Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Teja Roštan

Napovedovanje časovnih vrst z nevronskimi mrežami z dolgim kratkoročnim spominom

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Matej Guid
Somentor: dr. Tom Vodopivec

Ljubljana, 2018

Zahvala

Zahvaljujem se mentorju, doc. dr. Mateju Guidu, za vso znanje, ki sem ga pridobila pod njegovim mentorstvom. Zahvaljujem se svojemu somentorju dr. Tomu Vodopivcu za napotke in nasvete pri delu. Zahvaljujem se svoji družini, ker verjame vame, in Žigu, ker mi je vedno v oporo.

Teja Roštan, 2018

Moji družini in Žigu.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
LSTM	long short term memory neural networks	nevronske mreže z dolgim kratkoročnim spominom
API	application programming interface	aplikacijski programski vmesnik
ADF	augmented Dickey–Fuller test	prirejen Dickey–Fuller test
RNN	recurrent neural network	povratne nevronske mreže
ARIMA	autoregressive integrated moving average	integrirani avtoregresijski model s premikajočimi sredinami
AR	autoregression	avtoregresija
MA	moving average	premikajoča sredina
ARMA	autoregresive moving average	avtoregresijski modeli s premikajočimi sredinami
ARIMAX	autoregressive integrated moving average with explanatory variables	integrirani avtoregresijski model s premikajočimi sredinami z dodatnimi pojasnjevalnimi spremenljivkami
ANN	artificial neural networks	umetne nevronske mreže
FNN	feed forward neural networks	usmerjene nevronske mreže
CNN	convolutional neural networks	konvolucijske nevronske mreže
CEC	unit-constant error carousel	enotsko-konstantna napaka vrtiljaka

kratica	**angleško**	**slovensko**
BPTT	back propagation through time	vzvratno razširjanje skozi čas
RTRL	real time recurrent learning	v realnem času ponavljajoče se učenje
SGD	stohastic gradient descent	stohastični gradientni spust
ADAM	adaptive moment estimation	ocena prilagodljivega trenutka
STL	seasonal and trend decomposition using Loess	dekompozicija na sezonskost in trend z Loessovo metodo
TSA	time series analysis	analiza časovnih vrst
RMSE	root mean square error	korenjena srednja kvadratna napaka
MM	mixture model	mešanje verjetnostnih porazdelitev
MML	minimal message length	načelo najkrajšega sporočila
GMM	Gaussian mixture model,	mešanje Gaussovih verjetnostnih porazdelitev

Povzetek

Naslov: Napovedovanje časovnih vrst z nevronskimi mrežami z dolgim kratkoročnim spominom

Za napovedovanje časovnih vrst je dolgo veljalo načelo, da enostavne metode v napovednih točnostih presegajo metode strojnega učenja. Vendar pa enostavni modeli ne znajo izrabljati raznovrstnih medsebojnih odvisnosti in informacij, ki jih ponujajo časovne vrste, vsebinsko sorodne ali podobne tistim, ki so predmet napovedi. Pojav masovnih podatkov je povezan tudi z zbiranjem ogromnega števila časovnih vrst, vendar pa ob uporabi enostavnih, klasičnih metod, njihov visok potencial za izboljšanje natančnosti napovedi ostaja neizkoriščen.

Nevronske mreže so dobile priložnost, da zapolnijo omenjeno vrzel. Za delo z zaporednimi podatki so primerne povratne nevronske mreže, ki pri napovedih znajo izkoriščati medsebojne odvisnosti časovnih točk. Med njimi veljajo za še zlasti uspešne pri napovedovanju časovnih vrst tako imenovane *nevronske mreže z dolgim kratkoročnim spominom*. V delu smo se osredotočili na izgradnjo in optimizacijo tega tipa nevronskih mrež. Naš namen je bil izboljšati napovedno točnost pri napovedovanju časovnih vrst ter hkrati razumeti, zakaj in koliko k temu izboljšanju prispevajo posamezni dejavniki.

Napovedovali smo število klikov na oglase na družabnem omrežju Facebook. Najprej smo analizirali različne kombinacije obdelav časovnih vrst, za katere se je izkazalo, da bi lahko vplivale na izboljšanje natančnosti napovedi. Nevronske mreže smo učili na skupini sorodnih časovnih vrst in napovedi primerjali s klasičnimi pristopi napovedovanja časovnih vrst ARIMA, ARIMAX

in VAR. Raziskali smo tudi več možnosti za izboljšanje uspešnosti napovedovanja z nevronskimi mrežami s pomočjo uporabe podobnih časovnih vrst.

Po pričakovanjih se je izkazalo, da nevronske mreže z dolgim kratkoročnim spominom ob ustrezni obdelavi podatkov dosegajo višjo napovedno točnost kot klasični modeli. Pokazali smo, da je z uporabo podobnih časovnih vrst napovedi možno še dodatno izboljšati, vendar pa ta pristop vselej ne pomaga.

Ključne besede

analiza časovnih vrst, obdelava časovnih vrst, napovedovanje, nevronske mreže z dolgim kratkoročnim spominom, spletno oglaševanje, gručenje

Abstract

Title: Time series forecasting with long short-term memory neural networks

Time series forecasting was for a long time based on the principle that simple methods in forecast accuracy exceed machine learning methods. However, simple models cannot use the various mutual dependencies and information offered by time series, content-related or similar to those that are subject to prediction. The occurrence of massive data is also connected with the collecting of enormous amounts of time series, but in using simple, traditional methods, their high potential for enhancing forecast accuracy remains untapped.

The opportunity to bridge this gap comes with neural networks. To process sequential data, recurrent neural networks are used in forecasting that can use mutual dependencies of points in time. Among recurrent neural networks, *long short-term memory neural networks* are considered as especially successful in time series forecasting. The paper focuses on the building and optimization of this neural network type. Our purpose was to improve forecast accuracy in time series forecasting, understanding at the same time why and to what degree individual factors contribute to this improvement.

The forecasting was applied to the number of clicks on Facebook ads. First, we analysed various combinations of time series processing, discovering that they might influence forecast accuracy improvements. Neural networks learned using a group of related time series and the forecasts were compared to the traditional time series forecasting approaches ARIMA, ARIMAX and VAR. We also researched a number of options to improve forecast effective-

ness with neural networks using similar time series.

As expected, we found that with adequate data processing, long short-term memory neural networks achieve greater forecast accuracy compared to the traditional models. We demonstrated that forecasting can be further improved using similar time series, however, this approach is not always helpful.

Keywords

time series analysis, time series processing, forecasting, long short-term memory neural networks, digital advertising, clustering

Poglavje 1

Uvod

Pri napovedovanju časovnih vrst so klasične, statistične metode, kot je na primer ARIMA, tipično dosegale bolj natančne napovedi od metod strojnega učenja. Te metode so preproste in jih ni težko razumeti. Dolgo je veljalo načelo, da preproste metode presegajo napredne metode strojnega učenja [1]. To načelo se je zakoreninilo na vplivnem tekmovanju napovedovanja časovnih vrst M3, ki je bilo izvedeno leta 1999 [2]. Med naprednimi metodami so bile na slabem glasu predvsem metode nevronskih mrež, saj se na tekmovanjih, takrat in kasneje, niso najbolje odrezale. Preproste metode imajo visoko pristranskost (ang. *bias*) in nizko varianco (ang. *variance*). Bolj napredne in kompleksne metode, ki imajo lahko manjšo pristranskost, bodo trpele zaradi visoke variance in posledično slabše napovedovale [3].

Njihove slabe rezultate bi lahko pripisali posameznim časovnim vrstam, ki so za nevronske mreže prekratke, da bi lahko, kot napredni in kompleksni modeli, učinkovito modelirali [1]. Povratne mreže so medtem v panogah naravne obdelave jezika, prevajalnikov in prepoznave govora začele pridobivati pozornost in že prevladujejo pri implementacijah v gospodarstvu [4, 5, 6]. Količina informacij, ki se jih da uporabiti iz kratkih časovnih vrst, je omejena. Danes so že na voljo daljše časovne vrste, vendar na področju napovedovanja časovnih vrst to ne predstavlja rešitev težav, ki jih imajo nevronske mreže. Daljše časovne vrste pomenijo, da lahko gledamo dlje v preteklost,

preteklost pa je po navadi manj uporabna za napovedovanje, saj se lahko s časom ključni vzorci in odvisnosti spremenijo. Do nedavnega je tako veljalo, da dokler časovne vrste niso dolge in stabilne skozi čas, nevronske mreže ne bodo dobro napovedovale.

Pojav masovnih podatkov (ang. *big data*) pomeni tudi, da je na voljo veliko časovnih vrst, oziroma bolj pomembno – na voljo je veliko podobnih in sorodnih časovnih vrst. Klasični modeli ne znajo izrabiti raznovrstnih medsebojnih odvisnosti in informacij, ki jih ponujajo podobne časovne vrste. Zato puščajo visok potencial natančnih napovedi neizkoriščen, saj ne morejo ujeti odvisnosti vsebinskih informacij, ki jih nosijo podobne časovne vrste. Nevronske mreže so dobile priložnost, da zapolnijo vrzel. Leta 2016 so na tekmovanju napovedovanja časovnih vrst CIF2016 [7] zmagale povratne nevronske mreže, in sicer najbolj uspešna različica povratnih nevronskih mrež, ki se imenujejo nevronske mreže z dolgim kratkoročnim spominom (ang. *long short-term memory neural networks, LSTM*).

Zaradi splošne uspešnosti LSTM so jih poskusili uporabiti v raznih panogah, kjer so se razvile različne arhitekture LSTM mrež, kot tudi postopki za obdelavo samih časovnih vrst, ustvarjanja novih in gručenje najbolj podobnih [1]. Cilj magistrskega dela je ugotoviti, kako uspešne so LSTM pri napovedovanju časovnih vrst v panogi spletnega oglaševanja. LSTM smo primerjali s klasičnimi pristopi napovedovanja časovnih vrst, ARIMA [8], ARIMAX [9] in VAR [10] . Želeli smo ugotoviti, kako različne kombinacije obdelav časovnih vrst vplivajo na izboljšanje natančnosti napovedi. Poiskali smo pogosto uporabljene pristope obdelav in jih uporabili v različnih kombinacijah. Izkoristiti smo želeli dostop do masovnih podatkov in uporabiti poln potencial LSTM. Z uporabo podobnih časovnih vrst pri učenju LSTM smo želeli ugotoviti, ali bodo prispevale k natančnosti napovedi.

V poskusih smo se omejili na spletno oglaševanje v družbenem omrežju Facebook. Današnji obseg spletnega oglaševanja je velik. Posledično se zbirajo velike količine podatkov v obliki časovnih vrst. Domena spletnega oglaševanja na družbenem omrežju Facebook se nam zdi iz tehničnega vi-

dika primerna, ker ima na voljo veliko časovnih vrst različnih dolžin, ki so tudi relativno kratke. Na voljo je večje število sorodnih oziroma soležnih časovnih vrst, kot so število prikazov oglasov, klikov na oglase, všečkanje oglasov, deljenje oglasov in podobno. Facebook vsebuje ustrezen aplikacijski programski vmesnik (ang. *Application Programming Interface, API*) za zajemanje podatkov in statistik, kar je koristno za točnost podatkov. Poleg tega pa se časovne vrste ne obnašajo naključno, saj v tem omrežju veljajo številne zakonitosti. Ne nazadnje ima ta domena tudi zelo dobro splošno prepoznavnost.

V drugem poglavju je opisano, kaj časovne vrste so ter kako jih analiziramo. V tretjem poglavju smo se osredotočili na opis sorodnih del. Za tem smo v četrtem poglavju opisali klasične statistične modele za napovedovanje časovnih vrst, in sicer ARIMA, ARIMAX ter VAR modele. V delovanje umetnih nevronskih mrež smo se poglobili v petem poglavju, kjer smo predstavili njihov razvoj, ki je pripeljal do različice LSTM. Nadaljevali smo z opisi ubranih pristopov obdelave podatkov za optimiziranje LSTM v šestem poglavju. V sedmem poglavju smo predstavili zasnovo dveh poskusov, kjer smo uporabili pristope za izboljšanje natančnosti napovedi LSTM. Osmo poglavje smo namenili predstavitvi rezultatov. Povzetek rezultatov magistrskega dela in končne ugotovitve je mogoče najti v poglavju zaključek.

Poglavje 2

Časovne vrste

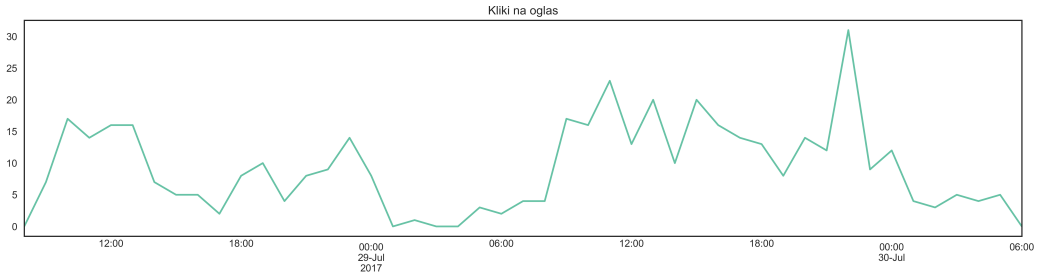
Časovna vrsta (ang. *time series*) je niz istovrstnih podatkov, ki se zbirajo skozi čas ob določenih časovnih razmikih ali izbranih trenutkih. Danes predstavljajo enega od najbolj pogostih tipov podatkov, s katerimi se srečujemo v vsakdanjem življenju. Srečamo jih pri vseh panogah družbe, kot so npr. finance, zbiranje podatkov o vremenu, kakšna je poraba električne energije, kot tudi spremljanje telesne mase. Podatki se zaradi avtomatiziranega zbiranja masovnih podatkov navadno merijo v enakih časovnih intervalih, npr. vsako uro, vsak dan, mesec, vsako leto, ni pa to pravilo.

Časovna vrsta z razliko enega samega podatka, ki pokaže statistično sliko pojava, tako predstavlja niz istovrstnih podatkov v enakih časovnih razmikih, ki opisuje sliko dinamike pojava. Zaradi te lastnosti časovnih vrst je pomembno upoštevati vrstni red podatkov, saj obstaja časovna odvisnost podatkov in bi sprememba vrstnega reda spremenila dinamiko oziroma pomen podatkov. Zaradi te lastnosti časovnih vrst analiza časovnih vrst igra veliko vlogo.

2.1 Analiza časovnih vrst

Z analizo časovnih vrst (ang. *time series analysis*) želimo določiti model, ki bi opisoval njihov vzorec oziroma zakonitosti njihovega gibanja. Tako

lahko opišemo pomembne značilnosti vzorca časovne vrste, pojasnimo, kako preteklost vpliva na prihodnost, in napovemo prihodnje vrednosti časovne vrste. Za primer analize časovnih vrst smo uporabili eno od skupin časovnih vrst, ki jih imamo na voljo. Slika 2.1 predstavlja eno od množice časovnih vrst klikov.



Slika 2.1: Primer časovnih vrst klikov na oglase.

Pri analizi časovnih vrst je dobro najprej odkrivati naslednjih lastnosti: ali je v podatkih opažen trend, ali je opažena sezonskost, kakšna je gostota klikov ali gostota spektralne moči, avtokorelacija, delna avtokorelacija in korelacija med soležnimi časovnimi vrstami.

2.1.1 Trend časovne vrste

Ali časovna vrsta vsebuje trend, se najbolj pogosto izračuna s prirejenim Dickey–Fuller testom (ang. *Augmented Dickey–Fuller test, ADF*). ADF test je statistični test, ki določa, kako močno je določen trend na časovni vrsti. Uporablja avtoregresijski model in optimizira informacijski kriterij skozi več zamikov časovne vrste. Ničelna hipoteza testa je, da časovna vrsta ni stacionarna, kar pomeni, da ima časovno odvisno strukturo – trend. ADF test časovne vrste iz slike 2.1 je predstavljen z rezultati na sliki 2.2.

Rezultati prikazujejo, da statistična vrednost ADF ni manjša od kritičnih vrednosti 1 % in 5 % intervala zaupanja, kar nakazuje, da je pri časovni vrsti s slike 2.1 prisoten trend z visoko stopnjo zaupanja.

```
ADF Statistic:  -2.630929
p-value:  0.086772
Critical Values:  1%:  -3.589
                  5%:  -2.930
                 10%:  -2.603
```

Slika 2.2: ADF test časovne vrste s slike 2.1, ki določa trend na časovni vrsti.

2.1.2 Sezonskost časovne vrste

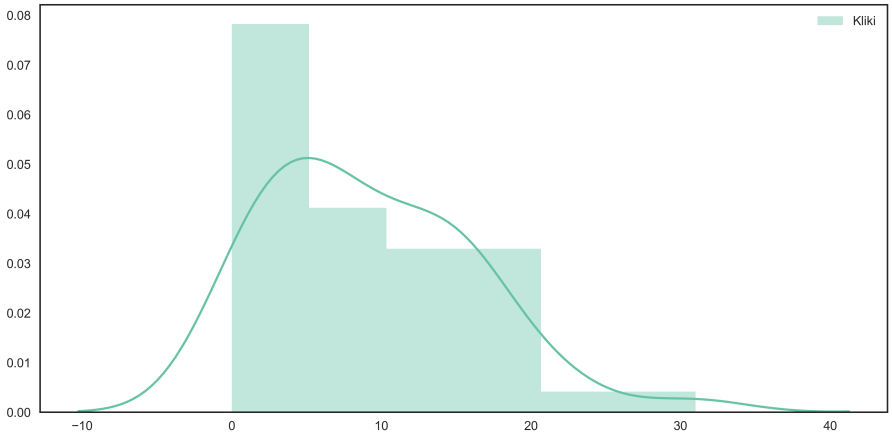
Sezonskost časovne vrste pomeni, da se v meritvah pojavljajo stalni ponavljajoči se vzorci vrhov in dolin, po navadi povezani s koledarskim časom, kot so leta, četrtletja, meseci in dnevi v tednu. Za računanje sezonskosti časovne vrste je uporabna metoda dekompozicija časovne vrste na sezonskost, trend in ostanek. Slika 2.3 prikazuje dekompozicijo časovne vrste na našem primeru časovne vrste klikov.

Dekompozicija je aditivna, kar pomeni, da je vsota kompozicij enaka originalni časovni vrsti. V primeru multiplikativne dekompozicije bi bil zmnožek dekompozicij enak originalnemu. Multiplikativno dekompozicijo je dobro uporabiti, ko se srečujemo z eksponentnimi časovnimi vrstami.

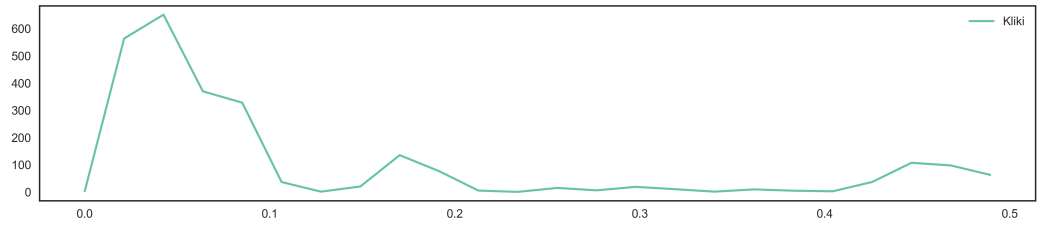
2.1.3 Gostota časovne vrste in gostota spektralne moči

Priporočeno je preveriti tudi gostoto klikov 2.4 ali gostoto spektralne moči 2.5. Slednja je zanimiva, saj prikazuje doprinos različnih frekvenčnih komponent varianci časovne vrste. Uporabili smo metodo Welch, ki časovno vrsto razdeli v segmente, vsakemu izračuna spekter in naredi povprečje rezultatov. V našem primeru je bolj zastopana predvsem nižja frekvenca časovne vrste, kar nakazuje na sezonskost.

Slika 2.3: Vrhnja časovna vrsta je časovna vrsta s slike 2.1. Naslednji trije grafi prikazujejo dekompozicijo časovne vrste na trend, ostanek in sezonskost.



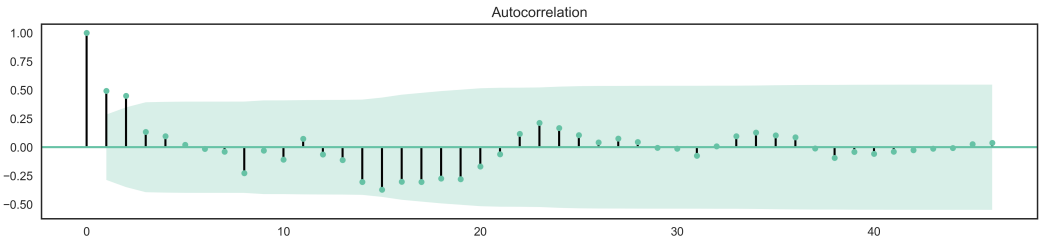
Slika 2.4: Slika prikazuje gostoto klikov časovne vrste iz slike 2.1.



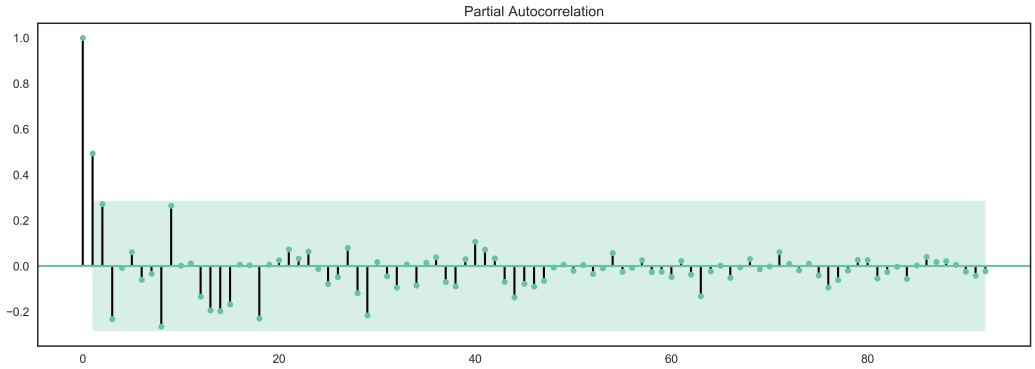
Slika 2.5: Slika prikazuje gostoto spektralne moči časovne vrste iz slike 2.1.

2.1.4 Avtokorelacija in delna avtokorelacija časovne vrste

Za napovedovanje časovnih vrst je dobro poznati moč in tip odnosa med časovno vrsto, ki jo napovedujemo, in njenimi zamiki. V statistiki temu pravimo korelacija in ker jo v tem primeru računamo nad lastno časovno vrsto pri različnih zamikih, jo imenujemo avtokorelacija. Korelacija ima vrednosti med -1 in 1. Predznak predstavlja negativno ali pozitivno korelacijo. Vrednost blizu nič predstavlja šibko korelacijo in vrednost bližje 1 ali -1 močno korelacijo. Slika 2.6 prikazuje avtokorelacijo časovne vrste klikov iz slike 2.1. Iz nje je opaziti šibko sezonskost, ki pojenja skozi čas v preteklost. Slika 2.7 medtem prikazuje delno avtokorelacijo (ang. *partial autocorrelation*), ki predstavlja korelacijo med y_t in $y_{t-zamik}$, potem ko smo odstranili linearne odvisnosti na y_{t-1}, y_{t-2}, $y_{t-zamik+1}$.



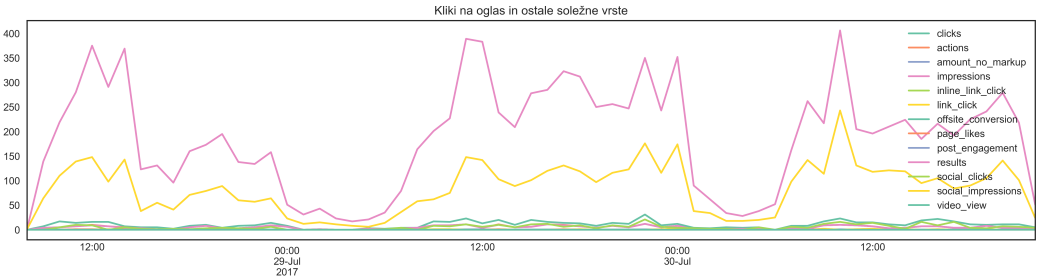
Slika 2.6: Slika prikazuje avtokorelacijo časovne vrste klikov iz slike 2.1.



Slika 2.7: Slika prikazuje delno avtokorelacijo časovne vrste klikov iz slike 2.1.

2.1.5 Korelacija med časovnimi vrstami

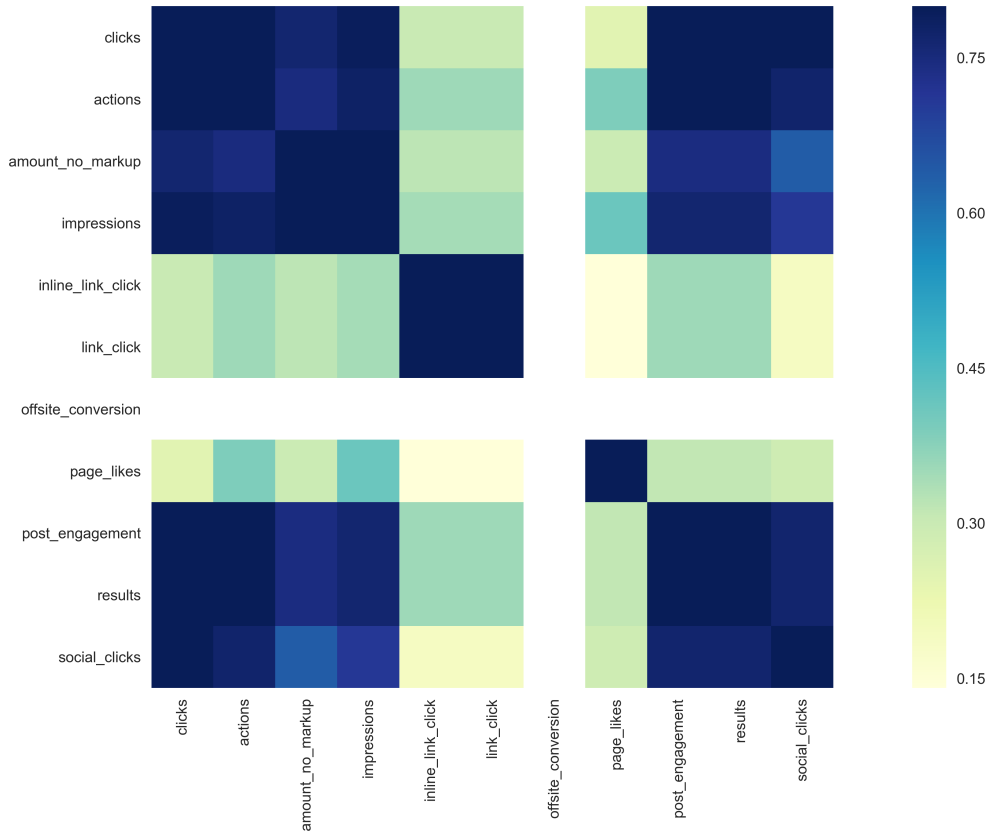
Poleg same časovne vrste klikov lahko izračunamo tudi korelacijo med soležnimi časovnimi vrstami. Kot je prikazano na sliki 2.9, ki povzame osnovne lastnosti skupine časovnih vrst, imamo poleg časovne vrste klikov (na sliki 2.8 označena pod *clicks*) 12 soležnih časovnih vrst. Korelacija med časovnimi vrstami je prikazana na matriki intenzitete 2.10 (ang. *heatmap*). Vidimo, da obstaja močna korelacija klikov z akcijami (ang. *actions)* in prikazi oglasov (ang. *impressions*), kar je pričakovano, saj se klik na oglas ne more zgoditi, če nimamo prikazanega oglasa. Prav tako lahko klik predstavlja eno od akcij.



Slika 2.8: Slika prikazuje časovno vrsto klikov na oglase iz slike 2.1 in njene soležne časovne vrste.

	count	mean	std	min	25%	50%	75%	max
clicks	63.00	10.24	6.77	0.00	5.00	10.00	15.00	31.00
actions	63.00	5.00	3.33	0.00	2.50	4.00	8.00	12.00
amount_no_markup	63.00	0.40	0.23	0.00	0.23	0.42	0.54	0.93
impressions	63.00	187.03	109.31	1.00	93.00	201.00	253.00	406.00
inline_link_click	63.00	0.29	0.55	0.00	0.00	0.00	0.00	2.00
link_click	63.00	0.29	0.55	0.00	0.00	0.00	0.00	2.00
offsite_conversion	63.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
page_likes	63.00	0.10	0.30	0.00	0.00	0.00	0.00	1.00
post_engagement	63.00	4.90	3.26	0.00	2.00	4.00	7.50	12.00
results	63.00	4.90	3.26	0.00	2.00	4.00	7.50	12.00
social_clicks	63.00	5.43	4.87	0.00	2.00	4.00	8.50	21.00
social_impressions	63.00	84.38	51.23	1.00	38.00	90.00	119.00	243.00
video_view	63.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Slika 2.9: Slika prikazuje opis ene časovne vrste klikov na oglase iz slike 2.1 in njenih soležnih časovnih vrst, ki jo je izdelala funkcija *describe* knjižnice *Pandas*.



Slika 2.10: Slika prikazuje matriko intenzitete korelacije med soležnimi časovnimi vrstami s slike 2.1.

Poglavje 3

Sorodna dela

Napovedovanje časovnih vrst (ang. *time series forecasting*) lahko delimo na dve kategoriji; na parametrično in neparametrično [11]. Med parametrične metode spada ARIMA, ki je visoko priznana metoda za gradnjo napovednih modelov. ARIMA se je izkazala za najbolj uspešno pri veliko različnih panogah družbe, kot so finance, okolje, promet ali energija, na primer uporaba ARIME za napovedovanje porabe energije na Kitajskem [12].

Med neparametrične metode spadajo umetne nevronske mreže. Predstavljajo skupino raznolikih, po navadi nelinearnih in kompleksnih modelov, ki so bili uspešno implementirani na mnogih področjih, kot so detekcija vida, obdelava naravnega jezika in prepoznava govora [13]. Medtem se nevronske mreže niso obnesle pri napovedovanju časovnih vrst. Glavni problem lahko vidimo v kompleksnosti nevronskih mrež, katere zahtevajo dolgo zgodovino časovnih vrst oziroma dovolj učnih primerov, da se lahko nevronske mreže naučijo natančno napovedovati naslednje vrednosti časovnih vrst [13]. Zbiranje masovnih podatkov jim je dalo novo priložnost.

V zadnjem času so v porastu dela, ki izkoriščajo današnje masovne podatke in skušajo s povratnimi nevronskimi mrežami (ang. *recurrent neural network, RNN*) napovedovati bolje, kot napoveduje ARIMA [14, 15, 16, 17, 18]. Njihov potencial na časovnih vrstah so odkrili, ker znajo delati z nelinearnimi podatki, ki so v resničnem svetu pogosti, in ker znajo odkrivati

soodvisnosti iz soležnih časovnih vrst. Izkazalo se je, da imajo RNN za namen napovedovanja časovnih vrst slabost v njeni arhitekturi, zaradi česar ne morejo delati s predolgimi časovnimi vrstami. Slabost RNN je, da ima v primeru predolgih časovnih vrst težavo z eksplozijo gradienta ali z izginotjem gradienta. Bolj podrobno sta ti težavi opisani v poglavju 5.

Ravno zaradi slabosti RNN sta Hochreiter in Schmidhuber [19] leta 1997 razvila različico povratnih mrež, ki se imenujejo nevronske mreže z dolgim kratkoročnim spominom (ang. *Long Short-Term Memory neural networks, LSTM*), katere niso občutljive na eksplozijo gradienta ali na izginotje gradienta v primeru dolgih časovnih vrst. V mnogih delih so v LSTM videli potencial, saj se tako kot RNN učijo na visokodimenzionalnih nelinearnih časovnih vrstah in znajo uporabljati soležne časovne vrste, poleg tega pa znajo ujeti dolge časovne odvisnosti v podatkih.

V raziskavi, kjer so napovedovali brezposelnost med mladimi v Italiji [17], so primerjali ARIMO z LSTM, kjer so pri LSTM uporabili dva različna vira podatkov. V ARIMI in LSTM so za napovedovanje brezposelnosti med mladimi v Italiji uporabili uradne podatke iz nacionalnih mesečnih anket. Kot pomožno časovno vrsto so pri LSTM uporabili tudi podatke o brezposelnosti v Italiji, ki so jih pridobili iz masovnih podatkov Google Trenda [17]. Ugotovili so, da uporaba spletnih podatkov glede iskalnih besedil na Google iskalniku oziroma splošneje – uporaba masovnih podatkov – lahko izboljša napovedovanje. Z uporabo masovnih podatkov in LSTM so izboljšali napovedovanje brezposelnosti med mladimi v Italiji, ki se je sicer merilo z ARIMO.

Ma in sodelavci [14] so se osredotočili na kratkotrajne napovedi hitrosti prometa v Pekingu. Odločili so se za uporabo LSTM, ker znajo učinkovito ujeti nelinearno dimaniko prometa. Pokazali so, da na njihovem primeru LSTM, v primerjavi z RNN, obvladajo problem izginotja gradienta pri vzvratnem postopku učenja in eksplozijo gradienta ter se tako uspešno naučijo dolge časovne odvisnosti v podatkih. Trdili so tudi, da znajo LSTM same določiti optimalne časovne zamike in to označili kot veliko pridobitev napram RNN.

Liu in sodelavci [20] so prav tako napovedovali kratkotrajne napovedi, in sicer, koliko energije je potrebne, da se vzpostavi ravnovesje med potrebo in ponudbo (ang. *load forecasting*). Iz njihovih podatkov so odkrili, da imajo časovne vrste zelo dolge periode. Uporabili so LSTM, ker znajo najbolje delati z dolgimi časovnimi odvisnostmi, kar se je pokazalo tudi na natančnosti napovedi. Pravijo, da LSTM posebno dobro delajo predvsem na podatkih z veliko različnimi periodami.

Li in sodelavci [15] so se pri napovedovanju onesnaženosti zraka poleg že opisanega v prejšnjih delih osredotočili tudi na veliko izboljšanje napovedovanja s pomočjo pomožnih zunanjih regresorjev oblike navideznih spremenljivk (ang. *dummy variables*), kot je dan v tednu, dan v mesecu, kot tudi s pomočjo vremenskih napovedi. Z LSTM so ujeli tudi prostorske korelacije v podatkih.

Tudi na področju financ, kjer so napovedovali gibanje delnic na trgu, so se LSTM izkazale bolje kot ARIMA, kar sta pokazala Fischer in Krauss [16].

Na področju spletnega oglaševanja se po navadi osredotočijo na napovedovanje prikazov oglasov ali klikov na oglase. Zhang in sodelavci [21] so upoštevali preteklo obnašanje in odvisnosti med različnimi prikazi oglasov. Uporabili so RNN, katere so se izkazale bolje kot linearna regresija, ki so jo uporabili za referenco. Dve leti kasneje so Chen in sodelavci [22] uporabili LSTM ravno zaradi uspešnosti mrež na drugih področjih in ker napram RNN nimajo težav pri izginotju gradienta ali eksploziji gradienta. Njihove časovne vrste pa so prav tako dolge in z dolgo časovno odvisnostjo. Slednje se je pokazalo tudi v natančnosti napovedovanja, saj so LSTM napovedale bolje, kot so napovedale RNN.

V naslednjem raziskovalnem delu so Liu in sodelavci [23] uporabili LSTM le za napovedovanje nizkofrekvenčnega dela časovnih vrst. Časovno vrsto so razslojili na višje frekvence in nižje frekvence. LSTM so uporabili na nižjih, saj jim na visokih, zaradi visokih stohastičnih karakteristik in zaradi kratke časovne odvisnosti LSTM, niso bolje napovedovale od ostalih nevronskih mrež. Po drugi strani so se Laptev in sodelavci [24] osredotočili ravno na napovedovanje visoko variančnih segmentov časovnih vrst s pomočjo LSTM.

V njihovem primeru je visoko nihanje časovnih vrst predvsem v času praznikov, to je času, ki je najbolj kritičen za odkrivanje napak, za načrtovanje in podobne naloge. Napovedovanje izrednih razmer je težavno, saj so odvisni od številnih zunanjih dejavnikov. LSTM so uporabili, ker znajo učinkovito uporabiti pomožne spremenljivke in ker imajo sposobnosti avtomatskega odkrivanja značilk (ang. *feature extraction*). Ugotovili so, da če vključijo velike masovne podatke skozi številne dimenzije, lahko LSTM modelirajo kompleksna medsebojna vplivanja, kar je kritično pri napovedovanju izrednih razmer. Najboljše napovedi posebnih dogodkov so dobili s posebno arhitekturo LSTM, katere poskusi so pokazali sposobnosti posploševanja in skalabilnosti. Iz njihovih izkušenj so določili tri kriterije: količina časovnih vrst, dolžina časovnih vrst in korelacija med njimi. Če so vsi visoki, potem so LSTM lahko dobra izbira, sicer je bolje izbrati klasične pristope za analizo časovnih vrst.

V naslednjem delu so se Shao in sodelavci [25] pri napovedovanju cen delnice odločili LSTM naučiti na gruči podzaporedij istega zaporedja, oziroma na lastni zgodovini cen delnice, za katero so tudi napovedovali. Menili so, da bodo tako bolje prikazali velike soodvisnosti med trendi, ki se dogajajo danes, s tistimi, ki so se zgodili davno v zgodovini. Za gručenje podzaporedij iste dolžine so uporabili metodo razvrščanja z voditelji (ang. *K-means*). Poskus z metodo gručenja se jim je izkazal bolje kot navaden pristop LSTM mreže, niso pa uporabili podobnih časovnih vrst drugih delnic. Smyl in Kuber [13] sta se osredotočila na velik pomen masovnih podatkov pri napovedovanju z LSTM. Z uporabo statističnih algoritmov časovnih vrst sta predobdelala, predvsem pa ustvarila veliko novih časovnih vrst, primernih za napovedovanje z LSTM.

V nedavno objavljenem delu so se Bandara, Bergmeir in Smyl [1] lotili učiti LSTM na skupini podobnih časovnih vrst. Uporabili so podatke CIF2016 tekmovanja in NN5 tekmovanja, kjer so bili podatki iz bančne industrije in umetno generirani podatki. Podatki so vključevali veliko časovnih vrst, saj so predpostavili, da je danes na voljo veliko podobnih časovnih

vrst. Potencial napovedovanja časovnih vrst je tako s klasičnimi univariatnimi napovednimi modeli neizkoriščen. Univariatni modeli so tisti modeli, ki za napovedovaje uporabljajo le eno časovno vrsto. Najbolj priljubljen med njimi je ARIMA. Če pa je množica časovnih vrst preveč raznolika, se lahko napovedi tudi poslabšajo. Predstavili so verjetnostni model z uporabo LSTM na podmnožicah časovnih vrst, ki so jih določili s tehnikami gručenja. Časovne vrste so gručili glede na izluščene značilke posameznih časovnih vrst. Uporabili so tudi vrsto obdelav podatkov za odstranjevanje vplivov sezonskosti, trendov in močne variance, katere so se do danes izkazale kot uspešni pristopi za izboljšanje napovedi časovnih vrst z LSTM. S tem pristopom so dosegli boljše rezultate kot pri izhodiščnih LSTM, ki so se naučile le na časovni vrsti, na kateri so napovedovale. Z izboljšanimi LSTM so zmagali na CIF2016 tekmovanju in se tako postavili na vrh pri napovedovanju časovnih vrst. Podoben pristop nameravamo uporabiti v našem delu in ga prilagoditi področju spletnega oglaševanja, katerega lahko opišemo kot nepredvidljivega. Za izluščenje značilk so se zgledovali po delu avtorjev Hyndman in sodelavci [26]. V tem delu so se osredotočili na veliko skupino časovnih vrst in iz njih poskusili izluščiti najmanj podobne, to so nenavadne časovne vrste, ki izstopajo. Za ta namen so izračunali vektor značilk za vsako časovno vrsto posebej. Značilke so vključevale sorazmerne časovne zamike, moč sezonskosti itd. Izluščene značilke bi po njihovem mnenju morale ujeti globalne informacije o časovnih vrstah.

Poglavje 4

Klasični statistični modeli za napovedovanje časovnih vrst

Za najpomembnejši klasični model za napovedovanje časovnih vrst velja integrirani avtoregresijski model s premikajočimi sredinami (ARIMA), medtem ko vektorska avtoregresija (ang. *vector autoregression, VAR*) zaradi svoje preprostosti predstavlja enega najbolj razširjenih klasičnih modelov za napovedovanje multivariatnih časovnih vrst [27]. Multivariatni modeli lahko za učenje uporabijo eno ali več časovnih vrst in napovedujejo eno ali več časovnih vrst.

4.1 ARIMA

Priljubljenost modela ARIMA (integrirani avtoregresijski model s premikajočimi sredinami) je posledica njegovih statističnih lastnosti in dobro znane metodologije Box-Jenkins [8]. Box-Jenkins metodologija predstavlja analizo časovnih vrst z namenom iskanja najbolj primernih parametrov, ki se jih uporabi v modelu ARIMA. Metodologijo predstavlja identifikacija modela, ocenjevanje parametrov modela, verifikacija ali preverjanje robustnosti modela in napovedovanje [28]. Zaradi njene priljubljenosti in uspešnosti na različnih področjih raziskovanja so nastale tudi knjižnice z dobro dokumentiranimi po-

stopki uporabe. Tako je v uporabi tudi pri manj veščih v matematični teoriji napovedovanja.

Modeli ARIMA so prilagodljivi, saj vključujejo druge vrste modelov časovnih vrst, vključno z avtoregresijo (ang. *autoregression, AR*), premikajočimi sredinami (ang. *moving average, MA*) in avtoregresijskimi modeli s premikajočimi sredinami (ang. *autoregresive moving average, ARMA*) [27]. Model ARIMA je zaradi lastnosti integriranja kot dodatek ARMA modelu med našteto zbirko modelov najbolj prilagodljiva. ARIMA z integriranjem, oziroma z diferenciacijo zgladi nestacionarne podatke v stacionarne, kar olajša napovedovanje in jo posledično naredi uspešnejšo [29].

Model AR (p) se nanaša na avtoregresivni model reda p. AR (p) se napiše kot:

$$y_t = c + a_1 y_{t-1} + ... + a_p y_{t-p} + u_t, \tag{4.1}$$

kjer so $a_1, ..., a_p$ parametri modela, c je konstanta in naključna spremenljivka u_t predstavlja beli šum.

Model MA (q) se nanaša na model premikajoče sredine reda q. MA (q) se napiše kot:

$$y_t = \mu + u_t + m_1 u_{t-1} + ... + m_q u_{t-q}, \tag{4.2}$$

kjer so $m_1, ..., m_q$ parametri modela, μ je pričakovana vrednost y_t (pogosto z vrednostjo 0) in $u_t, u_{t-1}, ..., u_{t-q}$ predstavljajo beli šum.

Model ARMA (p, q) se nanaša na avtoregresijski model s premikajočo sredino. Ta model vsebuje modela AR (p) in MA (q) in se ga napiše kot:

$$y_t = c + a_1 y_{t-1} + ... + a_p y_{t-p} + u_t + m_1 u_{t-1} + ... + m_q u_{t-q}. \tag{4.3}$$

ARIMA modeli so navadno označeni z ARIMA (p, q, d), kjer je p vrstni red modela AR, d je stopnja diferenciacije in q je vrstni red modela MA [12].

Modele ARIMA in njene zbirke modelov [8] se redko uporablja pri večdimenzionalnih multivariatnih časovnih vrstah, saj sledijo visoki računski

stroški [27].

4.2 ARIMAX

Ob uspešnih implementacijah ARIMA modelov napram standardnim ekonometričnim modelom so se ekonometriki leta 1973 [9] zaradi pomanjkanja dobre ekonomske teorije za ARIMA modelom odzvali z razvojem drugega razreda modelov, ki so prav tako vključevali avtoregresivno komponento in komponento premikajoče sredine Box-Jenkinsovega pristopa. Odzvali so se s pristopom pojasnjevalnih spremenljivk standardne ekonometrije. Najenostavnejši od teh modelov je integrirani avtoregresijski model s premikajočimi sredinami s pojasnjevalnimi spremenljivkami (ang. *explanatory variables*, X) (ARIMAX), ki je ARIMA model z dodatnimi pojasnjevalnimi spremenljivkami X, ki jih ponuja ekonomska teorija.

Enačba ARIMAX modela je podobna enačbi ARIMA modela oziroma ARMA modelu:

$$y_t = c \cdot X + a_1 y_{t-1} + ... + a_p y_{t-p} + u_t + m_1 u_{t-1} + ... + m_q u_{t-q}, \qquad (4.4)$$

kjer X predstavlja katero koli ekonometrično spremenljivko.

ARIMAX se je v nekaterih primerih izkazala kot bolj uspešen model za napovedovanje časovnih vrst [30]. Dodana vrednost ARIMAX napram ARIMA modela predstavlja vključevanje zunanjih regresorjev, kot je družbeno dojemanje časa oziroma koledarja, kot so dan v tednu, dan v mesecu, ure v dnevu, noč ali dan ter prazniki, katere je možno predvideti vnaprej. Takšne zunanje regresorje se pogosto vključuje v obliki navideznih spremenljivk (ang. *dummy variables*) [30, 15].

4.3 VAR

Model ARIMA uporablja le informacije iz podatkov ene časovne vrste, medtem ko VAR uporablja križno sorazmerje večih časovnih vrst. Modeli VAR

naravno razširjajo modele AR v multivariatno nastavitev s pomočjo vektorjev, ki ne upoštevajo odvisnosti med izhodnimi spremenljivkami [27]. Čeprav tudi ARIMAX uporablja dodatne časovne vrste, ni multivariaten model, saj so te časovne vrste pojasnjevalne spremenljivke, za katere moramo poznati prihodnost. VAR se lahko uči iz več časovnih vrst in jih več tudi napoveduje. V zadnjih letih je bil dosežen pomemben napredek pri različnih modelih VAR, vključno z eliptičnim modelom VAR za časovne vrste z dolgim repom, ki so ga predstavili Qiu in sodelovci [31] in strukturiranim modelom VAR za boljše interpretiranje odvisnosti med velikimi dimenzijskimi spremenljivkami, ki sta ga predstavila Melnyk in Banerjee [32].

Kljub temu zmogljivost modela VAR raste linearno s časovnim oknom in kvadratno s številom spremenljivk [27]. To pomeni, da je pri dolgih časovnih vrstah možnost prenasičenega učenja (ang. *overfitting*). To težavo so avtorji v delu [33] želeli rešiti z zmanjšanjem prvotno visokodimenzionalnih signalov v nižje dimenzionalne skrite predstavitve z uporabo regularizacije.

Težave VAR modela pri napovedovanju časovnih vrst lahko obravnavamo tudi kot standardne težave regresije s časovnimi parametri. Zato ni presenetljivo, da se tudi različni regresijski modeli (npr. linearna regresija) z različnimi funkcijami napake in pogoji regularizacije uporabljajo za napovedovanje časovnih vrst [27].

Model VAR reda d je definiran kot:

$$y_t = A_1 x_{t-1} + A_2 x_{t-2} + ... + A_d x_{t-d} + \epsilon_t, \tag{4.5}$$

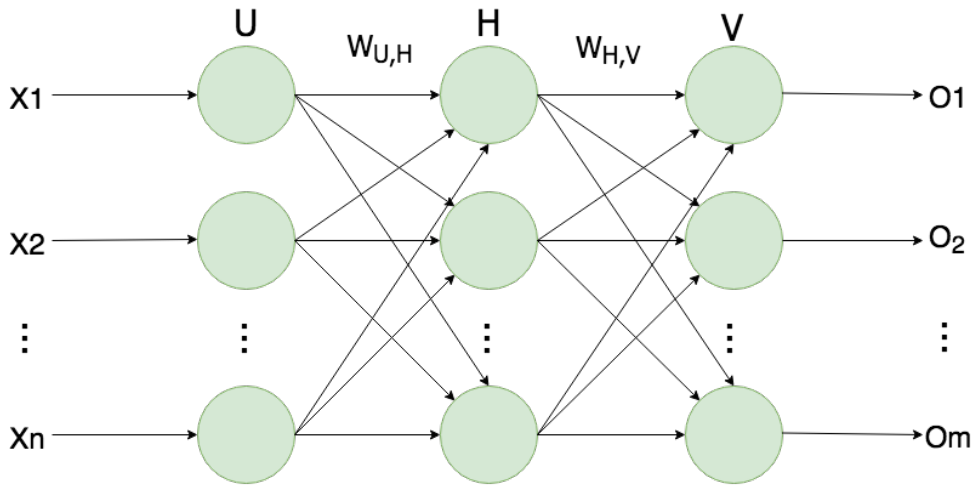
kjer so $A_1, ..., A_d \in \Re^{p \times p}$ parametri modela in $\epsilon_t \in \Re^p$ predstavlja šum naključne spremenljivke.

Poglavje 5

Umetne nevronske mreže

Danes eno izmed najbolj pogostih metod strojnega učenja predstavljajo arhitekture, ki temeljijo na umetnih nevronskih mrežah. Umetne nevronske mreže (ang. *Artificial Neural Networks, ANN*) so matematična predstavitev človeških nevronskih mrež, ki sta jih leta 1943 definirala nevropsiholog Warren McCulloch in logik Walter Pits [34]. Umetne nevronske mreže vsebujejo enote, ki se imenujejo nevroni. Koncept nevrona sta avtorja opisala kot biološko nevronsko celico, živečo v mreži nevronov. Vsak nevron sprejme vhodne signale iz izhodov drugih nevronov skozi medsebojne povezave. Vsaki povezavi je dodeljena utež in vsak nevron uporablja aktivacijsko funkcijo, ki določi izhodni signal. Vzorec povezav med nevroni predstavlja arhitekturo nevronskih mrež. Skica arhitekture je prikazana na sliki 5.1.

Nevronske mreže so v najbolj splošni arhitekturi organizirane v nivoje. Nivoji določajo skupine nevronov, ki so polno povezane s sosednjimi nivoji in po navadi niso povezane z nevroni znotraj istega nivoja. Vzorec vhodnih podatkov damo na vhodni nivo (ang. *input layer*), ki je zaporedno povezan z enim ali več skritih nivojev (ang. *hidden layer*). Računanje na skritih nivojih se izvaja na osnovi uteženih povezav. Zadnji, skriti nivo, je povezan z izhodnim nivojem (ang. *output layer*). Nevronov v vhodnem nivoju je torej toliko, kot je značilk v učnih podatkih, medtem ko je nevronov v izhodnem nivoju toliko, kot je vseh možnih ciljnih razredov. Nevroni izhodnih

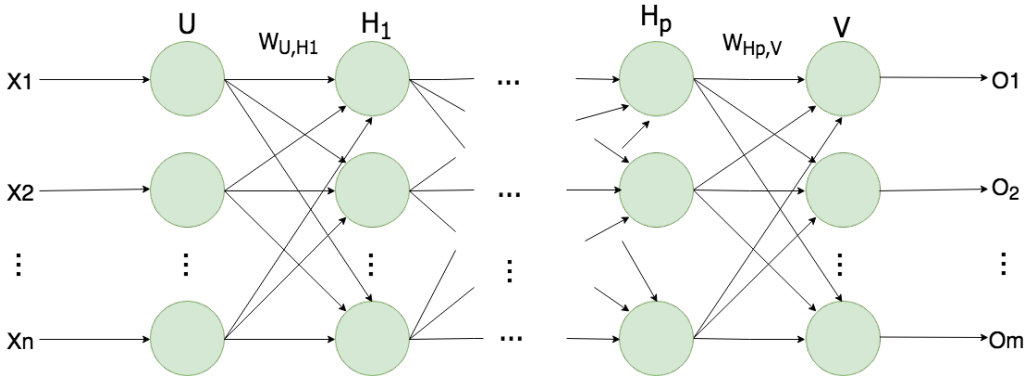


Slika 5.1: Slika prikazuje arhitekturo umetnih nevronskih mrež. Vhodni podatki $x_1, x_2, ..., x_n \in X$ gredo najprej skozi vhodni nivo U in naprej po uteženih povezavah $W_{U,H}$ do skrivnega nivoja H. Podatki potem potujejo iz skrivnega nivoja po uteženih povezavah $W_{H,V}$ do izhodnega nivoja V. Vsak nevron izhodnega nivoja vrne eno od vrednosti $o_1, o_2, ..., o_m \in O$, ki skupno predstavljajo končno napoved. Skrivnih nivojev je lahko več.

nivojev pošljejo zadnji signal kot napoved. Število skritih nivojev in uporabljenih enot na vsakem skritem nivoju uporabnik določi samostojno glede na kompleksnost problema in glede na podatke, ki jih ima v uporabi.

Izraz globokega učenja se nanaša na učenje z arhitekturo globokih nevronskih mrež in se od navadnih umetnih nevronskih mrež razlikuje z večjim številom skritih nivojev. Primer skice je prikazan na sliki 5.2. Teorija globokega učenja naslavlja prekletstvo dimenzionalnosti (ang. *the curse of dimensionality*) [35] s porazdeljenim računanjem. Globoke nevronske mreže v primerjavi z navadnimi nevronskimi mrežami lahko modelirajo globoke, kompleksne in nelinearne povezave z uporabo porazdeljene in hierarhične predstavitve značilk [36]. Do danes je globoko učenje doseglo veliko uspehov v domeni računalniškega vida, prepoznavi govora in obdelavi naravnega

jezika [11].

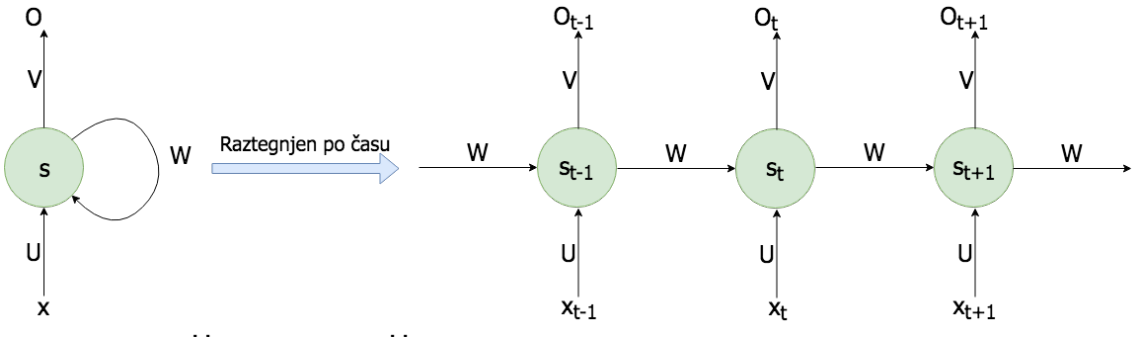


Slika 5.2: Slika prikazuje arhitekturo globokih nevronskih mrež. Vhodni podatki $x_1, x_2, ..., x_n \in X$ gredo najprej skozi vhodni nivo U in naprej po uteženih povezavah W_{U,H_1} do prvega skrivnega nivoja H_1. Podatki potem potujejo iz skrivnega nivoja H_1 po uteženih povezavah do naslednjega skrivnega nivoja H_2 in nadaljujejo do zadnjega skrivnega nivoja H_p, s katerega potujejo po uteženih povezavah $W_{Hp,V}$ do izhodnega nivoja V. Vsak nevron izhodnega nivoja vrne eno od vrednosti $o_1, o_2, ..., o_m \in O$, ki skupno predstavljajo končno napoved. Skrivnih nivojev je lahko poljubno veliko.

Tipični predstavniki nevronskih mrež so usmerjene nevronske mreže (ang. *feed forward neural networks, FNN*), konvolucijske nevronske mreže (ang. *convolutional neural networks, CNN*) in povratne nevronske mreže (ang. *Recurrent neural networks, RNN*).

5.1 Povratne nevronske mreže

RNN predstavljajo družino nevronskih mrež za obdelavo zaporednih podatkov, kot so sekvence genomskega zapisa, govor, besedila in časovne vrste. RNN mreža se od navadne nevronske mreže razlikuje po nevronih v skritem nivoju, kateri kot vhod, poleg novih signalov, sprejmejo tudi lasten, iz prejšnjega koraka vrnjen signal s časovnim zamikom. Na takšno skrito stanje



Slika 5.3: Na levi strani slike je prikazan nevron iz skritega nivoja RNN in povezave, ki grejo vanj in iz njega. Na desni strani slike imamo isti nevron, ki je raztegnjen po času. Prikazani so trije časovni koraki.

lahko gledamo kot na spomin nevronske mreže, saj lahko ujamejo informacijo o tem, kaj se je zgodilo v prejšnjih časovnih korakih. Slika 5.3 na levi strani prikazuje nevron skritega nivoja RNN in vse njegove povezave. Na desni strani slike 5.3 imamo isti nevron, raztegnjen po času za dva časovna koraka. RNN arhitektura lahko izkoristi vse razpoložljive vhodne podatke do trenutnega časa [11]. Matematični model RNN, ki z vhodnim vektorjem $v = (v_1, \ldots, v_T)$ izračuna skriti vektor $h = (h_1, \ldots, h_T)$, in izhodni vektor $y = (y_1, \ldots, y_T)$ z iteriranjem skozi $t = 1, ...T$, napišemo kot [18]:

$$h_t = \sigma(W_{i,h}x_t + W_{h,h}h_{t-1} + b_h), \tag{5.1}$$

$$y_t = W_{h_o}h_t + b_o, \tag{5.2}$$

kjer $W_{i,h}$ predstavlja matriko uteži med vhodnim in skritim nivojem, $W_{h,h}$ predstavlja matriko uteži skritega nivoja in $W_{h,o}$ matriko uteži med skritim in izhodnim nivojem. b_h in b_o predstavljata skritega in izhodnega pristranska vektorja (ang. *bias*), in σ predstavlja sigmoidno funkcijo [18]:

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \tag{5.3}$$

Dobra stran RNN arhitekture je, da svoje parametre U, V in W deli na vseh časovnih korakih, kar močno zmanjša končno število parametrov, ki jih je potrebno naučiti. Učenje RNN je podobno učenju navadnih nevronskih mrež, saj se prav tako uporablja algoritem za vzvratno razširjanje z manjšo razliko. Ker se parametri v nevronski mreži delijo skozi vse časovne korake, je gradient na vsakem izhodu odvisen ne le od izračuna trenutnega časovnega koraka, temveč tudi od prejšnjih korakov [20]. Modeli nevronskih mrež z RNN arhitekturo so tako izpostavljeni dvema težavama. Pri RNN je potrebno vnaprej določiti časovni zamik, ki bi prispeval k boljšim napovedim glede na dane podatke, kar pa zahteva precejšnje število poskusov za določitev najbolj optimalnega. Poleg tega RNN ne morejo zajeti dolgotrajnih odvisnosti v vhodnih zaporedjih, to je odvisnosti med koraki, ki so po času daleč narazen. Usposabljanje RNN na dolge časovne zamike je težavno tudi zato, ker se lahko pojavi izginotje gradienta ali eksplozija gradienta [15]. RNN zato kaže slabo delovanje pri modeliranju z dolgimi časovnimi vrstami, kar pomeni, da v skritih nevronih RNN arhitekture obstajajo pomanjkljivosti [20].

5.2 Nevronske mreže z dolgim kratkoročnim spominom

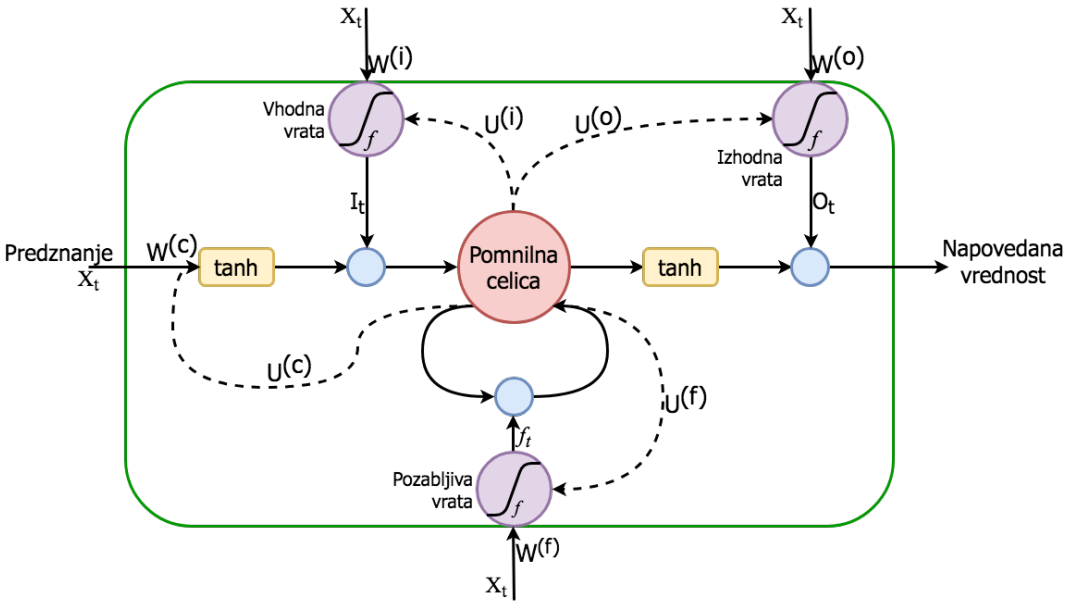
Za reševanje omenjenih težav RNN sta Hochreiter in Schmidhuber (1997) razvila posebno arhitekturo RNN z mehanizmom vrat, ki se imenuje nevronska mreža z dolgim kratkoročnim spominom (ang. *Long Short-Term Memory neural networks, LSTM*). V nasprotju s tradicionalnim RNN so LSTM sposobne učiti dolge časovne vrste, saj niso pod vplivom težave izginjajočega gradienta [15].

Hochreiter in Schmidhuber sta uvedla LSTM arhitekturo s ciljem, da bo model lahko modeliral dolgoročne časovne odvisnosti in določal optimalne časovne zamike časovnih vrst. Ker je RNN pokazal slabe napovedi pri soočanju z dolgimi časovnimi vrstami, se je LSTM štel za izboljšan pri-

stop.

LSTM se slabosti RNN izogne s posebno vrsto struktur skritih nevronov, ki jo imenujemo pomnilniški blok. Vsak blok je sestavljen iz ene ali več samopostrežnih oziroma vase povezanih pomnilniških celic in treh množilnih vrat. [18]. Primer pomnilniške celice je prikazan na sliki 5.4 in vsebuje naslednja množilna vrata: misc

- vhodna vrata, ki ščitijo stanje pomnilnika, shranjenega v vsaki pomnilniški celici, pred motnjami, ki jih povzročajo nepovezani vhodi. Analogija s pisanjem v pomnilno celico;
- izhodna vrata, ki ščitijo druge enote pred motnjami nepomembnih vsebin pomnilnika, shranjenih v celicah pomnilnika. Analogija z branjem iz pomnilne celice;
- pozabljiva vrata, ki omogočajo spominu, da pozablja nepomembne vsebine spominskih celic. Analogija s ponastavljanjem pomnilne celice.



Slika 5.4: Struktura pomnilne celice LSTM.

Vsak pomnilniški blok v svojem jedru vsebuje povratno, samopovezano, linearno, enotsko-konstantno napako vrtiljaka (ang. *unit-constant error carousel, CEC*). Aktivacija CEC predstavlja stanje celice [15, 14].

Samopovratna spominska celica lahko zaradi prisotnosti CEC zamaši katero koli zunanjo motnjo tako, da se nauči odpirati in zapirati vrata na način, ki ohranja konstantno napako mreže. Posledično ostane stanje nespremenjeno, ko se prestavimo iz ene časovne točke v naslednjo. Ravno to omogoča lastnost LSTM, da rešijo težavo izginjajočega gradienta [15, 14].

Pozabljiva vrata so bila medtem oblikovana tako, da se naučijo ponastaviti pomnilni blok, ko postane stanje zastarelo. Tako preprečijo, da bi stanje celic med nadaljnjo obdelavo še neobdelanih časovnih vrst brezmejno raslo in se tako izognejo eksploziji gradienta. Ko postane stanje zastarelo, pozabljiva vrata zamenjajo težo CEC z aktivacijo množilnih pozabljivih vrat [15, 14].

Učenje LSTM temelji na skrajšanem vzvratnim razširjanjem skozi čas (ang. *back propagation through time, BPTT*) in na spremenjeni različici v realnem času ponavljajočega se učenja (ang. *real time recurrent learning, RTRL*). Uporablja optimizacijsko funkcijo stohastičnega gradientnega spusta (ang. *stohastic gradient descent, SGD*) [14] ali njegovo nadgradnjo, oceno prilagodljivega trenutka (ang. *adaptive moment estimation, ADAM*). Funkcijo napake v našem primeru predstavlja zmanjšanje vsote kvadratnih napak.

Matematični postopek učenja LSTM

Enačbe spodaj opisujejo posodobitev pomnilne celice v LSTM nivoju ob vsaki časovni točki t. Celoten postopek smo povzeli po delu [16]. Uporabili smo naslednje notacije:

- x_t je vhodni vektor v času t;
- $W_{f,x}$, $W_{f,h}$, $W_{\tilde{s},x}$, $W_{\tilde{s},h}$, $W_{i,x}$, $W_{i,h}$, $W_{o,x}$ in $W_{o,h}$ so matrike uteži;
- b_f , $b_{\tilde{s}}$, b_i in b_o so vektorji;
- f_t , i_t in o_t so vektorji aktivacijskih vrednosti njim dodeljenih vrat;

- s_t in $\tilde{s}_t$ so vektorji stanj celic in vrednosti kandidatov;
- h_t je izhodni vektor LSTM.

Med naprej usmerjenim prehajanjem signalov se stanje celice s_t in izhodi LSTM nivojev h_t pri časovnem poteku t izračunajo po korakih, opisanih v nadaljevanju:

V prvem koraku LSTM nivo določa, katere podatke je potrebno odstraniti iz prejšnjega stanja celice s_{t-1}. Vrednosti aktivacije f_t pozabljivih vrat se tako pri časovni točki t izračunajo na podlagi trenutnega vhoda x_t glede na izhode spominskih celic iz prejšnje časovne točke h_{t-1} in glede na pristranskosti b_f pozabljivih vrat. Aktivacijska funkcija pomnoži vse aktivacijske vrednosti med 0 (popolnoma pozabil) in 1 (popolnoma zapomnil) z naslednjo enačbo:

$$f_t = sigmoid(W_{f,x}x_t + f_{f,h}h_{t-1} + b_f). \tag{5.4}$$

V drugem koraku LSTM nivo določa, katere informacije je potrebno dodati v stanja celic omrežja (s_t). Ta postopek obsega dve operaciji: najprej se izračunajo kandidatne vrednosti $\tilde{s}_t$, ki bi se lahko potencialno dodale v stanja celic, nato pa se izračunajo vrednosti vhodnih vrat po enačbi:

$$\tilde{s}_t = sigmoid(W_{\tilde{s},x}x_t + W_{\tilde{s},h}h_{t-1} + b_{\tilde{s}}), \tag{5.5}$$

$$i_t = sigmoid(W_{i,x}x_t + i_{i,h}h_{t-1} + i_i). \tag{5.6}$$

V tretjem koraku se nova stanja celic izračunajo na podlagi rezultatov prejšnjih dveh korakov s Hadamardovim produktom, ki je označen z znakom $\circ$:

$$s_t = t_t \circ s_{t-1} + t_t \circ s. \tag{5.7}$$

V zadnjem koraku se izvede izhod iz spominskih celic, kot je prikazano v naslednjih dveh enačbah:

$$o_t = sigmoid(W_{o,x}x_t + W_{o,h}h_{t-1} + b_o), \tag{5.8}$$

$$h_t = o_t \circ tanh(s_t). \tag{5.9}$$

Pri obdelavi vhodnega zaporedja časovnih vrst v LSTM mrežo so vhodne značilke predstavljene s posameznimi zaporednimi časovnimi točkami. Pri tem so vhodni podatki zaporedja v vsaki časovni točki t obdelani po postopku, ki je opisan zgoraj. Ko je zadnji element zaporedja obdelan, se vrne končni rezultat za celotno zaporedje časovnih vrst. Med učenjem se podobno kot pri navadnih nevronskih mrežah uteži prilagodijo z vzvratnim razširjanjem tako, da minimizirajo funkcijo napake [16].

Poglavje 6

Obdelava časovnih vrst za optimizacijo napovedovanja

Kot smo zasledili v pregledu sorodnih del, so v zadnjih letih LSTM največkrat uporabili za napovedovanje univariatnih časovnih vrst, to je, napovedovali so eno časovno vrsto. V ozadju raziskovalnih del je bilo tudi veliko razvoja na strani preobdelave podatkov za boljše napovedovanje z LSTM [1]. Ker nas v delu zanima, kako različni pristopi predobdelave časovnih vrst izboljšajo napovedi pri uporabi LSTM, so v nadaljevanju poglavja opisani pogosti postopki obdelave časovnih vrst za uporabo pri napovedovanju z LSTM mrežami, ki so se v raznih delih obnesli kot uspešni pri izboljšanju natančnosti napovedi.

6.1 Kodiranje z načinom ena naenkrat

V delih [15, 30] so nekatere časovne vrste preoblikovali v umetne spremenljivke, kodirane z načinom ena naenkrat (ang. *one-hot encoding*). Ta metoda vsaki možni vrednosti v časovni vrsti priredi svojo časovno vrsto na možni vrednosti 0 in 1. Časovna vrsta ima vrednost ena, ko se pri izvirni časovni vrsti priredi njena vrednost, sicer je vrednost 0. Kodiranje z načinom ena naenkrat je pogosta operacija, ki spremeni kategorične podatke v binarne. Na

primer, za indeksiranje meseca v letu imamo 12 različnih možnih vrednosti in s kodiranjem z načinom ena naenkrat dobimo 12-dimenzionalno časovno vrsto, kjer je mesec marec označen kot

$$0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0.$$

Takšen način kodiranja podatkov je značilen za več arhitektur nevronskih mrež in predvsem, ko z nevronskimi mrežami napovedujemo klasifikacijski problem. Ravno zato imamo v izhodnem nivoju toliko nevronov, kolikor je razredov, ki jih napovedujemo.

6.2 Logaritemska preslikava

V delih [1, 13] so se za obdelavo časovnih vrst odločili uporabiti logaritemsko funkcijo. Za korak predobdelave se vsako časovno vrsto pretvori v logaritemsko velikost in obliko (ang. *logarithmic scale*). Takšna obdelava podatkov je priljubljena za stabilizacijo variance v časovnih vrstah. Ker logaritem pri vrednosti 0 ni definiran oziroma je vrednost neskončna, se uporablja naslednja enačba:

$$w_t = \begin{cases} \log(y_t), & y \geq 0; \\ \log(y_t + 1), & y_t = 0; \end{cases} . \tag{6.1}$$

V fazi poobdelave podatkov, ko se želi napovedane vrednosti pretvoriti nazaj v izvorno velikost in obliko, se izračuna eksponentno funkcijo na napovedih:

$$y_t = \exp(w_t). \tag{6.2}$$

6.3 Odstranitev sezonskosti

Zgodnje raziskave kažejo, da so nevronske mreže primerne za učinkovito modeliranje osnovne sezonskosti in cikličnih vzorcev v časovnih vrstah zaradi univerzalnih lastnosti njihovih funkcij približevanja (ang. *approximation*),

kar pomeni sposobnost oceniti linearne in nelinearne funkcije [1]. Zadnje raziskave medtem trdijo, da je preobdelava časovne vrste z odstranitvijo sezonskosti potrebna zato, da dobimo natančne napovedi. V delu [37] so želeli pokazati, da odstranitev sezonskosti vpliva na izboljšanje napovedi, in sicer so to storili tako, da so primerjali napovedi iz podatkov, kjer so odstranili sezonskost s tistimi, ki jim niso, pri čemer so uporabili 68 različnih mesečnih časovnih vrst. Rezultati so pokazali, da je nevronska mreža, na kateri so bili odstranjeni vplivi sezonskosti, dosegala bolj natančne napovedi, kot pa je dosegala nevronska mreža, ki je napovedovala na podatkih, na katerih so ostali vplivi sezonskosti.

Sezonskost se iz časovnih vrst po navadi odstrani z metodami dekompozicije časovnih vrst na več komponent: trend, sezonskost in ostalo. Metode dekompozicije so lahko aditivne, kjer je vsota komponent izvorna časovna vrsta ali multiplikativne, kjer je produkt komponent izvorna časovna vrsta. Pogosti metodi za dekompozicijo sta STL (ang. *seasonal and trend decomposition using Loess*) in bolj naivna metoda z uporabo konvolucijskih filtrov. V našem poskusu smo uporabili naivno metodo, ker je dobro podprta v knjižnici *scikit-learn* [38] ter je namenjena za uporabo v programskem jeziku *Python*. Preprosta je za uporabo tako pri odstranjevanju sezonskosti kot pri prištevanju sezonskosti nazaj v časovno vrsto. Ko želimo napovedane vrednosti pretvoriti nazaj v izvorno obliko, njeno sezonskost samo prištejemo v primeru aditivne dekompozicije ali zmnožimo v primeru multiplikativne dekompozicije.

6.4 Odstranitev trenda

Če je v časovnih vrstah najdena sezonskost, pomeni, da časovna vrsta ni sezonsko stacionarna, ali splošneje, ni stacionarna. Enako velja za trend časovne vrste. Rečemo lahko, da nevronske mreže niso sposobne učinkovito modelirati nestacionarnih časovnih vrst, saj pretvorba časovne vrste iz nestacionarne v stacionarne zmanjša napovedno napako nevronskih mrež. Te

ugotovitve so najverjetneje povezane na omejitve, ki jih imamo pri dostopnosti podatkov. V nekaterih delih so se odločili, da bodo ustvarili stacionarne podatke z odstranitvijo sezonskosti [1], medtem ko so v naslednjem delu ugotovili, da jim je odstranitev trenda dala boljše rezultate kot pa odstranitev sezonskosti [24]. Odstranitev trenda se po navadi naredi z razliko prvega reda [17], njen inverz pa se naredi s seštevanjem razlik.

6.5 Normalizacija

Običajno je potrebna tudi normalizacija, saj nevronske mreže, ki uporabljajo splošč̌ljive funkcije (ang. *squashing functions*), kot sta hiperbolični tangens in sigmoidna funkcija, ne znajo dobro delati z vhodi, ki so precej zunaj intervala med -1 in 1. Tudi če nevronske mreže ne bi imele teh omejitev, je normalizacija potrebna zato, ker se učimo iz več časovnih vrst različnih amplitud [13]. Za normalizacijo se najpogosteje uporablja standardizacija, kar pomeni, da se podatkom odstrani povprečje in se jih pretvori v enotsko varianco – varianca podatkov je ena. Standardizira se vsako značilko posebej.

Poglavje 7

Opis poskusov

V naslednjem poglavju sta predstavljeni dve skupini poskusov, ki sta bili izvedeni z namenom, da najdemo dejavnike, ki vplivajo na uspešnost LSTM pri napovedovanju časovnih vrst. Klasični metodi ARIMA in VAR sta uporabljeni kot merilo uspešnosti LSTM, vztrajnostni model pa kot naivna metoda. Želeli smo preizkusiti različne pristope izboljšanja napovedovanja, saj smo želeli razumeti zakaj in koliko posamezen dejavnik vodi do bolj natančnih napovedi.

Za prve dejavnike izboljšanja natančnosti napovedovanja časovnih vrst smo se osredotočili na pristope obdelav časovnih vrst. Za njih smo se odločili, ker so se ti do danes najbolj izkazali pri izboljšanju natančnosti napovedovanja LSTM. V prvi skupini poskusov smo uporabili pristope obdelav časovnih vrst za izboljšanje napovedovanja, ki smo jih opisali v poglavju 6. Da je bila primerjava med LSTM in klasičnimi modeli pravična, smo tudi klasične modele učili na istih kombinacijah obdelovanja časovnih vrst za izboljšanje natančnosti napovedi.

V drugi skupini poskusov smo izkoristili masovne podatke in učnim časovnim vrstam dodali podobne časovne vrste z namenom, da napovedi LSTM še izboljšamo. Z novimi, večjimi množicami učnih časovnih vrst, smo učili LSTM, ki so se v prvem poskusu izkazale najbolje.

Vsi modeli iz prve in druge skupine poskusov imajo enake pogoje, in sicer

napovedujejo isto okno časovne vrste in imajo na voljo isto skupino časovnih vrst za učenje, zato so modeli iz prvega in drugega poskusa primerljivi. Preizkusili smo natančnost napovedovanja modelov tudi glede dolžine napovedi. Napovedovali smo eno uro, štiri ure in osem ur v prihodnost.

Poskuse smo izvedli v orodju Jupyter notebook [39]. Za gradnjo LSTM smo uporabili zmogljivo knjižnico Keras [40]. Za gradnjo klasičnih modelov smo uporabili dobro podprto knjižnico Pyflux [41]. Za analizo časovnih vrst smo uporabili knjižnico Statsmodel [42], razdelek TSA (ang. *time series analysis*).

Za evalvacijo smo uporabili kriterijsko funkcijo korenjeno srednjo kvadratno napako (ang. *root mean square error, RMSE*).

7.1 Kombinacije obdelav časovnih vrst

Od obdelav časovnih vrst smo uporabili logaritemsko transformacijo, odstranjevanje sezonskosti, odstranjevanje trenda in normalizacijo v naštetem vrstnem redu. Za logaritemsko transformacijo smo se zgledovali po enačbi 6.1. Za odstranjevanje sezonskosti smo uporabili naivno metodo, ki uporabi konvolucijske filtre. Za odstranjevanje trenda smo uporabili enkratno diferenciacijo ter za normalizacijo standardizacijo in se omejili na razpon med 0 in 1.

Potrebno je omeniti, da lahko pomembno vlogo igra tudi vrstni red obdelave podatkov. Za prvi korak obdelave podatkov je bila izbrana logaritemska funkcija zaradi njenega učinka, ki stabilizira varianco v podatkih in ker po obdelavi ohranja sezonskost in trend. Poleg tega se logaritemsko transformacijo pogosto uporabi pred odstranjevanjem sezonskosti tudi zato, ker logaritemska funkcija naredi časovno vrsto aditivno in večina metod za odstranjevanje sezonskosti uporablja aditivno dekompozicijo časovnih vrst. Nato uporabimo odstranjevanje sezonskosti, saj za tem trend še vedno ostane enak tudi po obdelavi. Za tem uporabimo odstranjevanje trenda in na koncu normalizacijo. Normalizacija je zadnji korak, ker želimo z njo omejiti razpon vrednosti

v časovnih vrstah, primeren za nevronske mreže. Tudi v primerih, ko ne uporabimo vseh obdelav časovnih vrst, se držimo istega vrstnega reda.

Napovedane časovne vrste je potrebno vrniti nazaj v izvorno velikost in obliko časovnih vrst, saj sicer napovedi nimajo pomena. Tukaj se držimo obratnega vrstnega reda, to je inverz normalizacije, seštevanje napovedi, da vrnemo trend, prištevek sezonske časovne vrste in uporaba eksponentne funkcije 6.2. Za postopek smo se zgledovali po delu Bandare in sodelavcev [1].

Na izboljšanje napovedi LSTM lahko vpliva tudi izbira skupine časovnih vrst, ki so uporabljene za učenje. Metode gručenja predstavljajo primeren pristop za iskanje podobnih časovnih vrst. Ker so časovne vrste dolge in jih je veliko, ni najbolj praktično neposredno gručiti samih časovnih vrst. Omejitev obsega značilk je zaželena, zato je primeren pristop zbiranje samoopisljivih značilk, ki bi znale čim bolje določiti razlike med gručami. Ker smo se zgledovali po delu Bandare in sodelavcev [1], smo tudi mi opisali značilke s pristopom iz dela, ki so ga predstavili Hyndman in sodelavci [26]. S predlaganimi značilkami so želeli ujeti večino dinamike, ki so v časovnih vrstah in so pogoste pri večini analiz časovnih vrst, kot so trend, sezonskost in avtokorelacija.

Uporabili smo mešanje verjetnostnih porazdelitev (ang. *mixture model, MM*) z načelom najkrajšega sporočila (ang. *minimal message length, MML*), to je tehniko Bayesovskih ocen točk, ki predstavlja najvišjo posteriorno verjetnostno porazdelitev vsake gruče. Ker smo v našem delu omejeni le na numerične časovne vrste, smo se lahko omejili na model za normalno porazdeljene časovne vrste. V delu je uporabljeno mešanje Gaussovih verjetnostnih porazdelitev (ang. *Gaussian mixture model, GMM*), kot so to naredili Bandara in sodelavci [1]. Prednost MM pred ostalimi tehnikami gručenja je, da je sposobno samostojno odkriti optimalno število gruč.

7.2 Podatki

Uporabljeni podatki so iz panoge spletnega oglaševanja družbenih omrežjih. V resničnem svetu se dogodke, kot so kliki, prikazi oglasa in ostalo pripadajočo vsebino ves čas beleži v dnevniško datoteko (ang. *log file*) s pripadajočo časovno oznako (ang. *timestamp*). Zaradi tega je naravna izbira, da se na teh podatkih uporabijo metode za analizo časovnih vrst [21].

Podatki, ki smo jih uporabili v poskusih, predstavljajo dnevniške datoteke, ki so bile zbrane na podlagi oglasov, prikazanih na družbenem omrežju Facebook. Podatki so se zbirali vsako uro, kar pomeni, da imamo za vsako uro v času dodane nove sveže podatke za vnaprej določene značilke. Imamo 13 značilk, ki so opisane v tabeli na sliki 7.1. Za vsako značilko lahko sestavimo svojo časovno vrsto. Tako imamo poleg časovne vrste, ki predstavlja klike na oglase, zbrane tudi druge časovne vrste, kot je število prikazov oglasov, število všečkov na oglase in podobno. Vseh skupin časovnih vrst imamo 60 ter jih ločimo po kampanji (ang. *campaign*) in ciljni publiki, ki je v podatkih označena kot skupina (ang. *group*), saj ima oglaševalska kampanja tipično več oglasnih skupin. Vse časovne vrste, ki smo jih zbrali za naše poskuse, so razpona približno enega leta, in sicer od 2. januarja 2017 do 19. januarja 2018. Skupine časovnih vrst so dolge od 63 do 305 ur, kar predstavlja pribl. 3 do 12 dni podatkov posameznih skupin časovnih vrst. Za napovedovanje smo si rezervirali zadnjih 16 ur vsake časovne vrste.

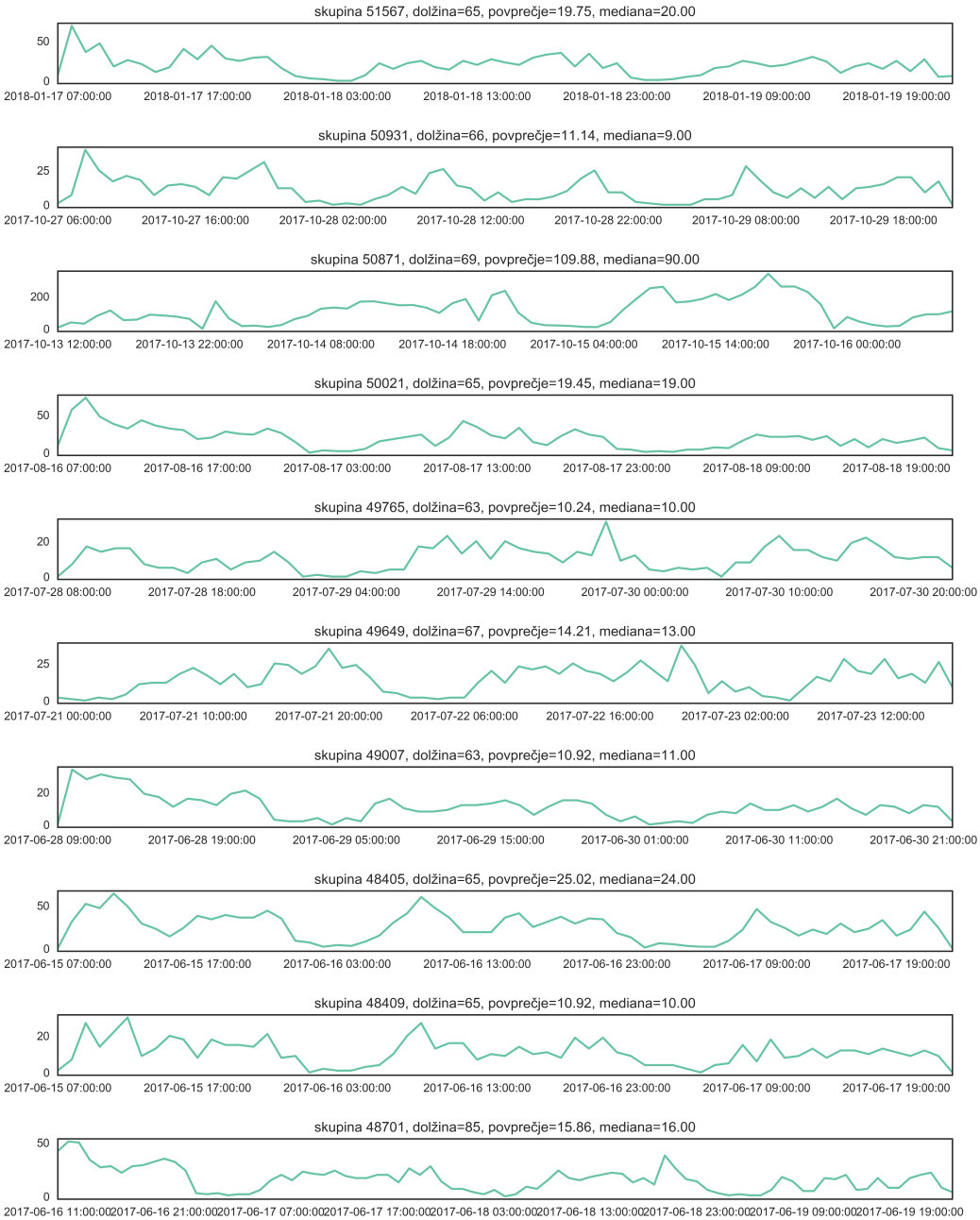
Iz vseh 60 skupin časovnih vrst smo poiskali 10 skupin, ki so mlajše in manjše. To smo naredili za drugo skupino poskusov, da smo povečali polje kandidatnih skupin za iskanje najbolj podobnih časovnih vrst za izboljšanje natančnosti napovedi LSTM modela. Poskuse smo izvajali na vsaki od desetih skupin posebej. Časovne vrste klikov teh 10 skupin so prikazane na sliki 7.2, ki prikazuje za vsako skupino tudi dolžino časovne vrste, povprečje klikov in mediano klikov. Tabela 7.1 medtem predstavlja izračunan ADF test na časovnih vrstah klikov.

	count	mean	std	min	25%	50%	75%	max
actions	7498.00	145.98	345.82	0.00	9.00	32.00	172.00	6779.00
amount_no_markup	7498.00	3.03	5.48	0.00	0.41	1.11	2.97	63.60
clicks	7498.00	23.01	27.47	0.00	6.00	15.00	29.00	341.00
impressions	7498.00	1662.42	3637.91	1.00	272.00	744.00	1619.00	98356.00
inline_link_click	7498.00	5.08	12.35	0.00	0.00	0.00	4.00	132.00
link_click	7498.00	5.08	12.35	0.00	0.00	0.00	4.00	132.00
offsite_conversion	7498.00	5.60	35.47	0.00	0.00	0.00	0.00	816.00
page_likes	7498.00	0.16	0.48	0.00	0.00	0.00	0.00	6.00
post_engagement	7498.00	143.41	334.61	0.00	8.00	28.00	172.00	6363.00
results	7498.00	144.27	333.60	0.00	8.00	31.00	173.00	6341.00
social_clicks	7498.00	14.14	19.63	0.00	2.00	8.00	18.00	249.00
social_impressions	7498.00	668.45	1020.84	0.00	106.00	349.00	894.00	17699.00
video_view	7498.00	134.06	335.42	0.00	0.00	0.00	170.00	6341.00

Slika 7.1: Slika predstavlja opisane značilke naših podatkov iz družbenega omrežja Facebook, ki jo je izdelala funkcija *describe* knjižnice *Pandas*. Za vsako od 13 značilk je izračunano število instanc *(count)*, povprečje *(mean)*, standardna deviacija *(std)*, najmanjša vrednost *(min)*, 25. percentil *(25 %)*, mediana *(50%)*, 75. percentil *(75 %)* in največja vrednost *(max)*.

	ADF	**p-vrednost**	**kritična vrednost 1%**	**kritična vrednost 5%**	**kritična vrednost 10%**	**Možnost trenda**
51567-8919	-4.133	0.00085	-3.575	-2.924	-2.6	Zelo majhna
50931-8805	-3.568	0.0064	-3.571	-2.923	-2.599	Majhna
50871-8843	-2.947	0.0402	-3.563	-2.919	-2.597	Srednja
50021-8653	-2.652	0.0827	-3.575	-2.924	-2.6	Velika
49765-8447	-2.631	0.0868	-3.589	-2.93	-2.603	Velika
49649-8561	-3.666	0.0046	-3.581	-2.927	-2.602	Zelo majhna
49007-8425	-2.723	0.0701	-3.585	-2.928	-2.602	Srednja
48405-8245	-3.082	0.0279	-3.578	-2.925	-2.601	Srednja
48409-8245	-3.432	0.0999	-3.589	-2.93	-2.603	Majhna
48701-8341	-3.071	0.0288	-5.53	-2.905	-2.59	Majhna

Tabela 7.1: ADF test časovnih vrst klikov, ki jih napovedujemo v poskusih.



Slika 7.2: Slika predstavlja časovne vrste klikov, ki smo jih uporabili za napovedovanje pri poskusih magistrskega dela. Prikazuje njihovo ime skupine, dolžino, povprečje in mediano.

7.3 Priprava napovednih modelov

V poskusih smo uporabili naiven vztrajnostni model kot spodnjo mejo in preverbo primernosti napovedi ostalih modelov, to so ARIMA, ARIMAX, VAR in LSTM. Tako kot LSTM smo tudi na ostalih modelih z izjemo vztrajnostnega modela uporabili na več načinov obdelane časovne vrste. Za obdelavo časovnih vrst smo uporabili vse možne kombinacije: ali ima narejeno logaritemsko preslikavo, ali ima odstranjeno sezonskost, ali ima odstranjen trend in ali ima normalizirane vrednosti. To nanese 16 možnih kombinacij. Ker imata ARIMA in ARIMAX znotraj svojega modela že sposobnost integriranja in iskanja sezonskosti, smo pri njima obdelavo z odstranjevanjem trenda in sezonskosti izpustili. V nadaljevanju so predstavljeni posamezni napovedni modeli.

7.3.1 Vztrajnostni model

Vztrajnostni model je najbolj splošen in naiven model za napovedovanje časovnih vrst. Model predpostavlja, da bo prihodnost enaka sedanjosti, kar pomeni, da napove zadnjo znano vrednost. V primeru, ko se napoveduje eno uro v prihodnost, za naslednjo uro napove enako število klikov na oglas, kot se je zgodilo v zadnji znani uri (enačba 7.1).

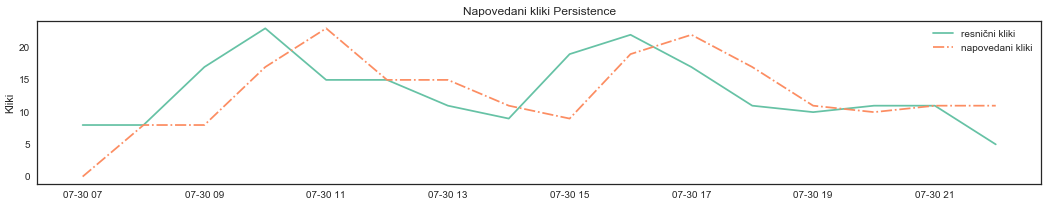
$$y_t = x_{t-1} \tag{7.1}$$

V primeru, ko se napoveduje 4 ali 8 ur v prihodnost, napove enako število klikov, kot se je zgodilo v zadnjih 4 ali 8 ur. Glej enačbo 7.2.

$$y_t + y_{t+1} + ... + y_{t+N} = x_{t-1} + x_{t-2} + ... + x_{t-(N+1)} \tag{7.2}$$

Primer napovedi ene ure v prihodnost je prikazana na sliki 7.3.

Ker nas ne zanima, kako samo obdelava časovnih vrst izboljša napovedi, jih pri vztrajnostnem modelu nismo uporabili. Za učenje potrebuje le tisto časovno vrsto, katero tudi napoveduje. To pomeni, da je univariaten model



Slika 7.3: Slika predstavlja resnično časovno vrsto klikov (modra črta) in napoved vztrajnostnega modela (rdeča črtkana črta). Vztrajnostni model napoveduje eno uro v prihodnost.

– ne omogoča uporabe soležnih časovnih vrst za dodatno znanje, ki bi ga te lahko prispevale.

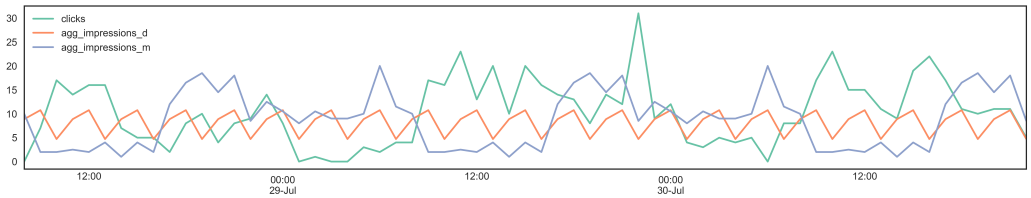
7.3.2 ARIMA

ARIMA je prav tako univariaten model, zato smo od učne množice časovnih vrst uporabili le časovno vrsto, ki prikazuje število klikov na oglase, ker to napovedujemo. ARIMO smo pripravili za naše poskuse tako, da se v začetku izvede metoda znana pod imenom *autoarima*, katera z določeno kriterijsko funkcijo poišče najbolj primerne parametre modela ARIMA p, d, q glede na našo že obdelano učno časovno vrsto. Izbrali smo privzeto kriterijsko funkcijo *autoarima* metode – informacijski kriterij Akaike (ang. *Akaike information criterion*). Pri izbranih parametrih se ARIMA model nauči in poda napovedi. Za tem je potrebno napovedi peljati skozi vzvratno obdelavo podatkov, da dobimo napovedi v izvirni obliki in velikosti. Ker ARIMA velja za uspešen model, smo pričakovali bolj natančne napovedi, kot jih naredi vztrajnostni model.

7.3.3 ARIMAX

Model ARIMAX je različica ARIMA modela, ki zna uporabiti dodatne časovne vrste, imenovane pojasnjevalne spremenljivke. To so tiste časovne vrste, pri katerih je znana tako zgodovina kot prihodnost. Za zunanji regresor se lahko

uporablja časovna vrsta, ki označuje uro v dnevu, dan v tednu ali koledar praznikov, saj je to znanje znano v naprej. Uporabijo se lahko tudi povprečja kot je zgodovinsko povprečje klikov na uro v dnevu ali povprečje klikov na dan v tednu ali mesecu. Uporabili smo vrsto zunanjih regresorjev, to je zgodovinsko povprečje klikov na oglase za vsako uro v dnevu in povprečje klikov na oglase za vsak dan v mesecu. Primer uporabljenih zunanjih regresorjev je predstavljen na sliki 7.4. Povprečja so se določila na podlagi klikov na oglase v učni množici. Ker je ARIMAX pravzaprav ARIMA s pojasnjevalnimi spremenljivkami, je uporabljen enak postopek učenja z enako metodo iskanja parametrov. Zaradi dodatnega znanja iz zunanjih regresorjev so pričakovane bolj natančne napovedi.



Slika 7.4: Slika predstavlja resnično časovno vrsto klikov (modra črta) in agregirano povprečje klikov glede na uro v dnevu (rdeča črta) in dan v mesecu (zelena črta), ki se uporabita kot zunanja regresorja pri modelu ARIMAX.

7.3.4 VAR

VAR je multivariaten model, zato lahko za učenje uporabi celotno skupino časovnih vrst. Tudi tukaj se uporabi isto kriterijsko funkcijo kot jo uporabi ARIMA za samodejno iskanje parametov. VAR ima en parameter – *zamik* (ang. *lag*). Model za učenje uporablja vse časovne vrste in jih vse tudi napove, a za naš poskus se uporabijo le napovedi klikov na oglase. Napovedi gredo tako kot pri ARIMA modelu in ARIMAX modelu skozi vzvratno obdelavo podatkov. Ker VAR zna povezati soodvisnosti časovnih vrst in v primeru, da je skupina časovnih vrst soodvisna, se pričakuje bolj natančne

napovedi kot jih dobimo z vztrajnostnim modelom.

7.3.5 LSTM

LSTM so multivariaten model in se učijo na celi skupini časovnih vrst. Znane so štiri strategije za napovedovanje časovnih vrst več korakov v prihodnost. Prva je neposredno napovedovanje več korakov v prihodnost in vključuje razvijanje modela za vsak časovni korak napovedovanja. Pri opazovanih vhodnih vrednostih v napovedi dobimo z enačbo 7.3.

$$\begin{aligned} napoved(t+1) &= model1(v(t), v(t-1), ..., v(t-n)) \\ napoved(t+2) &= model2(v(t), v(t-1), ..., v(t-n)) \end{aligned} \tag{7.3}$$

Ker moramo za vsak časovni korak učiti svoj model, postane učenje računsko in časovno zahtevno. Druga strategija je rekurzivno napovedovanje več korakov v prihodnost. Ta uporablja en model in napoveduje po en korak v prihodnost vsak korak, kjer uporabi prejšnjo napoved kot vhod za naslednjo napoved, kot prikazano v enačbi 7.4.

$$\begin{aligned} napoved(t+1) &= model(v(t), v(t-1), ..., v(t-n)) \\ napoved(t+2) &= model(napoved(t+1), v(t), ..., v(t-n)) \end{aligned} \tag{7.4}$$

Ker se pri tej strategiji prejšnje napovedi uporabijo kot opazovane vrednosti, dovolimo, da se napovedna napaka prenaša v naslednje časovne korake. Tretja strategija je rekurzivna neposredna hibridna strategija, ki kombinira prej opisani strategiji in ima enačbo 7.5.

$$\begin{aligned} napoved(t+1) &= model1(v(t), v(t-1), ..., v(t-n)) \\ napoved(t+2) &= model2(napoved(t+1), v(t), ..., v(t-n)) \end{aligned} \tag{7.5}$$

Večizhodna strategija medtem razvije en model, ki je zmožen napovedati celotno zaporedje napovedi v enem koraku, kot prikazano v enačbi 7.6.

$$napoved(t+1), napoved(t+2) = model(v(t), v(t-1), ..., v(t-n)) \tag{7.6}$$

Model večizhodne strategije je bolj kompleksen, saj se lahko nauči strukturo odvisnosti med vhodi in izhodi, kot tudi med izhodi samimi. Zaradi večje kompleksnosti se učijo počasneje in potrebujejo več učnih podatkov, da se izognejo prenasičenju (ang. *overfitting*). Za naše poskuse smo uporabili večizhodno strategijo.

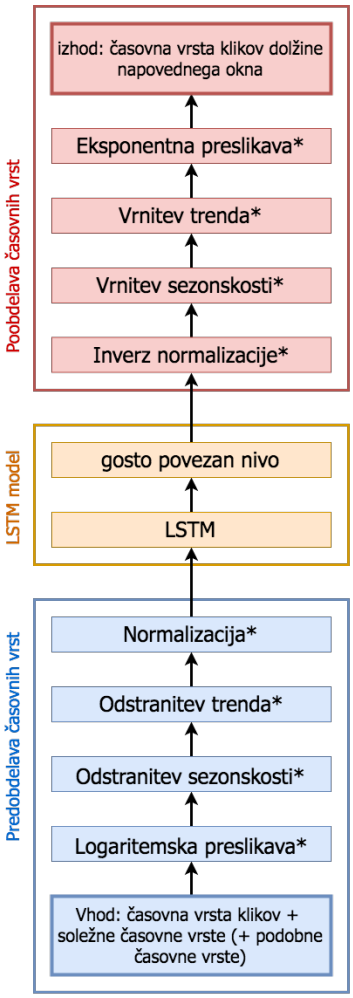
Model nevronske mreže smo zgradili s *Keras* knjižnico. Končno arhitekturo smo določili po več ročnih poskusih na različnih skupinah časovnih vrst. Pri gradnji smo si pomagali z delom Bandare in sodelovcev [1], po katerem smo se zgledovali. Arhitektura nevronske mreže, skupaj z obdelavo časovnih vrst, je predstavljena na sliki 7.5.

Pri LSTM ni uporabnih metod, ki bi znale dobro oceniti primerne metaparametre modela. Pred vsakim napovedovanjem je bila tako predhodno narejena optimizacija metaparametrov na validacijski množici. Validacijsko množico smo določili tako, da smo učno množico razdelili na novo učno in validacijsko časovno vrsto, kjer smo zadnjih 16 ur učne množice vzeli za validacijsko, kot je prikazano na sliki 7.6. Na sliki v primeru optimizacije parametrov učno množico predstavljajo kliki vse do prve modre točke in v primeru napovedovanja učno množico predstavljajo kliki vse do prve rdeče točke. Predmet optimizacije so parametri, predstavljeni v tabeli 7.2.

Za optimizacijo metaparametrov smo naredili 20 iteracij naključnih vrednosti in uporabili tiste metaparametre, ki so dosegli najbolj natančne napovedi na validacijski časovni vrsti. Za vsako kombinacijo obdelav podatkov smo učili 16 modelov in kot končno napoved uporabili povprečje napovedi. Slednje smo naredili zaradi stohastičnih lastnosti nevronskih mrež. Iz istega razloga smo za napovedovanje testne časovne vrste učili 40 modelov in končno napoved dobili s povprečenjem.

7.4 Napovedovanje

Najprej smo napovedovali po eno uro v prihodnost, vsako uro. Takšnemu načinu v angleščini pravijo *rolling* ali sprehod naprej (ang. *walk forward*) in

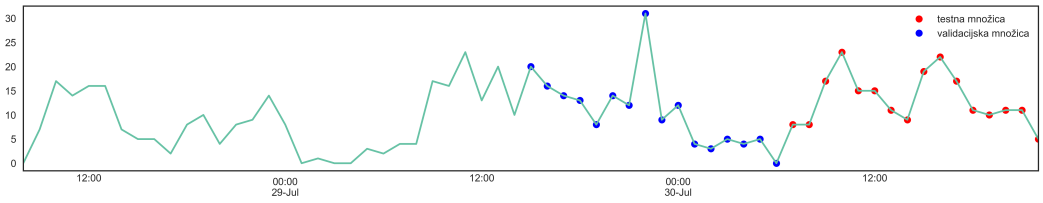


Slika 7.5: Pripravljena arhitektura mreže, ki vključuje predobdelavo časovnih vrst, učenje LSTM in poobdelavo časovnih vrst. *Prikazana obdelava je bila uporabljena le, kadar smo uporabili vse vrste obdelav časovnih vrst. Če se katero izloči, se vrstni red ohrani.

je prikazan na sliki 7.7. Drugi način je podoben prvemu, le da smo vsako uro napovedali štiri ure v prihodnost oziroma osem ur v prihodnost. Napoved štiri ure v prihodnost pomeni, da se napove štiri ure dolgo okno urnih napovedi, kar pomeni štiri napovedi. Natančnost napovedi smo preverili z odstopanjem seštevka napovedanih klikov.

Parameter	**Angleško ime**	**Najmanjša vrednost**	**Največja vrednost**
število nevronov	Number of neurons	10	80
dolžina dobe	Epoch size	10	60
dolžina paketov	mini-batch size	1	4
hitrost učenja na vzorec	learning rate per sample	0.001	0.04
največje število dob	maximum epochs	10	60
vcepitev Gaussovega šuma	Gaussian noise injection	0.0005	0.005
uteži L2 regularizacije	L2 regularization-weight	0.0005	0.0008
število posodobitev	Number of updates	1	50

Tabela 7.2: Parametri, katere smo nastavljali za optimizacijo LSTM.



Slika 7.6: Slika predstavlja resnično časovno vrsto klikov (črta), na kateri so označene točke klikov iz validacijske množice (modro) in testne množice (rdeče).

Slika 7.7: Slika predstavlja učno množico (modra) in testno množico (siva) za vsako naslednjo uro napovedovanja od zgoraj navzdol. V našem primeru smo uporabili zasidran način. Vir slike [43] .

Celoten postopek prvega poskusa je predstavljen v psevdokodi 1.

Algorithm 1 Psevdokoda celotnega poskusa (brez gručenja)

```
for model ∈ {Vztrajnostni model, ARIMA, ARIMAX, VAR, LSTM} do
    for vsaka možna kombinacija obdelav časovnih vrst do
        Razdelitev časovnih vrst na učno in testno množico
        Obdelava časovnih vrst
        Iskanje parametrov modela
        Učenje modela
        Napovedovanje modela
        Vzvratna obdelava napovedane časovne vrste
        Vrednotenje
        Beleženje rezultatov
    end for
end for
```

Gručenje časovnih vrst

Druga skupina poskusa predstavljaje uporabo podobnih časovnih vrst za učenje LSTM. V današnjem svetu masovnih podatkov je podobnih časovnih vrst veliko. V poglavju 2 je bilo omenjenih nekaj primerov uporabe podobnih časovnih vrst in tudi postopkov zbiranja podobnih časovnih vrst. Najbolj smo se zgledovali po delu Bandare in sodelavcev [1] in postopek prilagodili našemu poskusu. Držali smo se postopka, opisanega v psevdokodi 2.

S pomočjo gručenja smo poiskali podobne časovne vrste, in sicer tako, da smo za podobne časovne vrste šteli tiste časovne vrste, ki so se znašle v isti gruči kot časovna vrsta, ki jo napovedujemo. Podobne časovne vrste smo uporabili pri učenju LSTM. V drugem poskusu nismo preverili vseh kombinacij obdelanih časovnih vrst, kot smo storili v prvem poskusu, ampak vzeli le tisto obdelavo časovnih vrst, ki se je v prvem poskusu izkazala najbolje. To smo storili zato, da pokažemo na najbolj enostaven in prepričljiv način, ali uporaba podobnih časovnih vrst izboljša napovedi. Uporabili smo enake nastavitve učenja kot pri prvem poskusu, to je optimizacijo metaparametrov in učenje več modelov, katerih povprečje predstavlja končno napoved.

Algorithm 2 Psevdokoda gručenja podobnih časovnih vrst za LSTM

1: Imamo skupino časovnih vrst S_1..
2: **for** Za vsako skupino časovnih vrst S_x iz zbirke $S_{2..60}$ **do**
3: Odreži ure časovne vrste S_x, ki so se zgodile kasneje kot se je zgodila zadnja ura časovne vrste S_1.
4: **if** Časovna vrsta S_x enako dolga kot S_1 **then**
5: Dodaj skupino časovnih vrst v kandidatno množico K.
6: **end if**
7: **if** Časovna vrsta S_x daljša kot S_1 **then**
8: **for** $i = 1 : S_x$ - $lenght(S_1)$ **do**
9: Dodaj skupino časovnih vrst med uro i in $i+lenght(S_1)$ v kandidatno množico K.
10: **end for**
11: **end if**
12: **end for**
13: Iz vseh skupin časovnih vrst v množici K in skupini S_1, ki jo napovedujemo, vzamemo samo časovne vrste, ki predstavljajo klike na oglase K_{kliki}.
14: Izračunamo vektorje značilk V na klikih K_{kliki}, ki opisujejo njihove lastnosti. Značilke so opisane v tabeli 7.3 in so povzete po delu [26].
15: Opravi se postopek gručenja z GMM na vektorjih značilk V. Za ta postopek smo uporabili več metod.

Tabela 7.3: Zbrane značilke časovnih vrst za gručenje podobnih časovnih vrst v drugem poskusu, povzeti po delu [26].

Značilka	**Angleški izraz**	**oznaka**
spreminjanje variance v ostanku	changing variance in remainder	Lumpiness
spektralna entropija	spectral entropy	Entropy
avtokorelacija prvega reda	first order autocorrelation	ACF1
nivo zamika pri premikajočem oknu	level shift using rolling window	Lshift
sprememba variance	variance change	Vchange
število križnih točk	the number of crossing points	Cpoints
ravnih delov pri diskretizaciji	flat spots using disretization	Fspots
moč trenda	strength of trend	Trend
moč linearnosti	strength of linearity	Linearity
moč krivin	strength of curvature	Curvature
moč špičastosti	strength of spikiness	Spikiness
rezultat Kullback-Leiblerja	Kullback-Leibler score	KLscore
indeks največjega KL rezultata	index of the maximum KL score	Change.idx
povprečje	average	Avg
varianca	variance	Var

Poglavje 8

Rezultati

Poglavje rezultatov je razdeljeno na dve podpoglavji. Vsako podaja rezultate enega od dveh večjih poskusov.

Prvo podpoglavje je osredotočeno na prvo skupino poskusov, kjer se primerja uspešnost različnih modelov ter uspešnost modelov glede na kombinacijo obdelav časovnih vrst podanih v posamezen model. Uspešnost modelov se meri z natančnostjo napovedovanja, za katero je uporabljena kriterijska funkcija RMSE.

V drugem podpoglavju je predstavljena druga skupina poskusov, kjer se uporabi metodo gručenja za iskanje podobnih časovnih vrst, katere se doda k učenju LSTM z namenom izboljšanja natančnosti napovedi LSTM. Ostali parametri so pri drugem poskusu enaki kot pri prvem. Natančnost modela, ki je rezultat druge skupine poskusov, se tako lahko primerja z natančnostjo modelov iz prve skupine poskusov.

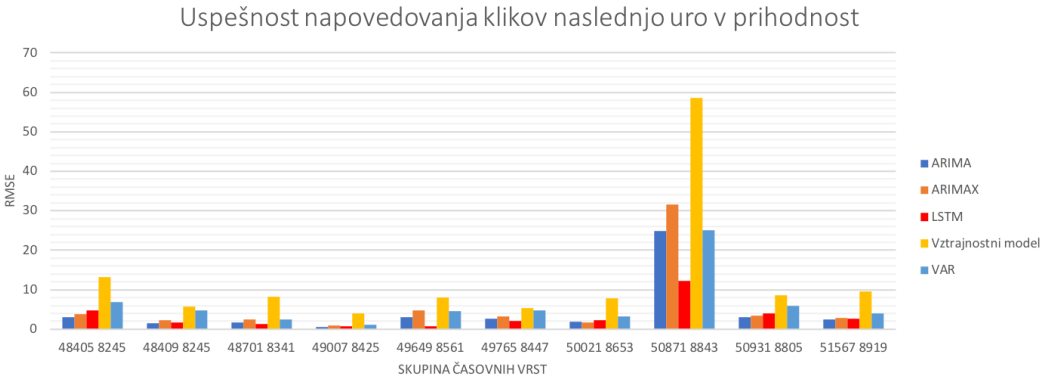
8.1 Prva skupina poskusov

S prvo skupino poskusov smo želeli preveriti uspešnost učnih modelov pri uporabi obdelav časovnih vrst in primerjati uspešnost napovedovanja LSTM z naivnim vztrajnostnim modelom ter s klasičnimi modeli ARIMA, ARIMAX in VAR. Preizkusili smo štiri vrste obdelav podatkov, to so logaritemska

preslikava podatkov, normalizacija, odstranitev sezonskosti in odstranitev trenda. Modeli so napovedovali vsako uro eno uro v prihodnost, štiri ure v prihodnost in osem ur v prihodnost.

8.1.1 Uspešnost posameznih modelov

Slike 8.1, 8.2, 8.3 prikazujejo natančnost napovedi klikov na oglas v naslednji uri, naslednjih štirih urah in naslednjih osmih urah za vsako od deset skupin posebej. Skupine so označene z univerzalnimi izkaznicami.

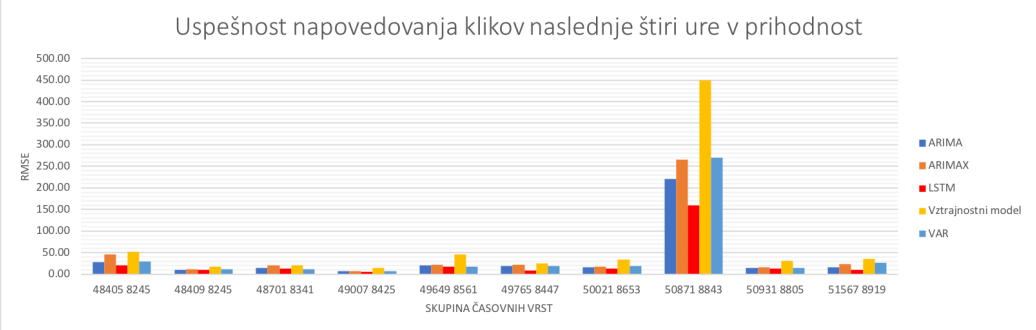


Slika 8.1: Napovedi modelov na testni časovni vrsti klikov za vsako od desetih skupin. Modeli so napovedovali število klikov na oglase v naslednji uri.

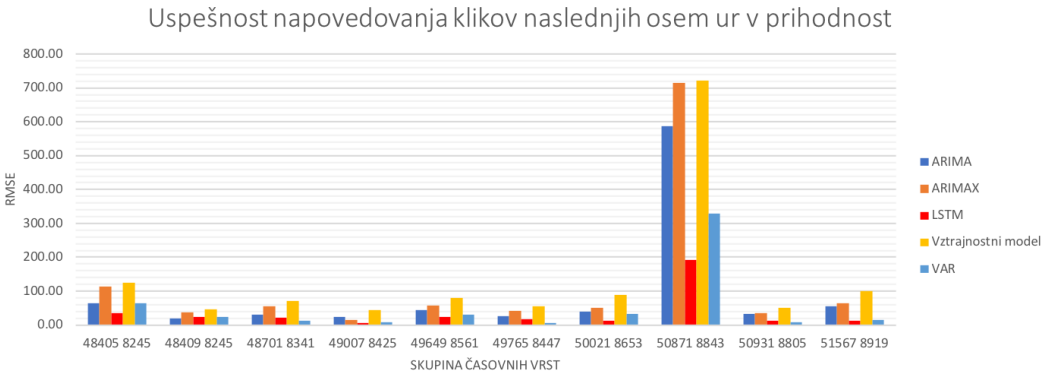
Naslednje slike 8.4, 8.5, 8.6 prikazujejo natančnost napovedi klikov na oglas v naslednji uri, naslednjih štirih urah in naslednjih osmih urah za vseh deset skupin skupaj. V ta namen smo najprej normalizirali RMSE znotraj vsake skupine, da so si bile skupine primerljive.

Iz prvih treh slik 8.1, 8.2 in 8.3 lahko opazimo, da so tako klasični modeli kot LSTM dosegli boljše napovedi kot naivni vztrajnostni model pri vsaki od desetih skupin in pri napovedovanju ene, štirih ali osem ur v prihodnost.

Iz slik 8.4, 8.5 in 8.6 vidimo, da so povprečno LSTM najbolj uspešno napovedovale klike na oglase pri napovedovanju ene, štirih in osem ur v prihodnost. Zanimiva opazka je tudi pri modelih ARIMA in VAR. Pri kratkih



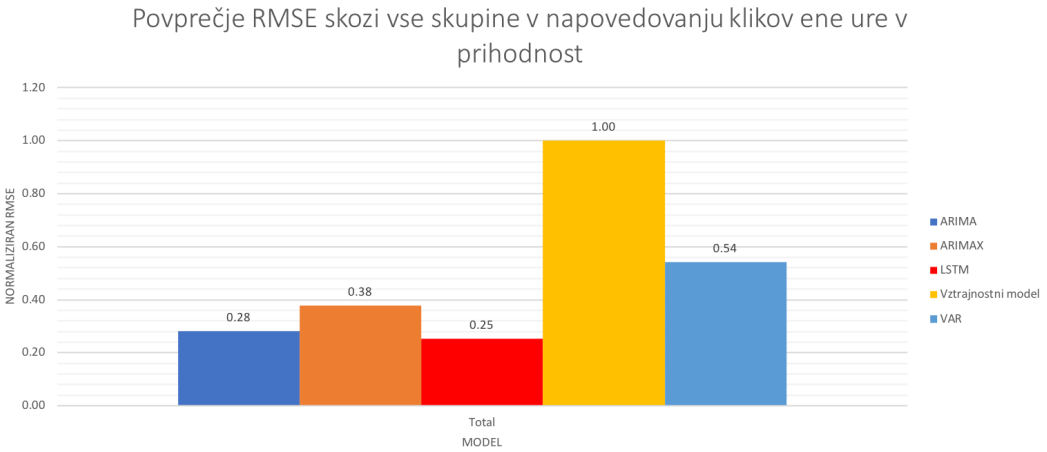
Slika 8.2: Napovedi modelov na testni časovni vrsti klikov za vsako od desetih skupin. Modeli so napovedovali število klikov na oglase v naslednjih štirih urah.



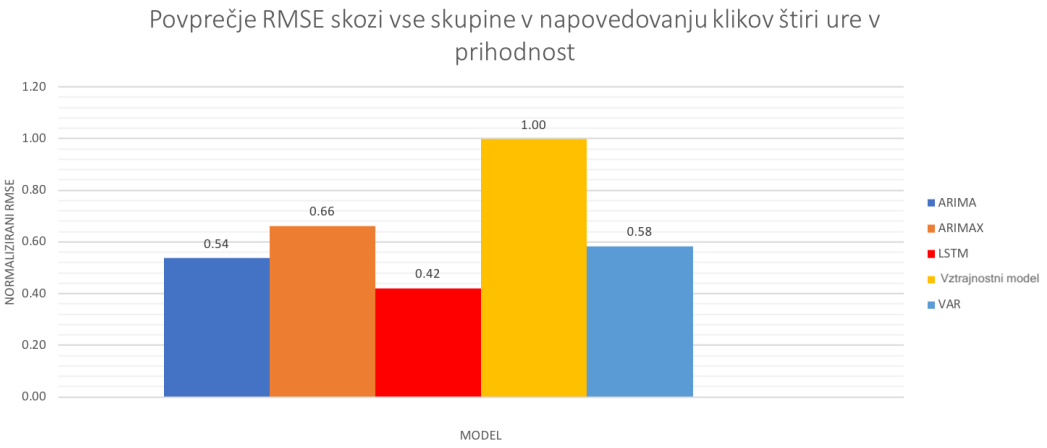
Slika 8.3: Napovedi modelov na testni časovni vrsti klikov za vsako od desetih skupin. Modeli so napovedovali število klikov na oglase v naslednjih osmih urah.

napovedih, to je napovedovanje eno uro v prihodnost, je ARIMA dosegla boljše rezultate kot VAR. Pri napovedovanju dolgih napovedi, to je osem urnih napovedi, sta se vlogi zamenjali. ARIMAX je bil tako pri kratkih kot dolgih napovedih v povprečju slabši od ARIMA modela.

Slike 8.7, 8.8 in 8.9 prikazujejo faktor izboljšanja LSTM napram najboljšemu od klasičnih modelov pri isti skupini. V legendi je podano, s katerim modelom je LSTM primerjan – kateri od preostalih modelov je bil najboljši.



Slika 8.4: Napovedi modelov na testni časovni vrsti klikov za vseh deset skupin skupaj. Modeli so napovedovali število klikov na oglase v naslednji uri.

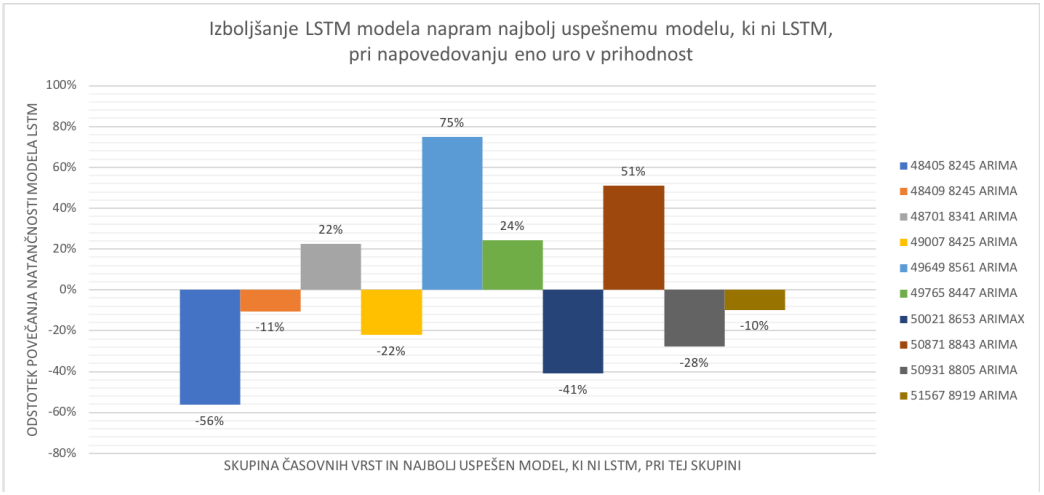


Slika 8.5: Napovedi modelov na testni časovni vrsti klikov za vseh deset skupin skupaj. Modeli so napovedovali število klikov na oglase v naslednjih štirih urah.

Faktor izboljšanja je določen z enačbo 8.1.

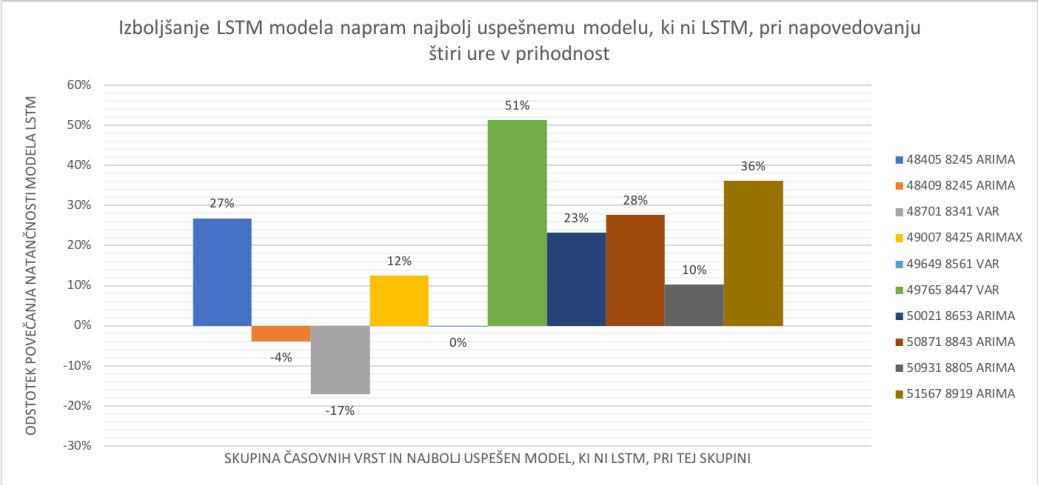
$$faktorIzboljsanja = (1 - \frac{RMSE_{LSTM}}{RMSE_{najboljsiKlasicni}}) \times 100 \qquad (8.1)$$

Slika 8.6: Napovedi modelov na testni časovni vrsti klikov za vseh deset skupin skupaj. Modeli so napovedovali število klikov na oglase v naslednjih osmih urah.

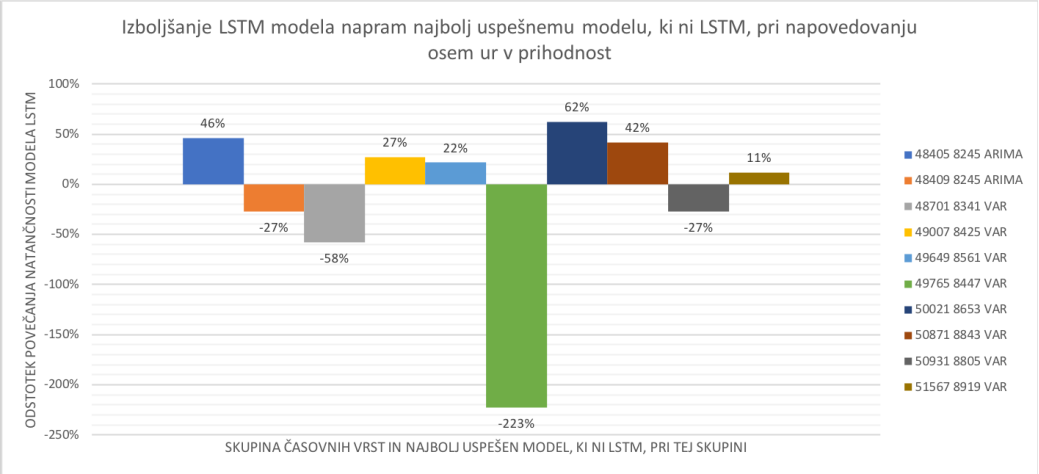


Slika 8.7: Slika prikazuje faktor izboljšanja LSTM napram najbolj uspešnemu klasičnemu modelu pri isti skupini. Ti modeli so napovedovali eno uro v prihodnost.

Pri napovedovanju ene ure v prihodnost lahko na sliki 8.7 vidimo, da so pri štirih skupinah od desetih LSTM dosegle najboljše napovedi. V ostalih



Slika 8.8: Slika prikazuje faktor izboljšanja LSTM napram najbolj uspešnemu klasičnemu modelu pri isti skupini. Ti modeli so napovedovali štiri ure v prihodnost.



Slika 8.9: Slika prikazuje faktor izboljšanja LSTM napram najbolj uspešnemu klasičnemu modelu pri isti skupini. Ti modeli so napovedovali osem ur v prihodnost.

primerih je v petih od šest primerov boljšo napoved dosegla ARIMA. Kljub temu, da je ARIMA v večih skupinah dosegla boljši rezultat kot LSTM, so

v povprečju najnižje napovedi dale LSTM, kot smo že pokazali na sliki 8.4.

Slika 8.8 prikazuje štiri ure dolge napovedi. V tem primeru so LSTM dosegle najboljši rezultat pri sedmih od desetih skupin. Za najbolj uspešen klasični model so se medtem izkazale vse tri metode, ARIMA, ARIMAX in VAR.

Na sliki 8.9 je prikazan faktor izboljšanja LSTM pri napovedovanju osem ur v prihodnost. LSTM so bile najuspešnejše v šestih od deset skupin. Kot smo že ugotovili s slike 8.6, tudi v tem primeru opazimo, da je najbolj uspešen klasičen model za osem ur dolge napovedi VAR. Izstopa rezultat skupine *49765-8447*, pri kateri sta tako VAR kot LSTM imela nizko normirano napako RMSE. Absolutna razlika je bila sicer majhna, vendar, ker se je VAR dobro obnesel, je relativna razlika med njima velika.

Opazimo lahko tudi, da so pri skupini *50871-8843*, LSTM v vseh treh dolžinah napovedi ostajale najuspešnejši model.

8.1.2 Prispevek metod obdelav časovnih vrst

V magistrskem delu nas je zanimalo tudi, koliko vpliva določena obdelava časovnih vrst na povečanje natančnosti napovedovanja. Osredotočili smo se na prispevek metod obdelav časovnih vrst pri LSTM.

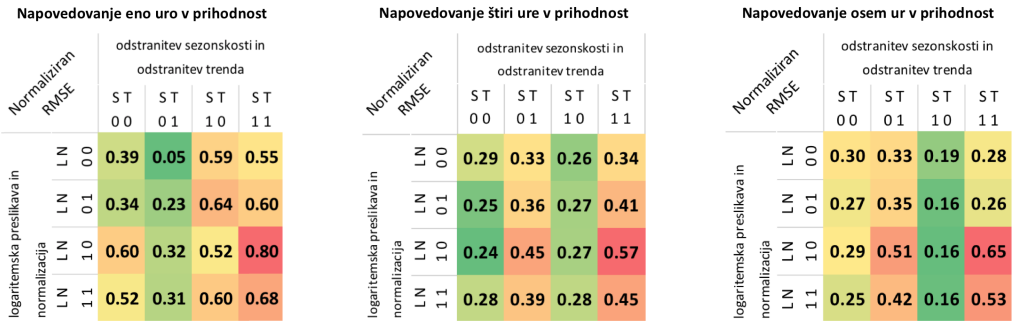
Slika 8.10 prikazuje tri matrike intenzitete (ang. *heatmaps*) kombinacij obdelav časovnih vrst, ki smo jih uporabili v našem poskusu pri napovedovanju ene, štiri ali osem ur v prihodnost. Tvorili smo pare metod obdelav in primerjali njihovo uporabo (vrednost 1) in neuporabo (vrednost 0). Prišli smo do naslednjih ugotovitev:

- V primeru napovedovanja eno uro v prihodnost je bil najnižji RMSE dosežen pri uporabi trenda in neuporabi sezonskosti. Na napoved je najslabše vplivala uporaba logaritemske preslikave in neuporaba normalizacije.

- V primeru napovedovanja štiri ure v prihodnost vidimo, da je najbolj pozitivno vplivala neuporaba trenda. Rezultati so bili še boljši, če ni-

smo uporabili sezonskosti. Pri tem je najbolj pomagalo, da smo dodali logaritemsko preslikavo in nismo vključili normalizacije.

- V primeru napovedovanja osem ur v prihodnost je bila najbolj uporabna obdelava z odstranitvijo sezonskosti brez odstranitve trenda.



Slika 8.10: Slika prikazuje tri matrike intenzitete. Levo prikazuje LSTM pri napovedovanju ene ure v prihodnost, na sredini pri napovedovanju štiri ure v prihodnost in desno osem ur v prihodnost. Da smo ustvarili dvodimenzionalne matrike, smo metode obdelav časovnih vrst poparili in primerjali njihovo uporabo (vrednost 1) in neuporabo (vrednost 0). RMSE je normaliziran, zato da smo ga lahko povprečili skozi vseh deset skupin napovedi.

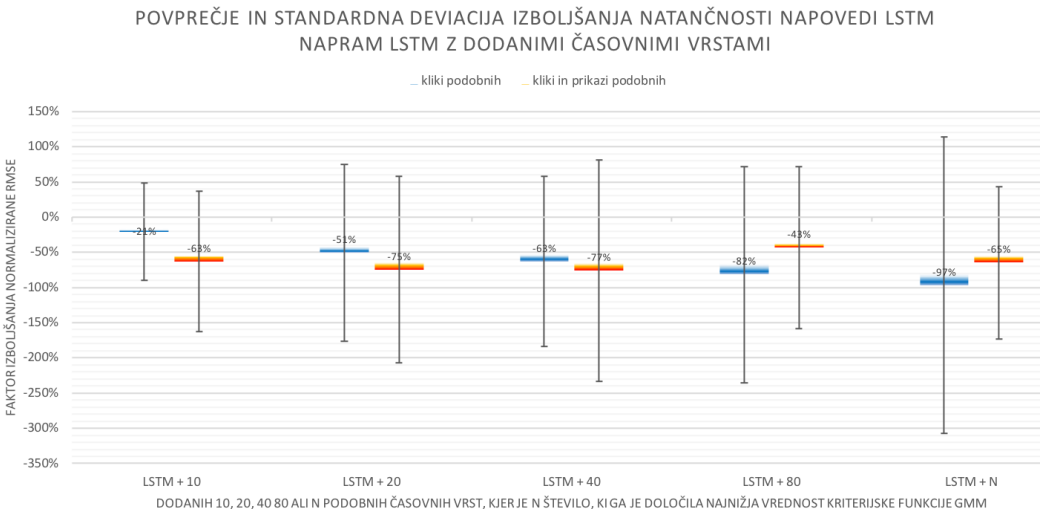
Iz ugotovitev lahko sklepamo, da na kratke napovedi, kot je napovedovanje eno uro v prihodnost, najbolj prispeva odstranitev trenda, pri dolgih napovedih, kot je napovedovanje osem ur v prihodnost, pa najbolj prispeva odstranitev sezonskosti. Ker vrednosti števila klikov niso bile značilno višje od 0, LSTM ni trpel zasičenja odvoda sigmoidne aktivacijske funkcije pri vzvratnem razširjanju in normalizacija ni bila obvezna. Ker število klikov skozi čas ni močno variiralo, logaritemska preslikava ni očitno izboljšala natančnosti napovedi.

8.2 Druga skupina poskusov

Pri drugi skupini poskusov smo želeli preizkusiti, ali bo dodajanje podobnih časovnih vrst v učno množico časovnih vrst prispevalo k izboljšanju natančnosti napovedovanja. Za iskanje podobnih časovnih vrst smo uporabili GMM. Poskusili smo več metod uporabe GMM za gručenje.

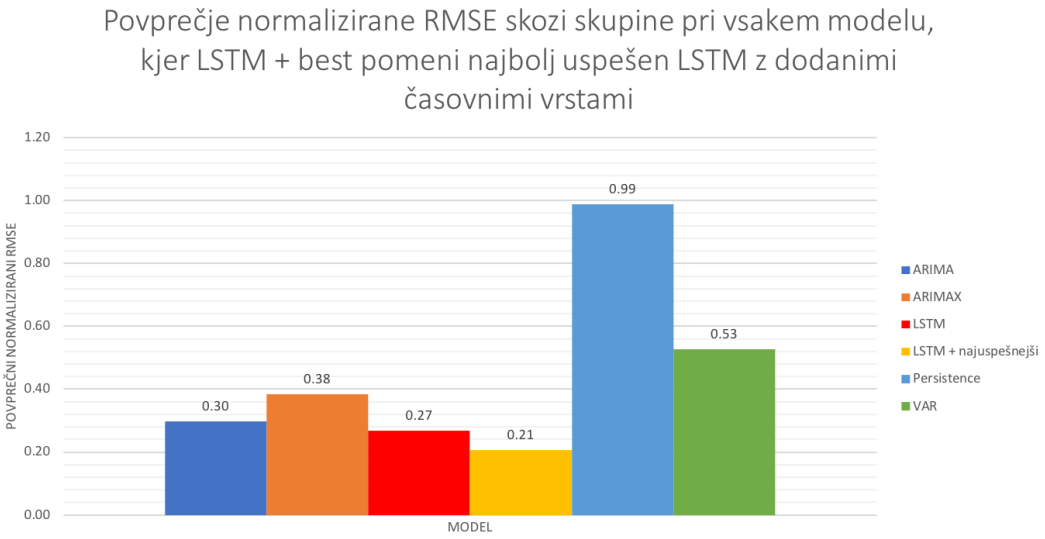
Prva metoda, ki smo jo preizkusili, je uporaba fiksnega števila dodanih časovnih vrst. S pomočjo GMM smo izračunali metriko podobnosti in uporabili 10, 20, 40, 80 in N najbolj podobnih časovnih vrst klikov, kjer je N število dodanih časovnih vrst, ki jih je določil Bayesovski informacijski kriterij. Preskusili smo tudi z dodajanjem soležnih časovnih vrst, ki predstavljajo število prikazov oglasov. S tem smo število dodanih časovnih vrst podvojili.

Zaradi časovnih omejitev in počasnega računanja LSTM smo se osredotočili na napovedovanje ene ure v prihodnost in prišli do rezultatov, prikazanih na sliki 8.11.



Slika 8.11: Slika prikazuje povprečni faktor izboljšanja in standardno deviacijo LSTM z dodanimi časovnimi vrstami napram LSTM iz prvega poskusa. Faktor izboljšanja je izračunan na LSTM z dodanimi časovnimi vrstami, kjer smo dodali fiksno število dodatnih časovnih vrst.

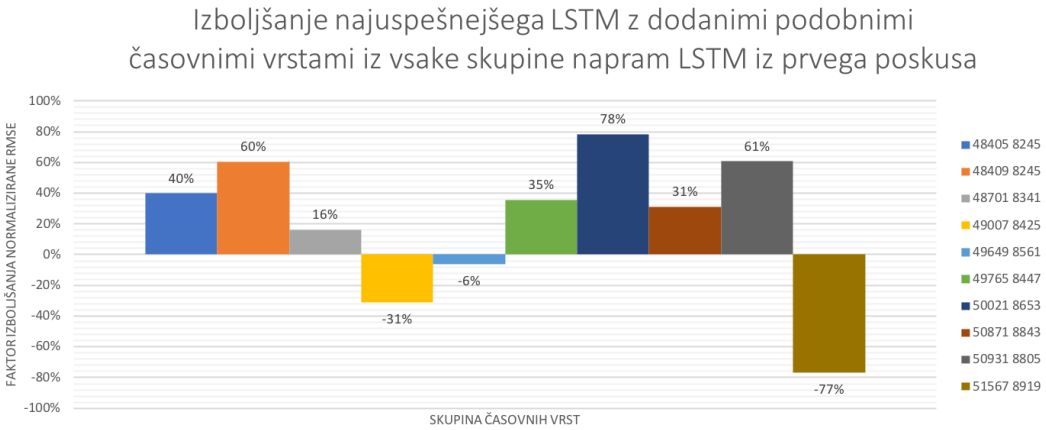
Rezultati na sliki 8.11 so nam pokazali, da povprečno pri nobenem fiksnem številu dodajanja časovnih vrst nismo dosegli izboljšanja. Standardne deviacije pa govorijo o tem, da so rezultati med skupinami močno variirali. Dodajanje časovnih vrst, ki predstavljajo število prikazov oglasov (spremenljivka *impressions* na sliki 7.1), ni prispevalo k povečanju natančnosti modela. Če upoštevamo le napovedi LSTM z dodanimi časovnimi vrstami, ki so dosegle najbolj natančne napovedi pri vsaki od skupin, dobimo rezultate, prikazane na slikah 8.12 in 8.13.



Slika 8.12: Slika prikazuje povprečni normaliziran RMSE pri modelih prvega poskusa in pri najbolj uspešnem modelu drugega poskusa iz vsake skupine.

Iz slik 8.12 in 8.13 je videti, da je mogoče doseči izboljšanje z dodajanjem časovnih vrst, kar smo uspeli doseči pri sedmih od deset skupin. Ker pa nismo našli fiksnega števila podobnih časovnih vrst, ki bi v povprečju izboljšale napovedi LSTM brez dodanih časovnih vrst, smo ubrali drug pristop. Določili smo fiksen prag vrednosti matrike podobnosti modela GMM. Ker dodajanje prikazov oglasov v prejšnjem poskusu ni obrodilo sadov in zaradi časovnih omejitev, smo jih pri tem poskusu izpustili.

Metrike podobnosti, ki jih je izračunal model GMM na vsaki skupini,



Slika 8.13: Slika prikazuje faktor izboljšanja LSTM z dodanimi časovnimi vrstami, ki je v skupini dosegel najnižji RMSE napram LSTM iz prvega poskusa.

so imele različno zalogo vrednosti pri različnih skupinah, zato smo najprej zalogo vrednosti normalizirali in omejili na interval od 0 do 100. Preizkusili smo več pragov in najprej preverili, koliko podobnih časovnih vrst to nanese, kar smo prikazali v tabeli 8.1.

Na podlagi tabele 8.1 in na podlagi števila podobnih časovnih vrst, ki so se pri fiksnem določanju najbolj izkazale, smo za poskus izbrali naslednje pragovne vrednosti:

$$0,01, \quad 0,02, \ 0,05, \ 0,08, \ 0,15, \ 0,20, \ 0,35 \ in \ 0,50$$

Rezultati so prikazani na sliki 8.14.

Slika 8.14 nam prikazuje, da tudi dodajanje časovnih vrst na podlagi praga vrednosti metrike podobnosti v povprečju ni izboljšalo napovedi.

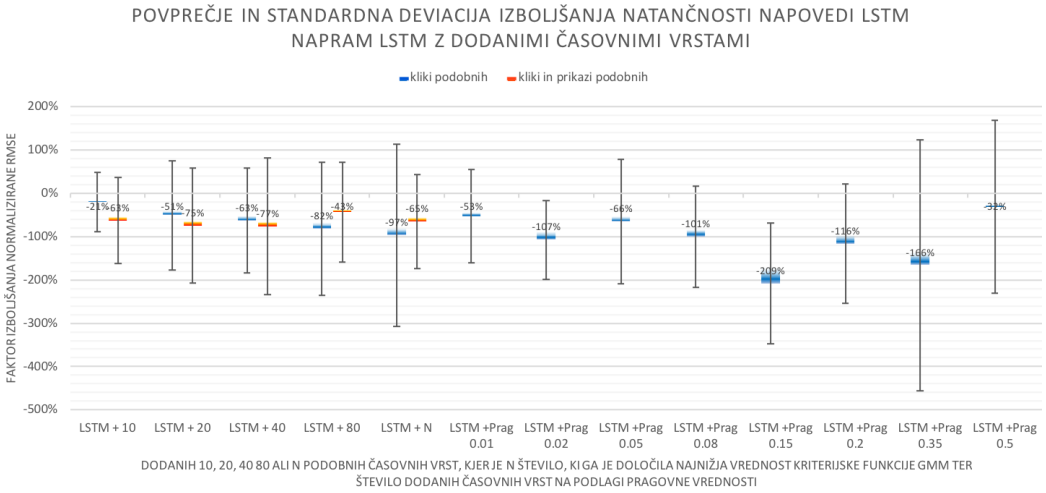
Pridemo torej do naslednjih ugotovitev:

- v povprečju posamezna metoda dodajanja podobnih časovnih vrst napoveduje slabše kot LSTM iz prve skupine poskusov;
- vsaka je bila sposobna v kakšni skupini narediti boljše napovedi in slabše napovedi kot LSTM iz prve skupine poskusov.

	0.01	**0.02**	**0.05**	**0.1**	**0.2**	**0.5**	**1**	**2**	**5**	**10**	**100**
51567	13	22	65	144	303	664	1298	2156	2930	3258	3658
50931	6	13	53	109	194	506	1043	1904	2878	3148	3600
50871	0	0	0	0	0	0	0	6	11	27	3650
50021	3	7	16	36	75	162	341	817	2709	2282	3650
49765	8	17	41	97	203	462	942	1711	2912	3343	3759
49649	14	24	49	94	191	492	949	1778	2837	3080	3543
49007	4	5	16	30	73	192	414	900	2855	3334	3561
48405	7	15	37	69	135	326	687	1361	2624	2836	3181
48409	9	16	50	98	199	499	970	1747	2506	2789	3181
48701	1	1	1	2	9	32	62	110	275	862	2319

Tabela 8.1: Tabela predstavlja število podobnih časovnih vrst, katerih vrednost metrike podobnosti je nižja ali enaka pragovni vrednosti. Število podobnih časovnih vrst pod pragom je izračunanih za vsako skupino posebej. Zadnji stolpec predstavlja celotno množico kandidatov podobnih časovnih vrst.

V naslednjem podpoglavju 8.2.1 smo se osredotočili poiskati možne vzroke, zakaj so metode v nekaterih skupinah napovedovale boljše in v nekaterih slabše kot LSTM iz prvega poskusa. Kaj so lastnosti tistih časovnih vrst, kjer je bilo dodajanje podobnih časovnih vrst uspešno in ali so kakšni ponavljajoči se vzorci, iz katerih lahko potegnemo sklepe?



Slika 8.14: Slika prikazuje povprečni faktor izboljšanja in standardno deviacijo LSTM z dodanimi časovnimi vrstami napram LSTM iz prvega poskusa. Faktor izboljšanja je izračunan na LSTM z dodanimi časovnimi vrstami, kjer smo dodali fiksno število dodatnih časovnih vrst in število dodatnih časovnih vrst, izračunanih na podlagi pragovne vrednosti metrike podobnosti.

SKUPINA	LSTM + 10	LSTM + 10 + prikazi	LSTM + 20	LSTM + 20 + prikazi	LSTM + 40	LSTM + 40 + prikazi	LSTM + 80	LSTM + 80 + prikazi	LSTM + N	LSTM + N + prikazi	LSTM + prag 0.01	LSTM + prag 0.05	LSTM + prag 0.2	LSTM + Prag 0.5
48405 8245	-13.47%	-6.15%	-4.69%	-8.64%	-16.72%	0.89%	-10.27%	-15.39%	-33.29%	39.80%	-10%	-19%	-13%	-18%
48409 8245	-8.28%	-10.01%	42.30%	-49.71%	-9.39%	-10.64%	-23.70%	34.70%	60.46%	-12.04%	73%	55%	-34%	-30%
48701 8341	0.79%	-66.40%	11.29%	-100.45%	6.17%	-5.85%	-9.32%	11.92%	16.11%	-57.89%	17%	15%	21%	5%
49007 8425	-73.81%	-237.12%	-219.58%	-376.13%	-280.76%	-462.48%	-381.95%	-30.98%	-233.14%	-166.66%	-26%	-356%	-535%	-146%
49649 8561	-6.30%	-90.71%	-113.01%	-184.94%	-190.50%	-46.83%	-246.24%	-216.06%	-118.42%	-299.94%	-308%	-192%	-191%	-153%
49765 8447	0.87%	35.43%	-25.98%	8.28%	-13.42%	31.74%	-7.37%	-8.55%	-3.90%	31.27%	-11%	-28%	16%	35%
50021 8653	43.75%	-61.51%	73.26%	76.83%	78.16%	-237.33%	69.45%	60.15%	-22.46%	-85.59%	13%	68%	58%	6%
50871 8843	-14.64%	30.98%	-11.09%	-2.28%	0.27%	-2.50%	-16.86%	40%	-16.32%	-2.78%				
50931 8805	53.21%	9.54%	51.05%	24.76%	18.96%	44.54%	60.67%	35.71%	28.61%	29.25%	41%	49%	41%	41%
51567 8919	-190.10%	-233.80%	-310.84%	-134.58%	-222.33%	-76.73%	-249.85%	-261.56%	-646.78%	-124.49%	-269%	-183%	-409%	-24%

Slika 8.15: Slika prikazuje, kolikšen je faktor izboljšanja LSTM pri posamezni metodi dodajanja podobnih časovnih vrst za vsako od deset skupin posebej. Metrika intenzitete je izračunana po vsaki metodi dodajanja podobnih časovnih vrst posebej, kar pomeni po stolpcih.

8.2.1 Analiza po drugi skupini poskusov

Zanimalo nas je, kako se faktor izboljšanja posamezne metode dodajanja podobnih časovnih vrst odraža na vsaki od deset skupin posebej. V ta namen smo pripravili matriko intenzitete, prikazano na sliki 8.15.

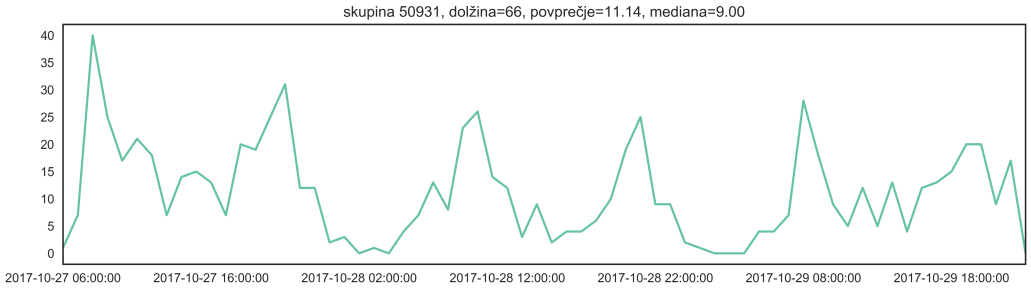
Iz slike 8.15 opazimo najprej, da pri skupinah, ki so na sliki označene z rdečo, to so *49007-8425*, *49649-8561* in *51567-8918*, nobena od metod ni dosegla izboljšanja napovedi napram LSTM iz prve skupine poskusov. Medtem so pri skupini, označeni z zeleno, to je *50931-8850*, vse metode dodajanja časovnih vrst dosegle izboljšanje. Ostale skupine in metode niso prikazovale vzorcev, na podlagi katerih bi lahko sklepali zaključke.

Za nadaljnjo analizo smo se tako osredotočili na primerjavo zelene skupine in rdečih skupin. Pripravili smo grafe njihovih časovnih vrst. Časovna vrsta klikov zelene skupine je na sliki 8.16 in časovne vrste klikov rdečih skupin na sliki 8.17.

Pregledali smo značilke časovnih vrst, prikazanih na sliki 8.18.

Pripravili smo tudi graf resnične testne časovne vrste klikov in njene napovedi različnih modelov za zeleno skupino *50931-8850* na sliki 8.19 in ene od rdečih skupin *51567-8918* na sliki 8.20.

Sliki 8.19 in 8.20 prikazujeta resnično časovno vrsto testnega okna in njene napovedi za zeleno in rdečo skupino. Vidimo lahko, da pri zeleni skupini



Slika 8.16: Slika prikazuje časovno vrsto klikov skupine *50931-8850* pri kateri so vse metode dodajanja podobnih časovnih vrst prispevala k izboljšanju LSTM iz prve skupine poskusov.

LSTM iz prve skupine poskusov ni dosegel natančnih napovedi. Slednje lahko vidimo tudi na sliki 8.7. Dodane časovne vrste so tako uspele napovedi močno izboljšati. Medtem se je LSTM iz prve skupine poskusov pri rdeči skupini bolj približal resnični časovni vrsti.

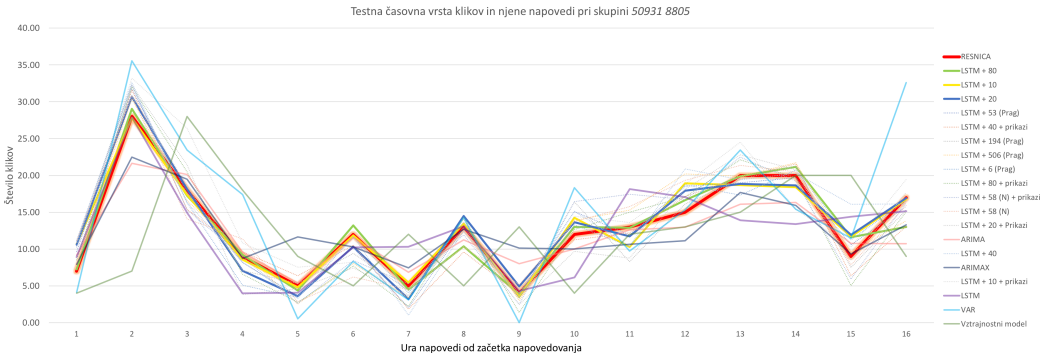
Za zadnji poskus smo preverili, ali lahko podobno pričakujemo tudi pri daljših napovedih, to je, ko smo napovedovali štiri ure v prihodnost in osem ur v prihodnost.

Prišli smo do rezultatov na sliki 8.21, iz katerih lahko sklepamo, da ugotovitve veljajo tudi pri daljših napovedih.

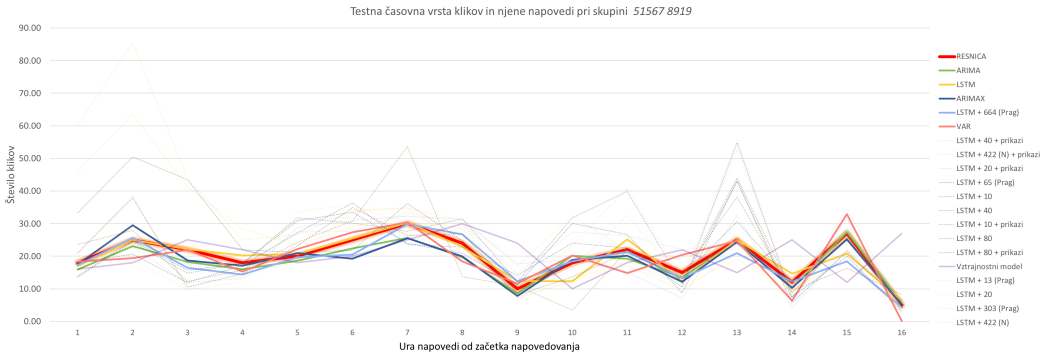
Slika 8.17: Slika prikazuje časovne vrste klikov skupin *49007-8425*, *49649-8561* in *51567-8918*, pri katerih so vse metode dodajanja podobnih časovnih vrst poslabšale LSTM iz prve skupine poskusov.

	49007–4250	49649–5610	51567–9190	50931–8050
Spreminjanje variance v ostanku	0.34	0.27	0.50	0.21
Spektralna entropija	0.77	0.87	0.85	0.87
Avtokorelacija prvega reda	0.69	0.63	0.47	0.58
Nivo zamika pri premikajočem oknu	1.14	1.33	1.08	1.07
Sprememba variance	0.75	0.87	0.76	0.72
Število križnih točk	4	16	8	10
Ravnih delov pri diskretizaciji	4	5	5	7
Moč trenda	0.66	0.75	0.56	0.18
Moč linearnosti	-3.83	2.35	-2.62	-3.00
Moč krivin	1.39	-1.51	0.77	1.42E-11
Moč spičastosti	0.0002	0.0001	0.0005	0.0007
Rezultat Kullback-Leiblerja	1.36	0.89	1.00	1.11
Indeks največjega KL rezultata	10	31	39	16
Povprečje	11.32	14.25	20	10.70
Varianca	76.48	87.03	195.25	87.36

Slika 8.18: Slika prikazuje značilke časovnih vrst klikov rdečih skupin *49007-8425*, *49649-8561* in *51567-8918*, pri katerih so vse metode dodajanja podobnih časovnih vrst poslabšale LSTM iz prve skupine poskusov ter zelene skupine *50931-8850*, pri kateri so vse metode dodajanja podobnih časovnih vrst prispevala k izboljšanju LSTM iz prve skupine poskusov.



Slika 8.19: Slika prikazuje testno časovno vrsto klikov in njene napovedi vseh modelov zelene skupine *50931-8850*, pri kateri so vse metode dodajanja podobnih časovnih vrst prispevale k izboljšanju LSTM iz prve skupine poskusov.



Slika 8.20: Slika prikazuje testno časovno vrsto klikov in njene napovedi vseh modelov rdeče skupine *51567-8918*, pri kateri so vse metode dodajanja podobnih časovnih vrst poslabšale LSTM iz prve skupine poskusov.

Metoda dodajanja podobnih časovnih vrst	Napovedovanje eno uro v prihodnost		Napovedovanje štiri ure v prihodnost		Napovedvanje osem ur v prihodnost	
	št. skupin uspešnejši	št. skupin slabši	št. skupin uspešnejši	št. skupin slabši	št. skupin uspešnejši	št. skupin slabši
LSTM exp2 10	4	6	3	7	4	6
LSTM exp2 20	4	6	4	6	3	7
LSTM exp2 40	4	6	5	5	4	6
LSTM exp2 80	2	8	3	7	5	5
LSTM exp2 N	3	7	6	4	3	7
LSTM exp2 N + prikazi	3	7	3	7	4	6
LSTM + najuspešnejši	7	3	9	1	7	3

Slika 8.21: Slika prikazuje matriko intenzitete o tem, pri koliko skupinah od desetih je metoda dodajanja podobnih časovnih vrst prispevala k izboljšanju natančnosti modela LSTM pri napovedovanju ene ure, štiri ali osem ur v prihodnost.

	ARIMA	ARIMAX	VAR	LSTM z optimizacijo parametrov	LSTM brez optimizacije parametrov
Čas izvajanja	6s	9s	2s	15min	1.5min

Tabela 8.2: Okviren čas učenja in napovedovanja časovne vrste klikov s slike 2.1 pri napovedovanju osem ur v prihodnost na posameznem modelu.

8.3 Razprava

V prvem poskusu smo primerjali LSTM s klasičnimi modeli za napovedovanje časovnih vrst, pri čemer smo uporabili obdelave časovnih vrst, ki so se v sorodnih delih (poglavje 3) izkazale za uspešne pri izboljšanju natančnosti napovedi LSTM. S poskusom smo potrdili, da so bile LSTM v povprečju najbolj uspešen model.

Zaradi potrebne optimizacije hiperparametrov LSTM in večkratnega poganjanja modela zaradi stohastične lastnosti nevronskih mrež, je bil čas, potreben za zanesljive napovedi LSTM, mnogo daljši kot pri napovedovanju s klasičnimi modeli. Tabela 8.2 prikazuje okviren čas, ki so ga potrebovali modeli ARIMA, ARIMAX, VAR, LSTM z optimizacijo metaparametrov in LSTM brez optimizacije metaparametrov pri napovedovanju klikov s slike 2.1 osem ur v prihodnost. Kljub paralelizaciji LSTM je bil čas učenja in napovedovanja približno 150-krat daljši od modela ARIMA in približno 450-krat daljši od VAR. Ker je razlika v računskem času velika, ga je potrebno upoštevati pri izbiri metode.

Potrdili smo tudi, da obdelava časovnih vrst izboljša napovedi. Za najbolj uspešno se ni izkazala uporaba vseh obdelav časovnih vrst hkrati. Katera kombinacija je bila najbolj uspešna, je bilo odvisno od skupin časovnih vrst, na podlagi katerih smo napovedovali,kot od dolžine napovedi. Iz ugotovitev lahko sklepamo, da moramo obdelavo časovnih vrst prilagoditi podatkom in tudi ciljem naših napovedi, kot je npr., kako daleč napovedujemo. Pri krajših napovedih je k uspehu najbolj pripomogla odstranitev trenda, medtem ko je pri daljših napovedih najbolj pripomogla odstranitev sezonskosti. Odstrani-

tev trenda je bolj pripomogla pri kratkih napovedih zato, ker se pri daljših napovedih napovedna napaka prenaša na naslednje ure napovedi, saj se pri računanju inverza trenda napovedi seštevajo. Medtem je bilo pri daljših napovedih pričakovati, da bo odstranitev sezonskosti močneje prispevala k natančnosti napovedi, ker bi jo LSTM sicer moral napovedati. Za LSTM pa se je že izkazalo, da sezonskosti ne napoveduje najbolje, kar smo omenili pri opisu sorodnih del (poglavje 3).

V drugi skupini poskusov smo poskusili napovedovanje LSTM še izboljšati. Uporabili smo gručenje z mešanjem Gaussovih verjetnostnih porazdelitev (GMM), da smo poiskali najbolj podobne časovne vrste in jih dodali v učno množico za učenje LSTM. Poskusi so pokazali, da je z dodajanjem podobnih časovnih vrst napovedi mogoče izboljšati, vendar pa je potrebno nadaljnje delo za boljše razumevanje okoliščin, ki vplivajo na izboljšanje napovedi.

Kot smo v prvi skupini poskusov uporabili naključno preiskovanje prostora hiperparametrov LSTM in mrežno preiskovanje metod obdelave časovnih vrst, bi lahko vključili tudi mrežno preiskovanje metod za dodajanje podobnih časovnih vrst.

Učenje bi lahko izvedli tudi v dveh fazah, in sicer da najprej preiščemo z naključnim preiskovanjem prostora hiperparametrov LSTM in z mrežnim preiskovanjem metod obdelave časovnih vrst. Nato bi na najbolj uspešnemu modelu dodali mrežno preiskovanje metod za dodajanje podobnih časovnih vrst. Tako bi čas izvajanja skrajšali.

Poglavje 9

Zaključek

Primerjali smo klasične modele ARIMA, ARIMAX in VAR z nevronskimi mrežami z dolgim kratkoročnim spominom (ang. *Long Short-Term Memory neural networks, LSTM*) pri napovedovanju časovnih vrst klikov na oglase v družbenem omrežju Facebook. Pri tem smo v sorodnih delih zbirali vse dosedanje znanje, ki je pripomoglo k izboljšanju natančnosti napovedi LSTM, saj smo želeli izkoristiti njihov poln potencial – uporabiti najnaprednejši (ang. *state-of-the-art*) model za napovedovanje časovnih vrst.

V magistrskem delu smo izvedli dve skupini poskusov. Za prvo skupino poskusov smo v sorodnih delih poiskali pristope obdelav časovnih vrst, ki so se do sedaj izkazali za uspešen način izboljšanja napovedi LSTM. Poskusili smo z vsemi kombinacijami obdelav, sestavljenih iz logaritemske preslikave podatkov, normalizacijo podatkov med vrednosti 0 in 1, odstranitev sezonskosti ter odstranitev trenda časovnih vrst. Z drugo skupino poskusov smo želeli natančnost LSTM še izboljšati z uporabo velikega števila podobnih časovnih vrst pri učenju modela. S slednjim pristopom smo želeli izkoristiti prednost nevronskih mrež – današnji obseg masovnih podatkov, s katerim so se LSTM že izkazale za uspešne pri računalniškem vidu in obdelavi jezika.

V prvi skupini poskusov smo ugotovili, da so LSTM s pomočjo obdelav časovnih vrst najbolj natančno napovedovale klike na oglase. Pri krajših napovedih je k uspehu najbolj pripomogla odstranitev trenda, medtem ko je

pri daljših napovedih najbolj pripomogla odstranitev sezonskosti. Odstranitev trenda je bolj pripomogla pri kratkih napovedih zato, ker se pri daljših napovedih napovedna napaka prenaša na naslednje ure napovedi, saj se pri računanju inverza trenda napovedi seštevajo. Medtem je bilo pri daljših napovedih pričakovati, da bo odstranitev sezonskosti močneje prispevala k natančnosti napovedi, ker bi jo LSTM sicer moral napovedati. Za LSTM pa se je že izkazalo, da sezonskosti ne napoveduje najbolje.

Za napovedovanje LSTM tako predlagamo, da se v primeru kratkih napovedi uporabi odstranitev trenda, v primeru dolgih in sezonskih podatkov pa odstranitev sezonskosti.

V drugi skupini poskusov smo najuspešnejšim LSTM iz prve skupine poskusov dodali podobne časovne vrste in primerjali natančnost napovedi. Podobne časovne vrste smo poiskali s pomočjo gručenja z mešanjem Gaussovih verjetnostnih porazdelitev (GMM). Gručili smo vektorje značilk, ki so opisovale časovne vrste. Uporabili smo več pristopov dodajanja podobnih časovnih vrst. Najprej smo fiksirali število časovnih vrst, ki se dodajo k učenju LSTM, potem smo uporabili Bayesovski informacijski kriterij, da smo dobili optimalno število gruč, ter uporabili tiste časovne vrste, ki so bile v isti gruči kot časovna vrsta, ki smo jo napovedovali. Za tem smo poskusili s podajanjem pragovne vrednosti metrike podobnosti – dodali smo le tiste podobne časovne vrste, ki so bile na podlagi metrike podobnosti bolj podobne od pragovne vrednosti.

Ugotovili smo, da je z dodajanjem podobnih časovnih vrst možno izboljšati napovedi, vendar ta pristop vedno ne pomaga. Vsak pristop je pri kakšni od skupin časovnih vrst, ki smo jih uporabili za napovedovanje, izboljšal napovedi in vsak je pri kateri drugi skupini napovedi tudi poslabšal. Potrebno je nadaljnje delo za boljše razumevanje okoliščin, ki vplivajo na izboljšanje napovedi.

Za najbolj natančne napovedi LSTM predlagamo, da se naredi preiskovanje prostora, kjer dimenzije prostora predstavljajo pristopi dodajanja podobnih časovnih vrst.

Literatura

[1] K. Bandara, C. Bergmeir in S. Smyl, "Forecasting across time series databases using long short-term memory networks on groups of similar series," *arXiv preprint arXiv:1710.03222*, 2017.

[2] S. Makridakis in M. Hibon, "The M3-competition: results, conclusions and implications," *International journal of forecasting*, vol. 16, no. 4, str. 451–476, 2000.

[3] W. Yan, "Toward automatic time-series forecasting using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 7, str. 1028–1039, 2012.

[4] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ in S. Khudanpur, "Recurrent neural network based language model," v *11th Annual Conference of the International Speech Communication Association*, str. 1045–1048, 2010.

[5] I. Sutskever, O. Vinyals in Q. V. Le, "Sequence to sequence learning with neural networks," v *Advances in neural information processing systems*, str. 3104–3112, 2014.

[6] A. Graves, A.-r. Mohamed in G. Hinton, "Speech recognition with deep recurrent neural networks," v *Acoustics, speech and signal processing (icassp), 2013 IEEE international conference on*, str. 6645–6649, IEEE, 2013.

[7] S. Smyl, "Cognitive toolkit helps win 2016 CIF international time series competition," 2016.

[8] G. E. Box, G. M. Jenkins, G. C. Reinsel in G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[9] C. Granger in P. Newbold, "Spurious regressions in econometrics," *Journal of Econometrics*, vol. 2, no. 2, str. 111 – 120, 1974.

[10] H. Lütkepohl, *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[11] Z. Zhao, W. Chen, X. Wu, P. C. Chen in J. Liu, "LSTM network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, str. 68–75, 2017.

[12] C. Yuan, S. Liu in Z. Fang, "Comparison of China's primary energy consumption forecasting by using ARIMA (the autoregressive integrated moving average) model and GM (1, 1) model," *Energy*, vol. 100, str. 384–390, 2016.

[13] S. Smyl in K. Kuber, "Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks," v *36th International Symposium on Forecasting*, 2016.

[14] X. Ma, Z. Tao, Y. Wang, H. Yu in Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, str. 187–197, 2015.

[15] X. Li, L. Peng, X. Yao, S. Cui, Y. Hu, C. You in T. Chi, "Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation," *Environmental Pollution*, vol. 231, str. 997–1004, 2017.

[16] T. Fischer in C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," FAU Discussion Papers in Economics 11/2017, Friedrich-Alexander University Erlangen-Nuremberg, Institute for Economics, 2017.

[17] A. Naccarato, S. Falorsi, S. Loriga in A. Pierini, "Combining official and Google Trends data to forecast the Italian youth unemployment rate," *Technological Forecasting and Social Change*, 2017.

[18] B. Cortez, B. Carrera, Y.-J. Kim in J.-Y. Jung, "An architecture for emergency event prediction using LSTM recurrent neural networks," *Expert Systems with Applications*, 2017.

[19] S. Hochreiter in J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, str. 1735–1780, 1997.

[20] C. Liu, Z. Jin, J. Gu in C. Qiu, "Short-term load forecasting using a long short-term memory network," v *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, str. 1–6, Sept 2017.

[21] Y. Zhang, H. Dai, C. Xu, J. Feng, T. Wang, J. Bian, B. Wang in T.-Y. Liu, "Sequential click prediction for sponsored search with recurrent neural networks.," v *AAAI*, str. 1369–1375, 2014.

[22] Q.-H. Chen, S.-M. Yu, Z.-X. Guo in Y.-B. Jia, "Estimating Ads' click through rate with recurrent neural network," v *ITM Web of Conferences*, vol. 7, str. 04001, EDP Sciences, 2016.

[23] H. Liu, X.-w. Mi in Y.-f. Li, "Wind speed forecasting method based on deep learning strategy using empirical wavelet transform, long short term memory neural network and Elman neural network," *Energy Conversion and Management*, vol. 156, str. 498–514, 2018.

[24] N. Laptev, J. Yosinski, L. E. Li in S. Smyl, "Time-series extreme event forecasting with neural networks at Uber," v *Int. Conf. on Machine Learning*, 2017.

[25] X. Shao, D. Ma, Y. Liu in Q. Yin, "Short-term forecast of stock price of multi-branch LSTM based on K-means," v *2017 4th International Conference on Systems and Informatics (ICSAI)*, str. 1546–1551, Nov 2017.

[26] R. J. Hyndman, E. Wang in N. Laptev, "Large-scale unusual time series detection," v *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*, str. 1616–1619, IEEE, 2015.

[27] G. Lai, W.-C. Chang, Y. Yang in H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," *arXiv preprint arXiv:1703.07015*, 2017.

[28] M. Tajnik, "Integrirani avtoregresijski modeli s premikajočimi sredinami za napovedovanje porabe električne energije," Master's thesis, Univerza v Mariboru, Fakulteta za naravoslovje in matematiko, 2016.

[29] Q. Yin, R. Zhang, Y. Liu in X. Shao, "Forecasting of stock price trend based on CART and similar stock," v *2017 4th International Conference on Systems and Informatics (ICSAI)*, str. 1503–1508, Nov 2017.

[30] W. Anggraeni, R. A. Vinarti in Y. D. Kurniawati, "Performance comparisons between ARIMA and ARIMAX method in moslem kids clothes demand forecasting: Case study," *Procedia Computer Science*, vol. 72, str. 630–637, 2015.

[31] H. Qiu, S. Xu, F. Han, H. Liu in B. Caffo, "Robust estimation of transition matrices in high dimensional heavy-tailed vector autoregressive processes," v *International Conference on Machine Learning*, str. 1843–1851, 2015.

[32] I. Melnyk in A. Banerjee, "Estimating structured vector autoregressive model," 2016.

[33] H.-F. Yu, N. Rao in I. S. Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," v *Advances in neural information processing systems*, str. 847–855, 2016.

[34] W. S. McCulloch in W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, str. 115–133, 1943.

[35] D. L. Donoho *et al.*, "High-dimensional data analysis: The curses and blessings of dimensionality," *AMS Math Challenges Lecture*, vol. 1, str. 32, 2000.

[36] W. Huang, G. Song, H. Hong in K. Xie, "Deep architecture for traffic flow prediction: deep belief networks with multitask learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, str. 2191–2201, 2014.

[37] M. Nelson, T. Hill, W. Remus in M. O'Connor, "Time series forecasting using neural networks: Should the data be deseasonalized first?," *Journal of forecasting*, vol. 18, no. 5, str. 359–367, 1999.

[38] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt in G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," v *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, str. 108–122, 2013.

[39] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. E. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. B. Hamrick, J. Grout, S. Corlay *et al.*, "Jupyter Notebooks-a publishing format for reproducible computational workflows.," v *ELPUB*, str. 87–90, 2016.

[40] F. Chollet *et al.*, "Keras." `https://github.com/fchollet/keras`, 2015.

[41] R. Taylor, "Pyflux: An open source time series library for Python." `http://pyflux.readthedocs.io/`, 2016.

[42] S. Seabold in J. Perktold, "Statsmodels: Econometric and statistical modeling with Python," v *9th Python in Science Conference*, 2010.

[43] Nicolas, "Strategy optimisation with walk forward analysis," 2017.