašenje člana I iz jednadžbe t_{off} ms nakon generiranja impulsa neurona. Jačine sinapsi manifestiraju se definicijom člana $I = s \cdot w$, gdje je s proizvoljna konstantna. U ovom slučaju, odabrano je $s = 25$ kako bi sinapse značajno doprinosile promjeni potencijala, a t_{on} i t_{off} su 2 i 3 ms respektivno. Gausijanski šum je dodan u jednadžbu (2.15) pri svakom koraku integracije u simulaciji.

Generiranjem podataka za $N = 20$ neurona dobiveno je 40 vremenskih nizova - 20 vremenskih nizova potencijala v i 20 vremenskih nizova u. Vremenska rezolucija je 1 ms, a vrijeme trajanja simulacije je 5 s. Stoga, za testiranje DKIvn modela imamo ukupno 40 vremenskih nizova i 5000 vremenskih trenutaka. Primjer generiranih podataka prikazan je na slici 6.1. Zadnji korak koji obuhvaća generiranje podataka je standardiziranje svakog vremenskog niza oduzimanjem srednje vrijednosti i normiranjem standardnom devijacijom:

$$x_{\text{stand}} = \frac{x - \mu_x}{\sigma_x}.$$

6.2 Pristup modeliranju podataka

Kako je glavni cilj ovoga modela prepoznavanje dinamike nekog sustava vremenskih nizova, naš zadatak je dobro razumjeti sustav koji želimo modelirati te dati modelu odgovarajući "rječnik" članova (funkcija) da bi s njima mogli opisati podatke. Konkretnije, moramo DKIvn model sastaviti od članova kojima želimo opisati podatke, a model će nam vratiti koeficijente koji stoje uz te članove i time nam omogućiti interpretaciju podataka. Ovakav pristup modeliranja podataka sličan je linearnoj ili logističkoj regresiji u strojnom učenju, iako koristimo model dubokog učenja, gdje prilagodbom modela na podatke dobivamo koeficijente uz članove. Za razliku od dubokog učenja, gdje prilagodbom kompleksne neuronske mreže na podatke ne dobivamo nikakav uvid na koji način su oni opisani, ovakvim pristupom dobivamo upravo to korištenjem kompleksnog modela.

Podaci koji će se koristiti za testiranje DKIvn modela generirani su Izhikievichevim modelom. Budući da znamo parametre kojima su generirani podaci, DKIvn će se sastojati od članova Izhikievichevog modela kako bismo provjerili je li ih moguće naučiti tijekom treninga modela.

Najprije je potrebno Izhikievichev model prilagoditi za analizu diskretnih vremenskih nizova. Stoga krećemo od jednadžbi (2.15) i (2.16) za mrežu od N neurona:

$$\begin{aligned}
&\frac{dv_i}{dt} = 0.04v_i^2 + 5v_i + 140 - u_i + \sum_j^N I_{ij}(t)\theta(v_j(t) - v_j^{prag}) \\
&\frac{du_i}{dt} = a_i(b_i v_i - u_i) \\
&\text{kada } v_i \geq v_i^{\text{prag}} : \begin{cases} v_i \leftarrow c_i \\ u_i \leftarrow u_i + d_i \end{cases},
\end{aligned} \tag{6.1}$$

gdje je sada eksplicitno dodana Heavisideova theta funkcija kraj ulazne struje I jer će u ovom slučaju taj član biti isključivo vezan uz sinaptičke veze s drugim neuronima i kao takav on će doprinositi samo kada neuroni generiraju impuls. Nadalje, dodani su indeksi i i j koji označavaju promatrani neuron i neuron izvora impulsa. Ove jednadžbe mogu se kompaktnije zapisati kao:

$$\frac{dX_i}{dt} = F(X_i, X_i^2) + \sum_j^{2N} I_{ij}(t)\theta(X_j(t) - X_j^{prag}), \tag{6.2}$$

gdje je uveden zapis blok-matricama

$$X_i = \begin{bmatrix} v_i \\ u_i \end{bmatrix}, \tag{6.3}$$

$$X_i^2 = \begin{bmatrix} v_i^2 \\ u_i^2 \end{bmatrix}, \tag{6.4}$$

$$F(X_i, X_i^2) = \begin{bmatrix} 140 + 5v_i + 0.04v_i^2 - u_i \\ a_i(b_i v_i - u_i) \end{bmatrix}, \tag{6.5}$$

$$I = \begin{bmatrix} I_{vv}(t) & I_{vu}(t) \\ I_{uv}(t) & I_{uu}(t) \end{bmatrix}. \tag{6.6}$$

Sljedeći korak je diskretizirati diferencijalne jednadžbe Eulerovom metodom:

$$X_{i,t} = X_{i,t-1} + \Delta t \left(F\left(X_{i,t-1}, X_{i,t-1}^2\right) + I_{ijl}\theta(X_{jl} - X_j^{prag}) \right), \tag{6.7}$$

a za Δt se može uzeti da je 1 jer je to vremenska rezolucija generiranih podataka. Iako bi Δt u Eulerovoj metodi trebao biti mala veličina, kasnije će se ispostaviti da je to dovoljno dobra aproksimacija. Također, dodan je novi indeks l koji označava prošle vremenske trenutke od $t-1$ do $t-1-l$.

Za kraj ostaje zapisati jednadžbe općenitije i dalje ih prilagoditi za DKIvn. Grupiranjem koeficijenata uz iste članove i dodavanjem prošlih trenutaka za svaki član koji sadrži X, ovaj vremenski niz, poštivajući Einsteinovu konvenciju o sumiranju, može se zapisati kao:

$$X_{i,t} = \alpha_i^{(0)} + \alpha_{ijl}^{(1)} X_{jl} + \alpha_{ijl}^{(2)} X_{jl}^2 + I_{ijl}\theta \left(X_{jl} - X_j^{prag} \right), \tag{6.8}$$

gdje je

- $\alpha_i^{(0)}$ konstantni član ili pristranost,
- $\alpha_{ijl}^{(1)}$ tenzor autokorelacijskih i kroskorelacijskih koeficijenata između linearnih članova i vremenskih nizova za svaki vremenski korak,
- $\alpha_{ijl}^{(2)}$ tenzor autokorelacijskih i kroskorelacijskih koeficijenata između kvadratičnih članova i vremenskih nizova za svaki vremenski korak,

- I_{ijl} konstantni član koji opisuje doprinose sinaptičkih veza između neurona u vremenu,
- θ_{jl} Heavisideova theta funkcija koja pali i gasi sinapse između neurona za svaki vremenski korak,
- X_j^{prag} prag potencijala neurona za generiranje impulsa.

Obratimo li pažnju na koeficijente uz članove, vidimo da oni trebaju biti:

$$\alpha_i^{(0)} = \begin{bmatrix} 140 \\ 0 \end{bmatrix} \tag{6.9}$$

$$\alpha_{ij0}^{(1)} = \begin{bmatrix} 1+5 & -1 \\ a_i b_i & 1-a_i \end{bmatrix}, \quad \alpha_{ijl}^{(1)} = 0 \text{ za } l \neq 0 \tag{6.10}$$

$$\alpha_{ij0}^{(2)} = \begin{bmatrix} 0.04 & 0 \\ 0 & 0 \end{bmatrix}, \quad \alpha_{ijl}^{(2)} = 0 \text{ za } l \neq 0 \tag{6.11}$$

te, ako pretpostavimo da je ulazna struja sinaptičkih veza konstantna i doprinosi samo u vremenskim koracima S, zadnji član je:

$$I_{ijk} = \begin{bmatrix} sw\delta_{lk} & 0 \\ d\delta_{1k} & 0 \end{bmatrix} \text{ za } l \in S, \text{ inače } I_{ijl} = 0, \tag{6.12}$$

gdje su elementi matrice ulaznih sinaptičkih struja. Svi neuroni imaju isti v^{prag}, stoga je

$$X_i^{prag} = \begin{bmatrix} -55 \\ 0 \end{bmatrix}. \tag{6.13}$$

Nakon kompaktnijeg zapisa i aproksimacije Eulerovom metodom, sada je jednadžbom (6.8) jasno prikazano od kojih članova će se sastojati DKIvn model. Definirat ćemo fokuser i modelator za koeficijente $\alpha^{(0)}$, $\boldsymbol{\alpha}^{(1)}$, $\boldsymbol{\alpha}^{(2)}$, $\boldsymbol{I}$ i X^{prag} na sljedeći način:

- $\alpha^{(0)} = \text{diag}(\boldsymbol{F}(1) \circ \boldsymbol{C}(\boldsymbol{F} \diamondsuit 1))$ - dijagonala matrice dimenzija $2N \times 2N$ čime se dobije vektor dimenzije $2N$. Ulaz u fokuser i modelator je vektor jedinica dimenzije $2N$ jer pristranost predviđanja ne ovisi o ulaznim varijablama.

- $\boldsymbol{\alpha}^{(1)} = \boldsymbol{F}(\boldsymbol{X}) \circ \boldsymbol{C}\left(\boldsymbol{F} \diamondsuit \boldsymbol{X}\right)$ - tenzor dimenzije $2N \times 2N \times L$, gdje je L širina prozora, odnosno broj prošlih vremenskih točaka koji uzimamo u obzir u ulazu modela $\boldsymbol{X}$ (ekvivalentno $\boldsymbol{Q}_{t-1}$ iz (5.17)). Budući da su $\boldsymbol{\alpha}_1$ koeficijenti uz linearni član, fokuser i modelator također ovise linearno o $\boldsymbol{X}$.

- $\boldsymbol{\alpha}^{(2)} = \boldsymbol{F}(\boldsymbol{X}^2) \circ \boldsymbol{C}\left(\boldsymbol{F} \diamondsuit \boldsymbol{X}^2\right)$ - tenzor dimenzije $2N \times 2N \times L$. Ulaz fokusera i modelatora su kvadratni članovi jer model predviđa koeficijente uz $\boldsymbol{X}^2$.

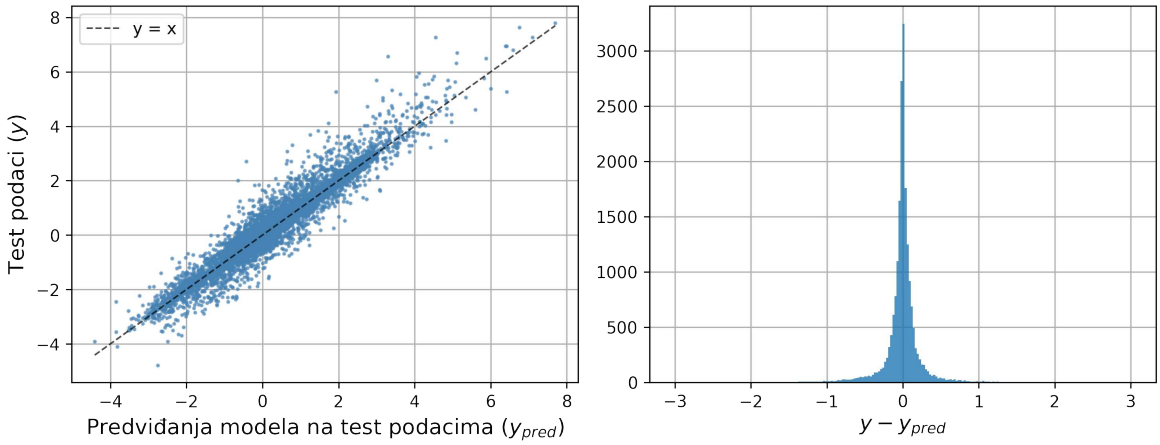
- $X^{prag} = \text{diag}(\boldsymbol{F}(1) \circ \boldsymbol{C}(\boldsymbol{F} \diamondsuit 1))$ - vektor dimenzije $2N$. Budući da prag postoji samo za prvih N vremenskih nizova, odnosno samo za potencijal, druga polovica koeficijenata neće se učiti prilikom učenja sinapsi maskiranjem elemenata theta funkcije.

- $\boldsymbol{I} = \boldsymbol{F}(\boldsymbol{\theta}) \circ \boldsymbol{C}(\boldsymbol{F} \diamondsuit \boldsymbol{\theta})$ - tenzor dimenzije $2N \times 2N \times L$, a $\boldsymbol{\theta}$ je aproksimacija theta funkcije:
$$\theta_{jl} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \sigma\left(10(X_{jl} - X_j^{prag})\right), \tag{6.14}$$
gdje je prvi član blok-vektor maska koja izolira samo doprinose potencijala, a drugi član aproksimacija theta funkcije sigmoidom. Kako doprinos ovoga člana ovisi o trenutku i indeksu neurona koji generira impuls, zapisano theta funkcijom, tako je i ulaz fokusera i modelatora aproksimacija te theta funkcije. To također znači da će se koeficijenti $\boldsymbol{I}$ učiti samo kada model "prepozna" (parametrom X^{prag}) da je pojedina sinaptička veza korištena.

Konačno, kompletan izlaz DKIvn modela je:

$$\begin{aligned} X_{i,t} = \text{diag}(&\boldsymbol{F}(\boldsymbol{1}) \circ \boldsymbol{C}(\boldsymbol{F} \diamondsuit \boldsymbol{1}))_i \\ &+ (\boldsymbol{F}(\boldsymbol{X}) \circ \boldsymbol{C}\left(\boldsymbol{F} \diamondsuit \boldsymbol{X}\right))_{ijl} X_{jl} \\ &+ \left(\boldsymbol{F}(\boldsymbol{X}^2) \circ \boldsymbol{C}\left(\boldsymbol{F} \diamondsuit \boldsymbol{X}^2\right)\right)_{ijl} \boldsymbol{X}_{jl}^2 \\ &+ (\boldsymbol{F}(\boldsymbol{\theta}) \circ \boldsymbol{C}(\boldsymbol{F} \diamondsuit \boldsymbol{\theta}))_{ijl} \theta_{jl} \end{aligned} \tag{6.15}$$

Ono po čemu se ovaj interpretativni model razlikuje od tipičnih modela dubokog učenja je da DKIvn ne računa direktno izlazne vrijednosti, nego indirektno koeficijentima uz zadane funkcije, a ti koeficijenti su predviđeni parametri koji zajedno sa zadanim funkcijama (ulaznim vrijednostima) analitički opisuju podatke prilagodbom



Slika 6.2: Lijevo: usporedba test podataka i predviđanja modela na test podacima prikazani su plavim točkama. Crna iscrtkana linija označava gdje su predviđanja i prave vrijednosti jednake. Desno: distribucija grešaka između pravih i predviđenih vrijednosti.

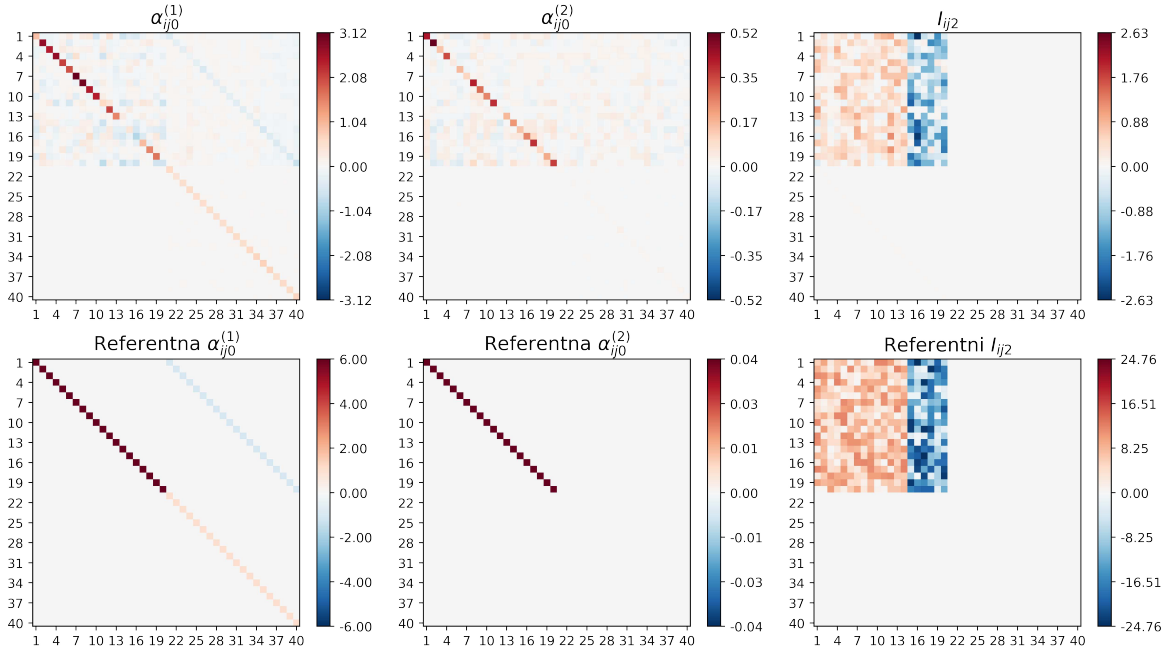
modela. U sljedećem poglavlju prikazani su rezultati i zaključci koji se mogu donijeti iz njih.

6.3 *Rezultati i interpretabilnost*

Kao što je ranije spomenuto, glavni cilj DKIvn-a je omogućiti korisniku interpretaciju podataka dešifriranjem njihove dinamike. Osim što je bitna točnost predikcija, također je bitna i usporedba između stvarnih i predviđenih parametara u modelu.

Za treniranje modela korišten je Adam optimizator s brzinom učenja od 0.0001, veličina primjera (batch size) od 256, a treniranje traje dokle god ima poboljšanja u srednjoj apsolutnoj grešci predviđanja (MAE, L1), što je ujedno i funkcija gubitka. Svi zadaci treninga i predviđanja izvedeni su na Intel® Xeon® Gold 6142 CPU-u s 252 GB RAM-a i NVIDIA GeForce RTX 2080 Ti GPU-om. Svi prikazani rezultati dobiveni su od prosjeka rezultata za 20 različitih iteracija treniranja modela, osim ako nije drugačije navedeno.

DKIvn model treniran je sa svrhom predviđanja sljedećeg koraka, stoga se njegove performanse ocjenjuju tim zadatkom. Za 20 iteracija prosječni L1 gubitak na test podacima je 0.135 sa standardnom devijacijom od 0.004 što ukazuje na stabilnosti modela tijekom treninga, a rezultati jedne takve iteracije treninga prikazani su na slici 6.2. Na lijevoj strani prikazana je usporedba test podataka i predviđanja modela gdje se vidi da su podaci dobro opisani. Na desnoj strani slike 6.2 je distribucija



Slika 6.3: Gornji red: prosječne vrijednosti parametara za 20 iteracija treniranja modela. Prikazani su samo najbitniji parametri, to jest $\alpha_{ij0}^{(1)}$, $\alpha_{ij0}^{(2)}$, I_{ij2}. Donji red: Referentne vrijednosti navedenih parametara iz (6.10), (6.11), (6.12). Vrijednosti svih parametara i njihovi rasponi prikazani su ljestvicama boja pokraj grafova.

pogrešaka predviđanja koja nalikuje Laplaceovoj distribuciji. Budući da se tijekom treninga minimizira L1 funkcija gubitka koja pretpostavlja Laplaceovu distribuciju varijabli, ovaj rezultat ima smisla.

Što se tiče interpretabilnosti podataka, DKIvn predviđa parametre slične onima u aproksimaciji Izhikievichevog modela (6.10), (6.11), (6.12). Njihova usporedba prikazana je na slici 6.3. U prvom stupcu slike za parametre $\alpha_{ij0}^{(1)}$ vidi se da DKIvn prepoznaje dijagonalne članove po blokovima, to jest prepoznaje da pojedini vremenski niz korelira isključivo s vlastitim prethodnim korakom. Točnije, DKIvn predviđa da stanje pojedinog neurona v_t i u_t evoluira kroz vrijeme neovisno o drugim neuronima i korelira s prethodnim vrijednostima v_{t-1} i u_{t-1}.

Slično se može zaključiti i u srednjem stupcu slike 6.3 za parametre $\alpha_{ij0}^{(2)}$, gdje DKIvn prepoznaje da potencijal pojedinog neurona v_t korelira s kvadratnom vrijednosti potencijala u prethodnom trenutku v_{t-1}^2. Također se može primijetiti da su predviđene vrijednosti veće za red veličine od referentnih vrijednosti. Moguće objašnjenje jest da su referentne vrijednosti slične kao i šum za parametre $\alpha_{ij0}^{(2)}$ pa se tijekom treninga parametri optimiziraju na vrijednostima red veličine iznad šuma.

U desnom stupcu slike 6.3 prikazani su parametri I_{ij2} koji rekonstruiraju dopri-

nose sinaptičkih veza i prepoznaju dvije skupine neurona. U ovome slučaju, predviđeni su parametri red veličine manji od referentnih vrijednosti. Razlog tomu može biti kaotična simulacija kada neuroni generiraju impuls. S jedne strane, zbog vremenske rezolucije od 1 ms, vrijednost potencijala neurona može vrlo brzo narasti pod utjecajem različitih doprinosa istovremeno koji se integriraju tijekom 1 ms te se gubi informacija o utjecajima između neurona. S druge strane, DKIvn uspijeva rekonstruirati sinaptičke veze bez obzira na to što ti parametri nisu aktivni cijelo vrijeme. Također, zadnji član u jednadžbi DKIvn modela (6.15) malo je kompliciraniji od ostalih dodatkom theta funkcije što također može imati utjecaj na stabilnost i točnost rezultata.

I za kraj, prosječni predviđeni pragovi potencijala svih 20 neurona za spomenutih 20 iteracija je:

$$\bar{v}^{prag} = (-55.1 \pm 0.5) \text{ mV} \tag{6.16}$$

što pokazuje da DKIvn može prepoznati uvjet praga i efektivno "paliti i gasiti" sinapse theta funkcijom u jednadžbi (6.15).

DKIvn model je iznimno prilagodljiv, točnije koeficijenti koji se predviđaju, i rezultati uvelike ovise o funkcijama kojima želimo opisati podatke. Budući da su nam, u ovom slučaju, poznate jednadžbe kojima su podaci generirani, pokušano je istim funkcijama opisati podatke i usporediti rezultate modela sa stvarnim vrijednostima. Kao što se vidi iz priloženog, DKIvn ima sposobnost prilagoditi se podacima i raditi dobra predviđanja. Istovremeno, adekvatnim odabirom funkcija, model nauči korelacije i odnose između vremenskih nizova koji su slični očekivanim vrijednostima iz aproksimacije Izhikievichevog modela. Model je prvi put ispitan na ovakvom tipu podataka koji imaju složeniju funkcijsku ovisnost i dva moda rada s uvjetom praga. Rezultati testiranja su obećavajući i istraživanje je produbilo razumijevanje sposobnosti modela te pridonijelo razvoju novih ideja.

7 Zaključak

Ovaj rad usmjeren je na ispitivanje mogućnosti nove arhitekture dubokog učenja DKIvn na multivarijatnim vremenskim nizovima. Glavna ideja ove arhitekture, to jest DKIvn modela, je omogućiti interpretaciju podataka. Ovaj model ujedinjuje fleksibilnost neuronskih mreža iz dubokog učenja s jasnim i analitičkim opisom podataka iz strojnog učenja. Arhitektura DKIvn modela uključuje dvije faze: Fokuser $\boldsymbol{F}$, koji filtrira šum i identificira najvažnije dijelove podataka relevantne za dinamiku sustava, i Modelator $\boldsymbol{C}$, koji generira koeficijente za opis evolucije sustava i omogućuje analizu uzajamnih utjecaja između vremenskih nizova, a predviđanje je umnožak ulaznih varijabli i odgovarajućih koeficijenata za svaki vremenski niz zapisan jednadžbom (5.17). Ovakav dizajn i lokalno predviđanje koeficijenata, umjesto direktnog predviđanja sljedećih vrijednosti, usmjerava model prema učenju korelacija između vremenskih nizova i međusobnih utjecaja. Sposobnost modela da pronađe stvarne odnose u podacima ovisi o našem odabiru funkcija koje želimo prilagoditi na podatke. Ako taj "rječnik funkcija" sadrži članove koji opisuju dinamiku između podataka, DKIvn će tijekom treninga naučiti koeficijente uz dane funkcije koji su usporedivi pravim vrijednostima.

U okviru ovog diplomskog rada, za primjer multivarijatnih vremenskih nizova uzeta je simulacija mreže neurona čije veličine opisuje Izhikievichev model jednadžbama (6.1). DKIvn model sastoji se od članova diskretne aproksimacije Izhikievichevog modela (6.8). Predviđanja sljedećih vrijednosti vremenskih nizova model računa jednadžbom (5.17), što se u ovom slučaju svodi na jednadžbu (6.15).

Točnost predviđanja sljedećih koraka vremenskih nizova, prikazana na slici 6.2, je dobra s prosječnom srednjom apsolutnom greškom za 20 iteracija treniranja od 0.135 i standardnom devijacijom od 0.004 što pokazuje da su rezultati stabilni. Osim točnih predikcija, model pruža interpretabilne parametre koji omogućuju bolje razumijevanje dinamike sustava. Predviđeni koeficijenti usporedivi su s očekivanim vrijednostima koeficijenata iz aproksimacije Izhikievichevog modela (6.10)-(6.13) čija je usporedba prikazana na slici 6.3. Posebno je značajno što model prepoznaje ključne karakteristike dinamičkog sustava, poput vremenske korelacije unutar nizova i doprinosa sinaptičkih veza, unatoč složenosti problema i potencijalnim ograničenjima, poput kaotičnosti simulacije i vremenske rezolucije. Prepoznati pragovi potencijala

dodatno potvrđuju da DKIvn uspješno rekonstruira ključne komponente ponašanja sustava. DKIvn ne samo da postiže visoku točnost predikcija, već također pridonosi razumijevanju dinamičkih procesa, čime se ispunjava njegov glavni cilj.

Ovi rezultati ukazuju na potencijal DKIvn-a za širu primjenu u analizi složenih vremenskih nizova. Do sada je DKIvn testiran samo na sintetičkim podacima gdje pokazuje odlične performanse, a ovi rezultati daju nadu da će biti moguće analizirati mreže neurona na stvarnim podacima. Ovaj diplomski rad doprinjeo je razumjevanju modela i daljnjim poboljšanjima DKIvn arhitekture.

Dodaci

Dodatak A Kod za generiranje podataka

```
import numpy as np
from brian2 import NeuronGroup, Synapses, SpikeMonitor, StateMonitor
from brian2 import ms, mV
from brian2 import defaultclock, run

np.random.seed(42)
Ne = 14
Ni = 6

re = np.random.normal(size=Ne)
ri = np.random.normal(size=Ni)
weights = np.hstack(
    [0.5*np.random.uniform(size=(Ne + Ni, Ne)),
     -np.random.uniform(size=(Ne + Ni, Ni))]
).T

defaultclock.dt = 1 * ms
eqs = """dv/dt = (0.04*v**2 + 5*v + 140 - u + I + I_noise)/ms : 1
         du/dt = (a*(b*v - u))/ms  : 1
         I : 1
         I_noise : 1
         a : 1
         b : 1
         c : 1
         d : 1
       """

N = NeuronGroup(
    Ne + Ni,
    eqs,
```

```
    threshold="v >= -55",
    reset="v = c; u += d",
    method="euler"
)

N.v = -65
N.I_out = 0.

N_exc = N[:Ne]
N_inh = N[Ne:]

spikemon = SpikeMonitor(N)
statemon_v = StateMonitor(N, 'v', record=True, when='after_thresholds')
statemon_u = StateMonitor(N, 'u', record=True, when='after_thresholds')
statemon_i = StateMonitor(N, 'I', record=True, when='after_thresholds')

a_val, b_val, c_val, d_val, sig = 0.015, 0.24, -65, 1, 0.05

N_exc.a = a_val*(1 + sig*re)
N_exc.b = b_val*(1 + sig*re)
N_exc.c = c_val*(1 + sig*re)
N_exc.d = d_val*(1 + sig*re)

N_inh.a = a_val*(1 + sig*ri)
N_inh.b = b_val*(1 + sig*ri)
N_inh.c = c_val*(1 + sig*ri)
N_inh.d = d_val*(1 + sig*ri)

N_exc.u = "b*v"
N_inh.u = "b*v"

S = Synapses(
    N,
    N,
```

```
    "w : 1",
    on_pre={"up": "I += 25*w", "down": "I -= 25*w"},
    delay={"up": 2 * ms, "down": 3 * ms}
)

S.connect()
S.w[:] = weights.flatten()

N_exc.run_regularly("I_noise = 1*randn()", dt=1 * ms)
N_inh.run_regularly("I_noise = 1*randn()", dt=1 * ms)

tfinal = 5000 * ms
run(tfinal)
```

Literatura

[1] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[2] Yanqiang Han, Imran Ali, Zhilong Wang, Junfei Cai, Sicheng Wu, Jiequn Tang, Lin Zhang, Jiahao Ren, Rui Xiao, Qianqian Lu, et al. Machine learning accelerates quantum mechanics predictions of molecular crystals. *Physics Reports*, 934:1–71, 2021.

[3] Hrvatska enciklopedija, mrežno izdanje. Živci, 2012-2025. Pristupljeno: 7.1.2025. `https://www.enciklopedija.hr/clanak/zivci`.

[4] Alan L Hodgkin i Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.

[5] Hugh R Wilson. Simplified dynamics of human and mammalian neocortical neurons. *Journal of theoretical biology*, 200(4):375–388, 1999.

[6] Larry F Abbott. Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain research bulletin*, 50(5-6):303–304, 1999.

[7] Eugene M Izhikevich. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003.

[8] Brian 2 development team. Interactive Brian 2 demo, 2013-2025. Pristupljeno: 7.1.2025. `http://mybinder.org/v2/gh/brian-team/brian2-binder/master?filepath=demo.ipynb`.

[9] Marcel Stimberg, Romain Brette, i Dan FM Goodman. Brian 2, an intuitive and efficient neural simulator. *elife*, 8:e47314, 2019.

[10] Simon JD Prince. *Understanding Deep Learning*. MIT press, 2023.

[11] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[12] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[13] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, i Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[14] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[15] arXiv: Scott Lundberg i Su-In Lee. A unified approach to interpreting model predictions. 2017. arXiv:1705.07874 [cs.AI].

[16] arXiv: Domjan Barić i Davor Horvatić. DCIts - Deep Convolutional Interpreter for time series. 2025. arXiv:2501.04339 [stat.ML].

[17] arXiv: Xuan-Hong Dang, Syed Yousaf Shah, i Petros Zerfos. seq2graph: Discovering dynamic dependencies from multivariate time series with multi-level attention. 2018. arXiv:1812.04448 [cs.LG].

[18] arXiv: Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, i Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. 2017. arXiv:1704.02971 [cs.LG].

[19] Akif Erdem Sağtekin i Sebastian Schmitt. Izhikievich 2003 Example, 2022. Pristupljeno: 1.8.2024. `https://brian2.readthedocs.io/en/stable/examples/frompapers.Izhikevich_2003.html`.