Generiranje imena naselja pomocu LSTM mreže

Antonio Čogelja Morena Granić Fran Lubina Iva Jurković Jakov Juvančić Matej Logarušić

***Sažetak*—Cilj projekta je LSTM rekurzivna neuronska mreža na razini znakova koja generira realistična imena naselja na različitim jezicima. Korištenjem LSTM mreže, koja je prilagođena za analizu sekvencijskih podataka, cilj je razviti model sposoban za učenje jezičnih obrazaca i struktura iz postojećih imena naselja. Svrha mreže je generiranje novih imena temeljenih na tim naučenim obrascima, pri čemu se zadržavaju jezične i strukturne zakonitosti specifične za taj kontekst. Željena točnost modela $\eta = 0.4$.**

***Index Terms*—Naselje, LSTM, rekurzivne mreže, neuronske mreže, generiranje**

I. Uvod

Ishod projekta je LSTM rekurzivna neuronska mreža na razini znakova koja generira realistična imena naselja na različitim jezicima.
Mreža radi sa vektorima koji predstavljaju slova neke abecede (ovisno o odabiru jezika) proširene specijalnim znakom npr. $\Sigma = \{\text{hrv. abeceda}\} \cup \{\text{spec. znakovi}\} \cup \{\langle END \rangle\}$.
Ulaz mreže je one-hot vektor $\hat{\mathbf{x}}^{(t)}$ dimezije $|\Sigma| = 30 + 0 + 1$.
Ulaz mreže je ustvari samo približno one-hot (vidi III-B)

$$\mathbf{x}_i^{(t)} = \begin{cases} 1, & \text{ako } i = j \\ 0, & \text{inače} \end{cases} \tag{1}$$

Izlaz dobiven na kraju pojedinog vremenskog koraka t je vektor vjerojatnosti pojave pojednog znaka abecende.

$$\hat{\mathbf{y}}^{(t)} = \begin{bmatrix} p(c_0^{(t)}|c^{(t-1)} \cdots c^{(0)}) \\ p(c_1^{(t)}|c^{(t-1)} \cdots c^{(0)}) \\ \vdots \\ p(c_{|\Sigma|}^{(t)}|c^{(t-1)} \cdots c^{(0)}) \end{bmatrix} \quad \text{Gdje} \quad c \in \Sigma \tag{2}$$

Vjerojatnosti su dobivene softmax funkcijom parametriziranom hiperparametrom temperature τ.
Na temelju tih vjerojatnosti se uzorkuje konačni izlazni vektor $\mathbf{y}^{(t)}$, odnosno t-ti znak u imenu naselja.

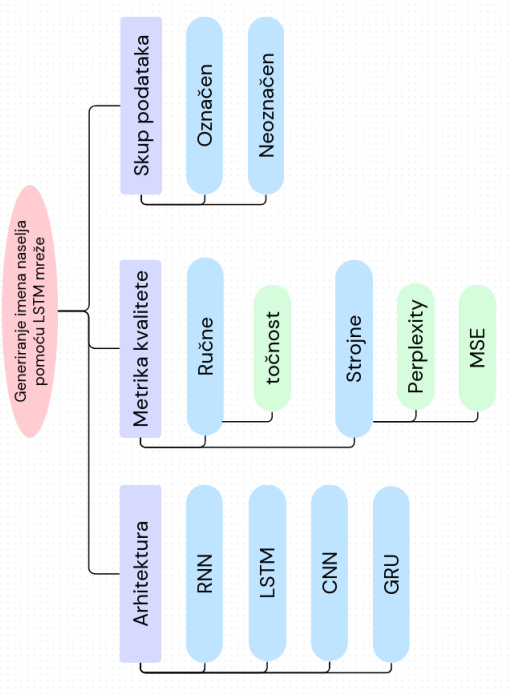
$$\mathbf{y}^{(t)} \sim \hat{\mathbf{y}}^{(t)} = \sigma_\tau(f(\mathbf{x}^{(t)}; \boldsymbol{\theta})) \tag{3}$$

$f(\mathbf{x}; \boldsymbol{\theta})$ predstavlja ukupno djelovanje ćelija modela nad njenim ulazom parametrizirano hiperparametrima modela $\boldsymbol{\theta} = \begin{bmatrix} |\hat{\mathbf{x}}| & |\mathbf{a}| & \mu & \tau \end{bmatrix}$ (opisani u poglavlju III-B). Temperaturno uzorkovanje je izabrano, jer omogućava eksperimentiranje i generiranje zanimljivih toponima. Izlaz mreže je niz znakova $\{\mathbf{y}^{(t)}\}\Big|_{t=0}^{T-1}$, odnosno ime naselja. Željena točnost modela η je

$$\arg\max_{\tau} \eta = 0.4$$

II. Pregled literature

U ovom poglavlju dajemo kratki pregled postojeće literature na način kako je to učinjeno u [1]. Svrstavanjem radova na temelju kriterija: pristupa dubokom učenju, funkciji pogreške/metrici kvalitete i skupu podataka.
Dobivamo taksonomiju na slici 1.
Razlog za odabir pojedine karakteristike dajemo u dotičnom poglavlju.



Slika 1. taksonomija rješenja za generiranje teksta

Nas samo zanimaju radovi na jezicima spomenutim u cjelini I na razini riječi sa primjenom generiranja jezičnih konstrukta.
U konačnici odabiremo karakteristike našeg rješenja, navedene u tablici III.

Nakon pretraživanja ukupno smo pronašli 9 relevantnih radova sa tražilica:

1) Google (5 radova)
2) IEEE Xplore (2 rada)
3) pretraživanje literature (2 rada)

Tablica I
DETALJI PRETRAŽIVANJA

		Komentar
tražilice	Google, Google Scholar, BASE, CORE, Science.gov, Semantic Scholar, Baidu scholar, RefSeek, CiteSeerX, ScienceOpen, The Lens, arXiv, AMiner, ACM, IEEE Xplore, Science Direct, Springer Link, Web of Science	Samo smo na tražilici Google našli relevatne radove (njih 5)
traženi pojam	(LSTM OR GRU OR CNN OR RNN OR Neural Network) AND ((city AND generator AND name) OR (generator AND name))	Napisan sintaksom i operatorima koje koristi Google, ali na ostalim tražilicama je korišten prilagođeni izraz.

Iako je problem kao takav složen (vidi poglavlje V), njegova izvedba je relativno jednostavna i teško primjenjiva na probleme u stvarnom svijetu te je zbog toga dostupno jako malo literature, koja je isključivo ograničena na hobi-projekte ([6], [7], [8], [9]) i projekte u sklopu fakultetskih kolegija ([4], [5]).

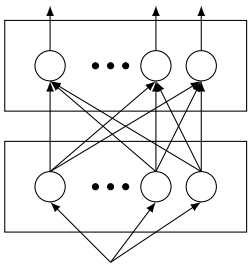
III. Opis implementirane LSTM mreže

Fokus projekta je treniranje i razvijanje neuronske mreže za generiranje realističnih imena naselja na različitim jezicima. Ukupno je dostupno generiranje imena na 5 jezika: engleski, francuski, hrvatski, španjolski, njemački. Moguće je generiranje toponima iz 7 različitih država (dodatno, SAD i Kanada). Korištenjem LSTM mreže, koja je prilagođena za analizu sekvencijskih podataka, cilj je razviti model sposoban za učenje jezičnih obrazaca i struktura iz postojećih imena naselja. Svrha mreže je generiranje novih imena temeljenih na tim naučenim obrascima, pri čemu se zadržavaju jezične i strukturne zakonitosti specifične za taj kontekst.

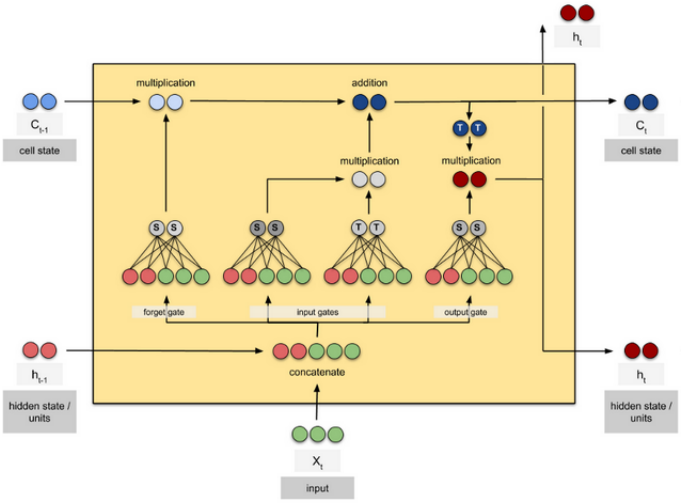
Nadalje uz ćeliju je dodatno razvijeno GUI sučelje koje ima mogućnost generiranja 25 različitih imena gradova na 6 dostupnih jezika.
Težine jednom naučene mreže se pohranjuju u trajnu memoriju, odnosno u datoteke u za to predviđenom direktoriju: `./saved_models`.

A. Arhitektura

Arhitektura „LSTM“ mreže se sastoji od više ćelija koji omogućavaju propagaciju podataka na način da uz svako propagiranje ažuriraju se dugoročna i kratkotrajna memorija. Ideja je da struktura omogućava ćelijama da „pamte“ informacije kroz duže sekvence. Na slikama je prikazana LSTM ćelija kako je izvedena u radnom okviru keras.
Obično se više jedinica u sloju povezuje kao na slici 2(a), kao što je to slučaj kod Dense slojeva u našoj mreži. Međutim, isto ne vrijedi za LSTM jedinice.



(a) više jedinica općenito

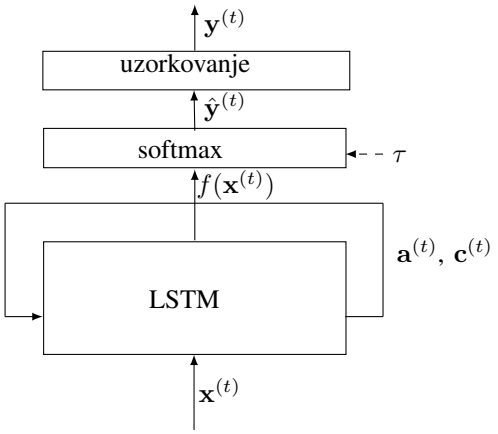


(b) više LSTM jedinica

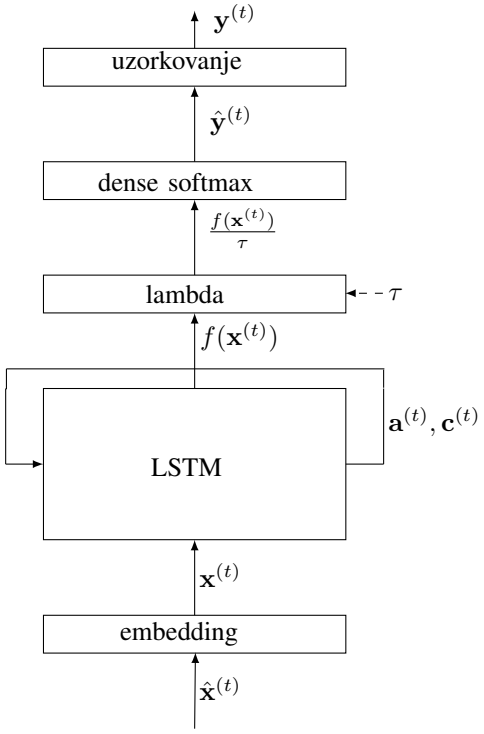
Slika 2.

Kao što vidimo na slici 2(b), parametar `units=` keras klase `LSTM` upravlja dimenzijama svih vektora osim ulaznog. Keras LSTM sloj se ne sastoji od više povezanih LSTM ćelija, već od jedne LSTM ćelije kojoj je povećana dimenzionalnost izlaznog prostora te vektora koji upravljaju stanjem ćelije.
U cjelini I je opisan konceptualni izgled mreže: njena najjednostavnija varijanta (slika 3(a)). Iz praktičnih razloga, implementacija izgleda nešto drugačije (slika 3(b)).

Također, modeli u Kerasu ne rade sa vektorima već sa tenzorima, te modelu ne dodajemo vektor koji predstavlja slovo, već 2D tenzor koji predstavljaju niz znakova. U teoriji modeli neuronskih mreža u Kerasu mogu raditi sa ulaznim vektorima varijabilne dimenzionalnosti. U praksi, rad sa vektorima fiksne duljine poboljšava performanse, osobito vrijeme treniranja. Razlog tomu je to što ulazni vektori fiksne težine omogućavaju stvaranje tenzora fiksnih oblika, a posljedično i stabilne težine.



(a) Konceptualna arhitektura LSTM ćelije



(b) stvarna arhitektura LSTM mreže

Slika 3.

B. Hipeparametri

1) Dimenzija ulaznog vektora: Dimenzija izlaznog vektora embedding sloja. Očekujemo $|\hat{\mathbf{x}}| = |\Sigma|$. Izlazni vektor nije u stvari one-hot, već kako se mreža trenira tako se tablica embedding layera prilagođava ta izazla vrijednost zadnjeg ascii znaka u tenzoru $\mathbf{x}^{(t)}$ odgovara one-hot vektoru tog slova.

2) Dimenzija skrivenog stanja: U trenutku t sadrži informacije iz koraka $\{t-1, ..., 0\}$, stoga očekujemo monotono padajuću funkciju gubitka u ovisnosti o $|\mathbf{a}|$.

3) Stopa učenja: Koristimo ADAM optimizator, koji se temelji na stohastičkom gradijentnom spustu, stoga stopa uravlja numeričkom stabilnosti optimizacije. Očekujemo konveksnu funckiju: prevelike stope učenja dovode do nestabilnosti, a premale se presporo miču prema optimumu. (vidi III-D)

$$\sigma_\tau(\mathbf{f_i}) = \frac{e^{\mathbf{f}_i/\tau}}{\sum_{j=0}^{|f|} e^{\mathbf{f}_j/\tau}} \tag{4}$$

Slika 4. Temperaturni softmax

4) Temperatura: temperaturom parametrizirani softmax čuva poredak vjerojatnosti klasa, ali smanjuje razliku između njih:

$$\mathbf{f}_i \geq \mathbf{f}_j \implies \sigma_\tau(\mathbf{f}_i) \geq \sigma_\tau(\mathbf{f}_j) \tag{5}$$

$$\mathbf{f}_i - \mathbf{f}_j \geq \sigma_\tau(\mathbf{f}_i) - \sigma_\tau(\mathbf{f}_j) \tag{6}$$

Što za izravnu posljedicu ima jednolikiji izbor konačnih klasa, odnosno "kretivniji" ispis, kada $\tau > 1$, odnosno "predvidljiviji" ispis kada $\tau < 1$.

Tablica II
HIPERPARAMETRI NAŠE MREŽE

Hiperparametar	***Vrijednost***	***komentar***
temperatura (τ)	0.00782	Daje mogućnost upravljanja "kreativnošću" generiranje.
dimenzija izlaznog prostora embedding sloja ($\|\mathbf{x}\|$)	32	
dimenzija skrivenog stanja ($\|\mathbf{a}\|$)	76	
stopa učenja (μ)	0.007572	
aktivacijska funckija	$\tanh$	zadano
povratna akt. funkcija	σ	zadano
bias	da	zadano
inicijalizator kernela	glorot jednoliki	zadano
inicijalizator povratne veze	glorot jednoliki	zadano
bias inicijalizator	zeros	zadano
forget bias	da	zadano
regularizacija kernela		zadano
regularizacija kernela povratne veze		zadano
bias regularizacija		zadano
kernel ograničenje		zadano
povratno ograničenje		zadano
bias ograničenje		zadano
dropout	0	zadano
povratni dropout	0	zadano

C. Ćelija

Ćelije su najbitniji dio naše mreže. Osim ulaza i izlaz ćelije imaju dugoročnu $\mathbf{c}$ i kratkoročnu memoriju $\mathbf{a}$. Ti su vektori svojstveni za ovakav tip mreže i razlikuju je od drugih RNN. Slika V prikazuje unutarnju shemu ćelije, $[\ ,\ ]$ označava operaciju konkatenacije.
Vidimo njene sastavne dijelove:

D. Treniranje

BPTT je korišten kao algoritam učenja. Kao funkcija gubitka koristi se kategorička unakrsna entropija.

$$L = -\sum_{t=0}^{|\Sigma|-1} \mathbf{z}_i^{(t)} \cdot log(\hat{\mathbf{y}}_i^{(t)}) \tag{7}$$

Tablica III
HIPERPARAMETRI NAŠE MREŽE

Komponenta	***komentar***
ulazni vektor $\mathbf{x}$	
izlazni vektor $\mathbf{y}$	
kratkoročna memorija $\mathbf{a}$	Kratkoročna memorija ili skriveno stanje, povezano težinskim vezama sa drugim komponentama, može se modificirati.
dugoročna memorija $\mathbf{c}$	Iako se dugoročna memorija može modificirati množenjem, a zatim kasnije zbrajanjem, ne postoje težine i bias koji mogu izravno modificirati memoriju. Nedostatak težina omogućuje dugoročnim sjećanjima da teku kroz niz odmotanih jedinica bez nestanka ili beskonačnog gradijenta.
Memorijski sklop	Specifičnost je da se kombiniraju ulaz i kratkoročna memorija pomnoženi sa prikladnim težinama te na kraju se dodaje bias. Ta funkcija prolazi kroz sigmoidnu aktivacijsku funkciju koja na kraju se množi sa dugoročnom memorijom. Ako je $\sigma([\mathbf{x}^{(t)}, \mathbf{a}^{(t)}] \cdot W_f + b_f) \approx 1$ pamtimo puno te $\mathbf{c}^{(t)}$ ostaje skoro nepromijenjen. U suprotnom ako je gornji izraz ≈ 0 dolazi do velike numeričke promjene (poništavanja).
Ulazni sklop	Vrijednost $\sigma([\mathbf{x}^{(t)}, \mathbf{a}^{(t)}] \cdot W_i + b_i)$ stvara potencijalno dugoročno sjećanje, a vrijednost $\tanh([\mathbf{x}^{(t)}, \mathbf{a}^{(t)}] \cdot W_m + b_m)$ određuje koji postotak tog sjećanja će se zapamtiti.
Izlazni sklop	Kombiniramo novostvorenu dugoročnu memoriju $\tanh(\mathbf{c}^{(t+1)})$ sa rezutatom sigmoide koji odlučuje u kojoj mjeri će se zapamtiti novostvoreno sjećanje. Na izlazu dobivamo novo kratoročno sjećanje, odnosno izlazni vektor $f(\mathbf{x}^{(t)})$ (slično kao ulazni sklop)

Gdje je $i \in [0, |\Sigma| - 1]$. $\hat{\mathbf{y}}^{(t)}$ je izlaz mreže, odnosno $\hat{\mathbf{y}}_i^{(t)}$ vjerojatnost da je idući znak i-ti znak abecede, a $\mathbf{z}^{(t)}$ je očekivani vektor. Inačica BPTT koju mi koristimo je u biti propagirani stohastički gradijentni spust.

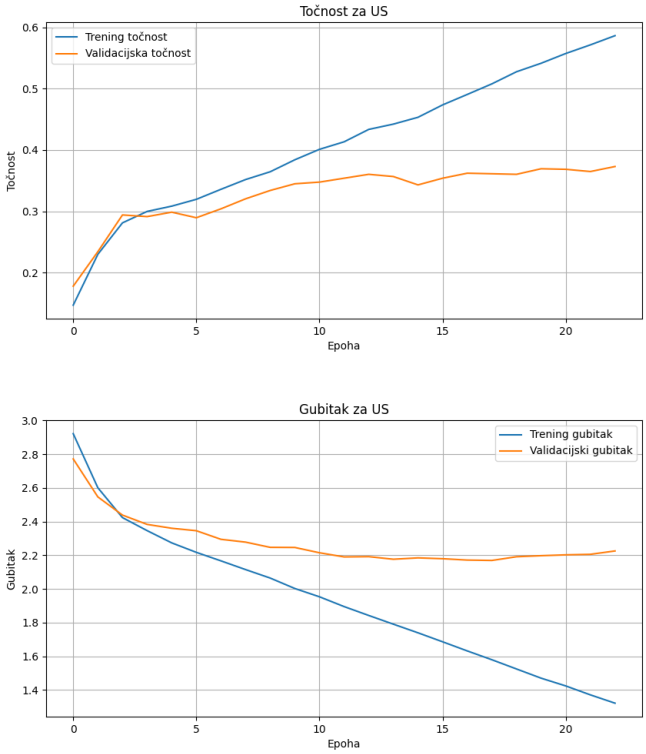
Pri treniranju koristimo parametre navedene u tablici IV.

Tablica IV
PARAMETRI PROCEDURE ZA TRENIRANJE

Parametar	***Vrijednost***
f-ja gubitka	kategorička unakrsna entropija
algoritam optimizacije	ADAM
metrika kvalitete	točnost

Pri treniranju koristimo podjelu skupova na skup za treniranje i testiranje (validaciju) u omjeru 4:1.
Uz hiperparametre $\begin{bmatrix} |\mathbf{a}| & \mu & |\hat{\mathbf{x}}| & \tau \end{bmatrix} = \begin{bmatrix} 32 & 76 & 0.007525 & 0.00728 \end{bmatrix}$ dobivamo slijedeće vrijednosti funkcije gubitka i točnosti po epohama.

Zbog ograničenja sa skupovima podataka te kako bi se omogućila usporedba veličina skupa za testiranje svih jezika je ograničena na $|\mathcal{D}_{test,L}| = \min_L |\mathcal{D}_{test,L}| = |\mathcal{D}_{test,GER}| \approx 600$.



Slika 5. Američki gradovi

Grafovi za ostale jezike izgledaju gotovo identično, greška na skupu za testiranje počinje rasti nakon 20-30 epohe. Najveća točnost je uvijek $40\% \pm 5\%$.

IV. Opis eksperimentalnih rezultata

A. optimiranje hiperparametara

Optimiranje je izvršeno strojno, pomoću razreda `RandomSearch`. Pokrenut je automatski postupak optimizacije nad 4 hiperparametra naše mreže, te su dobivene vrijednosti $\boldsymbol{\theta}_\mathbf{m} = \begin{bmatrix} |\mathbf{x}|_m & |\mathbf{a}|_m & \mu_m & \tau_m \end{bmatrix} = \begin{bmatrix} 32 & 76 & 0.007525 & 0.00728 \end{bmatrix}$. Vrijednosti optimuma $\hat{E}[L(\boldsymbol{\theta}_m)|\mathcal{D}_{test}] = 2.224$.

Vidimo da su naša predviđanja iz cjeline III-B bila većinom točna, sa iznimkom onog za skriveno stanje $|\mathbf{a}|$: nije riječ o monotono padajućoj funkciji pogreške u ovisnosti o tom hiperparametru, već o konveksnoj. Razlog tomu je moguće to što je riječ o kratkim nizovima znakova za koje postoji ograničena količina informacija koja se ima za zapamtiti između vremenskih koraka.

B. Usporedba rezultata

U tablici dajemo kratku usporedbu sa rezultatima u literaturi. Kao i u prethodnom cjelinama, korišten je skup podataka Američkih gradova (skup koji ima najmanju točnost na skupu za validaciju).

Tablica V

	[4]	[7]	Naša mreža
točnost	40%	$40\%^{a}$	

[a]na optimalnih $\tau = 0.5$.

Vrijedi napomenuti da, zbog svojstvenosti problema, točnost na ispitnom skupu predstavlja donju granicu stvarne točnosti modela. Naime moguće je da:

1) U skupu za ispitivanje postoje gradovi koji nisu u skupu za testiranje, a postoje u stvarnosti.
2) U skupu za ipistivanje postoje realistični, ali nepostojeći gradovi (cilj zadatka)

Budući da $|\mathcal{D}| \approx 600$, prva točka je vrlo izgledna. Stoga $E[L(\boldsymbol{\theta}_m)|\mathcal{D}_{test}]$ podcjenuje mogućnost generalizacije našeg modela.

V. Zaključak

Krajnji rezultat je mreža koja ima željenu točnost $\eta \approx 0.4$, svi projekti navedeni u literaturi dobivaju istu vrijednosti točnosti. Mišljenje je autora da je uzrok tako niskoj točnosti u samom problemu. Naime, iako su u toponimima uočljivi obrasci, također je prisutna velika raznolikost među imenima. Predviđanje realističnih imena je, stoga, izazovan zadatak.
Nije moguće proširiti skup podataka ta učenje, jer su svi postojeći gradovi već navedeni u skupu za učenje. Također je malo vjerojatno da će se na točnosti dobiti tako da se promijeni arhitektura mreže i/ili ćelije, jer je LSTM više nego dovoljno složena arhitektura za generiranje kratkih nizova znakova.
Kao moguće rješenje autori predlažu proširenje izvornog skupa podataka za učenje gradovima koji ne postoje. To se može ostvariti na bilo kojim, ili kombinacijom ova dva pristupa:

1) ručno: ljudi izmišljaju imena gradova te osiguravaju da već ne postoje.
2) strojno: LSTM mreža generira imena gradova, te pohranjuje u idući skup za učenje one nazive koji već ne postoje.

Ako pitamo ChatGpt 4 da generira 25 imena američkih gradova on to radi sa 80% točnosti međutim većina tih gradova je u skupu za učenje (CommonCrawl ili OpenWebText najvjerojatnije), a oni koji nisu su vrlo slični njima. Autori vjeruju da je upravo to razlog visokoj točnosti GPT te da bi takav pristup, implementiran na prethodno opisan način, znatno poboljšao točnost naše mreže.

Literatura

[1] Fatima, N., Imran, A S., Kastrati, Z., Daudpota, S M., Soomro, A (2022) "A Systematic Literature Review on Text Generation Using Deep Neural Network Models" IEEE Access, 10: 53490-53503, https://doi.org/10.1109/ACCESS.2022.3174108

[2] Eckhardt K. (2018, November 29). Choosing the right Hyperparameters for a simple LSTM using Keras. Towards data science. https://towardsdatascience.com/choosing-the-right-hyperparameters-for-a-simple-lstm-using-keras-f8e9ed76f046

[3] Karpathy A. (2015, May 21). The Unreasonable Effectiveness of Recurrent Neural Networks. Andrej Karpathy blog. http://karpathy.github.io/2015/05/21/rnn-effectiveness

[4] Randolph Z. (2020.) "Recursive Neural Network for Generating Novel Brand Names for Therapeutic Medicines". report. Department of Computer Science. Stanford. http://cs230.stanford.edu/projects_spring_2020/reports/38912979.pdf

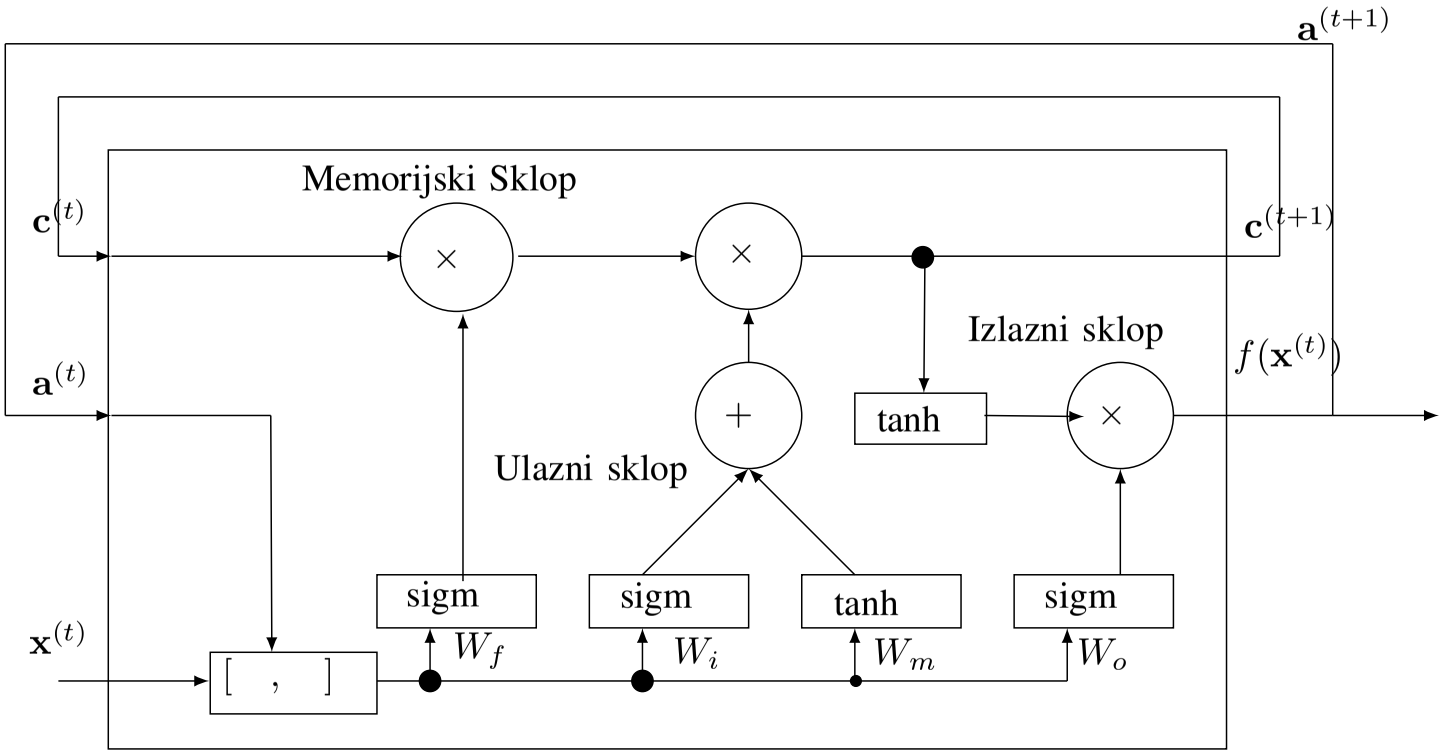
[5] Olah C. (2017, August 27). Understanding LSTM Networks. colah's blog. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[6] Dipanshu G. (2017, August 27). Master LSTM Networks With Python: A Guide Using TensorFlow and Keras. Medium. https://dipanshu10.medium.com/implementing-lstm-networks-with-python-a-guide-using-tensorflow-and-keras-915b58f502ce

[7] Rahalkar C. (2019, June 29). Name Generator Using Recurrent Neural Networks. Github. https://github.com/chaitanyarahalkar/Name-Generator-RNN

[8] Landy C. (2019, September 7). Look No More, The Data driven Baby Name generator. www.connorlandy.com. https://www.connorlandy.com/projects/rnn-name-generator

[9] Bosnali C. (2018, September 27). City-Name-Generation-using-LSTM-and-TF. Github. https://github.com/CihanBosnali/City-Name-Generation-using-LSTM-and-TF



Slika 6. Unutarnja arhitektura LSTM ćelije