



УНИВЕРЗИТЕТ У НОВОМ САДУ
ПРИРОДНО-МАТЕМАТИЧКИ ФАКУЛТЕТ
ДЕПАРТМАН ЗА МАТЕМАТИКУ И
ИНФОРМАТИКУ



Key Features Net i njene primene

Seminarski rad

Miloš Vujasinović

Novi Sad, jul 2020

Sadržaj

Sadržaj	1
Uvod	2
1 Poglavlje 1	3
1.1 Suština podataka i problem sa tradicionalnim autoenkoderom	3
1.2 Podpoglavlje 2	4
Zaključak	6
Literatura	7

Uvod

Autoenkodori su već godinama zlatni standard u smanjenju dimenzionalnosti podataka. Način na koji rade se pokazao kao veoma efikasan u otklanjanju šuma i dopunjavanju podataka koji su oštećeni. Motivisan datim primerima, ovaj rad pokušava da prikaže novi način gledanja na smanjene dimenzionalnosti podataka mašinskim učenjem. Zatim, primenom iznesenih ideja i nekih od principa koji se nalaze u osnovi autoenkodera se uvodi model neuronske mreže koji za cilj da iz podataka koji se prosleđuju modelu izdvoji najbitnije odlike za klasifikaciju. Kroz ovaj postupak se takođe razmatraju novi načini treniranja i evaluacije modela, a na kraju se rezultati datog modela porede sa rezultatima tradicionalnih autoenkodera.

1 Poglavlje 1

1.1 Suština podataka i problem sa tradicionalnim autoenkoderom

Autoenkoderi su neuronske mreže koje se treniraju da oponašaju identičko preslikavanje, to jest da ulaz preslikava na samog sebe. Ovo samo po sebi ne izgleda veoma korisno, međutim autoenkoderi, pored navedenih, sadrže i osobinu da je broj neurona u najmanjem skrivenom sloju mreže manji od broja neurona ulaznog i izlaznog sloja. Ovo za implikaciju ima da se prilikom prolaska podataka kroz mrežu u datom sloju nalazi reprezentacija podataka za čije je predstavljanje potreban manji broj memorijskih jedinica od originalnih podataka i ovo može da varira od nekoliko jedinica, pa do redova veličina manje. Dati skriveni sloj se naziva *usko grlo* (engl. *bottle neck*) i on deli neuronsku mrežu na dva dela: od ulaza do sebe i od sebe do izlaza. Ovi delovi se redom nazivaju *enkoder* i *dekoder*, a oni se koriste za preslikavanje originalnih podataka u reprezentaciju smanjenih dimenzija i nazad. U nastavku rada će se autoenkoderi definisati na formalniji način, ali za trenutne potrebe dato objašnjenje je dovoljno.

Prilikom treniranja autoenkodera se greška računa kao greška između svakog neurona ulaza i njemu odgovarajućeg neurona izlaza. Ovo je veoma intuitivno rešenje ako želimo da rezultati na izlazu budu naizgled što sličniji ulaznim podacima. Međutim dato preslikavanje je u praksi gotovo nemoguće napraviti tako da bude bde savršeno. Kao posledica dolazi do izvesne greške između originalnih podataka i rezultata mreže koji možda neizgledaju značajno ili su nam gotovo neprimetni, ali, teoretski, mogu značajno da utiču na rezultate klasifikacije ako bi se rezultati autoenkodera pustili kroz klasifikator i uporedili sa rezultatima klasifikacije originalnih podataka. Na primer, ako imamo klasifikator koji traga za odlikom koja je sadržana u veoma malom broju memorijskih jedinica u poređenju sa veličinom jedne instance podataka, a autoenkoder zaključi da bi uvrštavanje date odlike samo povećalo grešku, jer bi njenim uvrštavanjem bilo onemogućeno uvrštavanje neke druge odlike koja je iz pogleda greške autoenkodera važnija. Kao posledica, redukcija podataka ovim postupkom je ne potencijalno samo beskorisna, nego i čini podatke beskorisnim ako želimo da ih klasifikujemo ili, još gore, koristimo za treniranje klasifikatora.

Diskutovani problem svakako postavlja pitanje kako se može prevazići, ali pre davanja odgovora na njega, potrebno je da definišemo podelu koja će da omogući uočavanje ovog problema jasnijim i lakšim. Iz prikazanog rezonovanja se jasno može uvideti da se autoenkoderi mogu trenirati ili u svrhu očuvanja izgleda podataka ili u svrhu očuvanja onoga što podaci ustvari jesu, odnosno njihove *suštine*. Treba imati u vidu da podaci imaju gotovo bezbroj različitih suština u zavisnosti od toga iz kog se ugla posmatraju, to jest da se u zavisnosti od pitanja koje se postavlja o njima njihova suština menja i odatle možemo da spojimo ideju o suštini sa problemom klasifikacije tako što ćemo reći da pitanje koje postavlja klasifikacioni problem određuje suštinu podataka o kojoj u datom trenutku pričamo. Ove ideje se koriste za podelu definisanu u definiciji 1. Iz ove definicije je jasno da su tradicionalni autoenkoderi ustvari izgledni autoenkoderi, dok autoenkoderi kojima se teži u svrhu prevazilaženja diskutovanog problema su suštinski autoenkoderi, mada se i za njih, u zavisnosti od implementacije, postavlja pitanje koliko dobro mogu da očuvaju odlike bitne za klasifikaciju.

Definicija 1 (Podela autoenkodera po načinu treniranja) *Autoenkoder za čije se treniranje koristi znanje o samim podacima se naziva izgledni autoenkoder, dok autoenkoder za čije se treniranje koristi znanje o klasama kojima podaci pripadaju se naziva suštinski autoenkoder. Treba imati u vidu da date osobine nisu disjunktne, pa postoji i izgledno-suštinski autoenkoder.*

1.2 Treniranje suštinskih autoenkodera

Da bismo razmatrali kako se klase kojima podaci pripadaju mogu uključiti u treniranje autoenkodera definišimo prvo autoenkodere na formalniji način. Za ovo ćemo iskoristiti definiciju 2 preuzetu iz [1, Poglavlje 2].

Definicija 2 (Opšti autoenkoder *framework* [1]) $n/p/n$ autoenkoder je definisan kao t -toraka $n, p, m, \mathbb{F}, \mathbb{G}, \mathcal{A}, \mathcal{B}, \mathcal{X}, \Delta$ gde važi:

1. \mathbb{F} i \mathbb{G} su skupovi.
2. n i p su pozitivni celi brojevi. (Autor razmatra slučaj kada je $0 < p < n$ što se podrazumeva prilikom pravljenja autoenkodera)
3. $\mathcal{A} : \mathbb{G}^p \rightarrow \mathbb{F}^n$
4. $\mathcal{B} : \mathbb{F}^n \rightarrow \mathbb{G}^p$
5. $\mathcal{X} = \{x_1, \dots, x_m\}$ je skup od m (trening) vektora u \mathbb{F}^n . Ako se radi o spoljašnjem skupu ciljeva u \mathbb{F}^n označavamo ga sa $\mathcal{Y} = \{y_1, \dots, y_n\}$.
6. Δ je funkcija različitosti ili distorcije definisana nad \mathbb{F}^n

Da bismo bolje razumeli datu definiciju i bliže se upoznali sa terminologijom koja će se koristiti u nastavku prođimo kroz sve elemente date t -torke koja se koristi za predstavljanje autoenkodera. Kao što smo ranije pričali ideja iza autoenkodera je da nauči da slika ulaz na samog sebe, a da pri tome postoji skriveni sloj koji je manjih dimenzija od samog ulaza i na ovaj način se postiže smanjenje dimenzionalnosti podataka. n je ništa manje nego broj dimenzija ulaza kao i izlaza, dok je p broj dimenzija *uskog grla*¹, pošto se cilja da je broj dimenzija uskog grla manji i broja dimenzija ulaza zato se i pretpostavlja da je $0 < p < n$. Svaka memorijska jedinica u ulaznom i izlaznom sloju, odnosno uskog grlu sadrži podatke koji redom pripadaju skupovima \mathbb{F} i \mathbb{G} . \mathcal{B} je funkcija koja preslikava ulazne podatke u reprezentaciju smanjenih dimenzija i ona se naziva *enkoder*. \mathcal{A} radi obrnut proces od \mathcal{B} i preslikava reprezentaciju smanjenih dimenzija na izlaz i ova funkcija se naziva *dekoder*. Treba primetiti da se ovde delovi neuronske mreže posmatraju kao funkcije, jer oni to i jesu, i ova notacija će se u nastavku koristiti kada se govori o neuronskim mrežama i njenim delovima. Za treniranje autoenkodera su potrebni podaci i skup trening podataka je u datoj definiciji označen sa \mathcal{X} . Međutim, podaci se ne koriste samo za treniranje i možemo imati podatke koji se koriste za evaluaciju ili podatke koji se puštaju kroz autoenkoder u cilju smanjenja dimenzionalnosti. Ovakvi skupovi podataka su u definiciji označeni sa \mathcal{Y} . Na kraju, ostala je još da se objasni funkcija različitosti ili distorcije Δ i ona je ništa manje nego funkcija koja govori koliko se izlaz autoenkodera razlikuje od željenog izlaza. Ako ovo zvuči poznato, to je ništa manje nego što kod neuronskih mreža još znamo kao i funkcija greške ili *loss* funkcija.

Funkcija greške je jedna od glavnih karakteristika treninga neuronskih mreža i ona govori u kom smeru će se treniranje kretati. Kao takva ona je očigledna opcija za uključanje klasa kojima podaci pripadaju u trening i na taj način dobijanje suštinskog autoenkodera. Međutim, ovo nije tako lako izvesti kao što zvuči. Dok se kod tradicionalnih, izglednih, autoenkodera srednja kvadratna greška nudi kao idealna funkcija greške, kod suštinskih autoenkodera stvari su malo komplikovanije i zahtevaju se određeni trikovi u dobijanju željenog efekta. Glavni problem je što kod treniranja izglednih autoenkodera je vrednost koju mreža treba da proizvede poznata, dok kod suštinskih autoenkodera je poznata samo vrednost koja treba da se dobije kada se traženi

¹sloj u kome se prilikom prolaza podataka kroz autoenkoder nalazi reprezentacija podataka smanjenih dimenzija

rezultati puste kroz određenu funkciju. Na osnovu ovoga definicija 3 uvodi dva načina treniranja na koja se model mašinskog učenja može učiti.

Definicija 3 (Podela načina treniranja modela mašinskog učenja) *Ako je prilikom treniranja modela mašinskog učenja poznata vrednost koja se treba dobiti kao proizvod rada modela onda se dati proces treniranja naziva direktno treniranje. Nasuprot tome, treniranje u kome je poznata samo vrednost koja se treba dobiti nakon što se rezultati rada modela puste kroz određenu funkciju se naziva indirektno treniranje.*

Zaključak

Literatura

- [1] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In Isabelle Guyon, Gideon Dror, Vincent Lemaire, Graham Taylor, and Daniel Silver, editors, *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, volume 27 of *Proceedings of Machine Learning Research*, pages 37–49, Bellevue, Washington, USA, 02 Jul 2012. PMLR.