

/

--6 Duboko učenje

Priprema za ZI

sugondeze

Sadržaj

Teorija	4
Stari ispiti	21
MI 2016/17	21
ZI 2016/17	28
Rok 2016/17	38
h2_prob = sigmoid(torch.mm(v2, self.W) + self.h_bias)	
h2 = h2_prob.bernoulli() # ovo zapravo vraca gornji [1, 0, 0] dok mi h2_prob tražimo	45
MI 2017/18	46
ZI 2017/18	54
Optimizacijski algoritmi	55
SGD	55
SGD s momentom	55
AdaGrad	56
RMSProp	56
RMSProp + Nester moment	57
Adam	57

I

1.

2. Teorija

<p>1. Prednost algoritma AdaGrad nad osnovnom implementacijom SGD-a je:</p> <ul style="list-style-type: none"> (a) dodavanje momenta u procjenu trenutnog gradijenta (c) skaliranje komponenata gradijenata njihovom akumuliranim normom (c) skaliranje gradijenata eksponentijalnim pomicnim prosjekom norme (d) ugradnja momenta u praćenje gradijenta te skalaranja vremenski lokalnom procjenom prosječne norme <p>2. Povećanjem veličine mini-grupe dobivamo:</p> <ul style="list-style-type: none"> (a) veću brzinu evaluacije modela na ispitnim podatcima (b) manji broj manje preciznih ažuriranja parametara modela (c) veći broj preciznijih ažuriranja parametara modela (d) manji broj preciznijih ažuriranja parametara modela <p>3. Zašta potpuno povezani modeli s prosječnom unakrsnom entropijom imaju više jednakobnih lokalnih minimuma?</p> <ul style="list-style-type: none"> (a) zbog simetrije neurona istog skrivenog sloja (b) zbog simetrije neurona izlaznog sloja (c) zbog doprinosa regularizacije (d) zbog simetrije neurona susjednih skrivenih slojeva 	<p>4. Kako biste opisali odnos između gubitka "0-1" i unakrsne entropije?</p> <ul style="list-style-type: none"> (a) gubitak "0-1" je metoda izbora kad imamo malo podataka (b) gubitak "0-1" je nadomjestak unakrsne entropije (c) unakrsna entropija je derivabilni nadomjestak gubitka "0-1" (d) unakrsna entropija je metoda izbora kad imamo jako puno podataka <p>5. Kod uporabe lokalne normalizacije odziva (LRN), normaliziraju se:</p> <ul style="list-style-type: none"> (a) izlazi neurona s obzirom na odzive kada mreža radi ispravno odnosno neispravno (b) izlazi neurona s obzirom na odzive koje generira za različite uzorke za učenje (c) izlazi neurona s obzirom na izlaze njemu susjednih neurona u istom sloju (d) izlazi neurona s obzirom na izlaze njemu susjednih neurona u slojevima prije i nakon njega <p>6. Predtreniranje je postupak:</p> <ul style="list-style-type: none"> (a) koji se provodi kada je zadatak učenja neuronske mreže jednostavan (b) bez kojeg paralelizacija pri obradi podataka tijekom učenja mreže ne bi bila moguća (c) koji osigurava početne vrijednosti parametara mreže koje su bolje od slučajnih (d) koji osigurava da su težine smještene u raspon između -6 i +6
--	---

1. Ažurira se *ni* u svakom koraku prema jednadžbi:

$$ni_t = ni / \sqrt{\alpha_t + \epsilon}, \quad \alpha_t = \sum_{i=1}^t (dh/dW_i)^2$$

b)

2. Manji broj jer se više podataka obrađuje odjednom i parametri se ažuriraju za sve te podatke pa su zbog potencijalnog šuma u primjerima precizniji.

d)

3. **a)** Najbolje je objasniti na primjeru prepoznavanja slike. Recimo da se jedan neuron (nakon treniranja odnosno dolaska u minimum) specijalizirao za prepoznavanje vodoravnih rubova, a neki drugi za horizontalnih. Tak je svejedno koji neuron prepoznaće kakve rubove, odnosno u nekom drugom minimumu situacija bi bila obrnuta ali to nam nije bolje (ni losije,

/

vec isto) od trenutnog minimuma. Zbog te simetrije unutar sloja, za sto se koji neuron specijalizira, postoji vise jednakovrijednih lokalnih minimuma.

4. c)

5. Izlaz neurona se normalizira dijeljenjem sa sumom kvadrata izlaza n susjednih mapa značajki za iste pozicije u mapi (3. preza 69-73)

c)

6. c)

1. Do problema eksplodirajućeg gradijenta može doći ako se podatci uzastopno množe matricom težina za koju vrijedi: (a) da ima neke svojstvene vrijednosti koje su po iznosu oko 0 (b) da ima neke svojstvene vrijednosti koje su po iznosu dosta veće od 1 (c) da ima neke svojstvene vrijednosti koje su po iznosu dosta manje od 1 (d) da je singularna	2. Zašto pri optimizaciji radije promatramo log-izglednost nego izglednost? (a) log izglednost podatka se brže evaluira od njegove izglednosti (b) zbog boljih stat gradijenata nezavisnih podataka C) zbog aditivnosti gradijenata nezavisnih podataka (d) zbog aditivnosti gradijenata međusobno ovisnih podataka	3. Uzorkovanje manjeg broja uzoraka (mini-grupe) umjesto uporabe čitavog skupa uzoraka za učenje pri izračunu gradijenta opravданo je jer: (a) preciznost određivanja gradijenta s povećanjem broja uzoraka raste ispodlinearno (b) preciznost određivanja gradijenta ne ovisi o broju uzoraka mini-grupe (c) preciznost određivanja gradijenta nema nikakvog utjecaja na rad algoritma strojnog učenja fd) preciznost određivanja gradijenta raste kvadratno s brojem uzoraka	4. U kojoj od sljedećih situacija nam ne može pomoći učenje s momentom? (a) kod plitkih lokalnih minimuma (b) kod "kanjona" C) kod sedala (d) kod "ploha"
5. Zašto potpuno povezani modeli s prosječnom unakrsnom entropijom imaju više jednako dobrih lokalnih minimuma? (a) zbog simetrije neurona istog skrivenog sloja (b) zbog konveksnosti unakrsne entropije (c) zbog doprinosa regularizacije (d) zbog simetrije neurona susjednih skrivenih slojeva	6. Kod normalizacije nad grupom (engl. batch-norm), normalizirani se podatci na kraju provode kroz afinu transformaciju: (a) kako bi se povećala kompleksnost mreže u odnosu na onu bez normalizacije podataka C) kako bi se osiguralo da su podatci centrirani oko nule (c) kako bi se osiguralo da mreža ne izgubi ekspre- sivnost (d) kako bi se smanjilo standardno odstupanje podataka		

1. b)

2. <https://stats.stackexchange.com/a/289193>

c)

3. Broj podataka u mini-grupi poboljšava točnost gradijenta, no taj porast se smanjuje s brojem podataka.

a)

4. Što se smatra "kanjom" ovdje? Ja bi rekao da kanjon mozes zamisliti kao fukciju u obliku slova V. Rekao bih da je ovdje tocan odgovor d) jer kada imas ravnu plohu nijedna metoda

/

temeljena na gradijentu ti ne moze pomoci (gradijent ce uvijek biti 0) Nigdje nije receno da je ta ploha ravna i mislim da za blago nakrivljene plohe itekako pomaze. Ne pomaze itekako - moving average prati iznos gradijenta i biti ce mali. Akceleracija je ovdje nuzna. Moment je tu samo da preskoci neku grbu nakon lokalnog minimuma.

Uporaba momenta koristi se kao metoda za ubrzavanje učenja:

- kod malih ali konzistentnih gradijenata
- kod šumovite procjene gradijenata

Mislim da je ipak odgovor sedlo jer ce kod ravne plohe (nagnute) sigurno ubrzati ucenje, a sedlo ne predstavlja obicno problem kod obicnog SGD.

5. a)

6. Čista normalizacija umanjuje ekspresivnu moć, skaliranje i pomak (afina transformacija) rekonstruiraju početnu distribuciju.
- c)

-
1. Konvergencija kod CD algoritma nastupa kada:
 - (a) kada brojnik izraza za vjerojatnost ulaznog uzorka teži nuli
 - (b)** su pozitivni i negativni fazi gradijentnog izraza u ravnoteži
 - (c) su iscrpljeni svi hardverski resursi
 - (d) je postignuta zadovoljavajuća točnost

 2. Negativna faza gradijentnog izraza CD algoritma dolazi od:
 - (a) logaritamska srednja kvadratna pogreška
 - (b) donje varijacijske granice
 - (c)** partičijske funkcije
 - (d) brojnika izraza za vjerojatnost uzorka

 3. Treniranje novog RBM-a (nazovimo ga RBM2) nadodanog na već istrenirani RBM1 može poboljšati ukupne performanse zbog toga što:
 - (a) RBM2 duplificira efekt RBM1
 - (b) RBM1 time postaje nepristran
 - (c)** RBM2 ostvaruje bolji model skrivenog sloja RBM1
 - (d) RBM2 čini elemente skrivenog sloja RBM1 statistički nezavisnim

 4. Kad CD algoritma radi se alternirajuće Gibbsovo uzorkovanje vidljivog i skrivenog sloja kako bi:
 - (a) kada bi minimizirali srednju kvadratnu pogrešku
 - (b)** došli do uzorka iz stacionarne distribucije
 - (c) kako bi prekratili vrijeme
 - (d) kako bi aproksimirali varijancu

- 1. b)**
2. U negativnoj fazi se smanjuje vjerojatnost svih uzoraka. Particijska funkcija je suma svih energija pa je valjda to odgovor
 c)
3. Iz 4. laboratorijske vježbe “Teoretski, tako izgrađen DBN trebao bi povećati $p(v)$, što nam je i cilj.”
 c)
- 4. b)**

-
1. Tehnike regularizacije:
 - (a) povećavaju kapacitet modela
 - (b) smanjuju pristranost modela
 - (c) povećavaju pristranost modela
 - (d) povećavaju kompleksnost modela
 2. Može li korak gradijentnog spusta povećati gubitak?
 - (a) da, ali samo ako koristimo regularizaciju
 - (b) ne
 - (c) da, ali samo ako je mini-grupa prevelika
 - (d) da, ali samo ako je faktor pomaka prevelik
 3. Zašto mape značajki ponekad nadopunjavamo nulama?
 - (a) da smanjimo utjecaj rubova
 - (b) da dimenzije izlaza budu veće od dimenzija ulaza
 - (c) da smanjimo broj parametara mreže
 - (d) da dimenzije izlaza budu jednake dimenzijama ulaza

1. Smanjuje varijancu i malo povećava pristranost.

- c)
2. d)
3. d)

1. Tehnike regularizacije:

- (a) povećavaju kapacitet modela
- (b)** smanjuju pristranost modela
- (c) povećavaju pristranost modela
- (d) povećavaju kompleksnost modela

2. Koja je vremenska složenost računanja izlaza u svim vremenskim koracima u dvosmjernoj višeslojnoj povratnoj neuronskoj mreži (bidirectional RNN), gdje je dubina mreže D, a broj vremenskih koraka odmatanja mreže T?

- (a) $T + D - 1$
- (b)** $T \cdot D$
- (c) T^2
- (d) T

* Zašto potpuno povezani modeli imaju više jednakobrojnih lokalnih minimuma prosječne unakrsne entropije?

- (a)** zbog simetrije neurona istog skrivenog sloja
- (b) zbog simetrije neurona susjednih skrivenih slojeva
- (c) zbog konveksnosti unakrsne entropije
- (d) zbog simetrije neurona izlaznog sloja

1. c)

2. Potrebno mu je T da prođe kroz svaki sloj prije nego može ići na sljedeći.

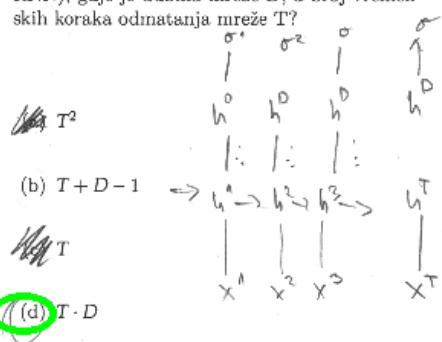
b)

Što nije da se slojevi mogu paralelno računat jer za 1. stanje 2. sloja ti treba samo 1. stanje 1. sloja, a ne zadnje? Onda bi odgovor bio T+D-1

Problem radi dvosmjernosti i mora se čekat da se oba smjera razmotaju

3. a)

- ① Koja je vremenska složenost računanja izlaza u svim vremenskim koracima u dvosmjernoj višeslojnoj povratnoj neuronskoj mreži (bidirectional RNN), gdje je dubina mreže D , a broj vremenskih koraka odmatanja mreže T ?



- ② Koliko ukupno parametara ima obična povratna neuronska mreža s dimenzijom izlaza 100, dimenzijom ulaza 100 te veličinom skrivenog stanja 1000?

(a) 1200	100	1000	+	10^5
(b) 12001100	1000	1000	+	10^6
(c) 1201100	1000	100	+	10^5
(d) 12000				

1. (Vidi prethodnu stranicu)

d)

2. $W_{ih} = (1000 \times 100)$, $W_{hh} = (1000 \times 1000)$, $W_{ho} = (1000 \times 100)$, $bias_{ho} = (100)$, $bias_{hh} = (1000)$

c)

- Dimenzije parametara: $W_{hh} \in \mathbb{R}^{h \times h}$, $W_{xh} \in \mathbb{R}^{h \times d}$, $W_{hy} \in \mathbb{R}^{y \times h}$, pri čemu nam je y kardinalitet izlaza (npr. broj klasa)

biasa ima onoliko koliko i izlaza, dakle h, y

-
1. Koji je odnos između vektora $a = \text{softmax}(x)$ i vektora $b = \text{softmax}(x - \max(x))$
- (a) u implementaciji preferiramo a jer se računa brže
 - (b) matematički, vrijedi $\|a\| < \|b\|$
 - (c)** matematički, vrijedi $a = b$
2. Softmax je invarijantan na dodavanje konstante
c)
3. Rano zaustavljanje ima regularizacijski efekt jer:
- (a) osigurava brzo učenje
 - (b) ne koristimo skup za validaciju
 - (c)** osigurava ograničenu normu vektora parametara modela
 - (d) osigurava rano pretraživanje prostora stanja
4. Težine ne stignu postati prevelike pa ne dolazi do over-fittinga.
c)

-
1. Kod uporabe normalizacije nad grupom (engl. batch-norm), normaliziraju se:

- (a) izlazi neurona s obzirom na odzive koje taj neuron generira za razlike uзорке за учење
- (b) izlazi neurona s obzirom na izlaze njemu susjednih neurona u istom sloju
- (c) izlazi neurona s obzirom na izlaze njemu susjednih neurona u slojevima prije i nakon njega
- (d) izlazi neurona s obzirom na odzive koje taj neuron generira kada mreža radi ispravno odnosno neispravno

2. Zašto potpuno povezani modeli imaju više jednakobrojnih lokalnih minimuma prosječne unakrsne entropije?

- (a) zbog simetrije neurona susjednih skrivenih slojeva
- (b) zbog doprinosa regularizacije
- (c) zbog simetrije neurona izlaznog sloja
- (d) zbog simetrije neurona istog skrivenog sloja

3. Tehnike regularizacije:

- (a) smanjuju pristranost modela
- (b) povećavaju kapacitet modela
- (c) povećavaju pristranost modela
- (d) povećavaju kompleksnost modela

1. 3. prezentacija, slajd 77.

a)

2. d)

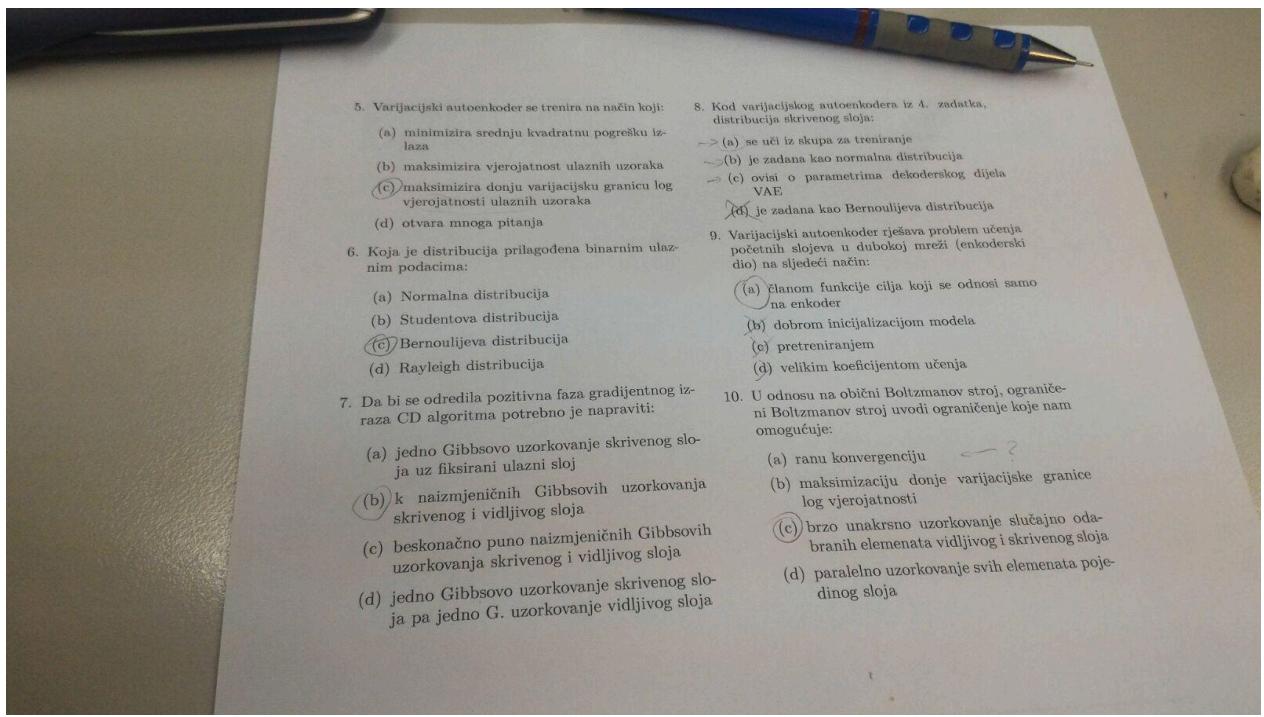
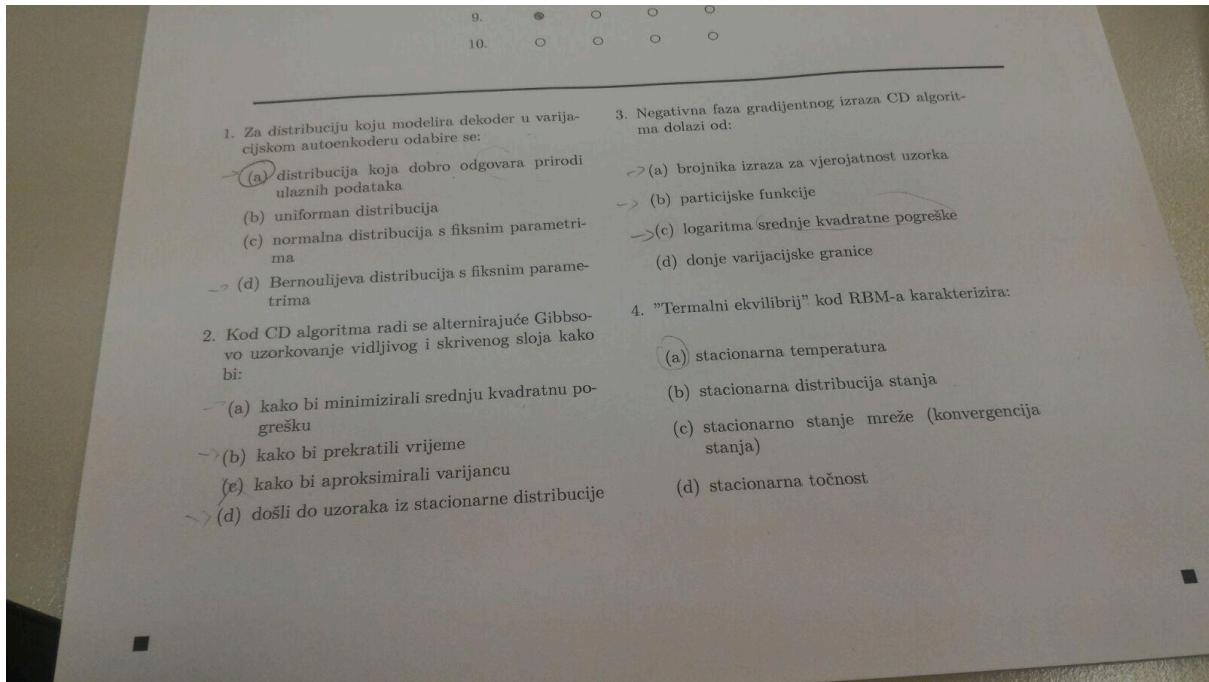
3. c)

1. Kako biste opisali odnos između gubitka "0-1" i unakrsne entropije?

- (a) gubitak "0-1" je metoda izbora kad imamo malo podataka
- (b) unakrsna entropija je metoda izbora kad imamo jako puno podataka
- (c) gubitak "0-1" je nadomjestak unakrsne entropije
- (d) unakrsna entropija je derivabilni nadomjestak gubitka "0-1"

2. Kod uporabe normalizacije nad grupom (engl. batch-norm), normaliziraju se:

- (a) izlazi neurona s obzirom na izlaze njemu susjednih neurona u istom sloju
- (b) izlazi neurona s obzirom na izlaze njemu susjednih neurona u slojevima prije i nakon njega
- (c) izlazi neuronsa s obzirom na odzive koje taj neuron generira za različite uzorke za učenje
- (d) izlazi neurona s obzirom na odzive koje taj neuron generira kada mreža radi ispravno odnosno neispravno



1a 2d 3b 4b 5c 6c 7a 8b 9a 10d

/

Nije 4b nego 4a (ali pise kod RBM-a). Eksplisitno piše na prezentaciji.

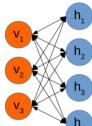
Thermal equilibrium

- Pojam posuđen iz fizike
- Ne odnosi se na stacionarno stanje mreže
- Odnosi se na stacionarnost funkcije gustoće vjerojatnosti
 - Konvergencija funkcije gustoće vjerojatnosti
 - Za zadanu temperaturu T
- Teško je odrediti kada je dosegnut

Također piše i za 7b, a ne 7a.

Pozitivna faza

- Vidljivi sloj je vezan (fixiran) na ulazne uzorke
- Odgovarajući skriveni sloj određuje se u jednoj iteraciji jer nema međusobnih veza u skrivenom sloju (ograničenje kod RBM)
- "Instantni termalni ekvilibrij!!"
- Može se interpretirati kao prilagodavanje distribucije modela vidljivim uzorcima za treniranje
 - Maksimizira se brojnik $p(v)$



i don't think..

jel 7. a ili b na kraju? a

1. "Termalni ekvilibrij" kod RBM-a karakterizira:
 - (a) stacionarna točnost
 - (b) stacionarna temperatura
 - (c) stacionarna distribucija stanja**
 - (d) stacionarno stanje mreže (konvergencija stanja)
2. Treniranje novog RBM-a (nazovimo ga RBM2) nadodanog na već istrenirani RBM1 može poboljšati ukupne performanse zbog toga što:
 - (a) RBM2 čini elemente skrivenog sloja RBM1 statistički nezavisnim
 - (b) RBM2 duplira efekt RBM1
 - (c) RBM2 ostvaruje bolji model skrivenog sloja RBM1**
 - (d) RBM1 time postaje nepristran
5. Negativna faza gradijentnog izraza CD algoritma dolazi od:
 - (a) brojnika izraza za vjerojatnost uzorka
 - (b) donje varijacijske granice
 - (c) logaritma srednje kvadratne pogreške
 - (d) particijske funkcije**
6. Kod varijacijskog autoenkodera iz 4. zadatka, distribucija skrivenog sloja:
 - (a) je zadana kao normalna distribucija**
 - (b) ovisi o parametrima dekoderskog dijela VAE
 - (c) se uči iz skupa za treniranje
 - (d) je zadana kao Bernouljeva distribucija
7. Kod CD algoritma radi se alternirajuće Gibbsovo uzorkovanje vidljivog i skrivenog sloja kako bi:
 - (a) kako bi prekratili vrijeme
 - (b) kako bi minimizirali srednju kvadratnu pogrešku
 - (c) kako bi aproksimirali varijancu
 - (d) došli do uzoraka iz stacionarne distribucije**
3. Negativnu fazu gradijentnog izraza CD algoritma možemo interpretirati kao:
 - (a) mjeru točnosti
 - (b) zaboravljanje onoga što model misli**
 - (c) minimizaciju srednje kvadratne pogreške
 - (d) regularizacijski član funkcije cijene
4. Da bi se odredila pozitivna faza gradijentnog izraza CD algoritma potrebno je napraviti:
 - (a) k naizmjeničnih Gibbsovih uzorkovanja skrivenog i vidljivog sloja
 - (b) beskonačno puno naizmjeničnih Gibbsovih uzorkovanja skrivenog i vidljivog sloja
 - (c) jedno Gibbsovo uzorkovanje skrivenog sloja pa jedno G. uzorkovanje vidljivog sloja
 - (d) jedno Gibbsovo uzorkovanje skrivenog sloja uz fiksirani ulazni sloj**
8. U odnosu na obični Boltzmanov stroj, ograničeni Boltzmanov stroj uvodi ograničenje koje nam omogućuje:
 - (a) maksimizaciju donje varijacijske granice log vjerojatnosti
 - (b) paralelno uzorkovanje svih elemenata pojedinog sloja**
 - (c) ranu konvergenciju
 - (d) brzo unakrsno uzorkovanje slučajno odabralih elemenata vidljivog i skrivenog sloja
9. Varijacijski autoencoder rješava problem učenja početnih slojeva u dubokoj mreži (enkoderski dio) na sljedeći način:
 - (a) glanom funkcije cilja koji se odnosi samo na enkoder**
 - (b) pretreniranjem
 - (c) dobrom inicijalizacijom modela
 - (d) velikim koeficijentom učenja
10. Za distribuciju koju modelira dekoder u varijacijskom autoenkoderu odabire se:
 - (a) normalna distribucija s fiksnim parametrima
 - (b) distribucija koja dobro odgovara prirodi ulaznih podataka**
 - (c) Bernouljeva distribucija s fiksnim parametrima
 - (d) uniforman distribucija

1.c, 2.c, 3. b, 4.d, 5.d, 6.a, 7.d, 8.b, 9.a, 10. b

1. Kod RBM-a, CD-k označava contrastive divergence algoritam, gdje je k:
- broj Gibbsovih uzorkovanja vidljivog i skrivenog sloja
 - parametar odabrane funkcije uvjetne vjerojatnosti skrivenog sloja
 - broj centroida klastera
 - koeficijent učenja
2. Denoising autoenkoderi ostvaruju regularizaciju:
- permutacijom ulaznih podataka
 - dodavanjem šuma u skriveni sloj
 - dodavanjem šuma u ulazne podatke
 - uklanjanjem šuma iz skrivenog sloja
3. Kod dubokih autoenkodera, problem dubokog učenja može se riješiti:
- učenjem bez nadzora
 - nelinearnim aktivacijskim funkcijama
 - povećanjem koeficijenta učenja
 - pohlepnim predtreniranjem slojeva
4. Svrha skrivenog sloja u Boltzmanovom stroju je:
- postizanje bolje separacije klasa
 - postizanje brže konvergencije
 - uključivanje korelacija višeg reda vidljivog sloja u model
 - efikasnijeg uzorkovanja stanja mreže
5. Kod treniranja varijacijskog autoenkodera dozvoljeno je u potpunosti odbaciti višestruka uzorkovanja slučajnih varijabli:
- ako na ulazu ne postoji šum
 - ako je veličina mini grupe za treniranje dovoljno velika
 - ako koristimo normalnu distribuciju
 - ako ne koristimo backpropagation algoritam
6. Može li se lokalni gradijent u algoritmu backpropagation odrediti za stohastičke neurone?
- uvijek
 - nikad
 - samo u skrivenom sloju
 - samo u poluskrivenom sloju

1. Može li se lokalni gradijent u algoritmu backpropagation odrediti za stohastičke neurone?
- uvijek
 - samo u vidljivom sloju
 - samo u skrivenom sloju
 - nikad
2. Efekt linearizacije enkodera s aktivacijskom funkcijom tanh, može se sprječiti
- povećanjem broja elemenata skrivenog sloja
 - vezanjem težina enkodera i dekodera
 - povećanjem veličine mini grupe za treniranje
 - transponiranjem matrice težina enkodera
3. Kodirajući dio autoenkodera s linearnim neuronima efektivno ostvaruje funkcionalnost:
- najbržeg spusta
 - maksimizacije vjerojatnosti ulaznih uzoraka
 - maksimizacije donje varijacijske granice
 - analize glavnih komponenti
4. Svrha skrivenog sloja u Boltzmanovom stroju je:
- uključivanje korelacija višeg reda vidljivog sloja u model
 - postizanje bolje separacije klasa
 - efikasnijeg uzorkovanja stanja mreže
 - postizanje brže konvergencije
5. Željena rijetkost u skrivenom sloju rijetkog autoenkodera može se ostvariti:
- normalizacijom težina u enkoderskom sloju
 - maksimizacijom vjerojatnosti ulaznih uzoraka
 - normalizacijom težina u dekoderskom sloju
 - korigiranjem pomaka u enkoderskom sloju
6. Denoising autoenkoderi ostvaruju regularizaciju:
- dodavanjem šuma u skriveni sloj
 - dodavanjem šuma u ulazne podatke
 - permutacijom ulaznih podataka
 - uklanjanjem šuma iz skrivenog sloja

- ✓ 1. Za aktivacijsku funkciju neuronske mreže ni u kojem slučaju nećemo koristiti:

- (a) $g(x|\alpha, \beta) = \beta + \alpha \cdot x$
 (b) $g(x) = \max(-0.1, x)$
 (c) $g(x|\alpha) = \max(0, x) + 0.001 \cdot \min(0, x)$
 (d) $g(x) = \max(0.1, x)$

- ✓ 2. Zašto su duboki modeli za visokodimenzionalne podatke prikladniji od alternativa?

- (a) zbog velikog kapaciteta
 (b) zbog mogućnosti modeliranja nelinearnih decizijskih funkcija
 (c) jer nisu skloni preučenju (engl. overfitting)
 (d) zbog kompozitne strukture

- ✓ 3. Zašto mape značajki ponekad nadopunjavamo nulama?

- (a) da dimenzije izlaza budu veće od dimenzija ulaza

- (b) da smanjimo utjecaj rubova

- (c) da smanjimo broj parametara mreže

- (d) da dimenzije izlaza budu jednake dimenzijama ulaza

- ✓ 4. Navedite izraz za ažuriranje jednog skivenog stanja povratne neuronske mreže:

- (a) $h[t] = \tanh(W^*h[t-1] + U^*x[t-1] + b)$
 (b) $h[t] = \text{ReLU}(W^*h[t-1] + b)$
 (c) $h[t] = W^*h[t-1]$
 (d) $h[t] = \tanh(W^*h[t-1] + U^*x[t] + b)$

- ✓ 5. Navedite nelinearnost koju koristi obična povratna neuronska mreža u laboratorijskoj vježbi:

- (a) tangens hiperbolni (\tanh)
 (b) zglobovica (ReLU)
 (c) logistička sigmoida
 (d) propusna zglobovica (leaky ReLU)

- ✓ 6. Vjerojatnosti na izlazu obične povratne neuronske mreže u svakom vremenskom koraku se računaju s izrazom:

- (a) $y_{\text{pred}}[t] = \text{softmax}(V^*h[t]+c)$
 (b) $y_{\text{pred}}[t] = W^*h[t] + U^*x[t] + b$
 (c) $y_{\text{pred}}[t] = V^*h[t]$
 (d) $y_{\text{pred}}[t] = \log(V^*h[t]+c)$

- ✓ 7. Navedite izraz za gradijent funkcije pogreške na izlazu u jednom koraku povratne neuronske mreže:

- (a) $dy[t] = y_{\text{pred}}[t] - 1$
 (b) $dy[t] = y_{\text{pred}}[t] - y[t]$
 (c) $dy[t] = y_{\text{pred}}[t] \cdot h[t]$
 (d) $dy[t] = \log(y_{\text{pred}}[t] - y[t])$

/

1. (1 point) Efekt linearizacije enkodera s aktivacijskom funkcijom $tanh$, može se spriječiti:

- | | |
|--|---|
| A) vezanjem težina enkodera i dekodera | C) povećanjem broja elemenata skrivenog sloja |
| B) transponiranjem matrice težina enkodera | D) povećanjem veličine mini-grupe za treniranje |

2. (1 point) Željena rijetkost u skrivenom sloju rijetkog autoenkodera može se ostvariti:

- | | |
|---|--|
| A) maksimizacijom vjerojatnosti ulaznih uzoraka | C) korigiranjem pomaka u enkoderskom sloju |
| B) normalizacijom težina u dekoder-skom sloju | D) maksimizacijom težina u enkoderskom sloju |

3. (1 point) Vjerojatnost jednog stanja Boltzmanovog stroja:

- | | |
|-----------------------------------|--|
| A) raste s rastom energije stanja | C) raste s donjom varijacijskom granicom |
| B) pada s rastom energije stanja | D) varira prema Gaussovoj distribuciji |

4. (1 point) Može li se lokalni gradijent u algoritmu backpropagation odrediti za stohastičke neurone?

- | | |
|---------------------------|-------------------------------|
| A) uvijek | C) nikad |
| B) samo u skrivenom sloju | D) samo u poluskrivenom sloju |

5. (1 point) Treniranjem RBM-a nastojimo:

- | | |
|---|--|
| A) maksimizirati energiju uzoraka za treniranje | C) maksimizirati log vjerojatnost uzoraka iz skupa za treniranje |
| B) postići maksimalnu generalizaciju | D) maksimizirati donju varijacijsku granicu vjerojatnosti uzoraka iz skupa za treniranje |

1

6. (1 point) Smanjenje broja elemenata skrivenog sloja kod klasičnog autoenkodera (u odnosu na ulazni sloj) ima ulogu:

- | | |
|--------------------------------|-------------------|
| A) unakrsne entropije | C) optimizacije |
| B) maksimizacije vjerojatnosti | D) regularizacije |

1. Stari ispiti

<MI 2016/17

21. studenog 2016.

Zavod za elektroniku, mikrelektroniku,
računalne i inteligentne sustave

Duboko učenje

pismeni ispit

1. Razmatramo klasifikacijsku konvolucijsku mrežu s arhitekturom:

- konvolucijski sloj bez nadopunjavanja: dvije jezgre 3×3 , korak 1, aktivacija ReLU;
- sažimanje maksimumom 2×2 ;
- pretvaranje u vektor;
- potpuno povezani sloj dimenzije 2 te aktivacijom softmaks.

Parametri mreže zadani su matricama (nema pomaka!):

$$k_1 = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, k_2 = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, W = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \cdot \ln(2).$$

Zadataci:

- Odredite veličinu ulazne slike.
- Odredite izlaze mreže ako je na ulazu zadana slika čiji su svi pikseli jednaki nuli osim piksela (2,3), (3,2) i (3,3) koji su postavljeni na jedan (indeksi redaka i stupaca počinju od nule).

(neka netko usporedi) -> Točno je, i meni

1

2.

2. Razmatramo klasifikacijsku mrežu s s arhitekturom:

- potpuno povezani sloj: dimenzija 2, aktivacija ReLU;
- potpuno povezani sloj dimenzije 2, aktivacija softmaxs.

Parametri mreže zadani su matricama:

$$W_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, b_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, W_2 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \ln(2), b_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Zadatci:

- Napišite jednadžbe modela kojeg implementira mreža.
- Odredite izlaze mreže ako je na ulazu zadan podatak $(5,2)$.
- Odredite gradijente po retcima matrice W_1 ako gubitak odgovara negativnoj log-izglednosti a željeni izlaz mreže je $(0,1)$.

22 (Treba provjera, u c) negativna log izglednost ispada $\ln(0)$)

Pogrešno je izračunat softmax iz $h_3 \rightarrow y$

Mislim da bi trebalo biti: $y = [32/33 \quad 1/33] \rightarrow$ kako to dobijes?

“kako to dobijes?” - primjeniš softmax funkciju nad $[1 \quad 0]$, dakle:

$y = [\exp(1) / (\exp(1) + \exp(0)) \quad \exp(0) / (\exp(1) + \exp(0))]$

Ispravite me ako grijesim :) +1 +1 +1

Ne nad $[1, 0]$ nego nad $h_3 = [5\ln(2), 0]$. Onda imas:

$y = [\exp(\ln(2^5)) / (\exp(\ln(2^5)) + \exp(0)), \exp(0) / (\exp(\ln(2^5)) + \exp(0))]$

$y = [32/33, 1/33]$

a) $h_1 = W_1 x + b_1$

$h_1 = \text{ReLU}(h_1)$

$h_2 = W_2 h_1 + b_2$

$y = \text{softmax}(h_2)$

b) $h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 5 \\ 2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 5 \\ -2 \end{bmatrix}$

$h_2 = \begin{bmatrix} 5 \\ -2 \end{bmatrix}$

$h_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 5 \\ -2 \end{bmatrix} \cdot \ln 2 + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 5 \ln 2 \\ 0 \end{bmatrix}$

$y = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Može netko riješit pod c)?

PLIIIZ neka netko potvrdi ovo!

1

c) Odredite grad po redima mat W_1 ako gubitak odgovara neg log izglednosti, a željeni izlaz je $(0, 1)$

$$\frac{\partial \mathcal{L}}{\partial \vec{e}_2} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}^T - \begin{bmatrix} 0 \\ 1 \end{bmatrix}^T = [1, -1] \quad \frac{\partial L}{\partial W_1} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \ln 2 \cdot [5 \ 2] = \begin{bmatrix} 10 & 4 \\ -10 & -4 \end{bmatrix} \ln 2$$

$$\frac{\partial \mathcal{L}}{\partial h_1} = \begin{bmatrix} 1 & -1 \\ -1 & -1 \end{bmatrix} \ln 2 \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \ln 2$$

$$\frac{\partial \mathcal{L}}{\partial s_1} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \ln 2 \cdot \begin{bmatrix} 1 & -1 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \ln 2$$

Čini mi se da bi dL/ds_1 bi trebao biti istih dimenzija kao i s_1 , a ne matrica 2×2 . $dL/dh_1 * dh_1/ds_1$ ($[1 \times 2] * [2 \times 2]$). I onda je na kraju dL/dW_1 vanjski produkt vektora dL/ds_1 i ulaza x .

3.

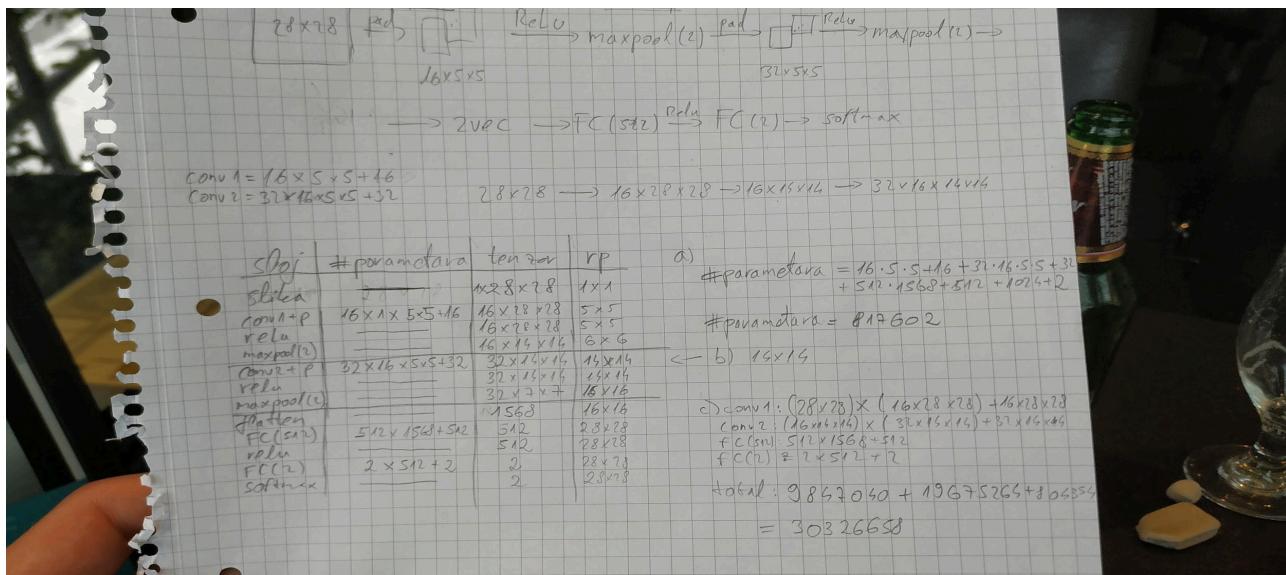
3. Razmatramo klasifikacijsku konvolucijsku mrežu koja je prilagodena obradi sivih slika dimenzija 28×28 . Arhitektura mreže je:

- konvolucijski sloj s nadopunjavanjem (rezolucija ostaje jednaka): 16 jezgara 5×5 , korak 1, aktivacija ReLU;
- sažimanje maksimumom 2×2 ;
- konvolucijski sloj s nadopunjavanjem (rezolucija ostaje jednaka): 32 jezgre 5×5 , korak 1, aktivacija ReLU;
- sažimanje maksimumom 2×2 ;
- pretvaranje u vektor;
- potpuno povezani sloj dimenzije 512, aktivacija ReLU;
- potpuno povezani sloj dimenzije 2, aktivacija softmax.

Zadatci:

- Koliko ukupno parametara ima ova mreža?
- Koliko ukupno receptivno polje imaju značajke iz drugog konvolucijskog sloja?
- Koliko bi parametara imala ova mreža kad bismo konvolucijske slojeve zamjenili potpuno povezanim slojevima s jednakim brojem izlaznih aktivacija.

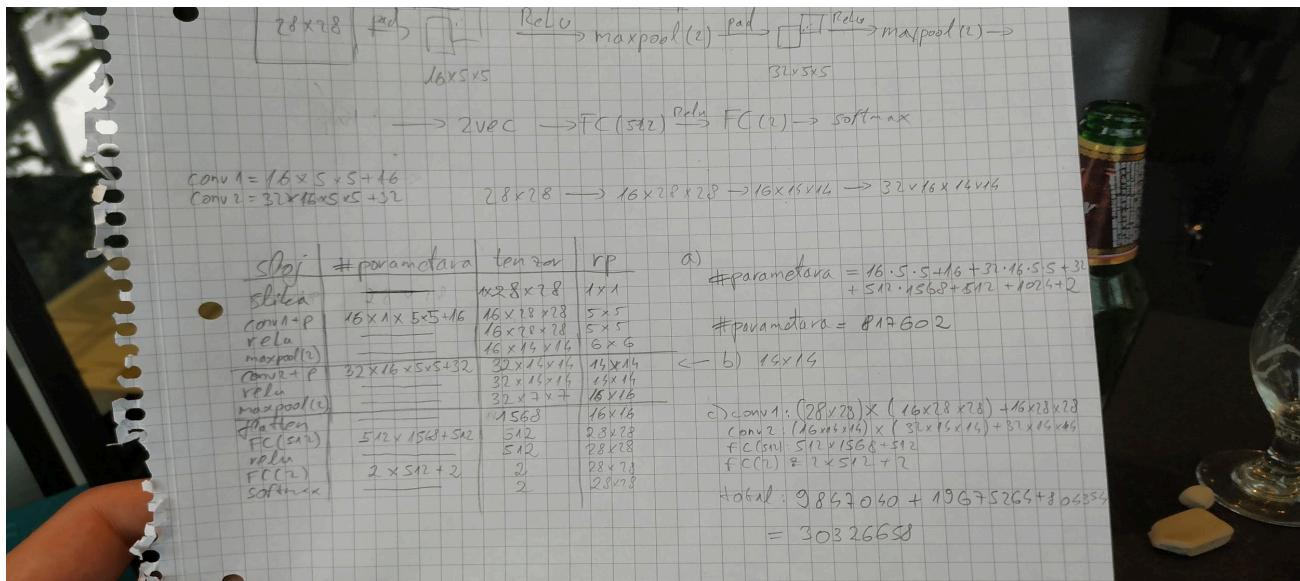
Video: <https://www.youtube.com/watch?v=BD8D5Dsx1y4>



- Zar ne fale ovdje parametri max pool slojeva u zbroju? maxpool nema parametara

1

- Maxpool nema parametara? - nema, to je samo sažimanje



zasto prvi maxpool povecava RP za 1, a drugi za 2?? zato sto svaki pool povecava sve buduce aktivacije puta 2

- nije li za conv2 kriv broj parametara u c)? trebalo bi ic $(16 \times 28 \times 28) * (32 \times 14 \times 14) + (32 \times 14 \times 14)$ jer je onda kompatibilno s prethodnim izlazom
- <https://fer.studosi.net/d/321-dubuce-2-laboratorijska-vjezba-20192020/10>

alternativno rjesenje za c) ako mi moze netko potvrdit jer mi se cini da je ovo sa slike krivo:

slika - $1 \times 28 \times 28 \rightarrow 784$

conv1 - $16 \times 28 \times 28 \rightarrow 12544$

conv2 - $32 \times 14 \times 14 \rightarrow 6272$

fc1 - $512 \rightarrow 512$

fc2 - $2 \rightarrow 2$

$$\text{uk_param} = (784 \cdot 12544 + 12544) + (12544 \cdot 6272 + 6272) + (6272 \cdot 512 + 512) + (512 \cdot 2 + 2) = 91742082$$

/

zas3

4.

(ovako nešto, nisam testirao, neka netko provjeri)

[backward_inputs](#) 1. linija ako se ne varam ide `* np.power(self.inverses, 2)` a ne `/`

4. Razmatramo sloj dubokog modela koji ulazni vektor \mathbf{q} preslikava u izlazni vektor \mathbf{p} , a može se opisati sljedećim jednadžbama: $p_i = w_1 \cdot 1/q_i + w_2 \cdot q_i^2$.

Predložite implementaciju tog sloja razredom koji nasljeđuje sučelje `Layer` iz prvog zadatka druge laboratorijske vježbe te implementira metode: `forward(self, inputs)`, `backward_inputs(self, grads)` i `backward_parameters(self, grads)`. U rješenju možete pretpostaviti da je veličina mini-grupe 1.

```
import numpy as np

class MyLayer(Layer):
    def __init__(self):
        super().__init__()
        self.w = np.random.randn(2)

    def forward(self, inputs):
        self.inverses = 1 / inputs
        self.squares = inputs * inputs
        return self.w * np.array([[self.inverses], [self.squares]])

    def backward_inputs(self, grads):
        grad = -self.w[0] / np.power(self.inverses, 2) + 2 * self.w[1] * np.sqrt(self.squares)
        return grads * grad

    def backward_parameters(self, grads):
        grad = np.array([[self.inverses], [self.squares]])
        return grads * grad
```

/

5.

5. Promotrite sljedeće dvije funkcije:

(a) $f(x) = \sqrt{1 + \max(0, x)^2}$

(b) $g(x) = \exp(x)/(1 + \exp(x))$

Odredite derivacije funkcija f i g obzirom na ulaze. Na temelju dobivenih izraza odredite koja od dvije funkcije bi bila prikladniji izbor nelinearnosti za skrivene slojeve dubokog modela.

5. a) $f(x) = \sqrt{1 + \max(0, x)^2} = f(a), a = 1 + \max(0, x)^2$

$$\frac{\partial f}{\partial a} = \frac{\partial}{\partial a} a^{\frac{1}{2}} = \frac{1}{2} a^{-\frac{1}{2}}$$

$$\frac{\partial a}{\partial x} = 2 \cdot \max(0, x) \cdot \frac{\partial \max(0, x)}{\partial x} = \begin{cases} 2 \cdot \max(0, x), & x > 0 \\ 0, & x \leq 0 \end{cases} = \begin{cases} 2x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

$$\frac{\partial f}{\partial x} = \frac{1}{2\sqrt{1+x^2}} \cdot 2x = \frac{x}{\sqrt{1+x^2}}, \quad x > 0$$

$$\frac{\partial f}{\partial x} = \begin{cases} \frac{1}{2\sqrt{1+x^2}} \cdot 0 = 0, & x \leq 0 \\ 1, & x > 0 \end{cases}$$

$$\frac{e^x}{e^x + 1} = \frac{e^x(e^x + 1) - e^x}{e^{2x}} = \frac{e^{2x} - e^{2x} + e^x}{(e^x + 1)^2} = \frac{e^x}{(e^x + 1)^2}$$

Prepostavljam da je prva funkcija bolja jer je gotovo ista kao ReLU, no derivabilna je u nuli.
Možda je i dobro primijetiti da je g(x) sigmoida.

/

6.

6. Napišite potpuni program za određivanje **korijena** kvadratne jednadžbe $a \cdot x^2 + b \cdot x + c = 0$ gradijentnim spustom. U vašem rješenju iskoristite mogućnost automatske diferencijacije koju nudi Tensorflow.

Podsjetnik: koristite operaciju koju vraća metoda minimize primjerka razreda `tf.train.GradientDescentOptimizer`.

6. Korišten je PyTorch umjesto TensorFlow-a. Zbog toga što se traži korijen jednadžbe koristio sam kvadrat funkcije kao funkciju gubitka (minimum gubitka je jednak nuli za korijen, ako je on realan).

```
import numpy as np
import torch

def find_roots(a, b, c):
    x = torch.randn(1, requires_grad = True)
    optimizer = torch.optim.SGD([x], lr = 0.1)
    for i in range(100):
        y = a*x*x + b*x + c
        loss = y*y
        loss.backward()
        optimizer.step()
        optimizer.zero_grad()
    return x

result = find_roots(torch.tensor(1), torch.tensor(-5) ,torch.tensor(6))
print(result)
```

/

ZI 2016/17

23. siječnja 2017.

Zavod za elektroniku, mikrelektroniku,
računalne i inteligentne sustave

Duboko učenje

pismeni ispit

1. Razmatramo sigmoidalnu transformaciju podataka u izlaznom sloju dubokog modela kojeg učimo optimiranjem i) kvadratnog gubitka ii) negativne log-izglednosti. Izvedite gradijente obaju gubitaka s obzirom na ulaze sigmoidalne funkcije. Koji od dvaju gubitaka je prikladniji za slučaj klasifikacije? Zašto? Napišite kod za računanje gradijenata pod numpyjem pod pretpostavkom da podaci dolaze u grupama proizvoljne veličine.

(Može netko provjerit ili pokazat lakši način?)

1

$$\mathcal{L} = - \sum_{i=0}^n y_i \cdot \log(\sigma(x_i))$$

$$a = y \cdot b$$

$$b = \log(c)$$

$$c = \sigma(x)$$

$$\frac{\partial a}{\partial b} = y$$

$$\frac{\partial b}{\partial c} = \frac{1}{c}$$

$$\frac{\partial c}{\partial x} = \sigma(x)(1 - \sigma(x))$$

$$\frac{\partial a}{\partial x} = y \cdot \frac{1}{\sigma(x)} \cdot \sigma(x)(1 - \sigma(x))$$

$$\frac{\partial a}{\partial x} = y \cdot (1 - \sigma(x))$$

$$\frac{\partial \mathcal{L}}{\partial x} = - \sum_{i=0}^n y_i - y_i \cdot \sigma(x_i)$$

(Može provjera?)

Iz prve prezentacije, slajd 36.:

Iz gradijenta kvadratne pogreške poi ulazima sigmoide vidi se da kada je sigmoida u zasićenju ($z \gg 0$ ili $z \ll 0$) gradijent funkcije gubitka je malen, bez obzira je li je $\sigma(z)$ blizu y ili ne: mreža ne može naučiti iz takvog primjera.

kvadratna nije prikladna za klasifikaciju jer primjeri daleko od decizionske plohe imaju ogroman gubitak pa outlier može prevagnuti nad 100 primjera blizu plohe.CIFAR

/

3. 4

2. Razmatramo učenje višerazredne logističke regresije optimiranjem unakrsne entropije stohastičkim gradijentnim spustom uz zadanu stopu učenja $\delta = 1$. Zadan je sljedeći skup za učenje: $x_1 = (1, 1, 1, 0)$, $y_1 = [1, 0]$, $x_2 = (0, 1, 1, 1)$, $y_2 = [0, 1]$. Početno stanje matrice težina i vektora pomaka zadano je s $\mathbf{W}^0 = \begin{bmatrix} 0.5 & -0.5 & -0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & -0.5 \end{bmatrix}$, $\mathbf{b}^0 = \begin{bmatrix} 0 & 0 \end{bmatrix}$.

- Odredite jednadžbe gradijenata svih parametara.
- Provode učenje podatkom (x_1, y_1) , bez korištenja ikakve regularizacije. Izračunajte gradiente i nove vrijednosti parametara.
- Provode jednu epohu učenja počevši od \mathbf{W}^0 i \mathbf{b}^0 , uz regularizaciju ispuštanjem (engl. dropout) ulaznih značajki s vjerojatnošću $p(\mu_{x_i}) = 0.5$, $i = 1, 2, 3, 4$. Prepostavite sljedeći raspored ispuštanja:
 - prva mini-grupa je (x_1, y_1) , ugašeni su 1. i 3. ulaz,
 - druga mini-grupa je (x_2, y_2) , ugašeni su 2. i 4. ulaz.Izračunajte gradiente i nove vrijednosti parametara.
- Izračunajte izlaz naučene mreže za ulaze $x_a = (0, 1, 0, 1)$ i $x_b = (1, 0, 1, 0)$.

4. PITANJE: Ne bi li matricu W (prije ulaza u softmax) trebalo pomnožiti skalarno s matricom koja je također dimenzija 2×4 , a sve su joj vrijednosti 0.5 (jer je to zadano kao onaj p(mi_x_i))?

to bi trebalo kod d dijela zadatka

A zašto ne i u c) dijelu?

c) je faza ucenja i tamo imas stvarni dropout u smislu da ces ili totalno srezat ulaz na vrijednost 0 ili ga ne dirat uopce. vidis da ti kaze u prvoj tockici da se gasi 1. i 3. ulaz, a u drugoj 2. i 4. ulaz. to znaci da ces bas te vrijednosti zamijenit s 0. e sad, da model nebi dobio hrpu informacije nenađano onda kad se radi inferencija se radi to skaliranje tezina.

$$2. L = - \sum_{i=1}^N y_i \cdot \log \left(\frac{z_i}{\sum_k z_k} \right)$$

$$= - \sum_{i=1}^N y_i (\log z_i - \log \sum_k z_k)$$

$$= - \sum_{i=1}^N y_i (z_i - \log \sum_k z_k)$$

$z = w \cdot x + b$

$$\delta = 1 \quad x_1 = (1, 1, 1, 0) \quad x_2 = (0, 1, 1, 1)$$

$$y_1 = [1, 0] \quad y_2 = [0, 1]$$

$$w^* = \begin{bmatrix} 0.5 & -0.5 & -0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & -0.5 \end{bmatrix}, b^* = [0 \ 0]$$

$$a) \frac{\partial L}{\partial z_i} = - (1 - z_i) = z_i - 1 = z_i - [y = i] \quad \frac{\partial L}{\partial s} = p - y$$

$$\frac{\partial z}{\partial w} = x_i$$

$$\frac{\partial z}{\partial b} = 1$$

$$\frac{\partial L}{\partial w} = (\text{softmax}(w \cdot x + b) - [[y = c_i]]) \cdot x$$

$$\frac{\partial L}{\partial b} = (\text{softmax}(w \cdot x + b) - [[y = c_i]])$$

$$z = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$$

$$\text{softmax}(z) = \begin{bmatrix} 0.269 \\ 0.731 \end{bmatrix}$$

$$\frac{\partial L}{\partial w} = \left(\begin{bmatrix} 0.269 \\ 0.731 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \cdot \begin{bmatrix} 1 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} -0.731 \\ 0.731 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} -0.731 & -0.731 & -0.731 & 0 \\ 0.731 & 0.731 & 0.731 & 0 \end{bmatrix}$$

$$\frac{\partial L}{\partial b} = \begin{bmatrix} 0.269 \\ 0.731 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.731 \\ 0.731 \end{bmatrix}$$

$$w \leftarrow w - \frac{\partial L}{\partial w} |_x = \begin{bmatrix} 1.231 & 0.231 & -0.231 & 0.5 \\ -0.231 & -1.231 & -0.231 & -0.5 \end{bmatrix}$$

$$b \leftarrow b - \frac{\partial L}{\partial b} |_x = \begin{bmatrix} 0.731 \\ -0.731 \end{bmatrix}$$

1

a) 1. MB: $x_1 = [0 \ 1 \ 0 \ 0]$

$$z = Wx + b = \begin{bmatrix} 0.5 & -0.5 & -0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & -0.5 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}$$

$$\text{softmax}(z) = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$\frac{\partial L}{\partial W} = \left(\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \cdot \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial L}{\partial b} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$$

$$W_{\text{new}} = \begin{bmatrix} 0.5 & 0 & -0.5 & 0.5 \\ 0.5 & -1 & 0.5 & -0.5 \end{bmatrix}$$

$$b_{\text{new}} = \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}$$

2. MB $x_2 = [0 \ 0 \ 1 \ 0]$

$$z = \begin{bmatrix} 0.5 & 0 & -0.5 & 0.5 \\ 0.5 & -1 & 0.5 & -0.5 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\text{softmax}(z) = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$\frac{\partial L}{\partial W} = \left(\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \cdot \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0.5 & 0 \\ 0 & 0 & -0.5 & 0 \end{bmatrix}$$

$$\frac{\partial L}{\partial b} = \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}$$

$$W_{\text{new}} = \begin{bmatrix} 0.5 & 0 & -1 & 0.5 \\ 0.5 & -1 & 1 & -0.5 \end{bmatrix}$$

$$b_{\text{new}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

d)

$$z_a = Wx + b = \begin{bmatrix} 0.5 & 0 & -1 & 0.5 \\ 0.5 & -1 & 1 & -0.5 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.5 \\ -1.5 \end{bmatrix}$$

$$\text{softmax}(z_a) = \begin{bmatrix} 0.88 \\ 0.12 \end{bmatrix}$$

$$z_b = \begin{bmatrix} 0.5 & 0 & -1 & 0.5 \\ 0.5 & -1 & 1 & -0.5 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.5 \\ 1.5 \end{bmatrix}$$

$$\text{softmax}(z_b) = \begin{bmatrix} 0.12 \\ 0.88 \end{bmatrix}$$

1

3.

$$\Delta W = \eta [\langle VH \rangle^o - \langle VH \rangle^s]$$

	V^o
$\begin{matrix} (1) & (1) \\ (1) & (0) \\ (1) & (1) \end{matrix}$	H^o
$\begin{matrix} (1) & (0) \\ (0) & (1) \\ (0) & (0) \end{matrix}$	$\rightarrow \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 2 \\ 2 & 1 & 1 \end{bmatrix}$

	V^o
$\begin{matrix} (1) & (1) \\ (0) & (0) \\ (1) & (1) \end{matrix}$	H^o
$\begin{matrix} (0) & (1) \\ (1) & (0) \\ (0) & (0) \end{matrix}$	$\rightarrow \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 2 & 2 & 0 \end{bmatrix}$

$$\Delta W = \eta \left(\begin{bmatrix} 2 & 1 & 2 \\ 2 & 1 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 0 \\ 2 & 2 & 0 \end{bmatrix} \right) = 0.01 \cdot \begin{bmatrix} 1 & 0 & 2 \\ 0 & -1 & 1 \end{bmatrix}$$

Na kraju se dijeli s brojem primjera u batchu.

$$w_{13} = 0.02 \cdot \frac{1}{3} = \frac{1}{150}$$

$$w_{23} = 0.01 \cdot \frac{1}{3} = \frac{1}{300}$$

<http://www.zemris.fer.hr/~ssegvic/du/duZadatciGen.pdf>

4. Za izlazni sloj varijacijskog autoenkodera odabrali smo Rayleighovu distribuciju:

$$p(x|\sigma) = \frac{x}{\sigma^2} e^{-x^2/(2\sigma^2)}$$

Izvedite izraz za drugu komponentu funkcije cilja varijacijskog autoenkodera:

$$\mathbb{E}_{q_\varphi(\mathbf{z}|\mathbf{x}^{(i)})} [\log(p_\theta(\mathbf{x}^{(i)}|\mathbf{z}))]$$

Prepostavite da je dovoljno samo jednom uzorkovati skriveni sloj jer su mini-grupe dovoljno velike. Skicirajte odgovarajući dekoder.

4.

(4) $\mathbb{E}_{q_\varphi(\mathbf{z}|\mathbf{x}^{(i)})} [\log(p_\theta(\vec{x}^{(i)}|\vec{z}))] \approx \frac{1}{K} \sum_{k=1}^K \log p_\theta(\vec{x}^{(i)}|\vec{z}^{(k)}) \approx \log p_\theta(\vec{x}^{(i)}|\vec{z})$

ZADANO JE DA \vec{z}
JE DOVOLJNO SAMO JEDNOM
UZORKOVATI

Za jedan element vektora \vec{x} vrijedi:

$$p(x|\sigma) = \frac{x}{\sigma^2} e^{-x^2/2\sigma^2}$$

Uz pretpostavku nezavisnosti elemenata vektora:

$$p(\vec{x}|\vec{\sigma}) = p(x_1, x_2, \dots, x_J | \vec{\sigma}) = \prod_{j=1}^J p(x_j | \sigma_j)$$

$$\begin{aligned} \log p(\vec{x}|\vec{\sigma}) &= \log \prod_{j=1}^J p(x_j | \sigma_j) \\ &= \sum_{j=1}^J \log \frac{x_j}{\sigma_j} e^{-x_j^2/2\sigma_j^2} \\ &= \sum_{j=1}^J \left[\log x_j - \log \sigma_j^2 - \frac{x_j^2}{2\sigma_j^2} \right] = \log p(\vec{x}|\vec{z}) \end{aligned}$$

<http://www.zemris.fer.hr/~sseqvic/du/duZadatciGen.pdf>

/

5.

5. Učiteljica Svjetlana suočena je s teškim problemom - učenici njenog 2.D razreda iznimno su nestrašni. Naime, kad stanu u red za prebrojavanje koliko ih je prisutno na nastavi, neki od njih se nakon što su prebrojni vrate na kraj reda, te ih zbog svoje zaboravljivosti Svjetlana ponovno prebroji, i ponekad po kraju (dugotrajnog) brojanja izgleda kao da je mnogo više djece na nastavi nego što ih ima u razredu - što je stvarno nemoguće. Svjetlana je u novinama pročitala o novoj metodi povratnih neuronskih mreža, te ih je odlučila iskoristiti za svoj problem.

Zadatak je postavila kao u nastavku: svaki indeks skrivenog stanja povratne neuronske mreže odgovara jednom učeniku, te na tom indeksu želi imati informaciju je li se učenik pojavio taj dan ili nije (0 ako nije, 1 ako je). Osim toga, kako bi skratila vrijeme brojanja, želi da povratna mreža na izlazu ispiše (1 - broj učenika koji nedostaju). Dakle, u slučaju da su svi učenici prisutni na izlazu je 1, ako fali jedan učenik, na izlazu je 0 a ako fali dvoje učenika, na izlazu je -1 itd.

U Svjetlaninom razredu je troje učenika, te je svakom od njih dodijeljen one-hot kod. Na ulaz mreže u slijedu dolaze one-hot kodovi učenika. Radi jednostavnosti, prijenosna funkcija na izlazu je identitet (umjesto funkcije softmax), a dimenzionalnost izlaza je 1. Skriveno stanje mreže je inicijalizirano na vektor nula. Ukoliko je poznato da je matrica \mathbf{U} jedinična, matrica \mathbf{W} dijagonalna, odredite vektore pristranosti \mathbf{b} i \mathbf{c} , matrice težina \mathbf{W} i \mathbf{V} te prijenosnu funkciju skrivenog sloja f .

5.

(5)

$$\text{Učenik A} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

$$\text{Učenik B} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

$$\text{Učenik C} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

$$h^+ = W_{hh} \cdot h^{+^{-1}} + W_{xh} \cdot x^+ + b_h$$

$$\sigma = W_{hy} \cdot h^+ + b_0$$

RADANO:

$$h^0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

$$W_{xh} = I$$

$$W_{hh} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$$

UNRSTIMO:

$$h^+ = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} h^{+^{-1}} + x^+ + b_h$$

$$\sigma = W_{hy} \cdot h^+ + b_0$$

Za slučaj kad nema učenika, na izlazu mora biti -2:

$$\sigma = -2$$

$$\underbrace{W_{hy} \cdot h^+}_0 + b_0 = -2$$

$$b_0 = -2$$

S obzirom da je zadano da je vektor h u slučaju da nema učenika nul-vektor, želimo element vektora povećati na 1 ako je učenik s tim indeksom prisutan. To možemo postići ako je $W_{hh} = I$. U vektoru h^+ se mogu prući i vrijednosti veće od 1, stoga je prikladna prijenosna funkcija $f = \min(h^+, 1)$. Kako bi prebrojali učenike, vektor W_{hy} mora biti jedinični vektor.

$$h^+ = \min(h^{+^{-1}} + x^+, 1)$$

$$\sigma = [1 \ 1 \ 1] \cdot h^+ - 2$$

$$(5) \text{ učenik A} = [1 \ 0 \ 0]$$

$$\text{učenik B} = [0 \ 1 \ 0]$$

$$\text{učenik C} = [0 \ 0 \ 1]$$

$$h^+ = W_{hh} \cdot h^{+-1} + W_{xh} \cdot x^+ + b_h$$

$$\sigma = W_{hy} \cdot h^+ + b_0$$

RADANO:

$$h^0 = [0 \ 0 \ 0]$$

$$W_{xh} = I$$

$$W_{hh} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$$

UVRSTIMO:

$$h^+ = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} h^{+-1} + x^+ + b_h$$

$$\sigma = W_{hy} \cdot h^+ + b_0$$

Za slučaj kad nema učenika, na izlazu mora biti -2:

$$\sigma = -2$$

$$\underbrace{W_{hy} \cdot h^+}_{0} + b_0 = -2$$

$$b_0 = -2$$

S obzirom da je zadano da je vektor h u slučaju da nema učenika nul-vektor, želimo element vektora povećati na 1 ako je učenik s tim indeksom prisutan. To možemo postići ako je $W_{hh} = I$. U vektoru h^+ se mogu pružati i vrijednosti veće od 1, stoga je prikladna prijenosna funkcija $f = \min(h^+, 1)$. Kako bi prebrojali učenike, vektor W_{hy} mora biti jedinični vektor.

$$h^+ = \min(h^{+-1} + x^+, 1)$$

$$\sigma = [1 \ 1 \ 1] \cdot h^+ - 2$$

cini mi se da postoji i jednostavnije rješenje:

$$W_{hh} = I$$

$$W_{hy} = [1 \ 1 \ 1]$$

$$b_h = [0 \ 0 \ 0]$$

$$b_o = -2$$

$$f(x) = x$$

ako netko može potvrditi bilo bi topčina \rightarrow nije dobro jer ne uzima u obzir ako se salju isti učenici više puta, $f(x)$ mora biti min

par pro tipova za ovakve zadatke:

- ak zelis ograniciti nesto odozgo - max(a, x)

- ak zelis ograniciti nesto odozdo - min(a, x)

Rok 2016/17

6. veljače 2017.

Zavod za elektroniku, mikroelektroniku,
računalne i inteligentne sustave

Duboko učenje
pismeni ispit

1. Razmatramo klasifikacijsku konvolucijsku mrežu za jednodimenzionalne podatke. Slojevi mreže su sljedeći:

F: konvolucijski sloj bez nadopunjavanja (dvije jezgre, korak 1, aktivacija ReLU);
 g: globalno sažimanje maksimumom;
 s: potpuno povezani sloj s dva izlaza i aktivacijom ReLU
 p: softmaks.

Početni parametri konvolucijskog sloja su zadani vektorima težina $w_{1A} = [-0.5 \ 0.5]$ i $w_{1B} = [0.5 \ 0.5]$ te pomakima $b_{1A} = 0$ i $b_{1B} = -0.5$. Početni parametri potpuno povezanog sloja zadani su matricom težina $\mathbf{W}_2 = \mathbf{I}$ i pomakom $\mathbf{b}_2 = \mathbf{0}$.

(a) Napišite jednadžbe modela kojeg implementira mreža.
 (b) Odredite izlaz mreže za ulazne podatke $\mathbf{x}_1 = [0 \ 1 \ 1 \ 0]$ i $\mathbf{x}_2 = [1 \ 1 \ 1 \ 0]$.
 (c) Skicirajte kod u Tensorflowu koji računa gubitak unakrsne entropije za \mathbf{x}_1 i \mathbf{x}_2 , ako su poznati razredi podataka $y_1=0$ i $y_2=1$.
 Pomoć: za konvoluciju koristite funkciju `tf.nn.conv1d(value, filters, stride, padding)`. Parametar `value` je 3D tenzor s dimenzijama broj podataka \times veličina vektora \times broj vektora značajki. Parametar `filters` je 3D tenzor s dimenzijama širina filtra \times broj ulaznih mapa \times broj izlaznih mapa. Uzmite u obzir da ova funkcija korelira podatke s *nezrcaljenim* filtrima (jednako kao i ostale Tensorflowove funkcije za konvoluciju).

(10 bodova)

1. Moj pokusaj, treba provjeriti (pogotovo b) to nisam niti siguran sto se ocekuje pa sam improvizirao

1

① $W_{1A} = \begin{bmatrix} -0.5 & 0.5 \\ 0 & 0 \end{bmatrix}$ $W_{1B} = \begin{bmatrix} 0.5 & 0.5 \\ 0 & -0.5 \end{bmatrix}$ $W_2 = I \quad b=0$

 $y_1 = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix}$

\downarrow conv

 $\begin{bmatrix} [1 & 0.5 & 0 & -0.5] \\ [0 & 0.5 & 0] \end{bmatrix}$

\downarrow relu

 $\begin{bmatrix} [0.5 & 0 & 0] \\ [0 & 0.5 & 0] \end{bmatrix}$

\downarrow max pool

 $\begin{bmatrix} [0.5] \\ [0.5] \end{bmatrix}$

$\downarrow W = J$

 $\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$

\downarrow relu

 $\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$

\downarrow conv

 $y_2 = \begin{bmatrix} 1 & 1 & 1 & 0 \end{bmatrix}$

\downarrow conv

 $\begin{bmatrix} [0 & 0 & -0.5] \\ [0.5 & 0.5 & 0] \end{bmatrix}$

\downarrow relu

 $\begin{bmatrix} [0 & 0 & 0] \\ [0.5 & 0.5 & 0] \end{bmatrix}$

\downarrow max pool

 $\begin{bmatrix} [0] \\ [0.5] \end{bmatrix}$

$\downarrow \cdot W$

 $\begin{bmatrix} 0 \\ 0.5 \end{bmatrix}$

\downarrow relu

 $\begin{bmatrix} 0 \\ 0.5 \end{bmatrix}$

a) Jednostavno modela: $conv_{-1} = X * W_{1A} + b_{1A}$
 $conv_{-2} = X * W_{1B} + b_{1B}$
 $net = \begin{bmatrix} max(\text{relu}(conv_{-1})) \\ max(\text{relu}(conv_{-2})) \end{bmatrix}$
 $net = \text{relu}(net \cdot W_2 + b_2)$

c) PyTorch

net = nn.Sequential(

nn.Conv2d(in_channels=1, out_channels=1, kernel_size=(1, 4), stride=1, padding=0),
nn.ReLU(),
nn.MaxPool2d(),
nn.Flatten(),
nn.Linear(in_features=2, out_features=1))

1. $x = \text{data}$
 $out = \text{net}(x)$
 $diff = out - y$
 $diff = diff \cdot diff$
 $entropy = \text{sum}(diff)$

Izgleda dobro, no fali softmax na izlazu mreže. Također, pretpostavljam da su htjeli da koristimo njihovu nomenklaturu **F g s p** prilikom opisivanja mreže.

/

2.

2. Razmatramo sloj dubokog modela koji ulazni vektor q preslikava u izlazni vektor p , a može se opisati sljedećim jednadžbama:

$$p_1 = w_1 \cdot q_1 \cdot q_2 + w_2 \cdot \max(q_1, q_2)$$
$$p_2 = w_1 \cdot q_1^2 + w_2 \cdot q_2^2$$

Napišite jednadžbe derivacija izlaza po ulazima, te derivacije izlaza po parametrima tog sloja.

Predložite implementaciju razreda koji bi podržao učenje parametara zadanog sloja kao što smo radili u drugoj laboratorijskoj vježbi. Razred treba imati sljedeće metode: `forward(self, inputs)`, `backward_inputs(self, grads)` i `backward_parameters(self, grads)`. U rješenju možete pretpostaviti da je veličina mini-grupe 1.

(10 bodova)

2. (ovako nesto, neka netko provjeri ako moze)

NekocBraca: Mislim da `backwards_input` nije dobar. Vektor inputa je dimenzija 1×2 pa bi takve trebale biti i dimenzije gradijenata (ako ne grijesim). Mislim da se gradijenti $dp1_q1$ i $dp2_dq1$ trebaju zbrojiti i onda dobijes gradijent za $q1$ (i naravno pomnoziti s `grads`).

Sto se tice samih derivacija, tu se vise manje slazem. Jedino kod `dp2_dq` ne bi li trebao imati vodecu 2 (derivacija od $q^2 = 2q$)?

blackbean: U pravu si, ispravit ću. Hvala! :)

```
import numpy as np
```

```
class Model:  
    def __init__(self):  
        self.w1 = np.random.randn()  
        self.w2 = np.random.randn()  
  
    def forward(self, inputs):  
        self.q1 = inputs[0]  
        self.q2 = inputs[1]  
        p1 = self.w1 * self.q1 + self.w2 * np.maximum(self.q1 * self.q2)  
        p2 = self.w1 * self.q1 * self.q1 + self.w2 * self.q2 * self.q2  
        return np.array([self.p1, self.p2])  
  
    def backward_inputs(self, grads):  
        dp1_dq1 = self.w1 * self.q2 + int(self.q1 > self.q2) * self.w2  
        dp1_dq2 = self.w1 * self.q1 + int(self.q1 < self.q2) * self.w2  
        dp2_dq1 = 2 * self.w1 * self.q1  
        dp2_dq2 = 2 * self.w2 * self.q2  
        return grads * np.array([dp1_dq1 + dp2_dq1, dp1_dq2 + dp2_dq2])  
  
    def backward_parameters(self, grads):  
        dp1_dw1 = self.q1 * self.q2  
        dp1_dw2 = np.maximum(self.q1, self.q2)  
        dp2_dw1 = self.q1 * self.q1  
        dp2_dw2 = self.q2 * self.q2  
        return grads * np.array([dp1_dw1 + dp2_dw1, dp1_dw2 + dp2_dw2])
```


3.

3. Zadana je potpuno povezana slojevita umjetna neuronska mreža arhitekture $3 \times 2 \times 1$ koja obavlja preslikavanje $(x_1, x_2, x_3) \rightarrow y$. Neuroni u skrivenom i izlaznom sloju imaju prijenosnu funkciju ReLU. Početno, sve težine inicijalizirane su na vrijednost 1 (uključivo i pragovi). Skup uzoraka za učenje sadrži dva uzorka: $(2, -1, 1) \rightarrow 5$, $(1, 0, 2) \rightarrow 7$. Mrežu učimo postupkom ADAM uz hiperparametre $\rho_1 = \rho_2 = 0.5$, $\eta = 1$, $\delta = 0$. Prilikom učenja, jedini promjenjivi parametri su težina između x_1 i prvog neurona skrivenog sloja te težina između prvog neurona skrivenog sloja i izlaznog neurona.

- (a) Odredite simbolički izraz za gradijent funkcije pogreške, ako se kao pogrešku koristi srednje kvadratno odstupanje.
(b) Provode dvije iteracije postupka učenja, pri čemu se u svakoj iteraciji koristi srednji skup uzoraka za učenje te funkcija pogreške iz (1).

(10 bodova)

Aj pliz jel zna netko ovo?

ima li netko rjesenje ovoga?

/

4.

4. Razmatramo problem jedičnog modela na razini slova iz laboratorijske vježbe. Problem rješavamo običnom povratnom mrežom koja u povratnoj vesti ima prijenosni funkciju tangens hiperbolni, a na izlazu funkciju softmax. Veličina vokabulara je 80, a veličina skrivenog sloja 200. Odredite dimenzije svih parametara povratne neuroneke mreže.
(4 boda)

72280 kaj?

mislim da treba prepostaviti da koristis one hot enkodiranje pa ti je onda reprezentacija 80-dim

1. sloj:

W_{xh} - 200×80

W_{hh} - 200×200

W_{yh} - 200×80

b_h - 200×1

b_o - 80×1

5.

5. Razmatramo običnu povratnu neuronsku mrežu koja u povratnoj vezi ima prijenosnu funkciju $\min(1, x)$, a na izlazu identitet bez nelinearnosti. Zadatak mreže je pobrojati jedinice i nule u binarnom broju pri čemu nam na ulaz dolazi proizvoljno dugačak niz jedinica i nula u one-hot reprezentaciji. Prijenosna matrica mreže je jedinična $\mathbf{W} = \mathbf{I}$, početno stanje mreže je $\mathbf{h}_0 = [0 \ 0]$, a za preostale parametre mreže t_i (tj. za elemente matrica \mathbf{U} i \mathbf{V} te vektora \mathbf{b} i \mathbf{c}) vrijedi: $|t_i| \leq 1$ i $t_i \in \mathbb{Z}$. Odredite vrijednosti ostalih parametara mreže.
(6 bodova)

$$\mathbf{U} = [-1]$$

$$\mathbf{V} = [-1]$$

$$\mathbf{b_h} = [0, 0]$$

$$\mathbf{h_y} = [0, 0]$$

$$\mathbf{Whh} = \mathbf{Why} = -1 * (\text{identitet matrica})$$

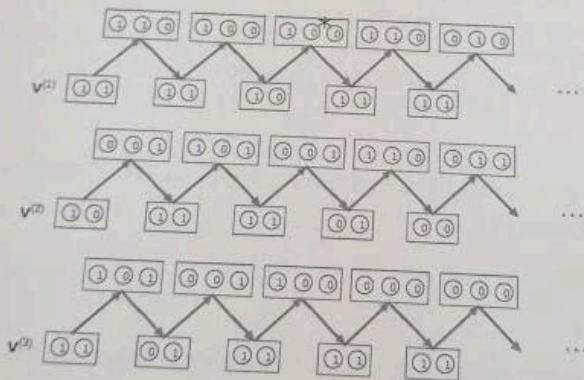
$$\mathbf{bh} = \mathbf{b0} = [0 \ 0]^T$$

6.

6. Razmatramo ograničeni Boltzmannov stroj s dva elementa u vidljivom i tri elementa u skrivenom sloju. Gibbsovo uzorkovanje za mini-grupu od tri uzorka $v^{(1)}$, $v^{(2)}$ i $v^{(3)}$ prikazano je na slici.

Zadani su koeficijent učenja $\eta=0.5$, trenutna matrica težina $W = \begin{bmatrix} 1/6 & -1/6 & 0 \\ 1/6 & -1/6 & 0 \end{bmatrix}^T$ te vektori pomaka vidljivog sloja $a = [1/6 \ 1/6]$ i skrivenog sloja $b = [1/6 \ 1/6 \ 0]$.

- Odredite vjerojatnosti $p(h_j = 1)$ za svaki element skrivenog sloja označenog zvjezdicom (*).
- Odredite novu matricu težina W i nove vektore pomaka a i b nakon jedne iteracije algoritma CD-2.



bodova)

- Vidim da nema rjesenja, mislim da se samo treba provest jedan step CD-a. U v2 imamo da je to v2 = [1, 0] s W gornji i dodajemo bias b. To ispade da je h2_probs = [2 / 6, 2/6, 0]. EkvivalentniT kod u pytorchu je:

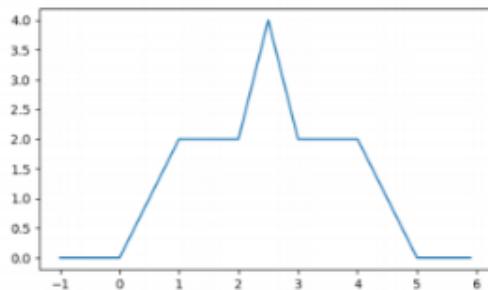
Treba još [2/6, 2/6, 0] provesti kroz sigmoidu.

`h2_prob = sigmoid(torch.mm(v2, self.W) + self.h_bias)`

`h2 = h2_prob.bernoulli() # ovo zapravo vraca gornji [1, 0, 0]
dok mi h2_prob tražimo`

MI 2017/18

1. Konstruirajte dvorazinski regresijski model koji kombinira affine transformacije i zglobnicu da bi aproksimirao funkciju zadatu slikom.



1. $h_1 = \max(2(x-0), 0)$
- $h_2 = \max(2(x-1), 0)$
- $h_3 = \max(4(x-2), 0)$
- $h_4 = \max(8(x-2.5), 0)$
- $h_5 = \max(4(x-3), 0)$
- $h_6 = \max(2(x-4), 0)$
- $h_7 = \max(2(x-5), 0)$

$$h = h_1 - h_2 + h_3 - h_4 + h_5 - h_6 + h_7$$

Kako se ovo određuje?

Želiš da ti svi h -ovi u zbroju opisuju ovaj gore graf. Problem je kad definiraš funkciju, onda ima početnu točku, ali ide u beskonačnost. Uzmimo na početku h_1 . Ona točno opisuje graf za x od 0 do 1 i kad bi stavili samo h_1 bez ostalih, onda bi on nastavio taj pravac u beskonačno. Ono što mi onda želimo s h_2 je da za x od 1-2 napravimo da je $h_1 - h_2 = 2$ (konstanta). Sada ako ostavimo samo $h_1 - h_2$ imamo graf koji izgleda kao ovaj gore osim što za $x > 2$ ukupni h je jednak i dalje 2, zato na $x = 2$ želimo dodati h_3 . Tako postepeno dodajemo i oduzimamo h_n funkcije dodavajući ih tako da su im početne točke na prelomnim mjestima ($0, 1, 2, 2.5, 3, 4, 5$) i s odgovarajućim brojem prije ($x-n$) opisujuemo točno graf za svaku od dionica.

Jedan od savjeta ako ne znate koji broj bi stavili ispred ($x-n$) je da izračunate za neki x iz idućeg intervala i vidite koliko trebate dodati/oduzete da dobijete istu vrijednost kao što je prikazano na grafu gore.

/

2.

2. Napišite jednadžbu gubitka negativne log-izglednosti za grupu od N podataka. Izvedite izraz za gradijente tog gubitka s obzirom na ulaz softmаксa u svim podatcima grupe.

Unakrsnu entropiju predikcije modela u podatku \mathbf{x}_i s obzirom na željenu distribuciju \mathbf{y}_i možemo izraziti jednadžbom: $\mathcal{L}_{\text{CE}} = -\sum_{j=1}^C y_{ij} \log p(Y = y_{ij} | \mathbf{x}_i)$. Odredite gradijent gubitka \mathcal{L}_{CE} s obzirom na ulaz softmаксa u podatku \mathbf{x}_i .

Jel ima itko ovo riješeno?

/

3.

3. Razmatramo klasifikacijski duboki model koji podatak \mathbf{x} preslikava u distribuciju \mathbf{p} :

$$\begin{aligned}\mathbf{h}_A &= \text{ReLU}(\mathbf{W}_A \cdot \mathbf{x} + \mathbf{b}_A) \\ \mathbf{s} &= \mathbf{W}_B \cdot \mathbf{h}_A + \mathbf{b}_B \\ \mathbf{p} &= \text{softmax}(\mathbf{s})\end{aligned}$$

- Odredite izlaze mreže ako je na ulazu zadan podatak $(2,4)$, a početni parametri su: $\mathbf{W}_A = [[-1,1]], \mathbf{b}_A=[-1], \mathbf{W}_B=[[1],[-1]] \cdot \ln(2), \mathbf{b}_B=[[1],[1]]$
- odredite gubitak negativne log-izglednosti ako je podatak označen distribucijom $y=(0,1)$;
- odredite gradijente gubitka negativne log-izglednosti s obzirom na težine \mathbf{W}_A i \mathbf{W}_B ;
- odredite parcijalnu derivaciju $\partial \mathbf{s} / \partial \text{vec}(\mathbf{W}_B)$ i komentirajte svrsishodnost njenog korištenja u praksi; napomena: $\text{vec}(\mathbf{W}_B)$ označava vektor sa svim elementima tenzora \mathbf{W}_B ;
- komentirajte zalihost modela: koji dio modela bi se mogao izostaviti bez utjecaja na kapacitet?

provjerite molim:

- $[[0.44], [0.66]] // [[0.8],[0.2]]$
- b)
- $0.4155 // 1.6094$
- $\mathbf{Wb} = [[0.8], [-0.8]], \mathbf{Wa} = [[2.218, 4.436]]$
- $d\mathbf{s}/d\text{vec}(\mathbf{W}) = [[\mathbf{ha}, 0], [0, \mathbf{ha}]], \mathbf{ha} = 1$
- zna li netko e)?

4.

-
4. Razmatramo konvolucijski duboki model koji 2D tenzor \mathbf{x} preslikava u vjerojatnost p :

$$\begin{aligned} \mathbf{h}_A &= \text{ReLU}(\text{conv}(\mathbf{x}, \mathbf{w}_A) + \mathbf{b}_A) \\ \mathbf{h}_B &= \text{conv}(\mathbf{h}_A, \mathbf{w}_B) + \mathbf{b}_B \\ s &= \text{GMP}(\mathbf{h}_B) \\ p &= \sigma(s \cdot \ln(2)) \end{aligned}$$

Pri tome ReLU označava zglobnicu, conv – konvoluciju bez nadopunjavanja, a GMP – globalno sažimanje maksimalnom vrijednošću. Tenzor kojeg proizvodi GMP ima prostorne dimenzije 1x1, dok broj kanala izlaza odgovara broju kanala ulaza.

Tenzor \mathbf{w}_A ima oblik 2x1x3x3. Prve dvije dimenzije odgovaraju broju kanala izlaza i ulaza. Posljednje dvije dimenzije odgovaraju retcima i stupcima po kojima provodimo konvoluciju i sažimanje. Početne vrijednosti elemenata \mathbf{w}_A su: $\mathbf{w}_{A0010}=-1$, $\mathbf{w}_{A0011}=1$, $\mathbf{w}_{A1021}=-1$, $\mathbf{w}_{A1011}=1$, a svi ostali elementi su 0.

Tenzor \mathbf{w}_B ima oblik 1x2x1x1 te jednaku organizaciju kao i \mathbf{w}_A . Početne vrijednosti elemenata \mathbf{w}_B su: $\mathbf{w}_{B0000} = -1$, $\mathbf{w}_{B0100} = 1$, a svi ostali elementi su 0.

Početne vrijednosti svih elemenata pomaka \mathbf{b}_A i \mathbf{b}_B su 0.

Ulazni tenzor \mathbf{x} ima oblik 4x4, $\mathbf{x}_{11} = \mathbf{x}_{12} = \mathbf{x}_{22} = 1$, a svi ostali elementi su 0. Točan razred podatka $y=1$.

Odredite izlaz modela te izračunajte gradijente negativne log-izglednosti s obzirom na elemente konvolucijske jezgre \mathbf{w}_B .

Svi indeksi počinju od nule.

BONUS: izračunajte gradijente negativne log-izglednosti s obzirom na elemente konvolucijske jezgre \mathbf{w}_A .

/

5.

5. Oblikujte funkciju koja implementira optimizacijski algoritam ADAM. Napišite ispitni kod koji uz pomoć te funkcije pronalazi minimum funkcije $x^4 - 4x$ tijekom 10000 iteracija, počevši iz $x=-2$. Zadatak riješite prema sljedećim uputama.
- Neka vas ne zbuni što se nigdje ne spominju podatci: njih smo izostavili kako bi zadatak bio jednostavniji;
 - tražena funkcija treba raditi samo za skalarne funkcije skalarne varijable;
 - neka argumenti funkcije budu:
 - početna vrijednost parametra (`theta`),
 - funkcija koja računa gradijent (`fgrad`), te
 - broj iteracija (`n`);
 - neka povratna vrijednost funkcije bude konačna vrijednost parametra;
 - nemojte komplikirati: naša implementacija ima 20 redaka Pythona od čega 5 redaka otpada na inicijalizaciju hiperparametara.

5.

```
import numpy as np

def perform_adam(theta, fgrad, n):
    r = 0
    s = 0
    fi_1 = 0.9
    fi_2 = 0.999
    t = 0
    eta = 0.001
    delta = 1e-8

    for i in range(n):
        g = fgrad(theta)
        s = fi_1*s + (1-fi_1)*g
        r = fi_2*r + (1-fi_2)*g*g
        t += 1
        s_ = s / (1-np.power(fi_1, t))
        r_ = r / (1-np.power(fi_2, t))
        delta_theta = - eta * s_ / (np.sqrt(r) + delta)
        theta += delta_theta

    return theta

def gradient(x):
    return 4*np.power(x, 3)-4

result = perform_adam(-2, gradient, 10000)
print(result)
```

d

6.

6. Razmatramo sloj dubokog modela koji ulazni vektor \mathbf{p} transformira u izlazni vektor \mathbf{q} koristeći skalarne parametre w_0 , w_1 i w_2 , a može se opisati jednadžbom: $q_i = w_0 \cdot p_{i-1} + w_1 \cdot p_i + w_2 \cdot p_{i+1}$. Odredite Jakobijan gubitka po ulazu i Jakobijan gubitka po parametrima ako je poznat Jakobijan gubitka po izlazu.

Napišite kod koji bi omogućio uklapanje sloja u duboki model proizvoljne složenosti. Sloj izrazite razredom koji implementira sučelje Layer iz prvog zadatka druge laboratorijske vježbe te implementira sljedeće metode: `forward(self, inputs)`, `backward_inputs(self, grads)`, te `backward_parameters(self, grads)`.

Izvedba treba osigurati da dimenzionalnost izlaznog vektora bude jednaka dimenzionalnosti ulaznog vektora.

Pomoć: numpyjev tenzor x proizvoljne dimenzionalnosti možemo nadopuniti nulama sljedećim pozivom: `np.lib.pad(x, 1, mode='constant')`.

ZI 2017/18

20. lipnja 2018.

Zavod za elektroniku, mikrelektroniku,
računalne i inteligentne sustave

Duboko učenje završni ispit

1. (10 bodova) Razmatramo klasifikacijski konvolucijski model koji je prilagođen obradi sivih slika dimenzija 5×5 . Arhitektura modela je:

- konvolucijski sloj bez nadopunjavanja: dvije jezgre 3×3 bez pomaka, korak 1, aktivacija ReLU;
- globalno sažimanje maksimumom;
- softmaks.

Konvolucijske jezgre inicijalizirane su kako slijedi: $\mathbf{w}_0 = -\mathbf{w}_1 = 0.5 \cdot \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix}$. Točne oznake podataka su $\mathbf{y}_0 = [1, 0]^\top$ i $\mathbf{y}_1 = [0, 1]^\top$.

Neka su na ulazu slike \mathbf{I}_0 i \mathbf{I}_1 čiji su pikseli $\mathbf{I}_0[2,1]$, $\mathbf{I}_0[2,2]$, $\mathbf{I}_0[2,3]$ te $\mathbf{I}_1[1,2]$, $\mathbf{I}_1[2,2]$, $\mathbf{I}_1[3,2]$ postavljeni na 1, dok su svi ostali pikseli postavljeni na 0 (indeksi kreću od nule).

Zadatci:

- Odredite dimenzije svih aktivacijskih tenzora te broj parametara modela.
- Odredite gradijente svih parametara u podatcima \mathbf{I}_0 i \mathbf{I}_1 .

(10 bodova)

Ima netko ovaj !!!?!?!!!?!?!?!

Pod a) ima primjer ranije

Jel bi ovo trebalo prvo napraviti forward za \mathbf{I}_0 , pa izračunati gradijente, pa za \mathbf{I}_1 , pa ponovo gradijente? Ili paralelno?

2. Optimizacijski algoritmi

2. Napišite jednadžbe optimizacijskih algoritama SGD, SGD s momentom, RMS prop i ADAM. Navedite memoriske zahtjeve za svaki od tih algoritama u ovisnosti o broju parametara n. Raspišite prve dvije iteracije minimizacije polinoma $f(x) = x^2 - 4x + 2$ algoritmom ADAM počevši od $x=0$. Poznati su sljedeći hiper-parametri postupka: $\rho_1=\rho_2=0.9$, $\delta=0$, te korak $\varepsilon=0.1$. (10 bodova)

SGD

Algorithm 8.1 Stochastic gradient descent (SGD) update at training iteration k

Require: Learning rate ϵ_k .

Require: Initial parameter θ

while stopping criterion not met **do**

 Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$.

 Compute gradient estimate: $\hat{\mathbf{g}} \leftarrow +\frac{1}{m} \nabla_{\theta} \sum_i L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)})$

 Apply update: $\theta \leftarrow \theta - \epsilon \hat{\mathbf{g}}$

end while

Broj parametara: learning_rate

Memorijski zahtjevi: 1 -> ne bi li u svakoj iteraciji trebalo mjesto u memoriji za g tj. n + 1?

SGD s momentom

$$\nu \leftarrow \alpha \cdot \nu - \epsilon \cdot \nabla_{\theta} \left(\frac{1}{m} \sum_{i=1}^m L(f(\mathbf{x}^{(i)}; \Theta), y^{(i)}) \right)$$

$$\theta \leftarrow \theta + \nu$$

Broj parametara: alpha, epsilon, v(n)

Memorijski zahtjevi: n -> isto ko u prethodnom pa bi bilo $2n + 1$?

al cini mi se ak je tak da se gotovo svi onda mijenjaju?

AdaGrad

Algorithm 8.4 The AdaGrad algorithm

Require: Global learning rate ϵ

Require: Initial parameter θ

Require: Small constant δ , perhaps 10^{-7} , for numerical stability

Initialize gradient accumulation variable $r = \mathbf{0}$

while stopping criterion not met **do**

 Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$.

 Compute gradient: $\mathbf{g} \leftarrow \frac{1}{m} \nabla_{\theta} \sum_i L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)})$

 Accumulate squared gradient: $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{g} \odot \mathbf{g}$

 Compute update: $\Delta\theta \leftarrow -\frac{\epsilon}{\delta + \sqrt{\mathbf{r}}} \odot \mathbf{g}$. (Division and square root applied element-wise)

 Apply update: $\theta \leftarrow \theta + \Delta\theta$

end while

Broj parametara: epsilon, delta, r(n)

Memorijski zahtjevi: n

RMSProp

Algorithm 8.5 The RMSProp algorithm

Require: Global learning rate ϵ , decay rate ρ .

Require: Initial parameter θ

Require: Small constant δ , usually 10^{-6} , used to stabilize division by small numbers.

Initialize accumulation variables $r = 0$

while stopping criterion not met **do**

 Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$.

 Compute gradient: $\mathbf{g} \leftarrow \frac{1}{m} \nabla_{\theta} \sum_i L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)})$

 Accumulate squared gradient: $\mathbf{r} \leftarrow \rho \mathbf{r} + (1 - \rho) \mathbf{g} \odot \mathbf{g}$

 Compute parameter update: $\Delta\theta = -\frac{\epsilon}{\sqrt{\delta + \mathbf{r}}} \odot \mathbf{g}$. ($\frac{1}{\sqrt{\delta + \mathbf{r}}}$ applied element-wise)

 Apply update: $\theta \leftarrow \theta + \Delta\theta$

end while

Broj parametara: epsilon, ro, delta, r(n)

Memorijski zahtjevi: n

RMSProp + Nesterov moment

Algorithm 8.6 RMSProp algorithm with Nesterov momentum

Require: Global learning rate ϵ , decay rate ρ , momentum coefficient α .

Require: Initial parameter θ , initial velocity v .

Initialize accumulation variable $r = \mathbf{0}$

while stopping criterion not met **do**

 Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$.

 Compute interim update: $\tilde{\theta} \leftarrow \theta + \alpha v$

 Compute gradient: $\mathbf{g} \leftarrow \frac{1}{m} \nabla_{\theta} \sum_i L(f(\mathbf{x}^{(i)}; \tilde{\theta}), \mathbf{y}^{(i)})$

 Accumulate gradient: $r \leftarrow \rho r + (1 - \rho) \mathbf{g} \odot \mathbf{g}$

 Compute velocity update: $v \leftarrow \alpha v - \frac{\epsilon}{\sqrt{r}} \odot \mathbf{g}$. ($\frac{1}{\sqrt{r}}$ applied element-wise)

 Apply update: $\theta \leftarrow \theta + v$

end while

Broj parametara: epsilon, ro, alpha, v(n), r(n)

Memorijski zahtjevi: 2n

Adam

Algorithm 8.7 The Adam algorithm

Require: Step size ϵ (Suggested default: 0.001)

Require: Exponential decay rates for moment estimates, ρ_1 and ρ_2 in $[0, 1]$.
(Suggested defaults: 0.9 and 0.999 respectively)

Require: Small constant δ used for numerical stabilization. (Suggested default: 10^{-8})

Require: Initial parameters θ

Initialize 1st and 2nd moment variables $s = \mathbf{0}$, $r = \mathbf{0}$

Initialize time step $t = 0$

while stopping criterion not met **do**

 Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$.

 Compute gradient: $\mathbf{g} \leftarrow \frac{1}{m} \nabla_{\theta} \sum_i L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)})$

$t \leftarrow t + 1$

 Update biased first moment estimate: $s \leftarrow \rho_1 s + (1 - \rho_1) \mathbf{g}$

 Update biased second moment estimate: $r \leftarrow \rho_2 r + (1 - \rho_2) \mathbf{g} \odot \mathbf{g}$

 Correct bias in first moment: $\hat{s} \leftarrow \frac{s}{1 - \rho_1^t}$

 Correct bias in second moment: $\hat{r} \leftarrow \frac{r}{1 - \rho_2^t}$

 Compute update: $\Delta\theta = -\epsilon \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$ (operations applied element-wise)

 Apply update: $\theta \leftarrow \theta + \Delta\theta$

end while

Broj parametara: epsilon, ro1, ro2, delta, t, s(n), r(n)

/

Memorijski zahtjevi: 2n

$$\begin{aligned}
 p_1 = p_2 &= 0.9 \\
 s &= 0 \\
 r &= 0.1 \\
 f(x) &= x^2 - sx + r \\
 x_0 &= 0 \\
 f'(x) &= 2x - s
 \end{aligned}$$

1. korak $s=0, t=0, r=0,$

$$\begin{aligned}
 \frac{df}{dx} \Big|_{x=0} &= 2 \cdot 0 - s = -s = g \\
 t' &= 1 \\
 s' &= 0.9s + 0.1g = -0.4 \\
 r' &= 0.9r + 0.1g + 0g = 1.6 \\
 \hat{s} &= \frac{s}{1-p_1^t} = \frac{-0.4}{1-0.9^1} = -4 \\
 \hat{r} &= \frac{r}{1-p_2^t} = \frac{1.6}{0.1} = 16 \\
 \Delta x &= -\epsilon \frac{\hat{s}}{\sqrt{\hat{r}' + \hat{s}}} = -0.1 \frac{-4}{\sqrt{16.42 + 0}} = 0.1 \\
 x' &= x + \Delta x = 0.1
 \end{aligned}$$

2. korak $s=-0.4, r=1.6, t=1$

$$\begin{aligned}
 g &= -3.8 \\
 t' &= t+1 = 2 \\
 s' &= 0.9 \cdot (-0.4) + 0.1 \cdot (-3.8) = -0.74 \\
 r' &= 0.9 \cdot 1.6 + 0.1 \cdot 16.44 = 2.74 \\
 \hat{s} &= \frac{-0.74}{1-0.81} = -3.89 \\
 \hat{r} &= \frac{2.74}{0.19} = 14.42 \\
 \Delta x &= -0.1 \frac{-3.89}{\sqrt{14.42 + 0}} = 0.102 \\
 x' &= x + \Delta x = 0.202
 \end{aligned}$$

Sad nisam siguran, ali moguce da je 2. korak krivi jer kada računaš vrijednost od s i r , uzimaš s_{kappa} i r_{kappa} , umjesto obične s i r .

Može li tko ovo provjeriti?

Meni r' u drugom ispadne 2.884 -> i meni i onda theta na kraju ispadne 0.199966

/

2. korak mi ispada:

$$s = -0.74$$

$$r = 2.884$$

$$s_{\hat{}} = -0.74/(1-0.9^2) = -3.895$$

$$r_{\hat{}} = -2.884/(1-0.9^2) = 15.17894$$

$$\delta = -0.1 * (-3.895)/(15.17894^{0.5}) = 0.0999$$

$$x = 0.1 + 0.0999 = 0.1999$$

/

3. Razmatramo učenje višerazredne logističke regresije $\hat{y} = \text{softmax}(\mathbf{W}\mathbf{x} + \mathbf{b})$ optimiranjem unakrsne entropije stohastičkim gradijentnim spustom uz zadanu stopu učenja $\delta = 1$. Zadan je sljedeći skup za učenje: $\mathbf{x}_0 = (2, 2)$, $\mathbf{y}_0 = (1, 0)$, $\mathbf{x}_1 = (-1, -1)$, $\mathbf{y}_1 = (0, 1)$. Početno stanje matrice težina i vektora pomaka zadano je s $\mathbf{W}^0 = \begin{bmatrix} 0.5 & -0.5 \\ 0.5 & -0.5 \end{bmatrix}$, $\mathbf{b}^0 = [0 \ 0]^\top$.

- Odredite jednadžbe gradijenata svih parametara.
- Provode jednu epohu učenja počevši od \mathbf{W}^0 i \mathbf{b}^0 , uz regularizaciju ispuštanjem (engl. dropout) ulaznih značajki s vjerojatnošću $p(\mu_{\mathbf{x}_i}) = 0.5, i = 0, 1$. Prepostavite sljedeći raspored ispuštanja:
 - prva mini-grupa je $(\mathbf{x}_0, \mathbf{y}_0)$, ugašena je dimenzija 0,
 - druga mini-grupa je $(\mathbf{x}_1, \mathbf{y}_1)$, ugašena je dimenzija 1.

Izračunajte gradijente i nove vrijednosti parametara.

- Nakon koraka učenja u prethodnom podzadatku, izračunajte izlaz naučenog modela za ulaz $\mathbf{x}_2 = (1, 0)$.

(10 bodova)

3.

Handwritten notes for problem 3:

$\hat{y} = \text{softmax}(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad \delta = 1$

$\mathbf{S} = \mathbf{W}\mathbf{x} + \mathbf{b}$

a) $\frac{\partial \mathcal{L}}{\partial \mathbf{S}} = \mathbf{P} - \mathbf{y}_{\text{obs}}$ $\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \frac{\partial \mathcal{L}}{\partial \mathbf{S}} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \frac{\partial \mathcal{L}}{\partial \mathbf{S}} \cdot \mathbf{X}^\top$

$\frac{\partial \mathcal{L}}{\partial \mathbf{b}} = \frac{\partial \mathcal{L}}{\partial \mathbf{S}} \cdot \underbrace{\frac{\partial \mathcal{L}}{\partial \mathbf{b}}}_{=1}$

b) $\mathbf{W}^0 = \begin{bmatrix} 0.5 & -0.5 \\ 0.5 & -0.5 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$

$\mathbf{S} = \begin{bmatrix} 0.5 & -0.5 \\ 0.5 & -0.5 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$

$\mathbf{y}_{\text{obs}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\frac{\partial \mathcal{L}}{\partial \mathbf{S}} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$

$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}$

$\mathbf{W}^1 = \mathbf{W}^0 - \delta \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & -1.5 \end{bmatrix}$

$\mathbf{b}^1 = \mathbf{b}^0 - \delta \frac{\partial \mathcal{L}}{\partial \mathbf{b}} = \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}$

1

2) $\vec{x} = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \quad y_{\text{obs}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

 $S = \begin{bmatrix} 0,5 & 0,5 \\ 0,5 & -1,5 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0,5 \\ -0,5 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \rightarrow \vec{y} = \begin{bmatrix} 0,73 \\ 0,27 \end{bmatrix}$
 $\frac{\partial f}{\partial s} = \begin{bmatrix} 0,73 \\ 0,27 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0,73 \\ -0,73 \end{bmatrix} \quad \frac{\partial f}{\partial w} = \begin{bmatrix} 0,73 \\ -0,73 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 \\ 0,73 & 0 \end{bmatrix} = \begin{bmatrix} -0,73 & 0 \\ 0,73 & 0 \end{bmatrix}$
 $\frac{\partial f}{\partial b} = \frac{\partial f}{\partial s} \cdot 1 \quad \hat{W} = \begin{bmatrix} 0,5 & 0,5 \\ 0,5 & -1,5 \end{bmatrix} - 1 \cdot \begin{bmatrix} -0,73 & 0 \\ 0,73 & 0 \end{bmatrix} = \begin{bmatrix} 1,23 & 0,5 \\ -0,23 & -1,5 \end{bmatrix}$
 $\vec{b} = \begin{bmatrix} 0,5 \\ -0,5 \end{bmatrix} - 1 \cdot \begin{bmatrix} 0,73 \\ 0,27 \end{bmatrix} = \begin{bmatrix} -0,23 \\ -0,77 \end{bmatrix}$

c)

 $\hat{W} = 0,5 \cdot W = \begin{bmatrix} 0,625 & 0,25 \\ -0,125 & -0,75 \end{bmatrix}$
 $S = \hat{W} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -0,23 \\ -0,77 \end{bmatrix} = \begin{bmatrix} 0,415 \\ -0,115 \end{bmatrix} + \vec{b} = \begin{bmatrix} 0,385 \\ -0,825 \end{bmatrix}$
 $\vec{y} = \begin{bmatrix} 0,78 \\ 0,22 \end{bmatrix}$

b) podzadatak mi se cini da bi b_2 trebalo biti:

$$\begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix} - \begin{bmatrix} 0.73 \\ -0.73 \end{bmatrix} = \begin{bmatrix} -0.23 \\ 0.23 \end{bmatrix}$$

ZASTO JE U C) POMNOZENO SA 0.5 PRIJE UNAPRIJEDNOG PROLAZA ?????? HVALA :)

neka intuicija je zato sto kod ucenja s vjerojatnoscu 0.5 gasis aktivacije pa je to kao "ocekivani" input kod faze ispitivanja. pogledaj si prezu regularizacija, slajd 47.

c) podzadatak bi isto trebao bit kriv zbog krivog b_2 iz b)

4.

4. Razmatramo običnu povratnu neuronsku mrežu s prijenosnom funkcijom tangens hiperbolni na povratnoj vezi te funkcijom softmax na izlazu.

Rješavamo problem jezičnog modela na razini slova iz laboratorijske vježbe, pri čemu nam je veličina vokabulara 10. Uz pretpostavku standardne preciznosti decimalnih brojeva od 4 bajta, odredite maksimalnu veličinu skrivenog sloja kako bi parametri modela stali u 624 bajta. Pri izračunu zanemarite vektore pristranosti b i c .

(4 bodova)

- Dimenzije parametara: $W_{hh} \in \mathbb{R}^{h \times h}$, $W_{xh} \in \mathbb{R}^{h \times d}$, $W_{hy} \in \mathbb{R}^{y \times h}$, pri čemu nam je y kardinalitet izlaza (npr. broj klasa)

AKO se koristi one hot enkodiranje za svako slovo onda:

$$4(10^*h + h^*h + 10^*h) = 624$$

$$h_1 = 6$$

ovo dolje je krivo. prema ovoj slici iz preze, W_{xh} je ima 10^*h težina, a ne 10 tezina.

Dakle $10^*h + h^*h \leq 156$

8.45 je jedno rjesenje za h (ako imamo = umjesto \leq) stoga je 8 maks velicina skrivenog sloja. Onda imamo 64 parametra u W_{hh} i 80 parametara u W_{xh} , ukupno 144. Ako povecamo h na 9, dobijemo vise od 156 pa jbg.

$$624/4 = 156$$

rijec je size 10 pa W_{xh} ima 10 tezina

W_{hh} je proizvoljne velicine, ali iste kao W_{hy}

$$146/2 = 73$$

odgovor je da skrivena moze bit max 73

paziti na ispitu ako bude bias

Ne bi li trebalo biti da je $W_{hx}=(10^*h)$, $W_{hh}=(h*h)$, $W_{hy}=(h*h)$?

//////////

/

Mislim da bi W_{hy} trebao biti iste dimenzije kao W_{hx} ?

Da

stavio sam sliku iz preze gore.

5. Razmatramo običnu povratnu neuronsku mrežu s identitetom u povratnoj vezi i bez izlaznog sloja, u kojoj su svi parametri skalari.

Ukoliko je poznato da je početno stanje $h^{(0)} = 64$, a ulazi u vremenskim koracima $x^{(1)} = z$, $x^{(2)} = z^2$ i $x^{(3)} = z^3$, odredite preostale parametre modela kako bi u skrivenom stanju mreže nakon trećeg koraka bila vrijednost polinoma $z + 2z^2 + 4z^3 + 1$.

(6 bodova)

$$\textcircled{5} \quad h^{(0)} = 64$$

$$x^{(1)} = z, \quad x^{(2)} = z^2, \quad x^{(3)} = z^3$$

$$\underline{h^{(3)} = z + 2z + 4z^3 + 1}$$

$$h^{(+)} = w_{hh} \cdot h^{(+-1)} + w_{xh} \cdot x^{(+)} + b_h$$

$$h^{(+)} = 64 w_{hh} + z w_{xh} + b_h$$

$$h^{(2)} = 64 w_{hh}^2 + z w_{hh} w_{xh} + w_{hh} b_h + z^2 w_{xh} + b_h$$

$$\begin{aligned} h^{(2)} &= 64 w_{hh}^2 + z w_{hh}^2 w_{xh} + w_{hh}^2 b_h + z^2 w_{hh} w_{xh} + w_{hh} b_h + z^3 w_{xh} + b_h \\ &= z + 2z^2 + 4z^3 + 1 \end{aligned}$$

$$\cancel{z^3 w_{xh}} = 4z^3$$

$$\boxed{w_{xh} = 4}$$

$$\boxed{h^{(+)} = \frac{1}{2} h^{(+-1)} + 4x^{(+)} - 4}$$

$$2z^2 = \cancel{z} w_{hh} w_{xh}$$

$$\boxed{w_{hh} = \frac{1}{2}}$$

$$\cancel{z} = \cancel{z} w_{hh}^2 w_{xh}$$

$$z = z \cdot \frac{1}{4} 4 w$$

$$1 = 64 w_{hh}^3 + w_{hh}^2 b_h + w_{hh} b_h + b_h$$

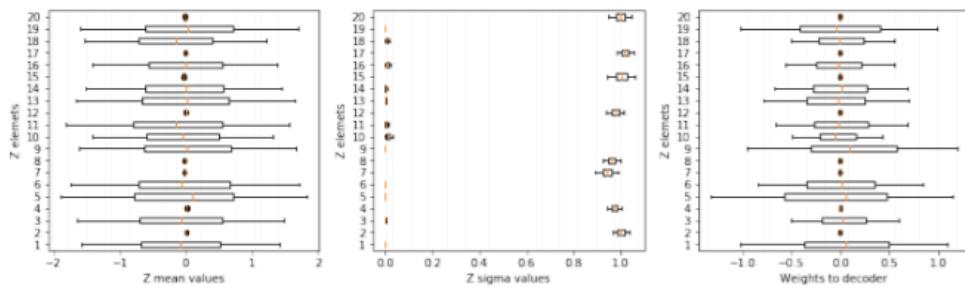
$$1 = 64 \cdot \frac{1}{8} + \frac{1}{4} b_h + \frac{1}{2} b_h + b_h$$

$$-7 = \frac{7}{4} b_h$$

$$\boxed{b_h = -4}$$

6. Razmatramo varijacijski autoenkoder (VAE).

- Opišite regularizacijsku komponentu funkcije cilja VAE (što je, na što utječe...).
- Opišite reparametrizacijski trik.
- Kako se zove efekt koji je jasno uočljiv na slici te čega je on posljedica? Na slici su vizualizirane distribucije od $\mu_z(\mathbf{x})$ i $\sigma_z(\mathbf{x})$ za 20 elemenata skrivenog sloja, za sve uzorke \mathbf{x} iz skupa za treniranje te od svih težina koje povezuju pojedini z sa dekoderom.



(10 bodova)

6. a) Regularizacijska komponenta funkcije cilja VAE je Kullback-Leibler divergencija između distribucija $q(z|x)$ i $p(z)$. Kad su te distribucije jednake, regularizacijski član jednak je nuli.

$$L(\theta, \phi, x^{(i)}) = -D_{KL}(q_\phi(z|x^{(i)}) \| p(z)) + E_{q_\phi(z|x^{(i)})} [\log(p_\theta(x^{(i)}|z))]$$

b) U svom originalnom obliku VAE uzorkuje iz slučajne varijable z određene modelom $p(z|\text{omega}, \mathbf{x})$. S obzirom da ne možemo propagirati pogrešku unatrag kroz nasumični čvor, rastavljamo z na zbroj determinističke i slučajne komponente, od kojih optimiramo samo determinističku.

c) Efekt se naziva **urušavanje komponenti (component collapsing)**. Događa se ako regularizacijski član uspije u svojoj namjeni i izjednači distribucije $q(z|x)$ i $p(z)$. U tom slučaju za svaki x vrijedi $q(z|x)=p(z)$. Takvi elementi z više ne nose nikakvu korisnu informaciju, pa ih dekoder niti ne upotrebljava.

**Ijudi imam osjecaj da cemo jebeno ovo napisat, svi cemo imat 5!!!
već sam predao papire za TVZ**

+1 za TVZ

:)

:)

:D

:')

#suza_za_palog_brata

/

Ima netko da sprema ovo sad za ljetni rok?

Ima ali ne znam koliko ce uspjesno bit

Ima li netko screenshot sa studioša kako se rješavaju oni zadaci s izračunom receptivnog polja?

imas donekle ok objasnjeno u videu sa zadatkom

<https://www.youtube.com/watch?v=BD8D5Dsx1y4&list=PLkOLgurQ4FfNPFTtKlccU91yFucsQGC4y&index=14>

tipa ako imas isti model koji on ima u videu:

slika uvijek ima 1x1 receptivno. kad mu lupis konvoluciju, svaka konvolucija se racuna iz 5 elemenata. relu ne mijenja nista i sad dolazi max pool. on se racuna iz 2 elementa, ali svaki od tih elemenata vidi po 5 elemenata iz originalne slike. posto ti je u toj konvoluciji stride bio jedan, kad si nacrtas vidjet ces da se preklapaju 4 elementa, a 2 se ne preklapaju i onda je to 6. poslije imas konvoluciju koja je opet s 5x5 filtrom pa sad ides opet na slican nacin gledat. svaki element u tom sloju se racuna na temelju 5 elemenata iz prethodnog. svaki u tom sloju se racuna kao max od 2 susjeda. tu vec razmatras 10 elemenata. opet u tih 10 gledas kako se oni racunaju a to je na temelju 5 elemenata iz ulazne slike. kad si nacrtas tih 10 vidjet ces da ih hrpa ima preklapanja, tj. ja taj dio obicno racunam kao da si pobrojim na temelju koliko se prvi racuna i onda brojim po jedan za sve ostale i onda to zbrojim ($5 + 9 = 14$). sljedeci pool opet ista prica: on se racuna na temelju 2 elementa iz prethodnog. U tom prethodnom gledas po 5 iz onog prije prethodnog (to ispadne 6 zbog preklapanja). Tih 6 se opet racuna na temelju 2 pa je to 12. na kraju tih 12 mozes racunat na istu onu foru da gledas koliko ti prvi uzima + po jedan za sve ostale ($5 + 11 = 16$). i onda logicno nakon flattenata imas potpuno povezani gdje sve aktivacije vide sve prethodne aktivacije pa je 28x28.

Hvala puno! :)

/

Btw postoji dosta jednostavna formula za računanje receptivnog polja u svakom sloju
šta sam iskopao iz dubina stackoverflowa:

$$r_0 = 1 + \sum_{l=1}^L \left((k_l - 1) \prod_{i=1}^{l-1} s_i \right)$$

where

- r_0 is the receptive field size after L convolutional / pooling layers,
- k_l is the kernel size of the convolutional / pooling operation at layer l ,
- and s_i is the stride rate at layer i .

provjeroeno funkcionira

Jesmo li preskočili išta iz dijela Optimizacije parametara modela?

jel ima netko mozda segviceve snimke s teamsa kak rjesava mi ili zi?

Jel na ovo misliš?

[https://drive.google.com/file/d/1YYppzRVisnclv94COSBV-ceF4wQePAm /view](https://drive.google.com/file/d/1YYppzRVisnclv94COSBV-ceF4wQePAm/view)

eee tocno to, hvala

ovogodisnji mi i zi, zajedno s teorijom, zahvale kolegi Stjepanu koji je to pretipkao u pdf:

https://www.mediafire.com/file/bw0rqcdi10fc02d/duboko_mi_teorija.pdf/file

https://www.mediafire.com/file/qslg0rijvwnvch0/duboko_zi_teorija.pdf/file

<https://www.mediafire.com/file/k9xs57mjd5x9dol/mi2022.pdf/file>

<https://www.mediafire.com/file/ypprly9xk7bzzbg/zi2022.pdf/file>

/

ispricavam se svima koji su sjebali pitalicu:

3. Razmatramo višerazrednu logističku regresiju s n značajki na ulazu. Ako prilikom učenja tog modela koristimo stohastičko izostavljanje značajki (dropout), jednim unaprijednim prolazom kroz tako naučeni model možemo dobiti:
- (a) aritmetičku sredinu predikcije $O(n^2)$ modela
 - (b) aritmetičku sredinu predikcije $O(n)$ modela
 - (c) aritmetičku sredinu predikcije $O(2^n)$ modela
 - (d) geometrijsku sredinu predikcije $O(2^n)$ modela**

u točnom odgovoru d) treba pisati: "geometrijsku sredinu predikcije $O(2^n)$ modela"

Hvala na ispitima!

jel zna netko zi 19/20 3. a)?

mozda konvolucija 3x3, sa strideom 3, 3 gdje je na (0, 0) element 1, ostalo 0, a fc gdje je bias -2.5, a W matrica 8x9 s tezinama:

[1 0 0 0 1 0 0 0 1]
[0 0 1 0 1 0 1 0 0]
[1 0 0 1 0 0 1 0 0]
[0 1 0 0 1 0 0 1 0]
[0 0 1 0 0 1 0 0 1]
[1 1 1 0 0 0 0 0 0]
[0 0 0 1 1 1 0 0 0]
[0 0 0 0 0 0 1 1 1]

ako moze netko potvrdit kak mu se ovo cini

Misli li se itko možda danas okušati u rješavanju ovogodišnjih ispita? Oćemo probati zajednički riješiti to? to bi bilo kul

/

**BOG NAM
POMOGAO
SUTRA!**

+1

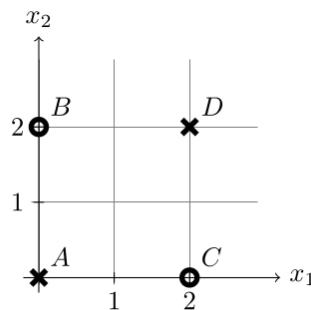
<3

Bome nam je pomogao! :)

MI 2021/22

- Dizajnirajte dvoslojni potpuno povezani model koji rješava problem sa slike. Zadani problem sadrži četir podatak koje je potrebno klasificirati u dva razreda: kruži i križi. Aktivacijska funkcija skrivenog sloja neka bude zglobnica, a izlaznog sloja sigmodia. Napišite sve parametre modela i pokažite da vaš model ispravno klasificira sve podatke pod pretpostavkom da granica klase definira vrijednost od 50%.

Pomoć: Razmotrite duboki model nad varijablama $z = x_2 - x_1$.



rj:

ovaj je zadatak odvratan, pogotovo ako ne znaš ono rjesenje iz primjera sa slajdova
 $W1 = [1 \ -1]$, $b1 = [2]$, $W2 = [1]$, $b2 = -1$

$$[1 \ -1] \quad [0] \quad [-2]$$

cini mi se da ovo rjesava zadatak, nazalost postupak mi je bio trial and error

- Napišite izravnu implementaciju funkcije koja obavlja unaprijedni prolaz i računa gubitak za višerazrednu logističku regresiju koristeći biblioteku numpy. Funkcija na ulazu prima jedan podatak x dimenzije (D, 1), prirodan broj y iz intervala [1, C] koji označava točni razred, te ostale potrebne parametre odgovarajući dimenzija (kojih?). Identificirajte dijelove vašeg koda u kojima može doći do numeričkih pogrešaka. Napišite robusnu implementaciju koja rješava navedene probleme.

Uputa: Obratite pažnju na mogući preljen u funkcijama exp i log.

rj:

def fp(x, w, b):

$$wx = \cancel{w} @ x + b$$

$$sfe = np.exp(wx - np.max(wx))$$

$$sfe = \cancel{sfe} / np.sum(sfe)$$

def loss(out, y):

$$\text{index} = np.argmax(y)$$

$$\text{element} = \cancel{\text{out}[index]} \quad \text{out}[index]$$

$$\text{return } -np.log(\text{element} + 10^{-8})$$

ovo smo frend i ja rješavali jučer, isto kao i zadatak ispod kojeg ću priložit, pa ako netko ima ideju što je krivo, nek samo nadopiše.

nema potrebe za np argmax(y) posto ti je on zadan kao oznaka iz [1, C], mozes samo pisati y - 1 za indeks, ostalo mi se sve cini ok

3. Zadana je sljedeća inicijalizacija modula, te odgovarajući unaprijedni prolaz. Na ulazu se nalazi jedna RGB slika dimenzija 32 x 32.

```

1 conv1 = nn.Conv2D(__, 16, kernel_size=3, stride=1, padding=0)
2 conv2 = nn.Conv2D(__, 24, kernel_size=3, stride=1, padding=0)
3 maxpool1 = nn.MaxPool2D(kernel_size=__, stride=__, padding=0)
4 conv3 = nn.Conv2D(__, 32, kernel_size=7, stride=1, padding=0)
5 conv_skip = nn.Conv2D(__, __, kernel_size=3, stride=__, padding=__)
6 avgpool1 = nn.AvgPool2D(kernel_size=2, stride=2, padding=0)
7 fc1 = nn.Linear(__, 256)
8 fc2 = nn.Linear(__, 4)

```

```

1 # img.shape == (1, C, H, W)
2 x = torch.ReLU(conv1(img))
3 x = torch.ReLU(conv2(x))
4 x = maxpool1(x)
5 assert(x.shape[-1] == 22)
6 assert(x.shape[-2] == 22)
7 x = torch.ReLU(conv3(x))
8 skip = torch.ReLU(conv_skip(img))
9 x = x + skip
10 x = avgpool1(x)
11 x = x.view(1, -1)
12 x = torch.ReLU(fc1(x))
13 x = fc2(x)

```

- (a) Dopunite inicijalizaciju zadanih slojeva hiperparametrima koji nedostaju, tako da se unaprijedni prolaz može izvršiti bez greške.
- (b) Napišite broj parametara svakog sloja, te dimenzije tenzora x nakon izvršene svake linije unaprijednog prolaza. Podrazumijevana vrijednost argumenta bias u konvolucijskim i potpuno povezanim slojevima je True.
- (c) Odredite receptivno polje izlaza sloja maxpool1.

rj:

3. INPUT = $3 \times 32 \times 32$

$\text{CONV}(3, 16, K=7 \times 7, S=1, P=0)$
 $\text{CONV}(16, 24, K=3 \times 3, S=1, P=1) \rightarrow$
 $\text{MAXPOOL}(K=5 \times 5, S=1, P=0)$
 $\text{CONV}(24, 32, K=7 \times 7, S=1, P=0)$
 $\text{CONV}(32, 24, K=3, S=5, P=8)$
 $\text{AVGPOOL}(K=2 \times 2, S=2, P=0)$

$$Fc1 = (3872, 256)$$

$$Fc2 = (256, 4)$$

	# PARAM	DIM	RP
CONV	$3 \cdot 16 \cdot 7 \cdot 7 + 3 \cdot 16 = 16 \cdot 26 \cdot 26$	$16 \cdot 26 \cdot 26$	$1+6=7 \times 7$
CONV	$24 \cdot 16 \cdot 3 \cdot 3 + 16 \cdot 24 = 24 \times 24 \times 24$	$24 \times 24 \times 24$	$3 \times 3 \times 9 \times 9$
MAXPOOL	—	$24 \times 16 \times 16$	13×13
CONV	$24 \times 32 \times 7 \times 7 + 24 \times 32 = 32 \times 10 \times 10$	$32 \times 10 \times 10$	19×19
CONV	$3 \times 32 \times 3 \times 3 + 3 \times 32 = 32 \times 10 \times 10$	$32 \times 10 \times 10$	$3 \times 3 \times 16 \text{ (IGNORE!!)}$
AVGPOOL	—	$32 \times 5 \times 5$	20×20
FC1	3872×256	3872×256	32×32
FC2	256×4	4	32×32

$$\frac{32+2^P-3}{2} + 1 = 22$$

$$\frac{32+2^P-3}{5} + 1 = 22$$

edit: zaboravio dodat biase u potpuno povezane slojeve

/

nije li u conv2 trebao ic padding = 0? -> moguće, mi smo rješavali sa prepisanog MI kojeg smo dobili od nekog, pa ja mislim da je pisalo da je padding 1, moguće slučajno skroz

vise vjerujem tom prepisanom jer postoji mogucnost da je ovaj u pdfu kriv, vec sam naisao na neke cudne stvari

u svakom slučaju ako netko zeli nek provjeri ovaj zadatak sa paddingom 1 da ne bi bilo da sam nes zajebo

4. Razmotrimo klasifikacijski model u 3 klase. Učenje provodimo na podacima koji su označeni indeksima $y \in 0, 1, 2, 3$. Indeksi 0, 1, 2 označavaju razrede unutar distribucijskih podataka. Na takve podatke primjenjujemo gubitak negativne log izglednosti \mathcal{L}_{nll} . Indeks 3 označava tzv. negativne podatke za učenje. To su izvandistribucijski podaci koji ne pripadaju niti jedom od 3 poznata razreda. Na takve podatke primjenjujemo gubitak KL divergencije \mathcal{L}_{kl} . Takav je model opisan sljedećim jednadžbama:

$$\mathbf{s} = \mathbf{W}\mathbf{x} + \mathbf{b} \quad (1)$$

$$p = softmax(\mathbf{s}) \quad (2)$$

$$\mathcal{L} = \begin{cases} \mathcal{L}_{nll}(p_y), & y \in 0, 1, 2 \\ \mathcal{L}_{kl}(\mathbf{p}_1 uniform), & y = 3 \end{cases} \quad L_{kl}(\mathbf{p}, \mathbf{q}) = -\sum_i q_i \log\left(\frac{p_i}{q_i}\right) \quad (3)$$

- (a) odredite gradijente KL divergencije po logitima $\frac{\partial \mathcal{L}_{kl}}{\partial s}$ za zadanu proizvoljnu distribuciju q.
- (b) Pojasnite odnos između gubitka negativne log izglednosti i gubitka unakrsne entropije.
- (c) Pojasnite odnos između gubitka unakrsne entropije i KL divergencije.
- (d) Pojasnite funkciju gubitka koristeći prethodna razmatranja.
- (e) Izračunajte gubitak i gradijente po parametrima u podacima $\mathbf{x}_1 = \begin{bmatrix} -5 \\ 2 \end{bmatrix}$, $y_1 = 0$, $\mathbf{x}_2 = \begin{bmatrix} 6 \\ 8 \end{bmatrix}$, ako su zadani parametri: $\mathbf{W} = \begin{bmatrix} 1 & 1 & -1 \\ -1 & 1 & 1 \end{bmatrix}$, $\mathbf{b} = \mathbf{0}$.

rj:

jel bi pod b) odgovor bio da se cross entropy koristi kad imamo meke oznake, dok je klasična negativna log izglednost kad su tvrde oznake, i također cross entropy je uprosječen?

na strojnom je isla prica da je negativna log izglednost sablona prek koje mozes dobit gubitak ako znas po kojoj se distribuciji ravna izglednost, ali brijem da je ovdje shema da je unakrsna entropija za dvoklasni problem a NLL za proizvoljan broj klasa

/

Mislim da si mozd samo pobrko pojam negativne log izglednosti i gubitka negativne log izglednosti. Mislim vise manje je jedno samo izvedenica drugog. Ne znam isto kad kaze unakrsna entropija u zadatku onda valjda mislim na ovu generalno visekласну tak da im onda to nebi bila razlika da je jedan za visekласni a drugi za binarni slučaj.

ima smisla, mozda je onda trebalo skuzit da je NLL "iznad" unakrsne entropije u hijerarhiji? definitivno je izvedenica od NLL-a, da

Mislim da je zapravo glavna razlika činjenica da se jedna pogreška uprosječava a druga ne. Isto se može primjetit da zapravo ako ignoriramo uprosječavanje i imamo tvrde oznake(one hot vektore) zapravo cross entropy postaje NLL.

<https://stackoverflow.com/questions/50913508/what-is-the-difference-between-cross-entropy-and-log-loss-error>

Segvicev take:

$$\mathcal{L}_{\text{NLL}}(\mathbf{W}, \mathbf{b} | \mathcal{D}) = - \sum_i \log P(Y = y_i | \mathbf{x}_i).$$

Obratite pažnju na to da $P(Y = y_i | \mathbf{x}_i)$ označava vjerojatnost kojom model podatak \mathbf{x}_i klasificira u točan razred y_i . Ovako izražen gubitak favorizira parametre \mathbf{W} i \mathbf{b} koji točnim oznakama podataka za učenje pridružuju što veće vjerojatnosti. U praksi nas ništa ne sprječava da umjesto negativne log-izglednosti parametara optimiramo neki drugi gubitak. Međutim, za slučaj klasifikacije, tj. kada su predikcije kategoričke, negativna log-izglednost vrlo dobro funkcioniра u praksi te je najdosljedniji izbor kad učimo na čvrstим oznakama (za meke oznake treba nam unakrsna entropija). Kada negativnu log-izglednost podijelimo brojem podataka, dobivamo izraz koji odgovara prosječnoj unakrsnoj entropiji između točne i prediktirane distribucije po svim podatcima (provjerite to za vježbu!):

$$\mathcal{L}_{\text{CE}}(\mathbf{W}, \mathbf{b} | \mathcal{D}) = - \frac{1}{N} \sum_i \log P(Y = y_i | \mathbf{x}_i).$$

onda je to to.

also nije mi jasno "kada nll podijelimo brojem podataka, dobivamo izraz koji odgovara PROSJEČNOJ unakrsnoj entropiji". nije li onda negativna log izglednost isto sto i unakrsna entropija?

/

jos nesto sto sam nasao

Dosljednija formulacija gubitka u probabilističkom okruženju jest negativna log-izglednost parametara modela:

$$J(\theta) = -\frac{1}{N} \sum_i \log P_{\text{model}}(y_i | \mathbf{x}_i, \theta).$$

Može se pokazati da je negativna log-izglednost specijalni slučaj unakrsne entropije (ekvivalent KL udaljenosti).

d) Pojednostavite funkciju gubitka*

Koliko sam skuzio ovdje se Qi iz nazivnika moze maknuti s obzirom da se tako i tako makne kad deriviramo funkciju <- ovo je dobro, ovaj video lijepo objasni te sheme <https://www.youtube.com/watch?v=Pwgpl9mKars>

e) zadatak mi je cudan jer nije moguce napraviti Wx , netko je zajebo kod prepisivanja cini mi se -> krivo je prepisano W bi trebao bit W^T

Fali $y^2=3$

/

5. Razmotrite klasifikacijski model zadan nizom slojeva:

$$\mathbf{s}_1 = \text{Conv1D}(\mathbf{x}, \mathbf{w}_1, \mathbf{b}_1) \quad (4)$$

$$\mathbf{h}_1 = \text{ReLU}(\mathbf{s}_1) \quad (5)$$

$$\mathbf{g}_1 = \text{concat}(\mathbf{h}_1, \mathbf{x}) \quad (6)$$

$$\mathbf{s}_2 = \mathbf{w}_2 + \mathbf{g}_1 \quad (7)$$

$$\mathbf{p} = \sigma(\mathbf{s}_2) \quad (8)$$

Izračunajte gradijente gubitka binarne unakrsne entropije obzirom na sve parametre modelam ako je zadan ulazni podatak $\mathbf{x} = [1 \ 0 \ -1]^\top$, parametri modela $\mathbf{w}_1 = [1 \ 1]^\top$, $b_1 = 0$, $\mathbf{w}_2 = [1 \ 1 \ 0 \ 0 \ 1 \ 1]^\top$, te točna oznaka podataka $y = 0$.

Napomena: Operacija concat označava konkatenaciju dva vektora dimenzija n1 i n2; rezultat te operacije je novi vektor s n1 + n2 elemenata, gdje prvih n1 elemenata odgovara prvome vektoru, a preostalih n2 elemenata drugome vektoru.

rj:

ZI 2021/22

1. Binarni klasifikacijski model zadan je nizom slojeva

- Potpuno povezani sloj ($\mathbf{W}_x + \mathbf{b}$) s parametrima \mathbf{W}_1 i \mathbf{b}_1 .
- Normalizacija nad grupom (eng. batchnorm) bez afine transformacije
- Aktivacija zglobnicom
- Potpuno povezani sloj ($\mathbf{W}_x + \mathbf{b}$) s parametrima \mathbf{W}_2 i \mathbf{b}_2 .
- Aktivacija sigmoidom

Provredite jedan korak učenja modela gubitkom binarne entropije za minigrupu sastavljanu od podataka $\mathbf{x}_1 = [1 \ 1]^\top$ i $\mathbf{x}_2 = [-1 \ 3]^\top$ s odgovarajućim točnim oznakama $y_1 = y_2 = 1$.

Početne vrijednosti parametara modela su $\mathbf{W}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\mathbf{b}_1 = [0 \ 0]^\top$, $\mathbf{W}_2 = [1 \ 1]$ te $b_2 = -1$. Za izračun statistika koristite pristrane procjenitelje (hint: N u nazivniku). Prilikom unatražnog prolaza kroz sloj normalizacije nad grupom, statistike grupe promatrajte kao konstante.

Prepostavite da je stopa učenja 1.

rj:

radi li se ovdje backprop zasebno za oba podatka i onda samo zbrojim gradijente?

Da zbrojiti pa podjeliti sa dva.

da samo nek netko odgovori jer nisam siguran jel se onda mora taj zbroj podjeliti sa dva ili ne

S obzirom da kaze da je rjec o minigrupi uzima se prosjek gradijenata primjeraka.

Jel se da nekome samo ukratko objasniti što se tu dešava? Nije li kod batchnorma da se uzme prosjek izlaza ili tako nešto?

/

2. Razmatrite sustav za pretraživanje slika. Zadana su dvodimenzionalna metrička ugrađivanja skupa podataka u obliku matrice \mathbf{X} , te njihovi identiteti \mathbf{y} . Na ulaz sustava dolazi upit identiteta $y_i = 1$, za kojeg smo izračunali vektor ugrađivanja $\mathbf{x}_i = [1 \quad 1]^\top$.

Zadaci:

- Izračunajte sličnost podataka \mathbf{x}_i s ostalim podacima iz zadano skupa, ako za mjeru sličnosti koristimo recipročnu vrijednost L1 udaljenosti vektora ugrađivanja.
- Koristeći izračunate sličnosti skicirajte krivulju preciznosti i odaziva za zadani ispit i odredite prosječnu preciznost (AP).

Zadano je:

$$\mathbf{X} = \begin{bmatrix} 1 & 0 \\ 3 & -1 \\ -0.5 & 0.5 \\ 1 & 4 \end{bmatrix} \text{ i } \mathbf{y} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}.$$

rj:

jel tu treba paziti da slicnost promijenimo u udaljenost?

Vrednovanje zadatka

jel moze netko objasniti medukorake ovog:

<http://www.zemris.fer.hr/~ssegvic/du/projects/rep22.pdf>

koji dio te muci tocno?

šta radi sa udaljenostima? jel ih konvertira u slicnosti i onda na temelju toga odluce je 1 ili 0, a i osim toga kak odjednom ima u trećem koraku u tablici 2 clana, ali zato ima 1 TP, 1 FN, i 1 FP? ne kuzim

aj cek

prvo ih sortiras po udaljenosti gdje ti onda udaljenost "inducira" rangiranje. znaci udaljenost x5 od x1 je 1 on je ocito najblizi, udaljenost x3 od x1 je sljedeca po redu itd. mislim da istu stvar mozes raditi i sa slicnostima ali obrnuto, ono sto je najslicnije ce bit prvo u sortiranom. 1 ili 0 odluce je po tome koja je pripadajuca oznaka u

/

sortiranom. kad racunas ove FP TP itd uvijek prvih i uzmes da su pozitivi, ostali da su negativi i onda racunas te pokazatelje na temelju tih predikcija i stvarnih oznaka.
treci korak:

sortirano je: [x5, x3, x4, x2]

oznake su : [1, 0, 1, 0]

poz pred : [1, 1, 0, 0]

i sad racunas TP, FP, FN

TP = x5 = 1 itd..

mislim da mi je sad jasno, nisam kuzio ta rangiranja i fore, tnx!!!

nemas frke :)

btw da nam je ulazna labela bila 0, onda bi u tablici postavljali sve primjere redom na predikciju 0? umjesto ovak kak trenutno radimo da je 1. -> ne, mislim da uvijek ide ovak

kak vam ispada ovo? jel dobio netko isto?

a)

$$s(x_0, x_1) = 1$$

$$s(x_0, x_2) = 1/4$$

$$s(x_0, x_3) = 1/2$$

$$s(x_0, x_4) = 1/3$$

to sam dobio isto ko i ti

b)

poz	TP	FP	FN	R	P
x1..x1	1	0	2	1/3	1
x1..x3	1	1	2	1/3	1/2
x1..x4	2	1	1	2/3	2/3
x1..x2	3	1	0	1	3/4

ovako nesto? tako je i meni isto ispalio sad

/

$0.33*1 + 0.33*0.66 + 0.33*0.75 = 0.805$ (upisujte bas trecine u kalkulator, inace greska bude pristojno velika)

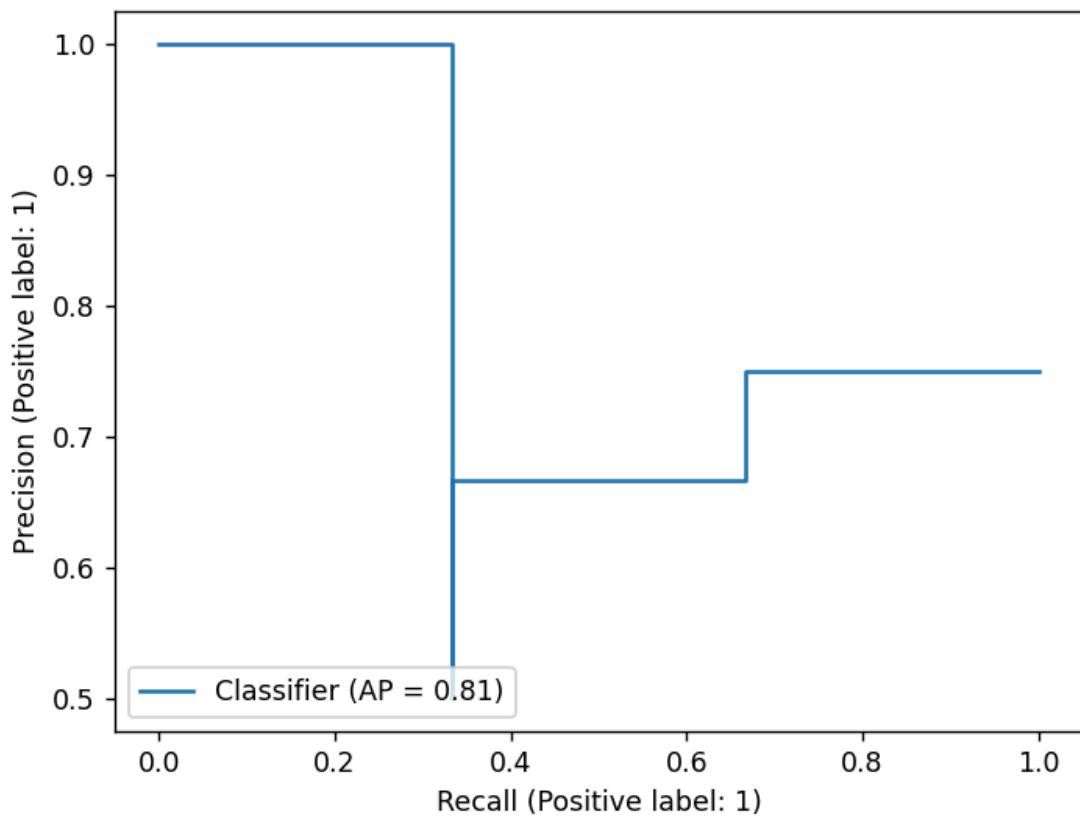
Znači mi u ovom zadatku gledamo labelu ulaznog podatka i postepeno povećavamo radijus područja u kojem označujemo primjere klasom ulaznog primjera.?

da, upravo tako -> nije bas da gledas labelu ulaznog podatka, gore sam to isto spomenuo

e koja je formula za AUPRC?

ja si uvijek crtam ovaj dole graf i onda iz njega lomim podrucje na pravokutnike, i na kraju sam sumiras sve ispod krivulje

ahaaaaaaa oke oke jebeno



tooo asu <3

/

3. Razmatramo konvolucijski klacifikacijski model zadan nizom slojeva:

- 1D konvolucija bez nadopunjavanja s korakom 1, s dvije jezgre $\mathbf{w}_{11} = [1 \ 0 \ -1]$, $\mathbf{w}_{12} = [1 \ 1 \ -1]$ bez pomaka
- Aktivacija zglobnicom
- Globalno sažimanje prosjekom
- Potpuno povezani sloj s parametrima $\mathbf{W}_2 = \begin{bmatrix} \ln 2 & 0 \\ 0 & \ln 2 \end{bmatrix}$ i $\mathbf{b}_2 = [0 \ \ln 2]^\top$
- Aktivacija softmaksom

Na ulazu modela nalazi se podatak $\mathbf{x} = [4 \ -1 \ 2 \ 0 \ 1]$ koji pripada razredu $y = 1$.

- (a) Napravite unaprjedni prolaz kroz model za zadani podatak \mathbf{x} .
- (b) Izračunajte gradijent gubitka unakrsne entropije po parametrima konvolucijskog sloja.

rj:

ja -> 

konvolucija backprop -> 

Ako ima neko da je skuzio kako se lako pamti backprop formule za sve vrste konvolucije ljubim ga.

lowkey moras pamtit jednu stvar:

ako ide gradijent po parametrima onda sigurno nema tih parametara u izrazu ->

ulaz * gradijenti_otprije

ako ide gradijent po ulazu onda sigurno ima parametara u izrazu ->

gradijenti_otprije * flip(parametri)

sto se paddanja tice, samo pamti da ti dimenzionalnost mora stimat u ovisnosti po cemu radis gradijent ->

ako radis gradijent po parametrima onda mora bit ista dimenzionalnost kao i taj parametar

ako radis gradijent po ulazima onda mora bit ista dimenzionalnost kao i taj ulaz

ne moras zasebno pamtit formule za razlicite dimenzionalnosti konvolucije, isto sto radis u jednoj dimenziji, radis i u dvije

Fala lepa, :*

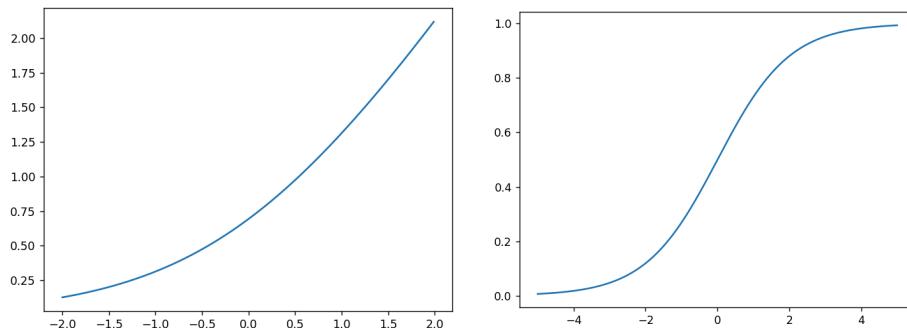
4. Razmotrimo aktivacijsku funkciju $\text{softplus}(x) = \ln(1 + ex)$.

- Skicirajte graf zadane funkcije i njene derivacije.
- Napišite jakobijan aktivacije zadanom funkcijom za 4-dimenzijski ulaz.
- Za zadanu aktivacijsku funkciju implementirajte sučelje **Layer** po uzoru na 2. Laboratorijsku vježbu. Implementirani sloj treba moći raditi s tensorima različitih redova i dimenzija. Ulaz u funkciju **forward** će biti tipa **np.ndarray**.

rj:

$$f(x) = \ln(1 + e^x), f(x)' = e^x/(1 + e^x) = 1/(e^{-x} + 1) \rightarrow \text{sigmoid}$$

a)



b)

$$y_i = \ln(1+e^{(x_i)})$$

$$\begin{aligned} J = [& \frac{dy_1}{dx_1} \quad \frac{dy_1}{dx_2} \quad \frac{dy_1}{dx_3} \quad \frac{dy_1}{dx_4}] = [\text{sigmoid}(x_1) \quad 0 \quad 0 \quad 0] \\ & [\frac{dy_2}{dx_1} \quad \frac{dy_2}{dx_2} \quad \frac{dy_2}{dx_3} \quad \frac{dy_2}{dx_4}] = [0 \quad \text{sigmoid}(x_2) \quad 0 \quad 0] \\ & [\frac{dy_3}{dx_1} \quad \frac{dy_3}{dx_2} \quad \frac{dy_3}{dx_3} \quad \frac{dy_3}{dx_4}] = [0 \quad 0 \quad \text{sigmoid}(x_3) \quad 0] \\ & [\frac{dy_4}{dx_1} \quad \frac{dy_4}{dx_2} \quad \frac{dy_4}{dx_3} \quad \frac{dy_4}{dx_4}] = [0 \quad 0 \quad 0 \quad \text{sigmoid}(x_4)] \end{aligned}$$

c)

nisam bas siguran za bw_inputs

```
class Softplus:
    def forward(self, inputs):
        self.x = inputs
```

```

    /
return np.log(1 + np.exp(self.x))

def backward_inputs2(self,grads):
    nazivnik = 1 + np.exp(-1 * self.inputs)
    dfdx = 1 / nazivnik
    return grads @ dfdx

# -> nebi li grads bio (m, n, 1), a dfdx (m, n, n), m zbog batcha? mislim da se moze
jednostavnije ako napravis m (n, 1) vektora koje elementwise mnozis s gradsim.
uostalom ovo predlozeno rjesenje onda uopce nebi ni bilo ispravno jer bi dfdx bilo
(m, n, 1), a ak se mnozi matricno dat ce m skalara na izlazu, pa se ne poklapa
dimenzija derivacije s dimenzijom tog sloja.

def backward_params(self, grads):
    pass

```

5. Oblikuj model obične povratne neuronske mreže s prijenosnom funkcijom zglobnice za evidentiranje promjene stanja bankovnog računa. Za taj zadatak koristit će te tradicionalni vector skrivenog stanja, pri čemu će se u pravom element pamtitи ukupan iznos dostupan za isplatu, na drugom elementu će biti zapisano dopušteno prekoračenje, dok treći element služi kao zastavica koja provjerava jeli račun u “minusu” (0 ako je, inače pozitivan broj). Početno skriveno stanje računa je:

$$\mathbf{h}^{(0)} = \begin{bmatrix} 10000 \\ 5000 \\ 765 \end{bmatrix}^\top \quad (1)$$

- (a) Odredite preostale parameter (W_{hh}, W_{xh}, b_h) kako bi mreža obavljala navedenu funkcionalnost.
- (b) Provedite unaprijedni prolaz s navedenim parametrima za iduće ulaze:

$$\mathbf{x}^{(1)} = \begin{bmatrix} 1000 \\ 0 \end{bmatrix}^\top, \mathbf{x}^{(2)} = \begin{bmatrix} 1000 \\ 3500 \end{bmatrix}^\top, \mathbf{x}^{(3)} = \begin{bmatrix} 0 \\ 4000 \end{bmatrix}^\top \quad (2)$$

- (c) Razmotrimo pristup istom problemu sa dvoslojnom LSTM mrežom, uz jednake dimenzionalnosti ulazne reprezentacije i skrivenog stanja. Odredite ukupan broj parametara takve mreže.

da jos nadodam nesto što nije rpepisano u pdf:
prvi element = pozitivno stanje racuna + dopusteno prekoracanje

/

Slucajem u kojem korisnik prekoraci dozvoljeni minus nije se potrebno baviti.
Dvodimenzionalni ulazni vektor mreze sadrzi iznos uplate na prvom elementu te
iznos isplate na drugom elementu.

rj:

- (a) ovdje je $Whh = [[1, 0, 0], [0, 1, 0], [1, -1, 0]]$, $Wxh = [[1, -1], [0, 0], [1, -1]]$ i $bh = [0 \ 0 \ 0]$
- (b) samo $h(t) = \text{ReLU}(Whh h(t-1) + Wxh x(t) + bh)$ i cepaj, ReLU ce resetirati negativni broj u h_3 na 0 -> **xt ti je 2 dim, u tvom rjesenju bi bio 3dim, jel meni nesto promaklo tako je popravio sam sad**
- (c) dvoslojni LSTM = 4 * RNN parametara (forget, input, output gateovi i current state) ali PAZIT kod dvoslojnog da je ulaz u sljedeci LSTM skriveno stanje dimenzije a ne dimenzije ulaza x

prvi sloj: $gate1(t) = Wg_{hh} h(t-1) + Wg_{xh} x(t) + bg_h$

drugi sloj onda u sredini nije Wg_{xh} nego Wg_{hh} i prijasnji $h(t)$ iz nizeg sloja

$Wxh - x * h$ parametara

$Whh - h * h$ parametara

$bh - h$ parametara

znaci $4 * (\underline{xh} + \underline{hh} + h + \underline{hh} + \underline{hh} + h) = 156$? (nek me ispravi nek ak sam zezno) -> **dobro je**

jedno pitanje vezano za ovaj zadatak, video sam da se u starim ispitima cesto spominje notacija iz goodfellowa. Jel moramo to pamtit ili mislite da je ovo (Whh Wxh , bh , bo , Wyh) dovoljno?

– da spomenuto je u prezenciji da je promijenjena notacija sa notacije od one u knjizi goodfellowa ali evo neka bude kao cheat sheet ovdje -> **mislim da u novim ispitima su prestali koristit staru notaciju i u potpunosti presli na ovu noviju iz prezentacija**

- **[!!] Notacija u knjizi: $W := W_{hh}$, $U := W_{xh}$, $V := W_{hy}$, $b := b_h$, $c := b_o$**

Rok 2021/22

1. Razmatramo klasifikacijski konvolucijski model na jednodimenzionalne podatke. Slojeni su slijedeći:

F: konvolucijski sloj bez nadopunjivanja (2 jezgre, korak 1, aktivacija relu)

g: globalno sažimanje maksimumom

s: potpuno povezani sloj s dva izlaza

p: softmax

Pocetni parametri konv. su: $w_{1A} = [-0.5 \ 0.5] \quad w_{1B} = [0.5 \ 0.5]$

Ako $b_{1A} = 0, b_{1B} = -0.5$, pocetni parametri potpuno povezani sloja su $w_2 = I, b_2 = 0$

a) napisite jednadzbe modela kojeg implementira mreza

b) odredite izlaz mreze za $x_1 = [0 \ 1 \ 1 \ 0] \quad x_2 =$

$[1 \ 1 \ 1 \ 1]$. Pretpostavite da se konvolucija provodi koracijom s razmakom filtra

c) Odredite gradijente s obzirom na konvolucijske parametre (za x_1 i $y_1 = t$ (tasa 1))

3. Razmatramo duboki model zadani listom slojeni kao ito

smo imali u 2. labu su. Napisati implementaciju

sloja Softmax Cross Entropy With Logits

4. Razmatramo jezični model na razini slova, gdje je zadatik predvidjeti iduće slovo na temelju prethodnih. Problem rješavamo običnim povratnim modelom koji u povratnoj verziji ima proporsnu funkciju tanh, a na izlazu softmax. Način

modela dovodimo naiz znakova kodičenih jedinicnim kodom.

Model treba moći raditi s 80 različitim znakovima. Skriveni sloj ima 200 dimenzija. Odredite dimenzije svih parametara povratne neuronske mreže,

5. Razmatramo običnu povratnu mrežu koja u povratnoj

vezici ima prijenosnu funkciju $\min(1, x)$, a na izlazu identitet bez nelinearnosti. Zadatak može je pobrojati jedinice i nule u binarnom broju pridruženu na ulaz dolazi proizvoljno dugacak mreži jedinica i duž u one-hot reprezentaciji. Prijenosna mreža je jedinica $W_{nn} = I$, početna stanje mreže je $h_0 = [0 \ 0]$, a za preostale parametre mreže t_i (t_j , W_{xh} , W_{yh} , b_h i b_o) vrijedi $|t_i| \leq 1$ i $t_i \in \mathbb{Z}$. Odredite vrijednosti ostalih parametara mreže.

6. Razmatramo model za ugradnjive dvodimenzionalne podskupove u jednodimenzionalni metrički prostor: Slojevi su:

h_1 : potpuno povezani sloj s dva izlaza i aktivacijskom relu

s_2 : potpuno povezani sloj sa jednim izlazom

model vremeno troglim gubitkom $L(a, p, n) = \max(d(a, p) - d(a, h) + m, 0)$

parametri su: $W_1 = I$, $W_2 = \begin{bmatrix} 0.4 & 0.6 \end{bmatrix}$, $b_1 = 0$, $b_2 = 0$

$d = L$ 1 metar, $m = 1$

a) napravite jednadžbe modela kojeg implementira mreža pod pretpostavkom da na ulaz modela dolaze triple (x_a, x_p, x_n)

b) Odredite iznos gubitka za ulaznu triple

$$x_a, x_p, x_n = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

c) Odredite gradientne gubitke s obzirom na sve parametre modela

2. Zadana je funkcija $f(x) = x^4 - 2x^3 + 5$. Povisite pre 2 iteracije gradientnog spusta počevši iz

$w_0 = -1$ uz $\delta = 0.1$. Bi li vasi postupak drugačije konvergiranje nego uvećanje inicijalizacije ako pretpostavimo da je korak dovoljno mali. Ako da naredite koju vam

moguća gresnja bi' grad. spust mogao pronaći. Bi li vasi postupak bolje konvergiranje negom drugim optim. algoritmom? Ako da naredite zašto i sticijete taj algoritam.