Razvoj rekurentne neuronske mreže i primena na analizi vremenskih serija

Seminarski rad u okviru kursa
Računarska inteligencija
Matematički fakultet

Kristina Pantelić, 91/2016, kristinapantelic@gmail.com
Nevena Mesar, 107/2015, @.com

18. maj 2020.

Sažetak

Danas je upotreba neuronskih mreža za rešavanje diverziteta računarskih problema u širokoj upotrebi. Za različite probleme koriste se različite vrste neuronskih mreža. U ovom radu čitalac će se upoznati sa terminom neurona, neuronske mreže, rekurentne neuronske mreže i primenom rekurentne neuronske mreže na analizi vremenskih serija.

Sadržaj

1 Uvod

Algoritmi neuronskih mreža su nastali kao pokušaj da se oponaša način na koji ljudski mozak uči tj. način na koji ljudski mozak klasifikuje i spoznaje stvari. Neuronske mreže su razvijene tako da simuliraju neurone tj. mreže neurona mozga[4]. U tradicionalnoj neuronskoj mreži svi ulazi i izlazi su nezavisni jedni od drugih, ali u slučajevima kada se, na primer, zahteva predviđanje sledeće reči u rečenici, zahteva se da imamo informaciju o prethodnoj reči i da je na neki način zapamtimo[2]. Ovaj problem je bio motivacija za pojavljivanje i razvijanje rekurentne neuronske mreže. Rekurentna neuronska mreža (RNN) ima "memoriju" koja pamti sve informacije koje je prethodno izračunala tj. naučila. Treba imati na umu da tradicionalna neuronska mreža takođe pamti informacije, ali samo one koje je naučila tokom treninga. Na primer, ako klasifikator nauči zavisnosti ulaznih od izlaznih podataka tokom treninga, koristiće to znanje da klasifikuje test instance. Dok RNN takođe uči prilikom treninga, dodatno, ona pamti informacije naučene iz prethodnih trening instanci da bi generisala izlazne podatke[3]. Ovo svojstvo pamćenja informacija kroz vreme omogućava RNN mreži predikciju u oblasti vremenskih serija[2], kojima u ovom radu posvećujemo pažnju u kontekstu primene RNN.

Na samom početku ovog rada upoznaćemo se sa modelom naše rekurentne neuronske mreže, a u daljem tekstu ćemo se upoznati sa korišćenim skupom podataka vremenskih serija i primenom RNN na skup.

2 Rekurenta neuronska mreža

Postoji više vrsta neuronskih mreža koja, svaka zbog svojih specifičnih karakteristika, daju dobre rezultate na specifičnim poljima istraživanja podataka. Vrsta mreže koja je tema našeg rada je rekurentna neuronska mreža, a naša koncentracija je na razvijanju Jordanove strukturne rekurentne neuronske mreže (eng. *Jordan SRNN*).

Jordanova rekurentna neuronska mreža je zasnovana na konceptu koji podrazumeva ponovno korišćenje dobijenih izlaznih vrednosti mreže kao dodatnih informacija za usmeravanje pretrage i izračunavanje novih vrednosti rezultata. Pored ulaznih instanci, dodaju se novih k vrednosti koje predstavljaju vrednosti k rezultata. Dakle, kopija izlaznog sloja se sprovodi na ulaz.

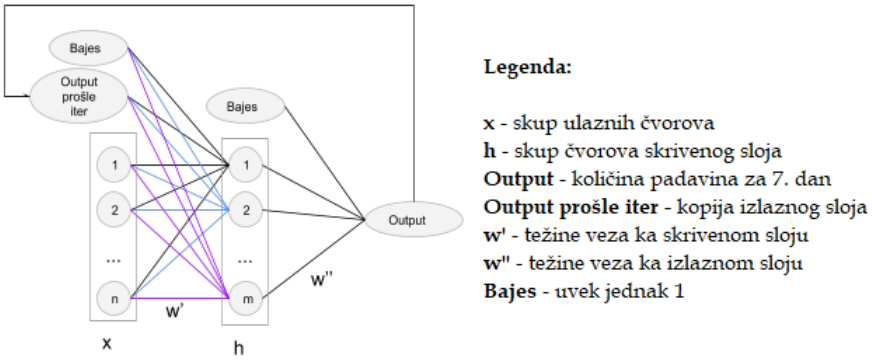
Naš cilj je konstruisati rekurentu neuronsku mrežu sa jednim skrivenim slojem koja će za 6 unetih uzastopnih dana prognozirati količinu padavina za 7. dan. U nastavu ćemo predstaviti model ove mreže u skladu sa brojem naših izlaznih vrednosti (jedan izlazni podatak) i odgovarajuće oznake korišćene u kôdu za implementaciju mreže.

2.1 Model rekurentne neuronske mreže

Neuronska mreža se sastoji od 3 sloja koju čine ulazni, skriveni i izlazni sloj. Svaki od slojeva ima odgovarajući broj neurona tj. promenljivih. Ulazni sloj se sastoji od $n+p$ ulaznih promenljivih $x_1^{(l)}$, $x_2^{(l)}$, ..., $x_n^{(l)}$,$o_1^{(l-1)}$, $o_2^{(l-1)}$, ..., $o_p^{(l-1)}$, uz dodatnu promeljivu $x_0^{(l)}$, koja se naziva bajes i čija je vrednost uvek jednaka 1, za svako l. Promenljiva l u superskriptu promenljive x_i predstavlja redni broj instance u skupu odabranih instanci za treniranje mreže. Radi jednostavnosti sve ulazne promenljive označavamo sa $x_i \in [0, n+p]$. Skriveni sloj čine m neurona $h_1^{(l)}$, $h_2^{(l)}$, ..., $h_m^{(l)}$ uz do-

datnu promenljivu $h_0^{(l)} = 1$, za svako l, koja predstavlja bajes za skriveni sloj mreže. Izlazni sloj čine p neurona, čiji su izlazi $o_1^{(l)}$, $o_2^{(l)}$, ..., $o_p^{(l)}$. Svi neuroni ulaznog sloja su spojeni sa svim neuronima skrivenog sloja, osim bajesa, vezama čije su težine w'_{ij}, $0 \leq i \leq n+p$, $0 \leq j \leq m$. Svi neuroni skrivenog sloja su povezani sa svim neuronima izlaznog sloja vezama sa težinama w''_{jk}, $0 \leq j \leq m$, $1 \leq k \leq p$.

Nakon završavanja algoritma, cilj je da izlazne vrednosti budu što bliže izlaznim vrednostima y_1, y_2,...,y_p.



Slika 1: Jordanova RNN sa jednim skrivenim slojem

Proces učenja se sastoji od više iteracija, gde se svaka iteracija izvršava u nekoliko koraka. Najpre se, za sve j, $1 \leq j \leq m$, izračunava vrednost:

$$u'_j = \sum_{i=0}^{n+p} x_i w'_{ij} \tag{1}$$

i $h_j = f(u'_j)$, gde je $f(x) = (1 + e^{-x})^{-1}$ sigmoidna funkcija. Sigmoidna funkcija predstavlja aktivacionu funkciju neurona skrivenog i izlaznog sloja. Analogno kao kod skrivenog sloja, određuju se vrednosti izlaza iz neurona izlaznog sloja. Tako se dobija

$$u''_k = \sum_{ij=0}^{m} h_j w''_{jk} \tag{2}$$

i $o_k = f(u''_k)$. Nakon odgovarajućeg broja iteracija, očekujemo da će vrednosti o_k biti približno jednake željenim izlaznim vrednostima y_k. Greška izlaznog sloja neurona k je definisana:

$$E_k = \frac{1}{2}(y_k - o_k)^2. \tag{3}$$

Pri ažuriranju vrednosti w''_{jk}, važi $w''_{jk} = w''_{jk} + w''_j k$, gde je

$$\Delta w''_{jk} = -\eta \frac{\partial E_k}{\partial w''_{jk}} + \alpha w''_{jk}.$$

Parametri η i α, respektivno, mere uticaj parcijalnog izvoda greške E_k po w''_{jk} i prethodne vrednosti w''_{jk}. Kako je E_k funkcija od o_k, o_k funkcija od u''_k, a u''_k funckija čiji je jedan od argumenata w''_{jk}, prema pravilu o izvodu složene funkcije važi

$$\frac{\partial E_k}{\partial w''_{jk}} = \frac{\partial E_k}{\partial o_k}\frac{\partial o_k}{\partial u''_k}\frac{\partial u''_k}{\partial w''_{jk}} = -(y_k - o_k)f'(u''_k)h_j = -(y_k - o_k)o_k(1-o_k)h_j$$

Iz poslednjeg reda sledi da je

$$\Delta w''_{jk} = \eta(y_k - o_k)o_k(1-o_k)h_j + \alpha w''_{jk}$$

Greška neurona skrivenog sloja predstavlja sumu svih odgovarajućih grešaka neurona izlaznog sloja i određena je sa

$$E_j = \frac{1}{2}\sum_{k=1}^{p}(y_k - o_k)^2$$

.

Pri ažuriranju vrednosti w'_{ij}, važi $w'_{ij} = w'_{ij} + \Delta w'_{ij}$, gde je

$$\Delta w'_{ij} = -\eta\frac{\partial E_j}{\partial w'_{ij}} + \alpha w'_{ij}$$

.

Vrednost $\frac{\partial E_j}{\partial w'_{ij}}$ se može izračunati prema pravilu o izvodu složene funkcije na sledeći način:

$$\frac{\partial E_j}{\partial w'_{ij}} = \frac{\partial E_j}{\partial o_k}\frac{\partial o_k}{\partial u''_k}\frac{\partial u''_k}{\partial h_j}\frac{\partial h_j}{\partial u'_j}\frac{\partial u'_j}{\partial w'_{ij}}$$

$$= -\sum_{k=1}^{p}(y_k - o_k)f'(u''_k)w''_{jk}f'(u'_j)x_i$$

$$= -\sum_{k=1}^{p}(y_k - o_k)o_k(1-o_k)w''_{jk}h_j(1-h_j)x_i$$

Sledi da je

$$\Delta w'_{ij} = \eta x_i h_j(1-h_j)\sum_{k=1}^{p}(y_k - o_k)o_k(1-o_k)w''_{jk} + \alpha\Delta w'_{ij}[1]$$

2.2 Algoritam rekurentne neuronske mreže

Na osnovu prethodnog opisa, algoritam učenja se izvršava u sledećih osam koraka[1]:

1. Ulazni podaci se sastoje od parova vektora ($x_1^{(l)}$, $x_2^{(l)}$,..., $x_{n+p}^{(l)}$) i (y_1, y_2,..., y_p), gde x_i predstavljaju ulaze uključujući kopiju rezultata iz prethodne iteracije algoritma, a y_k željene izlaze za svaku ulaznu instancu. Skup ulaznih podataka se proširuje bajesom $x_0^{(l)} = 1, \forall l$ iz skupa podataka.
2. Inicijalizovati parametre η, α i odrediti kriterijum zaustavljanja. Inicijalizovati početne vrednosti za w'_{ij} i w''_{jk} i postaviti $\Delta w'_{ij} = \Delta w''_{jk} = 0$. Izabrati aktivacionu funkciju f (ovde je korišćena sigmoidna funkcija).
3. Izabrati novi par ulaznog i izlaznog vektora (novi test primer iz skupa za učenje).
4. Odrediti vrednosti u'_j i h_j. Postaviti $h_0^{(l)} = 1, \forall l$ iz skupa podataka.
5. Odrediti vrednosti u''_k i o_k. Ukoliko je ispunjen kriterijum zaustavljanja, prekinuti izvršavanje.

6. Odrediti $\Delta w''_{jk}$ i ažurirati vrednosti w''_{jk}.
7. Odrediti $\Delta w'_{ij}$ i ažurirati vrednosti w'_{ij}.
8. Preći na korak 3.

Literatura

[1] Model rekurentne neuronske mreže. on-line at: http://poincare.matf.bg.ac.rs/~stefan/ri/index.htm.

[2] Uvod u RNN. on-line at: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/.

[3] Uvod u RNN i poređenje sa tradicionalnom NN. on-line at: https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce.

[4] Sara Gavran. Veštačke neuronske mreže u istraživanju podataka: pregled i primena. Master's thesis, Univerzitet u Beogradu, Matematički fakultet, 2016.

A Dodatak