Razvoj LSTM neuronske mreže i primena nad problemom sekvencijalnog učenja

Seminarski rad u okviru kursa
Računarska inteligencija
Matematički fakultet

Nevena Soldat, Milena Kurtić
nevenasoldat@gmail.com, mimikurtic67@gmail.com

11. april 2020.

Sažetak

U okviru ovog rada predstavljen je primer implementacije LSTM neuronske mreže u programskom jeziku Python, uz korišćenje biblioteka za rad kao što su pandas, numpy, keras i druge. Problem koji rešavamo jeste predviđanje broja obolelih osoba, kao i broja smrtnih slučajeva od virusa koji je zahvatio čitav svet. Videćemo koje su prednosti navedene vrste neuronskih mreža, kao i koja su njena ograničenja. Takođe ćemo se poslužiti jednom od dobro poznatih statističkih metoda koje rade sa ovom vrstom podataka i uporedićemo dobijene rezultate oba modela.

Sadržaj

1 Uvod

Rekurentne neuronske mreže (eng. Recurrent Neural Networks - RNN) predstavljaju arhitekturu mreža specijalizovanu za obradu sekvencijalnih podataka, poput rečenica prirodnog jezika i vremenskih serija. One su konstruisane sa idejom da se modeluje zavisnost među instancama. Eksperimenti su pokazali da ih je veoma teško trenirati efikasno. Naime, prilikom ažuriranja težina, može doći do toga da njihova promena bude toliko mala da nema efekta (dovodi do problema nestajućih gradijenata), odnosno toliko velika da su promene prevelike (dovodi do problema eksplodirajućih gradijenata). Ovaj problem se prevazilazi upotrebom duge kratkoročne memorije (eng. long short term memory - LSTM), što je složena jedinica mreže sa specifičnom strukturom koja omogućava kontrolu čitanja i upisa u jedinicu. Upravo ova vrsta jedinice dovela je do ključnih uspeha rekurentnih neuronskih mreža, i predstavlja standardni izbor prilikom formulisanja modela rekurentne mreže [1].

1.1 Primena na vremenske serije

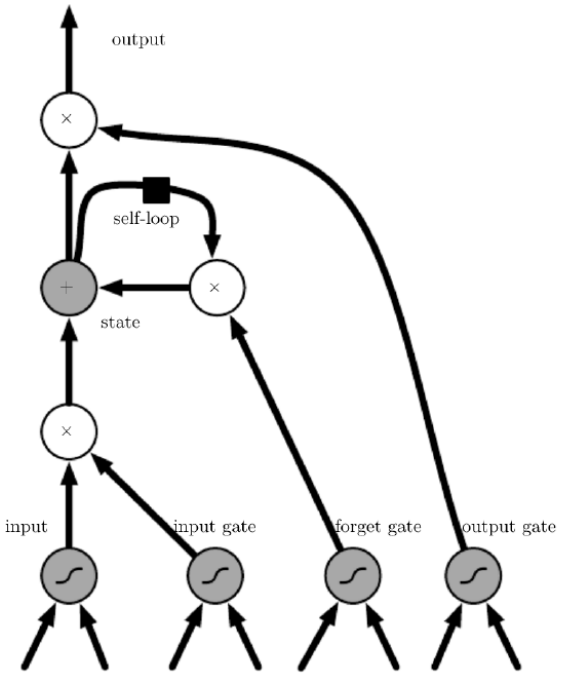
Predviđanje vremenskih serija se može opisati kao proces koji izvlači korisne informacije iz vrednosti koje su se realizovale u nekom prethodnom trenutku, i na osnovu njih predviđa buduće vrednosti. Nailazimo na veliku primenu ove tehnike u oblastima poput vremenske prognoze, planiranja transporta, odnosno regulisanja saobraćaja. Metode predviđanja koje su bazirane na neuronskim mrežama stiču veliku popularnost jer je dokazano da mogu biti podjednako dobre kao klasične statističke metode [2]. Analiza vremenskih serija i predviđanje su predmet intenzivnog proučavanja poslednjih 40 godina. Statistički model ARIMA (eng. Autoregressive Integrated Moving Average) je zbog svoje uspešnosti često korišćen za analizu procesa koji su vremenski zavisni. Međutim, glavno ograničenje ovog modela je to što ima tendenciju da teži ka srednjim vrednostima.

2 LSTM neuronske mreže

Osnovna ideja LSTM-a je postojanje takozvane ćelije koja čuva skriveno stanje, uz kontrolu pisanja, čitanja i zaboravljanja, koja se vrši na osnovu naučenih pravila. Na slici 1 prikazana je struktura LSTM jedinice, koja prikazuje ćeliju u centru i nekoliko kapija koje kontrolišu prethodno pomenute aspekte. Svaka od kapija ima strukturu kao jedinica standardne rekurentne mreže i na osnovu ulaza koje dobija i pridruženih im parametara odlučuje o tome da li i u kolikoj meri dozvoljava izvršavanje operacije koju kontroliše. Kapija koja kontroliše zaboravljanje odlučuje šta je bitno sačuvati iz prethodnog koraka. Ulazna kapija odlučuje koje informacije treba dodati iz trenutnog koraka. Izlazna kapija određuje sledeće skriveno stanje. Prednost LSTM jedinice je to što ćelija ne mora da prihvata ulazne signale, i stoga može dugo da čuva informaciju o dalekim delovima sekvence.

3 Opis problema

Cilj ovog rada je demonstriranje upotrebe LSTM neuronske mreže na problem predviđanja vremenskih serija. Kako je predviđanje toka pandemije virusa u trenutku pisanja ovog rada jedna od najektuelnijih tema,



Slika 1: Struktura LSTM jedinice

podaci koje koristimo predstavljaju broj obolelih, kao i broj žrtava zaraze ovim virusom. Zadatak je predvideti kako će se ovi brojevi menjati kroz dane koji slede. Kako bismo pokazali uspešnost ove metode, uporedićemo je sa ranije pomenutom metodom ARIMA.

3.1 Opis baze podataka

Za potrebe ovog projekta korišćena je baza podataka COVID19 Global Forecasting koja se u trenutku razvijanja ovog projekta svakodnevno ažurira (može se pronaći na sledećoj adresi: https://www.kaggle.com/c/covid19-global-forecasting-week-3). Baza sadrži podatke o broju osoba koje su potvrđeno zaražene virusom COVID-19, broj osoba koje su preminule, datume po danima počevši od 22.1.2020, kao i pokrajine i države na koje se date brojke odnose. Podaci su smešteni u fajl pod nazivom train.csv.

4 Rešenje problema

Problem predviđanja toka pandemije virusa rešavamo pomoću LSTM neuronske mreže. Program se sastoji iz sledećih delova:

- **Preprocesiranje**
- **Konfiguracija modela**
- **Testiranje modela**

4.1 Preprocesiranje

Kako bismo primenili LSTM neuronske mreže moramo da sredimo podatke koje imamo. To, u ovom slučaju, podrazumeva popunjavanje nedostajućih vrednost kao i normalizaciju podataka.

Nedostajuće vrednosti imamo u koloni Province_State. Te vrednosti popunjavamo sa vrednostima iz kolone Country_Region, što je prikazano na slici 2.

```
def fillState(state, country):
    if state == "empty":
        return country
    return state

train['Province_State'].fillna("empty", inplace = True)
train['Province_State'] = train.loc[:,['Province_State', 'Country_Region']]
                          .apply(lambda x: fillState(x['Province_State'], x['Country_Region']), axis = 1)
```

Slika 2: Popunjavanje nedostajućih vrednosti

Kada mreža koristi podatke koji su u velikom rasponu vrednosti, za velike ulaze može doći do velikog usporavanja učenja kao i konvergencije, stoga je potrebno skalirati date podatke. Postoje dva nacina za skaliranje vrednosti:

- **Normalizacija**: Ponovno skaliranje vrednosti podataka tako da su svi u opsegu od 0 do 1.
- **Standardizacija**: Reskaliranje vrednosti podataka tako da je srednja vrednost 0 a standardna devijacija 1.

Za potrebe ovog projekta korišćena je normalizacija (Standardizaciju ima smisla koristiti kada bi imali Gausovu raspodelu). Normalizaciju datog skupa podataka vršimo pomoću biblioteke scikit-learn i objekta MinMaxScaler.3

```
Confirmed = Confirmed.values
Fatalities = Fatalities.values

Confirmed = Confirmed.astype('float32')
Fatalities = Fatalities.astype('float32')

scaler_c = MinMaxScaler(feature_range = (0,1))
scaler_f = MinMaxScaler(feature_range = (0,1))

Confirmed = scaler_c.fit_transform(Confirmed)
Fatalities = scaler_f.fit_transform(Fatalities)
```

Slika 3: Normalizacija podataka

Kako bi podaci bili u odgovarajućem obliku, vršimo transformaciju na sledeći način. Odvajamo podatke za broj obolelih od podataka za broj smrtnih slučajeva. U oba slučaja kolona za datum postaje indeks, dok različite države smeštamo po kolonama. Tako imamo za svaku državu broj slučajeva uređene po danima. Početni datum je 22.01.2020, a broj država u ovom trenutku je 184. Model koji gradimo radi sa brojem instanci koji je jednak broju dana za koje imamo podatke, i paralelno vrši predviđanje za sve države.

4.2 Konfiguracija modela

Kako podaci koje imamo predstavljaju vremenske serije, sami moramo formirati ulaz, odnosno izlaz iz mreže. Ulaz je predstavljen podacima za tri dana uzastopno, dok je izlaz broj za četvrti dan. Funkcija split_sequences (sequences, n_steps_in, n_steps_out) obavlja taj zadatak. LSTM sloj zahteva da ulazni podaci budu u sledećem obliku - (n_samples, n_steps, n_features)[1], odnosno da imaju tri dimenzije. Prethodno opisana funkcija nam baš to omogućava.

Sledeći korak je podela podataka na trening i test skup. Kako funkcija train_test_split podelu vrši na nasumičan način, mi je ne možemo koristiti. Vremenske serije zahtevaju da ostane održano uređenje podataka. Podelu radimo ručno, 80% podataka je odvojeno za trening, a 20% za test.

4.2.1 Model za broj potvrđenih slučajeva

4.2.2 Model za broj smrtnih slučajeva

4.3 Testiranje modela

5 Zaključak

Literatura

[1] Mladen Nikolić i Anđelka Zečević. *Mašinsko učenje*. 2019.

[2] Sima Siami-Namini and Akbar Siami Namin. Forecasting economics and financial time series: Arima vs. lstm. *arXiv preprint arXiv:1803.06386*, 2018.

A Dodatak

[1] n_samples predstavlja broj uzoraka, n_steps broj koraka, a n_features broj izlaza