Učenje sličnosti

metrička ugrađivanja složenih podataka

Siniša Šegvić
UniZg-FER

Plan

- motivacija za učenje sličnosti
- sijamsko učenje
- trojni gubitak
- detalji izvedbe i vrednovanje
- primjene: stereoskopija, samonadziranje

Uvod : granice klasifikacijske paradigme

Kada klasifikacijski modeli mogu postati nepraktični:

- razredi ne postoje ili nisu poznati
- nepraktično veliki broj razreda

Primjeri primjena:

- stereoskopska korespondencija
- samonadzirano učenje
- praćenje, asocijativno pretraživanje, biometrijska verifikacija

[bicanic19fusion]

UVOD : GRANICE KLASIFIKACIJSKE PARADIGME (2)

Za takve primjene najpraktičnije prediktirati sličnost primjera

Lijevi par: različit. Desni par: sličan

[chopra05cvpr]

Ideja: ugraditi podatke u prikladni vektorski prostor

- standardne metrike (L2, ...) modeliraju sličnost među podatcima
- kratki naziv: metrička ugrađivanja

Uvod : metrička ugrađivanja

Prije nego što nastavimo dalje, moramo se zapitati:

- zašto ne bismo mjerili sličnost u originalnom prostoru podataka?

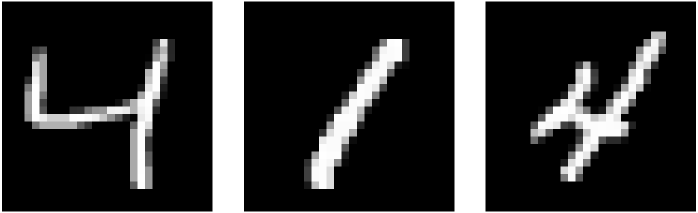
Odgovori:

- zato što udaljenosti u visokodimenzionalnim vektorskim prostorima nemaju smisla (prokletstvo dimenzionalnosti)
- zato što vektorske reprezentacije složenih podataka tipično nisu pogodne za njihovu usporedbu

Uvod : metrička ugrađivanja - primjer

Promotrimo udaljenosti između znamenki skupa MNIST

- L2 udaljenost između prve i druge znamenke: 121.4
- L2 udaljenost između prve i treće znamenke: 133.2
- L2 udaljenost između druge i treće znamenke: 114.9



[lecun98pieee]

Zaključak: ne postoji jaka veza između udaljenosti u originalnom prostoru i semantičke sličnosti!

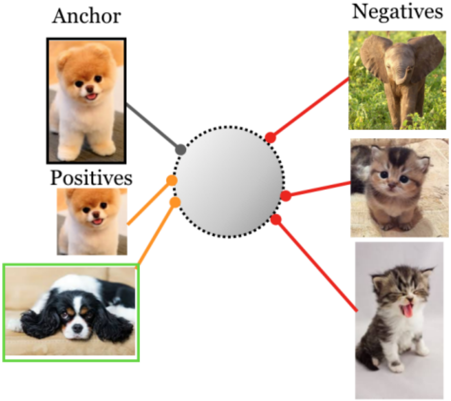
Uvod : metrička ugrađivanja - primjer (2)

```
import torch
import torchvision
mnist = torchvision.datasets.MNIST('data', download=True)
print(mnist.targets[:10])
# tensor([5, 0, 4, 1, 9, 2, 1, 3, 1, 4])
print(torch.sqrt(torch.sum((mnist.data[2]-mnist.data[3])**2,
      dtype=torch.float)))
# tensor(121.4)
print(torch.sqrt(torch.sum((mnist.data[2]-mnist.data[9])**2,
      dtype=torch.float)))
# tensor(133.2)
print(torch.sqrt(torch.sum((mnist.data[3]-mnist.data[9])**2,
      dtype=torch.float)))
# tensor(114.9)
import cv2
cv2.imwrite('m2.png', mnist.data[2].numpy())
cv2.imwrite('m3.png', mnist.data[3].numpy())
cv2.imwrite('m9.png', mnist.data[9].numpy())
```

Uvod : metrička ugrađivanja - cilj

Ugraditi podatke u (relativno) niskodimenzionalni prostor gdje će standardna metrika modelirati sličnost među podatcima

Ako reprezentacije normiramo (tj. smjestimo ih na hipersferu), udaljenost možemo mjeriti skalarnim produktom



[khosla20neurips]

Uvod : metrička ugrađivanja - prednosti

Podatke možemo asocirati iako u trenutku učenja nismo vidjeli sve razrede

Modelu je teže prenaučiti se:

- klasifikacija: O(N) podataka za učenje
- sličnost: O(N^2), O(N^3) ili O(N^B) podataka za učenje

Vrlo korisne reprezentacije mogu se naučiti i u samonadziranom kontekstu gdje zahtijevamo da podatak bude sličan perturbiranom sebi a različit od ostalih podataka; npr. SimCLR [chen20icml].

Metrička ugrađivanja mogu ponekad pomoći i u klasičnim nadziranim zadatcima [khosla20neurips].

METRIKE: POJMOVI

Razmatramo skup X te preslikavanje $d : X \times X \to R$.

Kažemo da je d metrika, a (X, d) - metrički prostor akko:

1. $d(a, b) \geq 0 \quad \forall a, b \in X$ (pozitivnost),
2. $d(a, b) = 0 \iff a = b \quad \forall a, b \in X$ (strogost),
3. $d(a, b) = d(b, a) \quad \forall a, b \in X$ (simetričnost),
4. $d(a, b) \leq d(a, c) + d(c, b) \quad \forall a, b, c \in X$ (nejednakost trokuta).

Ova definicija dobro se uklapa u koncept sličnosti podataka.

Aksiomi su redundantni: npr. pozitivnost i simetričnost slijede iz strogosti i nejednakosti trokuta.

U praksi najčešće učimo pseudo-metriku koja relaksira strogost

- teško osigurati $d(a, b) \neq 0 \; \forall a \neq b$, zahtijevamo samo $d(a, a) = 0$

METRIKE : STANDARDNI IZBORI

Euklidska metrika:

$$d_E(a, b) = \sqrt{(a-b)^\top (a-b)} \sim (a-b)^\top (a-b)$$

Ako su podatci normirani, skalarni produkt odgovara kosinusnoj sličnosti te inducira isto rangiranje kao i Euklidska metrika:

$$\begin{aligned} d_E(a, b) &\sim (a-b)^\top (a-b) \\ &= a^\top a - 2 \cdot a^\top b + b^\top b = 2 - 2 \cdot a^\top b \\ &\sim -a^\top b \end{aligned}$$

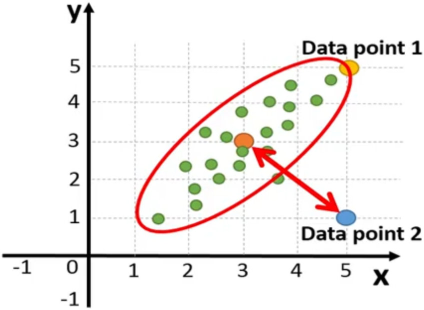
Mahalanobisova metrika (M odgovara inverznoj kovarijanci podataka):

$$d_M(a, b) \sim (a-b)^\top \cdot M \cdot (a-b)$$

METRIKE : MAHALANOBIS

Mahalanobisova izohipsa označena je crvenom bojom

- Po Mahalanobisu, žuti podatak puno je bliži narančastom od plavog podatka
- možemo hipotetizirati da plavi podatak ne pripada zelenima
- Euklidska metrika ne podržava takvo zaključivanje!



[tilestats-yt]

METRIKE : MAHALANOBIS (2)

Matrica M je realna i simetrična $\Rightarrow$ može se dijagonalizirati: $M = W^\top W$

Korištenje Mahalanobisove metrike možemo interpretirati kao plitko ugrađivanje u Euklidski metrički prostor W:

$$\begin{aligned}
d_M(a, b) &\sim (a - b)^\top \cdot M \cdot (a - b) \\
&\sim (W \cdot (a - b))^\top \cdot (W \cdot (a - b)) \\
&\sim (W \cdot a - W \cdot b)^\top \cdot (W \cdot a - W \cdot b)) \\
&\sim d_E(Wa, Wb)
\end{aligned}$$

Postoje analogni plitki pristupi koji uzimaju u obzir informaciju o pripadnosti podataka simboličkim razredima (Fisher LDA)

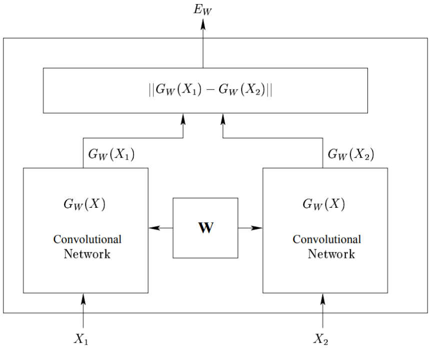
Logični korak dalje: zamijeniti W **dubokim modelom** f_θ

- ono što je logično danas, nije bilo logično 2005...

Sijamsko učenje : ideja

Naučiti model G_W koji ugrađuje podatke X u prostor gdje euklidska metrika E_W odražava sličnost među podatcima

Sijamsko učenje: parovi podataka prolaze kroz dva primjerka modela.



[chopra05cvpr]

Primjerci dijele parametre W, a gradijenti odgovarajućih parametara dviju grana se akumuliraju.

Sijamsko učenje : gubitak

Sijamsko učenje koristi neku od varijanti kontrastnog gubitka

Kontrastni gubitak ovisi o tome jesmo li na ulaze sijamskih modela doveli primjerke istog razreda

$$L(\theta) = \sum_{y_q = y_p} L_{\text{pos}}(\theta | x_q, x_p) + \sum_{y_q \neq y_n} L_{\text{neg}}(\theta | x_q, x_n)$$

Gubitak L_{pos} tjera primjerke istih razreda da se približe:

$$L_{\text{pos}}(\theta | x_q, x_p) = \| f_\theta(x_q) - f_\theta(x_p) \|^2$$

Gubitak L_{neg} tjera različite primjerke da se udalje [hadsell06cvpr]:

$$L_{\text{neg}}(\theta | x_q, x_n) = [\max(0, m - \| f_\theta(x_q) - f_\theta(x_n) \|)]^2$$

Sijamsko učenje : gradijenti

Pogledajmo gradijente gubitaka L_{pos} i L_{neg} s obzirom na metrička ugrađivanja $f_q = f_\theta(x_q)$:

$$\frac{\partial L_{\text{pos}}}{\partial f_p} = 2 \cdot (f_p - f_q)$$

$$\frac{\partial L_{\text{neg}}}{\partial f_n} = 2 \cdot \underbrace{\max(0, m - \|f_q - f_n\|)}_{\text{iznos}} \cdot \underbrace{\frac{f_q - f_n}{\|f_q - f_n\|}}_{\text{smjer}}$$

Ovi gradijenti potiču približavanje ugrađivanja pozitivnih parova te udaljavanje negativnih parova sve dok oni ne postanu udaljeniji od m.

Gradijent $\frac{\partial L_{\text{neg}}}{\partial f_n}$ je vektor:

- usmjeren je od negativa prema sidru: $\frac{f_q - f_n}{\|f_q - f_n\|}$
- iznos opada kako udaljenost raste prema m: $\max(0, m - \|f_q - f_n\|)$

Sijamsko učenje : gradijenti (2)

Domaći rad:

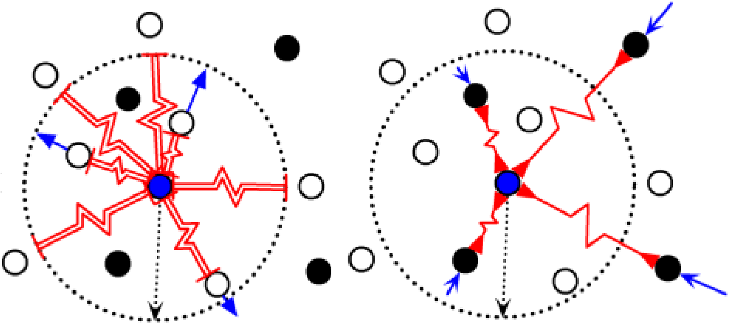
- izvesti gradijente kontrastnog gubitka $\frac{\partial L_{\text{pos}}}{\partial f_p}$ i $\frac{\partial L_{\text{neg}}}{\partial f_n}$;
- usporediti gradijent $\frac{\partial L_{\text{neg}}}{\partial f_n}$ s gradijentom alternativne formulacije:

$$L_{\text{neg2}}(\theta|x_q, x_p) = \max(0, m^2 - \|f_\theta(x_q) - f_\theta(x_p)\|^2)$$

SIJAMSKO UČENJE : GRADIJENTI (3)

Dinamiku kontrastnog gubitka možemo ilustrirati sustavom mehaničkih opruga (sila opruge proporcionalna je udaljenosti)

- crni i bijeli krugovi predstavljaju pozitivne i negativne primjere s obzirom na plavi podatak
- negativi izvan radijusa m ne osjećaju odbijanje plavog podatka



[hadsell06cvpr]

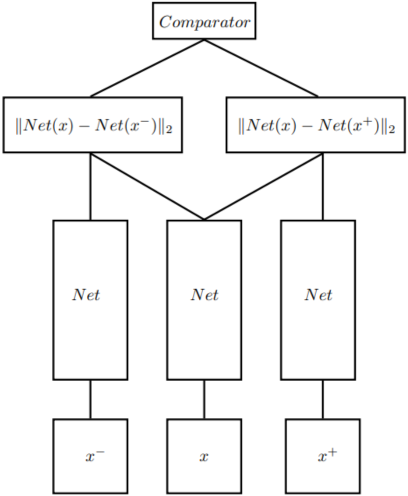
TROJNO UČENJE : IDEJA

U sijamskom učenju jedan te isti podatak uspoređujemo i s negativnim i s pozitivnim primjerima.

Pri tome ugrađivanje promatranog primjera trebamo izračunati dva puta

Taj problem adresira trojno učenje:

- referentno ugrađivanje uspoređujemo s pozitivnim i negativnim primjerom



[hoffer15iclrw]

Tri primjerka modela dijele parametre i ravnopravno doprinose gradijentima gubitka

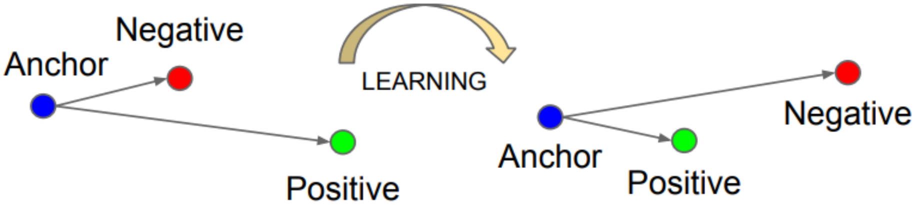
Trojno učenje : gubitak

Trojno učenje tipično koristi trojni gubitak

Trojni gubitak spaja obje komponente kontrastnog gubitka u jedan izraz:

$$L(\theta) = \sum_i \max(0, \|f_\theta(x_{ia}) - f_\theta(x_{ip})\| - \|f_\theta(x_{ia}) - f_\theta(x_{in})\| + \alpha)$$

Trojni gubitak privlači referentni i pozitivan podatak te odbija referentni i negativan podatak:



[schroff15cvpr]

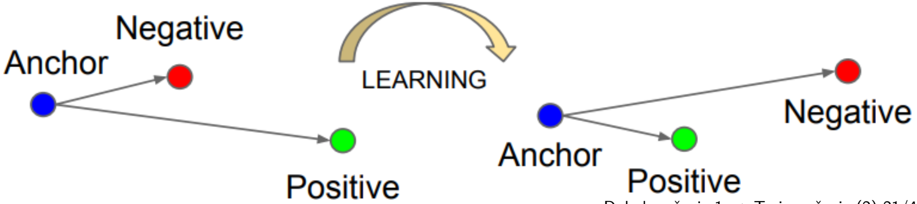
Trojno učenje : gradijenti

Pogledajmo gradijente trojnog gubitka s obzirom na ugrađivanja $f_a = f_\theta(x_{ia})$, $f_p = f_\theta(x_{ip})$ i $f_n = f_\theta(x_{in})$:

$$\frac{\partial L}{\partial f_p} = [\![\|f_a - f_p\| + \alpha > \|f_a - f_n\|]\!] \cdot \frac{f_p - f_a}{\|f_p - f_a\|}$$
$$\frac{\partial L}{\partial f_n} = [\![\|f_a - f_p\| + \alpha > \|f_a - f_n\|]\!] \cdot \frac{f_a - f_n}{\|f_a - f_n\|}$$

Ti gradijenti testiraju je li udaljenost od f_p do f_a manja od $\|f_n - f_a\| - \alpha$.

- ako nije, gradijenti potiču približavanje f_p i udaljavanje f_n:



Detalji : meka zglobnica

Klasični trojni gubitak zanemaruje trojke kod kojih je negativ dalji od pozitiva za više od margine.

Prednost tog pristupa jest onemogućavanje prenaučenosti: kada se podaci dovoljno dobro rasporede --- učenje prestaje

Međutim, s druge strane, takav ziheraški pristup može dovesti do lošije generalizacije

Stoga ponekad možemo poboljšati generalizaciju na način da čvrstu zglobnicu ReLU(x) = $[\ x\ ]_+$ zamijenimo njenom mekom varijantom:

$$\text{softplus}(x) = \ln(1 + e^x)$$

Detalji: verzija sa skalarnim produktom

Dogovorimo skraćenu notaciju ugrađivanja s obzirom na x_p, x_p, x_n:

$$f_a = f_\theta(x_{ia})$$
$$f_p = f_\theta(x_{ip})$$
$$f_n = f_\theta(x_{in})$$

Ako su latentne reprezentacije normirane, $\|f_a\| = \|f_p\| = \|f_n\| = 1$, gubitak možemo izraziti i kroz kosinusnu sličnost:

$$L(\theta) = \sum_i \max(0, f_\theta(x_{ia})^\top f_\theta(x_{in}) - f_\theta(x_{ia})^\top f_\theta(x_{ip}) + \alpha)$$

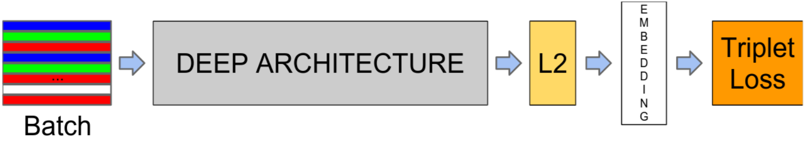
Detalji : formiranje trojki

Pojednostavnjeni izraz za osnovni trojni gubitak [hermans17arxiv]:

$$L_3(\theta) = \sum_{y_a = y_p \neq y_n} [\alpha + D_{ap} - D_{an}]_+$$

Formiranje trojki za učenje vrlo je važan izvedbeni detalj

- glavni problem je u učinkovitosti učenja
- broj svih trojki raste s $O(N \cdot \overline{N_p} \cdot \overline{N_n})$



[schroff15cvpr]

Detalji: formiranje trojki (2)

Ponekad koristimo čvrsti trojni gubitak na slučajnim grupama (eng. batch-hard [hermans17arxiv]):

- referentni podatak a povezujemo s najtežim pozitivom i najtežim negativom [schroff15cvpr].

$$L_{BH}(\theta) = \sum_a [\alpha + \max_{y_a = y_p} D_{ap} - \min_{y_a \neq y_p} D_{an}]_+$$

Ponekad trojke formiramo unaprijed [zbontar15cvpr]:

- svaki primjer koristimo kao referentni (sidro)
- ako je praktično, tražimo teške pozitive i negative s obzirom na ažurni skup parametara [schroff15cvpr]

Detalji : n parova

Pretpostavimo da u grupi imamo N parova podataka [sohn16neurips]:

- svi parovi dijele sidro: $\mathbf{x}_a$
- samo jedan par je pozitivan: $(\mathbf{x}_a, \mathbf{x}_p)$
- svi preostali parovi su negativni (ima ih $N-1$): $(\mathbf{x}_a, \mathbf{x}_{ni})$

Gubitak definiramo tako da raste kad je sidro slično negativima a pada kad je sidro slično pozitivima:

$$\mathcal{L}_{\text{N-pairs}}(\mathbf{x}_a, \mathbf{x}_p, \{\mathbf{x}_{ni}\}) = \log\left(1 + \sum_{i=1}^{N-1} e^{f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_{ni}) - f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_p)}\right)$$

Za $n=2$, $\mathcal{L}_{\text{N-pairs}}$ vrlo je sličan trojnom gubitku sa skalarnim produktom.

Detalji : n parova (2)

Nakon nekoliko jednostavnih koraka $\mathcal{L}_{\text{N-pairs}}$ svodimo na sljedeći oblik:

$$\begin{aligned}
&\mathcal{L}_{\text{N-pairs}}(\mathbf{x}_a, \mathbf{x}_p, \{\mathbf{x}_{ni}\}) = \\
&\quad = -\log \frac{\exp(f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_p))}{\exp(f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_p)) + \sum_{i=1}^{N-1} \exp(f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_{ni}))} \\
&\quad = -\log \frac{\exp(f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_p))}{\sum_{i=1}^{N} \exp(f_\theta(\mathbf{x}_a)^\top f_\theta(\mathbf{x}_i^{\text{all}}))}
\end{aligned}$$

Vidimo da je $\mathcal{L}_{\text{N-pairs}}$ ekvivalentan standardnoj unakrsnoj entropiji nad softmaksom vektora sličnosti parova podataka.

Gubitak n parova ekvivalentan je infoNCE gubitku [vandenoord18arxiv]:

- infoNCE dolazi do iste jednadžbe optimiranjem zajedničke informacije.

DETALJI: N PAROVA (3)

Gubitak n parova može se poopćiti i na slučaj kad imamo više pozitivnih i više negativnih primjera u grupi $\{\mathbf{x}_i, y_i\}_{i=1}^{B}$ (eng. soft nearest neighbours) [frosst19icml]:

$$\mathcal{L}_{\text{snn}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{1}{|\{y_j = y_i, j \neq i\}|} \frac{\sum_{j \neq i, y_j = y_i} e^{f_\theta(\mathbf{x}_i)^\top f_\theta(\mathbf{x}_j)/\tau}}{\sum_{y_i \neq y_k} e^{f_\theta(\mathbf{x}_i)^\top f_\theta(\mathbf{x}_k)/\tau}}$$

- hiper-parametar τ (temperatura) modulira entropiju izlaza.

Stroža varijanta tog gubitka zahtijeva da **svaki** pozitiv iz grupe bude sličniji od negativa [khosla20neurips]:

$$\mathcal{L}_{\text{sum-out}} = -\frac{1}{B}\sum_{i=1}^{B} \frac{1}{|\{y_j = y_i, j \neq i\}|} \sum_{j \neq i, y_j = y_i} \log \frac{e^{f_\theta(\mathbf{x}_i)^\top f_\theta(\mathbf{x}_j)/\tau}}{\sum_{y_i \neq y_k} e^{f_\theta(\mathbf{x}_i)^\top f_\theta(\mathbf{x}_k)/\tau}}$$

- ova varijanta generalizira bolje iako $\mathcal{L}_{\text{snn}} > \mathcal{L}_{\text{sum-out}}$ (Jensen)

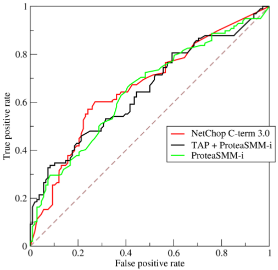
Detalji : vrednovanje

Naučene mjere sličnosti induciraju rangiranje:

- fiksiramo podatak x_a, sortiramo sve druge podatke prema padajućoj sličnosti: $s_{ai} = -d(f_\theta(x_a), f_\theta(x_i))$
- ako rangiranje savršeno generalizira (to obično nije slučaj) onda su svi pozitivi sortirani prije negativa

Kvalitetu rangiranja mjerimo površinom ispod krivulja PR ili ROC

- veća površina $\Rightarrow$ bolji model



[Wikipedia]

AUROC odgovara vjerojatnosti da slučajni pozitiv rangira ispred slučajnog negativa.

Nepogodno za nebalansirane probleme:

- za takve probleme preferiramo AUPR.

Detalji : vrednovanje (2)

Evo kako formirati krivulje za podatak x_a:

- postavljamo prag na svaki indeks i: podatci koji su sličniji od s_{ai} su pozitivi (ima ih P_i), a ostali su negativi (ima ih N_i);
- uz pomoć oznaka dobivamo brojnosti
  - točnih pozitiva - TP_i
  - lažnih pozitiva - FP_i
  - lažnih negativa - FN_i
  - i točnih negativa - TN_i
- sada možemo (i dalje za taj isti prag i) odrediti relevantne metrike:
  - odziv $R_i = TPR_i = TP_i/(TP_i+FN_i)$
  - preciznost $P_i = TP_i/(TP_i+FP_i)$
  - udio lažnih pozitiva = $FPR_i = FP_i/(TN_i+FP_i)$
- krivulju preciznosti i odziva (PR) čine točke (R_i,P_i)
- krivulju ROC čine točke (FPR_i,TPR_i)

Detalji : zadatak

Zadani su podatci x_1 do x_5.

Poznato je da su identiteti podataka redom Y=[1, 0, 0, 1, 1]

Poznato je da udaljenosti od podatka x_1 iznose:

d(x_1, X) = [0.0, 5.0, 2.0, 3.0, 1.0]

Odredite površinu ispod P-R krivulje (skraćeno AUPR) za predikcije modela u podatku x_1

Napomena: AUPR često nazivamo i prosječnom preciznošću (eng. average precision, AP)

Detalji : zadatak - rješenje

Rangiranje podataka je: $[x_5^{(1)}, x_3^{(0)}, x_4^{(1)}, x_2^{(0)}]$

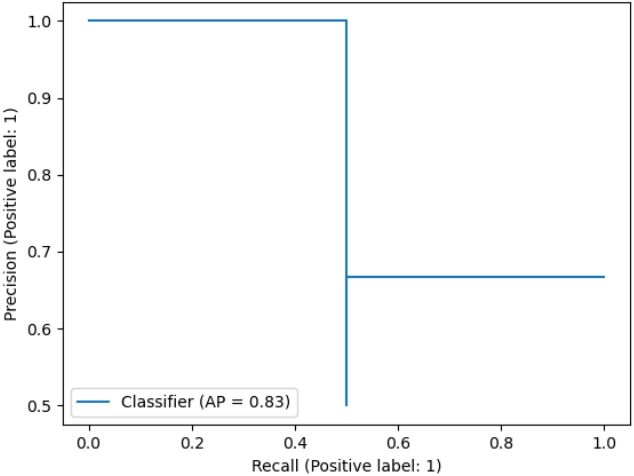
Postavljamo prag na svaki podatak i mjerimo preciznost P=TP/(TP+FP) i odziv R=TP/(TP+FN):

pozitivne predikcije	TP	FP	FN	P	R
x_5, x_3, x_4, x_2	2	2	0	0.5	1.0
x_5, x_3, x_4	2	1	0	**0.7**	**1.0**
x_5, x_3	1	1	1	0.5	0.5
x_5	1	0	1	**1.0**	**0.5**

Za niti jedan prag nemamo R=0. Zato dogovorno dodajemo točku (R=0, P=preciznost za najmanji R).

Ako za isti R imamo više P-ova — smijemo izabrati bolji.

Detalji : zadatak - graf



Rješenje: AUPR = $(0.5\text{-}0)\times 1 + (1\text{-}0.5)\times 0.66 = 0.83$

Detalji: zadatak - kod

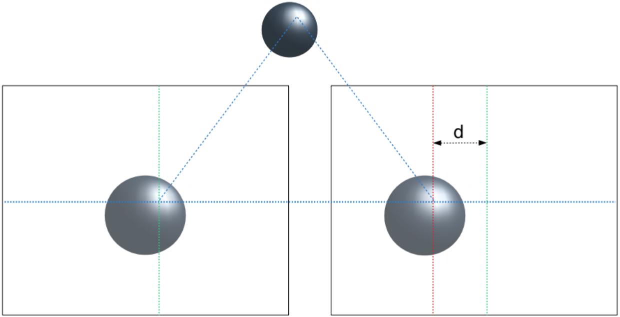
```python
import numpy as np
from sklearn.metrics import average_precision_score
from sklearn.metrics import PrecisionRecallDisplay
import matplotlib.pyplot as plt

y_true = np.array([0, 0, 1, 1])
y_scores = np.array([-5, -2, -3, -1])
print(average_precision_score(y_true, y_scores))

PrecisionRecallDisplay.from_predictions(y_true, y_scores)
plt.show()
```

STEREO : ZADATAK

Za svaki piksel lijeve slike tražimo korespondentni piksel u desnoj slici:



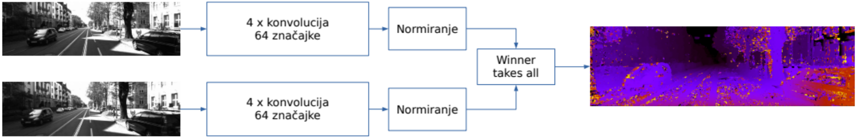
[orsic17ms]

Pretpostavljamo kalibrirani slučaj: korespondencija je u istom retku

- tražimo gusto polje horizontalnih pomaka (dispariteta)
- ako znamo disparitet, širinu vidnog polja kamere i udaljenost među kamerama - možemo odrediti dubinu tog dijela scene u metrima

STEREO : IDEJA

Ugraditi piksele obje slike u metrički prostor konvolucijskim modelom $f_\theta : \mathbb{R}^{3\times H\times W} \rightarrow \mathbb{R}^{F\times H\times W}$, F=64 [zbontar15cvpr].



[orsic17ms]

Formirati gusti volumen cijene V oblika D×H×W:

- $V_{ijd} = \mathrm{cost}(f_\theta(I_L)_{i,j}, f_\theta(I_R)_{i,j+d})$

U svakom pikselu odrediti najbolji disparitet (winner takes all):

- $D_{ij} = \arg\min_d V_{ijd}$

STEREO: DETALJI

Trojni gubitak izražavamo skalarnim produktom nad normiranim metričkim ugrađivanjima isječaka 9×9:

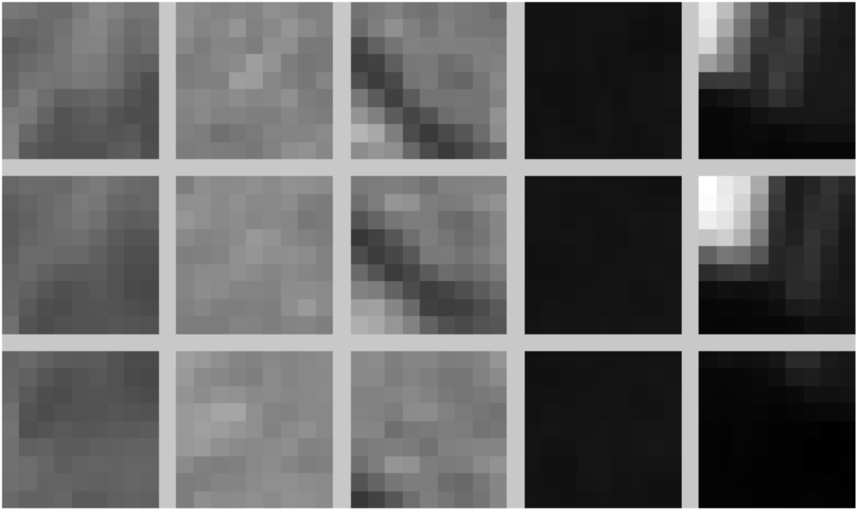
- tri primjerka modela f_θ dijele parametre
- svaki primjerak pamti aktivacije i računa gradijente
- ukupni gradijent za svaki parametar dobivamo agregacijom doprinosa primjeraka modela

Model f_θ je ekvivarijantan: $4\times$conv3x3 bez sažimanja:

- ulazni podatci za učenje imaju dimenzije $128\times3\times9\times9$
- pri učenju ne koristimo nadopunjavanje (pri zaključivanju - da)
- ugrađivanja imaju 64 dimenzije, $f_\theta : \mathbb{R}^{3\times9\times9} \to \mathbb{R}^{64}$
- pikseli slika se normiraju na $N(0, \mathbf{I})$ pri učenju i zaključivanju
- zaključivanje primjenjujemo na cjelokupne slike $2\times3\times$H$\times$W

STEREO : TROJKE

Stupci prikazuju trojke za učenje:

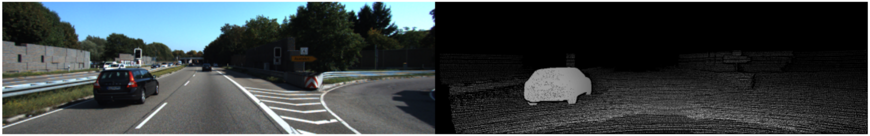


[orsic17ms]

STEREO : EKSPERIMENTI

Skup podataka KITTI [geiger13ijrr]:

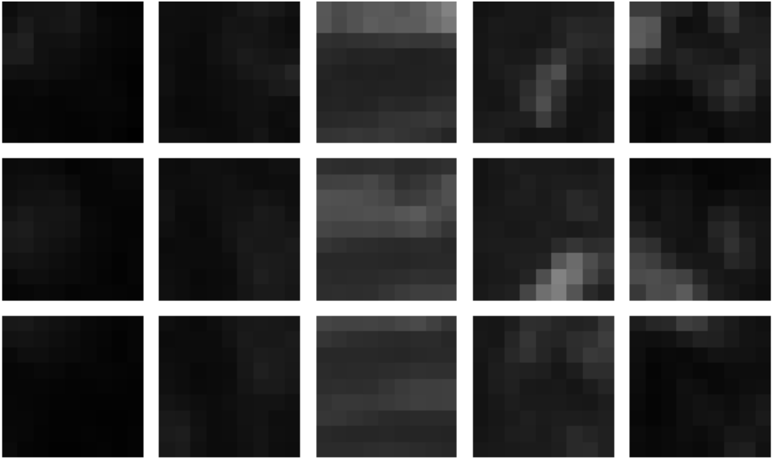
- 200 rektificiranih slika 1382 x 512
- skup za učenje: 80% slika (ostatak - skup za validaciju)
- točni dispariteti izmjereni LIDAR-om u 30% piksela
- gusti dispariteti na automobilima dobiveni fitanjem CAD modela



[orsic17ms]

STEREO : GREŠKE

Mnogi pikseli nemaju korespondenciju zbog "stereoskopske sjene":



[orsic17ms]

Retci prikazuju i) sidro, ii) korespondenciju, iii) najsličniji negativ.

STEREO : TOČNOST

Stereoskopske metode na KITTI-ju uspoređujemo prema postotku točnih dispariteta s tolerancijom ± 3 piksela.

Eksperimentalna točnost je solidna, iako lošija od stanja tehnike:

Model	Točnost - treniranje	Točnost - testiranje
Sive ulazne slike	84.99%	82.32%
Sive ulazne slike - BN	74.51%	71.61%
Ulazne slike u boji	**85.50%**	**82.84%**
Ulazne slike u boji - BN	74.40%	71.33%

[orsic17ms]

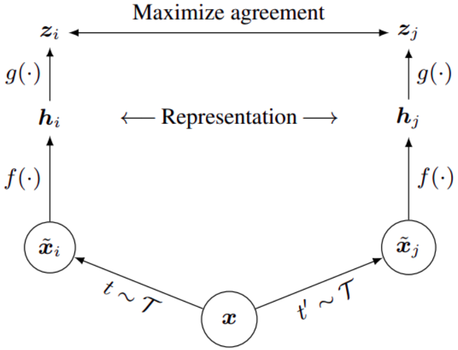
Velika prednost ovakvih pristupa: otpornost na prenaučenost.

Sastavna komponenta novijih pristupa koji rješavaju preostale izazove:

- učenje na neoznačenom videu [liu20cvpr]
- popunjavanje područja bez korespondencija analizom konteksta

SAMONADZIRANJE : ZADATAK

Naučiti korisne reprezentacije bez korištenja semantičkih oznaka.



[simclr20icml]

- f - konvolucijska okosnica
  - npr. ResNet-50
- g - projekcijski modul
  - npr. 2× FC(128)
- $t, t' \in T$ - slučajne perturbacije
  - npr. izrezivanje, rastresanje boje, zaglađivanje

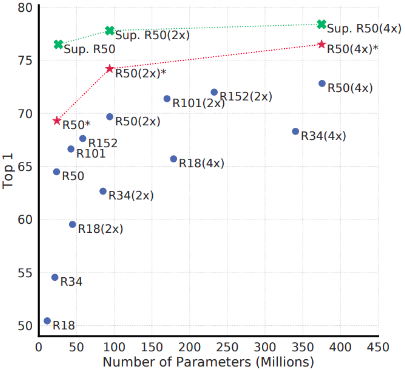
Prednosti: možemo učiti bez oznaka, bolji prijenos učenja!

Nedostatci: dugotrajno učenje, velike grupe, trivijalna rješenja?

SAMONADZIRANJE : SIMCLR

SimCLR: Simple Contrastive Learning of visual Representations

- gubitak N parova: za svaki pozitivni par imamo 2(B-1) negativa
- zaključivanje: odbaciti g, koristiti f za prijenos učenja
- evaluacija: nad f naučiti višerazrednu logističku regresiju
- ostale primjene: predtreniranje + ugađanje, polunadzirano učenje



- veći modeli i veće grupe (4096) bolje uče
- SimCLRv2 + linear gotovo jednako dobar kao i nadzirano učenje
- za najbolje rezultate (*) trebamo 10× duže učiti

SAMONADZIRANJE : SIMCLR (2)

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}, t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^{\top} \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i,j)$ **as** $\ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

[simclr20icml]

Gubitak ima 2B članova:

- svaki član je $\ell_{NP}(i, j)$
  - i : sidro
  - j : pozitiv
  - $k \neq i$: negativi
- $\ell_{NP}(i, j)$ je izražen s obzirom na kosinusne sličnosti s_{ij} i s_{ik}
- negativi su svi preostali podatci minigrupe
  - $k \in [1..2N], \ k \neq i$

Zaključak

Kvantificiranje sličnosti jedan od temeljnih zadataka strojnog učenja

Metrička ugrađivanja primjenjujemo kad klasifikacija nije praktična

- broj razreda prevelik ili unaprijed nepoznat
- malo ili nimalo označenih podataka za učenje

Danas metrička ugrađivanja tipično ostvarujemo dubokim modelima koje učimo s prikladnim kontrastnim gubitcima.

Primjene:

- praćenje osoba, verifikacija lica, stereo
- učenje s malenim brojem označenih primjeraka (few-shot)
- samonadzirano učenje
- detekcija izvandistribucijskih podataka.