UMETNA INTELIGENCA IN VEJICE

Računalništvo in informatika

Raziskovalna naloga

Avtorica: Nadezhda Komarova
Letnik: 2.

Mentor: Gregor Anželj, prof. – Gimnazija Bežigrad
Somentor: prof. dr. Marko Robnik-Šikonja – Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

Ljubljana, 2020

Gimnazija Bežigrad

Zahvala

Najprej bi se rada zahvalila mojima mentorjema prof. dr. Marku Robniku-Šikonji in gospodu Gregorju Anželju, ki sta me vseskozi vodila pri raziskovalnem delu.

Prav tako se iskreno zahvaljujem profesorjem Gimnazije Bežigrad, ki so me spodbujali pri raziskovalnem delu.

Hvaležna sem tudi svoji družini, ki me je zelo podpirala ravno takrat, ko mi je bila izdelava raziskovalne naloge najtežja.

Kazalo

Seznam prilog

Pri raziskovalni nalogi sem uporabila naslednjo programsko kodo, napisano v programskem jeziku Python:

```
import os
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.layers import Embedding, LSTM, Dense, \\
     Dropout, Bidirectional
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import regularizers
import tensorflow.keras.utils as ku
import numpy as np
import matplotlib as plt

directory = "\ccGigafida-text.zip\ccGigafidaV1_0-text"
files = os.listdir(directory)

tokenizer = Tokenizer()

data_c = []
sequences = []
labels = []

for elem in files:
    handle = open(elem, "r")
    for line in handle:
        data_c.append(line)
        t_list = tokenizer.texts_to_sequences([line])[0]
        for i in range(1, len(t_list)):
            sequences.append(t_list[:i+1])
            this_app = []
            for j in range in t_list[:i+1].size():
                if j == t_list[:i+1].size() - 1:
                    this_app.append(0)
                    continue
                if t_list[:i+1][j+1] == ',':
                    this_app.append(1)
                else:
                    this_app.append(0)
            labels.append(this_app)
    handle.close()

tokenizer.fit_on_text(data_c)
num_words = len(tokenizer.word_index) + 1
```

```
max_num = max([len(x) for x in sequences])
sequences = np.array(pad_sequences(sequences, \\
                    maxlen = max_num, padding='pre'))

labels = ku.to_categorical(labels, num_classes=num_words)

model = Sequential()
model.add(Embedding(num_words, 100, input_length=max_num))
model.add(Bidirectional(LSTM(150, return_sequences = True)))
model.add(Dropout(0.2))
model.add(LSTM(100))
model.add(Dense(num_words/2, activation='relu', \\
          kernel_regulizer=regularizers.l2(0.01)))
model.add(Dense(num_words, activation='softmax'))
model.compile(loss='categorical_crossentropy', \\
              optimizer='adan', metrics=['accuracy'])
print(model.summary())

his = model.fit(sequences, labels, epochs = 100, verbose=1)
accur = his.history['acc']
loss = his.history['loss']

plt.plot(range(len(accur)), 'b', laber='Training acc.')
plt.title('Training acc.')
plt.figure()
plt.plot(range(len(accur)), loss, 'b', label='Training loss')
plt.title('Training loss')
plt.legend()
plt.show()
```

Povzetek

V raziskovalni nalogi sem preučevala učinkovitost uporabe globokih nevronskih mrež pri napovedovanju ustreznosti postavljanja vejic v slovenskem jeziku. Uporabila sem globoko nevronsko mrežo, ki je bila sestavljena iz BLSTM celic (*angl. Bidirectional Long Short-Term Memory*). Učinkovitost izbire slednje sem izmerila s točnostjo napovedi in interno oceno napake nevronske mreže.

Problema sem se lotila s strojnim učenjem, ker se je njegova uporaba pri svetovnih jezikih izkazala ja bolj učinkovito od predhodnih pristopov. Za slovenski jezik rešitev na osnovi globokih nevronskih mrež še ne obstaja. Tema je pomembna, ker je ustrezno postavljanje vejic najpogostejša napaka slovenskih piscev.

Kot podatkovno množico sem uporabila korpus ccGigafida, ki vsebuje 100 milijonov besed v standardni slovenščini. Oblikovala sem model, ki napove, na katerih mestih v stavku bi najverjetneje morale stati vejice.

Ključne besede: nevronske mreže, strojno učenje, jezikovni modeli.

1. Uvod

Za raziskovalno nalogo na področju umetne intelligence v jeziku sem se odločila, ker me je zanimala povezava med računalniško tehnologijo in naravnim jezikom. Raziskovalni problem je pomemben, ker obstoječe rešitve za popravljanje vejic v stavku še niso uporabne. Cilj moje raziskovalne naloge je s pomočjo modelov strojnega učenja napovedati pravilnost vejic v stavku.

Zahtevnost napovedi vejic v slovenskih povedih je v tem, da ni mogoče postaviti enoličnih in enostavnih pravil. Razlog je povezanost vejic s skladnjo. Obstaja sicer nekaj veznikov, ki bi uvajajo podredja in služijo kot preprosti indikatorji tega, da bi morala pred njimi stati vejica. To so na primer: ker, zato, ki, ko, da in če.

Slabost tovrstnega pristopa je, da takih pravil ne moremo uporabiti v vsakem primeru. V več primerih bi z njihovo uporabo vstavili preveč ali premalo vejic, na primer pri vrinjenih stavkih.

Problem postavljanja vejic v slovenščini poskušam rešiti pomočjo nevronskih mrež. Slednje so v zadnjem času doživele ogromen napredek in so rešile več problemov, ki so se zdeli nerešljivi s pomočjo predhodnih pristopov. Dosegle so več uspehov na področju razumevanja naravnega jezika, predvsem pri strojnem prevajanju, pri razpoznavanju govora in generiranju besedil.

Globoke nevronske mreže pri razumevanju naravnega jezika posnemajo ljudi. Tako kot mi se naučijo kontekstualne primernosti besed, ki je še posebej koristna na področju napovedovanja naslednje besede. Problem pravilne rabe vejic v slovenščini je soroden problemu napovedovanja naslednjih besed oziroma generiranju besedil, ki je eden temeljev razumevanja naravnega jezika.

Pri raziskovalnem delu sem za učenje nevronske mreže uporabila množico zanesljivih besedil iz velike zbirke slovenskih besedil Gigafida. Z njeno pomočjo sem ustvarila pripomoček za postavljanje vejic v slovenščini.

Za pisanje kode programa za učenje nevronske mreže sem uporabljala programski jezik Python in knjižnico Keras, ki v ozadju kliče knjižnico Tensorflow. Naša globoka nevronska mreža, prilagojena slovenski skladnji, uporablja LSTM celice (*angl. Long Short-Term Memory*), ki spadajo med rekurentne nevronske mreže.

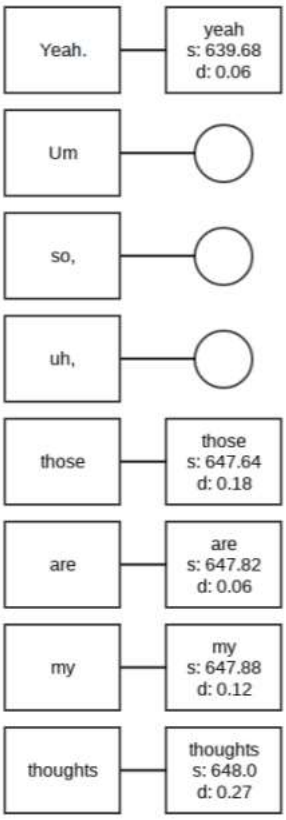
V nadaljevanju raziskovalne naloge bom v 2. razdelku podrobno predstavila teoretično ozadje teme, delovanje modelov strojnega učenja in delovanje uporabljenih jezikovnih modelov. V rezdelku 3 bom predstavila podatkovne množice in postopek testiranja modela z evalvacijo in razširitvijo v nadaljne raziskave. V 4. razdelku bodo predstavljeni rezultati raziskovalne naloge.

2. Teoretični del

V teoretičnem delu bom na začetku opisala, kaj je bilo na področju napovedovanja ločil z uporabo globokih nevronskih mrež narejeno do sedaj. Nato bom predstavila zgradbo in delovanje nevronskih mrež ter jezikovnih modelov. Opisala bom postopek napovedovanja s pomočjo modelov strojnega učenja in predstavila svoj pristop.

2.1. Obstoječa dela

V zadnjem času je nastalo več znanstvenih člankov o napovedovanju uporabe ločil, vendar delujejo za druge jezike, predvsem za angleški jezik. Uporaba vejic v slovenščini je skladenjska in zahteva drugačen pristop.



Slika 1: Delo z jezikovnimi zaporedji

Leta 2018 so [4] raziskali napovedovanje ločil v pogovornem sporočanju (Punctuation Prediction Model for Conversational Speech). Pri svojem delu so uporabili dve globoki nevronski mreži za označevanje zaporedij (*angl. sequence labelling models*).

Prva mreža je bila tipa BLSTM (*angl. Bidirectional Long Short-Term Memory*), druga pa konvolucijska (*angl. CNN, Convolutional Neural Network*). Podatkovna množica, na kateri sta bila modela naučena je bil Fisherjev korpus [4], ki vključuje podatke o ločilih v besedilih.

Konvolucijska mreža CNN je dosegla večjo natančnost (*angl. precision*), mreža BLSTM pa večji priklic (*angl. recall*). Natančnost je delež relevantnih primerov izmed dobljenih primerov, med tem ko je priklic – delež primerov označenih za

relevantne izmed vseh relevantnih primerov v testni množici. Drugi model globoke nervonske mreže je naredil pri testiranju manj napak.

Avtorji [4] so se lotili ločil pri problemu prepoznavanja govora, saj ena največjih težav tega področja prav to, da avtomatski sistemi za razpoznavanje govora (*angl. automatic speech recognition – ASR systems*) niso sposobni postavljanja ločil.

To naredi nastalo besedilo težje razumljivo za bralca, oteži pa tudi procesiranje drugim sistemom, ki delajo z naravnim jezikom.

Izdelan primoček vnese pravilna ločila v povedi brez ločil. Uporabili so dvostranske rekurentne nevronske mreže (*angl. bidirectional reccurent neural network*). Za podatkovano množico so uporabili WikiText Long Term Dependency Language Modelling Dataset [4], nato pa še manjši Europarl v7 corpus.

S prvo so uspeli doseči boljše rezultate, in sicer napako z vrednostjo 31,4% in F_1 mero 78,5%. F_1 mera je mera za točnost testa v statistični analizi binarne klasifikacije.

Pham in sod. (2020) so izdelali sistem za vietnamski jezik.

2.2. Opis metodologije – nevronske mreže

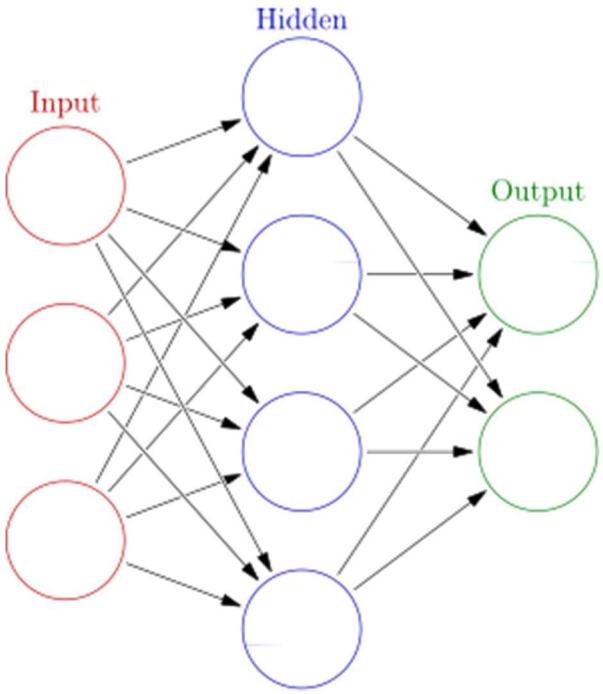
Strojno učenje z globogimi nevronskimi mrežami je v zadnjem desetletju močno napredovalo. Metode, ki jih uporabljam, sicer vedno niso tako nove.

Nevronske mreže so bile predstavljene že leta 1944 s strani Warrena McCullougha in Waltera Pittsa, oba sta delovala na Univerzi v Čikagu. [4]

Umetni nevroni skušajo posnemati delovanja bioloških nevronov z uporabo računalniške tehnologije. Defenicija neronske mreže se glasi:

»Umetna nevronska mreža je računalniški učni sistem, ki uporablja mrežo funkcij z namenom, da razume in prevede podatke in določene oblike v željen izhod.«

V nadaljevanju bom pojasnila zgradbo in delovanje nevronske mreže.



Slika 2: Zgradba preproste nevronske mreže

Nevronsko mrežo na sliki sestavljajo trije sloji: vhodni, ki je sestavljen iz treh nevronov, skriti sloj (štirije nevroni) in izhodni sloj (dva nevrona). Vsaka povezava med nevroni ima svojo utež. Utežena vsota vhoda vsakega nevrona se izračuna tako, da se izhod nevronov prejšnjega nivoja, iz katerih izhaja povezava k danemu nevronu pomnoži z utežjo povezave. Formula za izračun utežene vsote je

x_j ... *izhod nevrona prejšnjega sloja*

w_j ... *utež*

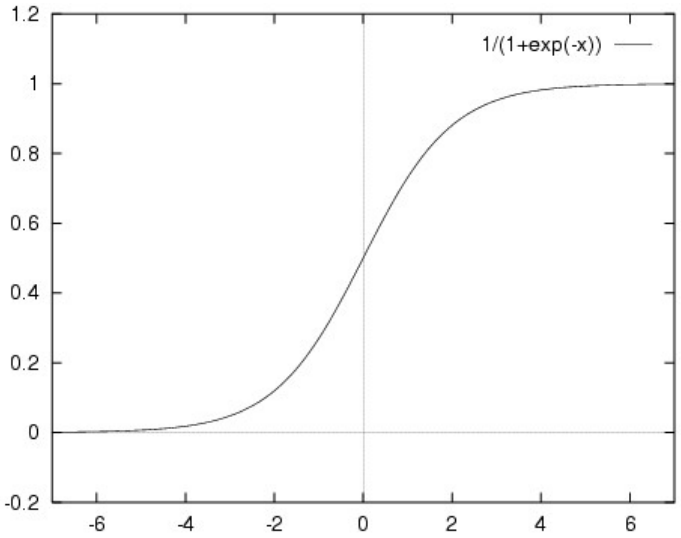
$$s = \sum_{j=1}^{N} x_j w_j$$

N ... *število vhodov*

Izhod bo umetni nevron prenesel na naslednje sloje in ga določimo s pomočjo aktivacijske funkcije. Cilj aktivacijske funkcije je pretvoriti vhodni signal nevrona v njegov izhodni signal. Obstaja več vrst aktivacijskih funkcij, najbolj razširjene so:

1. Logistična

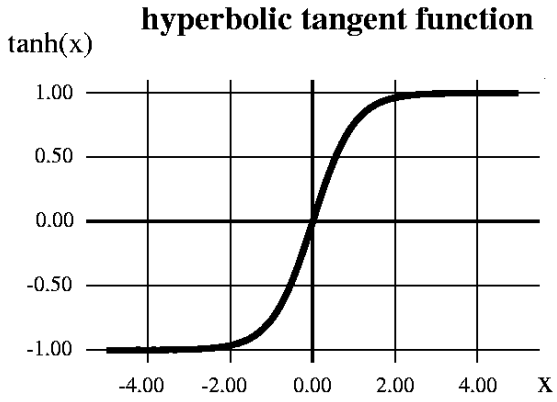
$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$



Slika 3: Graf logistične funkcije

2. Tanh (hiperbolični tangens)

$$f(x) = \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



Slika 4: Graf hiperboličnega tangensa

3. ReLu (Rectified linear units)

$$f(x) = x^+ = \max(0, x)$$

Logistična funkcija in hiperbolični tangens (tanh) sta predstavnici skupine sigmoidnih funkcij.

2.3. Jezikovni modeli

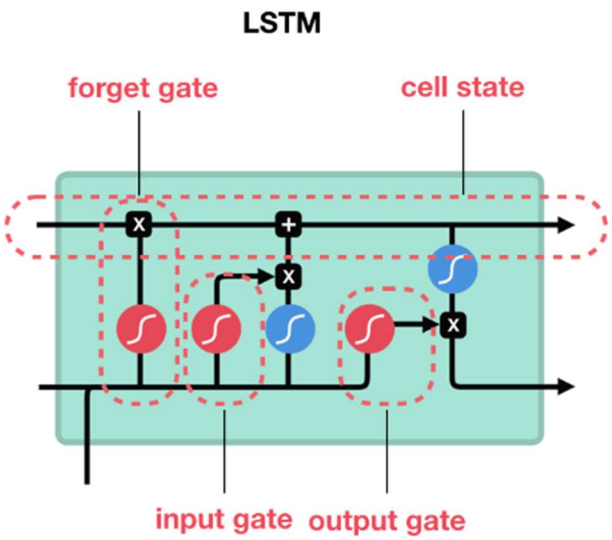
Jezikovni modeli v strojnem učenju ocenijo verjetnost pojavitve naslednje ali prejšnje besede. Uporabni so na mnogih področjih obdelave naravnega jezika.

Cilj jezikovnega modela je izračunati verjetnost naslednje besede.

$$P(w_N | w_1, w_2, w_3, ..., w_{N-1})$$

LSTM nevronske mreže

Težava navadnih rekurentnih nevronskih mrež (RNN) je njihovo »vidno polje«, ki ni dovolj široko, zato lahko pomembne informacije, ki so bile obdelane na začetku besedila, pozabijo.



Slika 5: Model LSTM

Rešitev te težave so nevronske mreže LSTM (*angl. Long Short-Term Memory*) na sliki 5. Imajo notranje mehanizme (»vrata« – *angl. gates*), ki regulirajo pretok informacije. »Vrata« se naučijo, katere informacije je pomembno ohraniti na dolgi rok in katere pozabiti.

2.4. Napovedovanje s pomočjo modelov strojnega učenja

Za uporabo besedil z nevronskimi mrežami je besede potrebno pretvoriti (vložiti) v številsko obliko. Pogosto je za to uporabljen algoritem (*angl. embedding algorithm*) word2vec ali GloVe.

Pri svojem delu sem uporabila vložitve tipa word2vec.

2.5. Opis mojega pristopa

Za učenje globokih nevronskih mrež sem uporabila knjižnici v programskem jeziku Python, in sices Keras in Tensorflow.

Kot nevronsko mrežo sem uporabila BLSTM mrežo (*angl. Bidirectional Long-Short Term Memory*) z aktivacijskima funkcijima ReLu in softmax. Model je imel 6 slojev, pri čemer so se na vhod pošiljala različno dolga zaporedja besed iz zbirke besedil. Po vsaki besedi v stavki je bilo označeno, ali mora na tem mestu stati vejica. Izhod je verjetno zaporedje prisotnosti vejic za različnimi besedami. V LSTM sloju je bilo 100 enot, v BLSTM sloju pa 150 enot.

Parametra učenja sta bila točnost in interna ocean napake nevronske mreže.

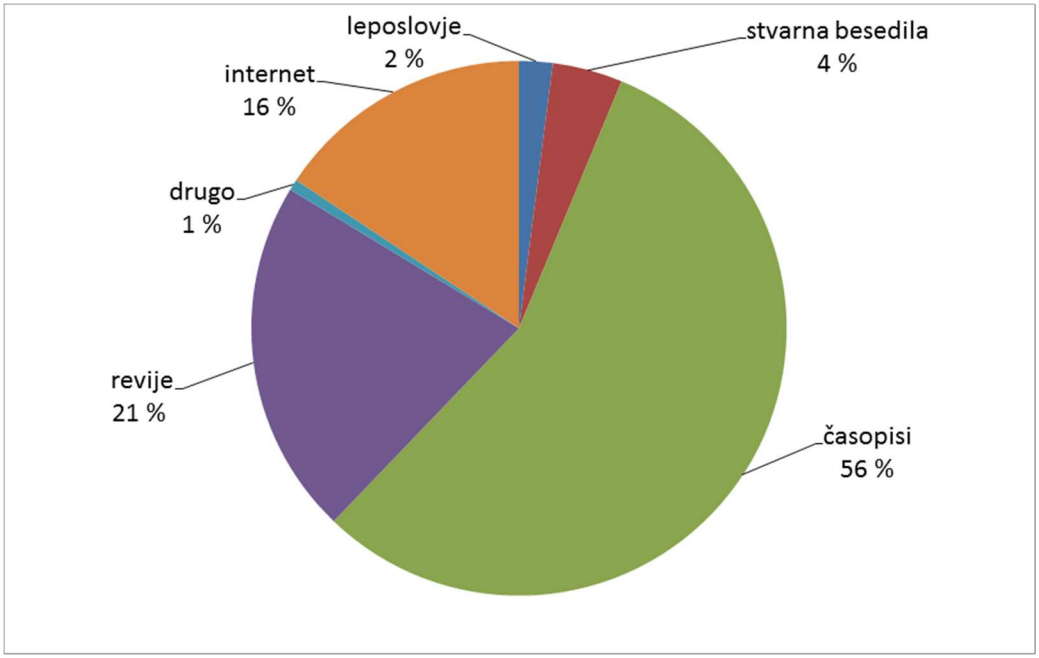
3. Eksperimentalni del

3.1. Podatkovna množica

Za učenje modela sem izbrala korpus za slovenščino ccGigafida (s 100 milijoni besed). To predstavlja 9% korpusa Gigafida, referenčnega korpusa slovenskega jezika.

Korpus, ki sem ga uporabila sestavlja 31.722 datotek. Vsaka izmed njih vključuje informacije o viru (npr. časopisni članki), letu izida, besedilni vrsti, naslovu in avtorju, če je ta znan.

Korpus je tako opremljen z morfosintaktičnimi opisi (*angl. PoS-tagged*) in lematiziran. Format, v katerem je podatkovna množica na voljo, je XML TEI format (*angl. Text Encoding Initiative P5*).



Slika 6: Vsebinska sestava jezikovanega korpusa Gigafida

3.2. Postopek testiranja modela

Napovedni model sem pripravila tako, da sem podatkovno množico – korpus razdelila na dva dela: prvi del je postal množica podatkov za učenje, drugi del pa množica podatkov za testiranje.

Pri testiranju sem merila naslednji dve količini: natančnost (*angl. accuracy*) in izguba (*angl. loss*).

Kategorična prečna entropija je funkcija izgube, ki se uporablja za kategorizacijske probleme.

$$L(y,\hat{y}) = -\sum_{j=0}^{M}\sum_{i=0}^{N}(y_{ij} * log(\hat{y}_{ij}))$$

$\hat{y}$...*napovedana vrednost*

y...*dejanska vrednost*

4. Rezultati

Pri testiranju je moj napovedni model strojnega učenja dosegel spodnje rezultate. Število epoh učenja sem nastavila na 100.

Epoha	Točnost	Interna ocena napake nevronske mreže
1	0.0207	6.9122
10	0.0599	5.6765
20	0.1418	4.7746
30	0.2293	3.9821
40	0.3388	3.2913
50	0.4618	2.7405
60	0.5581	2.3100
70	0.6383	1.9589
80	0.6900	1.7176
90	0.7323	1.4956
100	0.7564	1.3614

Tabela 1: Rezultati testiranja jezikovnega modela — učna množica

Učenje je trajalo približno en dan, uporabljen procesor je bil Intel® Core™ i5-7200U CPU @ 2.50GHz 2.71 GHz. Model najde vejice za vezniki ko, če, da, ker, saj in zato; pogosto jih najde tudi pri vrinjenih stavkih.

5. Razprava

Iz dobljenih rezultatov testiranja jezikovnega modela z uporabo tehnologije globokih nevronskih mrež lahko zaključim, da je bil dani model uspešnejši od klasičnih pristopov brez nevronskih mrež.

Raziskovalno nalogo bi lahko v nadaljevanju nadgradila s primerjavo več vrst nevronskih mrež, npr. LSTM in CNN ter z večjo učno množico.

Prav tako bi bilo smiselno v orodje vgraditi urejevalnik besedil (na primer LibreOffice), tako da širše uporaben za slovenski jezik.

6. Zaključek

Oblikovala sem model strojnega učenja za napovedovanje ločil v slovenskih stavkih. Ključne so bile BLSTM celice, ki so omogočile napoved verjetnosti pojavitve vejice po vsaki besedi v danem stavku.

Uporabljena metodologija se je izkazala za učinkovito. Koristno bi bilo model z nekaj spremembami preizkusiti tudi na drugih ločilih slovenskega jezika – dvopičje, narekovaj in podobno, ne samo na vejicah.

Do boljših rezultatov bi lahko prišla tudi z uporabo večje podatkovne množice, npr. kar s celotnim korpusom Gigafido.

7. Literatura

[1] *History: The 1940's to the 1970's.* Pridobljeno z: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html

[2] Juin C., Wei, R., D'Haro. L. in Banchs, R. (2017). *Punctuation prediction using a bidirectional recurrent neural network with part-of-speech tagging.* 1806-1811. 10.1109/TENCON.2017.8228151.

[3] Pham, T., Nguyen, N., Pham, Q., Cao, H. in Nguyen, B. (2020). *Vietnamese Punctuation Prediction Using Deep Neural Networks*. Chatzigeorgiou A. et al. (eds) SOFSEM 2020: Theory and Practice of Computer Science. SOFSEM 2020. Lecture Notes in Computer Science, vol 12011. Springer, Cham.

[4] Zelasko, P. et al. (2018). *Prediction Model for Conversational Speech.* arXiv:1807.00543.

7.1. Viri slik

Slika 1: Zelasko, P. et al. (2018). *Prediction Model for Conversational Speech.* arXiv:1807.00543.

Slika 2: *What is a Neural Network*? Pridobljeno s: https://deepai.org/machine-learning-glossary-and-terms/neural-network.

Slika 3: *Activation functions and it's types-Which is better?* Pridobljeno s: https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f.

Slika 4: *Activation functions and it's types-Which is better?* Pridobljeno s: https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f.

Slika 5: *Illustrated Guide to LSTM's and GRU's: A step by step explanation*. Pridobljeno s: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21.

Slika 6: *GIGAFIDA.* Pridobljeno s: http://www.slovenscina.eu/korpusi/gigafida.