Regularizacija dubokih modela

Josip Krapac i Siniša Šegvić

Pregled

- Regularizacija
- Regularizacija normom vektora parametara modela
- Regularizacija generiranjem podataka i unošenjem šuma
- Regularizacija ranim zaustavljanjem
- Polunadzirano i višezadaćno učenje, dijeljenje parametara
- Regularizacija baggingom i dropout

Regularizacija: pregled

Glavni izazov u strojnom učenju: osigurati da model radi dobro ne samo na podacima za učenje nego i na novim podacima.

Tehnike regularizacije: smanjenje greške na skupu za testiranje, uz moguće povećanje greške na skupu za učenje.

Duboki modeli omogućavaju primjenu raznih tehnika regularizacije.

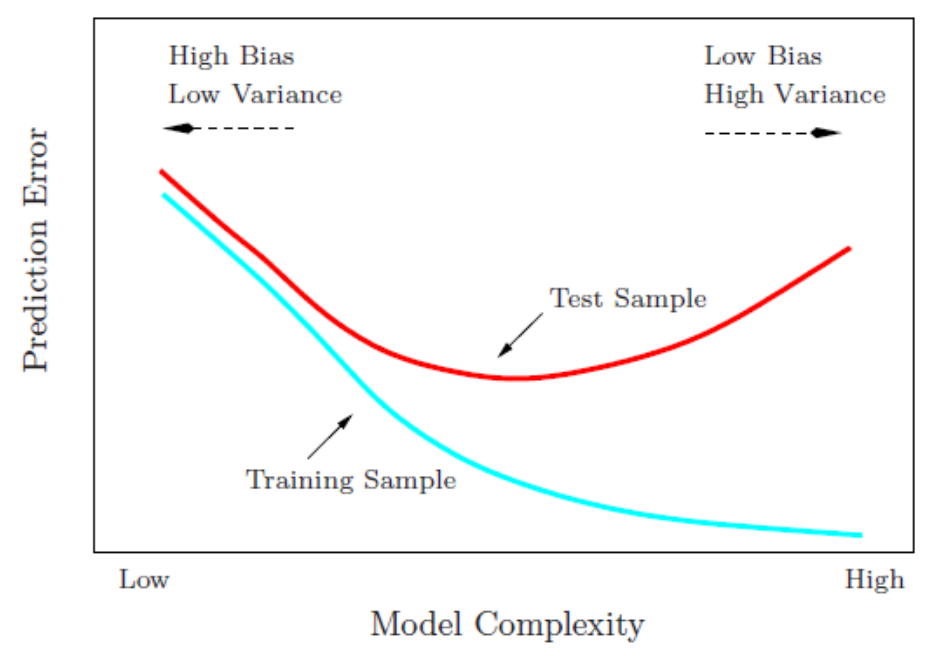
Jedan od najvažnijih otvorenih izazova: vrlo aktivno područje istraživanja.

Regularizacija: pregled

Tehnike regularizacije u strojnom učenju povećavaju pristranosti i smanjuju varijancu modela.

Krajnji cilj: otežavanje prenaučenosti.

U praksi: najbolju generalizacijsku pogrešku postiže model vrlo velikog kapaciteta na koji su primjenjene odgovarajuće tehnike regularizacije.



Pregled

- Regularizacija
- Regularizacija normom vektora parametara modela
- Regularizacija generiranjem podataka i unošenjem šuma
- Regularizacija ranim zaustavljanjem
- Polunadzirano i višezadaćno učenje, dijeljenje parametara
- Regularizacija baggingom i dropout

Regularizacija normom vektora parametara modela

Jedna od najstarijih metoda regularizacije: modificirati gubitka $J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y})$ dodavanjem norme vektora parametara $\Omega(\boldsymbol{\Theta})$:

$$\widetilde{J}(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(\boldsymbol{\Theta})$$

$\alpha \in [0, \infty]$ određuje relativni doprinos regularizatora Ω.

Minimizacija regularizirane funkcije gubitka $\widetilde{J}$ smanjuje i J i Ω.

Obično regulariziramo samo **težine**, tj. Ω ne djeluje na **pomak**.

Odabirom funkcije Ω preferiramo određene klase modela.

Intuicija: regularizator povlači vektor težina $\boldsymbol{w}$ prema ishodištu.

Regularizacija L_2 normom vektora parametara modela

Promotrimo funkciju cilja regulariziranu s $\Omega(\boldsymbol{\Theta}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$

$$\widetilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) + \frac{\alpha}{2}\|\boldsymbol{w}\|_2^2$$
$$\nabla_{\boldsymbol{w}}\widetilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) + \alpha \boldsymbol{w}$$

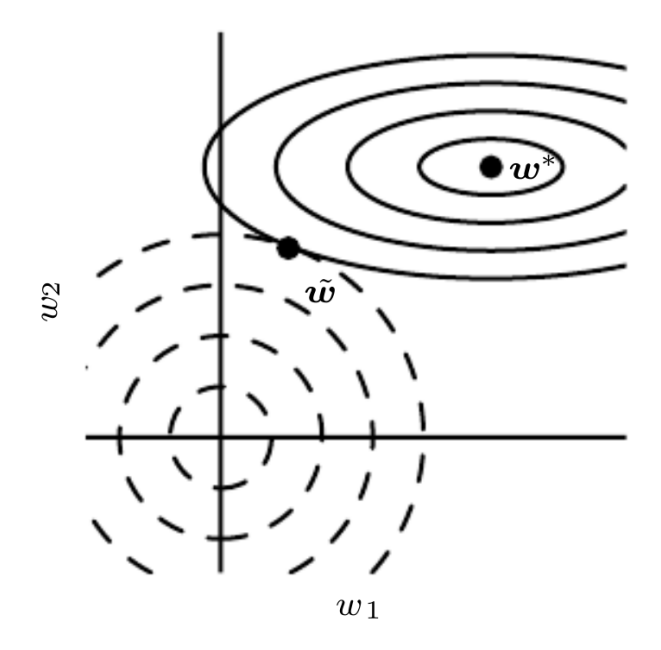
Korak gradijentnog spusta sada je (”weight decay”):

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \epsilon\alpha\boldsymbol{w}^t - \epsilon\nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})|_{\boldsymbol{w}=\boldsymbol{w}^t}$$
$$\boldsymbol{w}^{t+1} = (1 - \epsilon\alpha)\boldsymbol{w}^t - \epsilon\nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})|_{\boldsymbol{w}=\boldsymbol{w}^t}$$

Regularizacija L_2 normom vektora parametara modela

Razvijmo ne-regulariziranu funkciju gubitka J u Taylorov red oko minimuma $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} J(\boldsymbol{w})$:

$$\widehat{J}(\boldsymbol{w}) = J(\boldsymbol{w}^*) + \frac{1}{2}(\boldsymbol{w} - \boldsymbol{w}^*)^\top \boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}^*)$$



Regularizacija L_2 normom vektora parametara modela

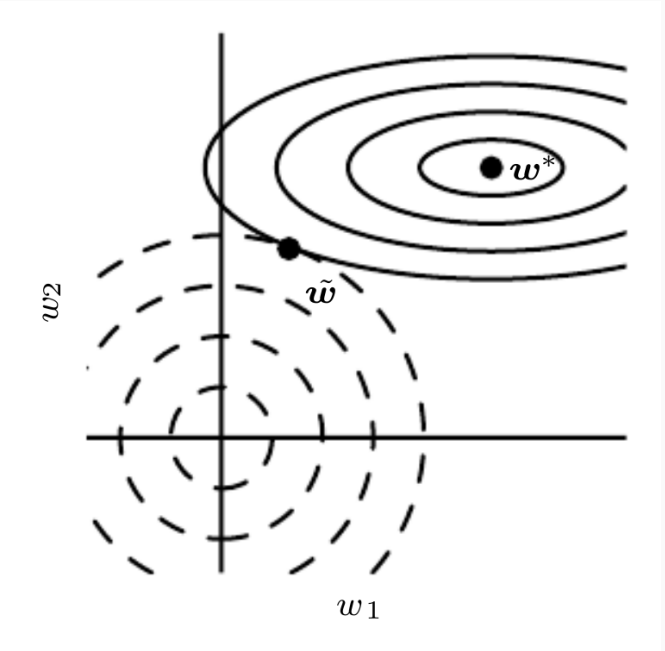
Dodajmo regularizacijski član i pogledajmo gdje se pomakne minimum

$$\nabla_{\boldsymbol{w}} \left(\widehat{J}(\widetilde{\boldsymbol{w}}) + \frac{\alpha}{2} \|\widetilde{\boldsymbol{w}}\|_2^2) \right) = 0$$

$$\nabla_{\boldsymbol{w}} \widehat{J}(\widetilde{\boldsymbol{w}}) + \alpha \widetilde{\boldsymbol{w}} = 0$$

$$\boldsymbol{H}(\widetilde{\boldsymbol{w}} - \boldsymbol{w}^*) + \alpha \widetilde{\boldsymbol{w}} = 0$$

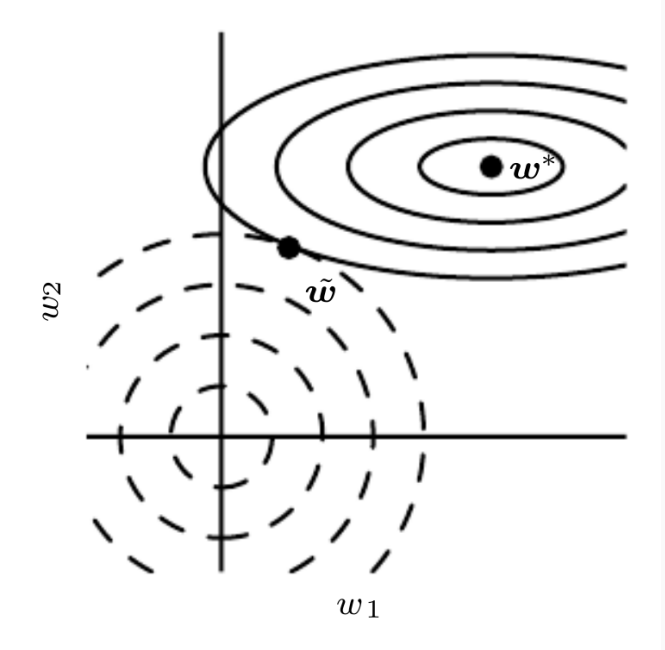
$$\widetilde{\boldsymbol{w}} = (\boldsymbol{H} + \alpha \boldsymbol{I})^{-1} \boldsymbol{H} \boldsymbol{w}^*$$



Regularizacija L_2 normom vektora parametara modela

Pogledajmo što se dešava kada α raste. Uvid je lakši u prostoru razapetom svojstvenim vektorima ($\boldsymbol{Q}$) matrice $\boldsymbol{H} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top$

$$
\begin{aligned}
\tilde{\boldsymbol{w}} &= (\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q} + \alpha\boldsymbol{I})^{-1}\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}\boldsymbol{w}^* \\
&= \left[\boldsymbol{Q}(\boldsymbol{\Lambda} + \alpha\boldsymbol{I})\boldsymbol{Q}^\top\right]^{-1}\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^* \\
&= \boldsymbol{Q}(\boldsymbol{\Lambda} + \alpha\boldsymbol{I})^{-1}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^* \\
&= \boldsymbol{Q}\cdot\mathrm{diag}\left(\frac{\lambda_i}{\lambda_i + \alpha}\right)\cdot\boldsymbol{Q}^\top\boldsymbol{w}^*
\end{aligned}
$$



Regularizacija L_2 normom vektora parametara modela

Pogledajmo doprinos projekcije $\boldsymbol{w}^*$ na i-ti svojstveni vektor $\boldsymbol{H}$:

$$\widetilde{\boldsymbol{w}}^{(i)} = \boldsymbol{q}_i \frac{\lambda_i}{\lambda_i + \alpha} \boldsymbol{q}_i^\top \boldsymbol{w}^*$$

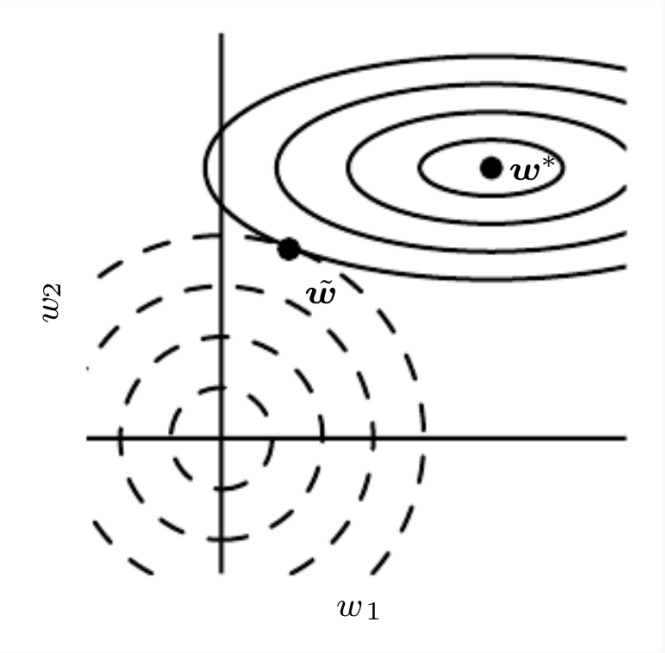
Vidimo da $\widetilde{\boldsymbol{w}}^{(i)}$ postaje tim različitiji od $\boldsymbol{w}_i^*$ što je λ_i manji, odnosno što je funkcija cilja manje strma u smjeru $\boldsymbol{q}_i$

Regularizacija L_2 normom vektora parametara modela

Za slučaj dijagonalne matrice $\boldsymbol{H}$ ($\boldsymbol{Q} = \boldsymbol{I}$):

$$\tilde{\boldsymbol{w}} = \text{diag}(\frac{\lambda_i}{\lambda_i + \alpha}) \cdot \boldsymbol{w}^*$$

$$\tilde{w}_i = \frac{\lambda_i}{\lambda_i + \alpha} w_i^*$$



Vidimo da $\tilde{w}_i$ postaje tim različitiji od w_i što je λ_i manji, odnosno što se $\widehat{J}$ manje mijenja po odgovarajućoj osi koordinatnog sustava

Regularizacija identificira parametre koji ne utječu na funkciju cilja i priteže ih prema ishodištu

Regularizacija L_2 normom vektora parametara modela

Pogledajmo utjecaj L_2 regularizacije na linearnu regresiju sa srednjom kvadratnom pogreškom odstupanja kao funkcijom cilja.

Rješenje u ne-regulariziranom slučaju: $\boldsymbol{w} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$

Rješenje u regulariziranom slučaju: $\boldsymbol{w} = (\boldsymbol{X}^\top \boldsymbol{X} + \alpha \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$

Matrica $\boldsymbol{X}^\top \boldsymbol{X}$ je proporcionalna kovarijacijskoj matrici $\frac{1}{n} \boldsymbol{X}^\top \boldsymbol{X}$.

Efekt regularizacije: prividno povećavanje varijance podataka. Kao da smo oko svakog podatka $\boldsymbol{x}$ generirali nove podatke izvlačenjem iz normalne distribucije s srednjom vrijednosti koja odgovara podatku $\boldsymbol{x}$ i varijancom $\alpha \boldsymbol{I}$.

Regularizacija L_1 normom vektora parametara modela

L_1 regularizator: $\Omega(\boldsymbol{\Theta}) = \sum_i |w_i| = \|\boldsymbol{w}\|_1$.

Promotrimo gradijent regularizirane funkcije cilja:

$$\nabla_{\boldsymbol{w}} \widetilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) + \alpha \text{sgn}(\boldsymbol{w})$$

Korak gradijentnog spusta je:

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \epsilon\alpha \text{sgn}(\boldsymbol{w}^t) - \epsilon \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})|_{\boldsymbol{w}=\boldsymbol{w}^t}$$

Doprinos regularizacije ovisi samo o predznaku $\boldsymbol{w}$.

Regularizacija L_1 normom vektora parametara modela

Isto kao i za L_2 normu: promatramo rastav u Taylorov red u minimumu ne-regularizirane funkcije gubitka, dodamo regularizacijski član i promatramo gdje se pomakne minimum.

Dodatno, pretpostavljamo da je matrica $\boldsymbol{H} = \boldsymbol{\Lambda}$ dijagonalna:

$$\widetilde{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \sum_i \left(\frac{1}{2} \lambda_i (w_i - w_i^*)^2 + \alpha |w_i| \right)$$

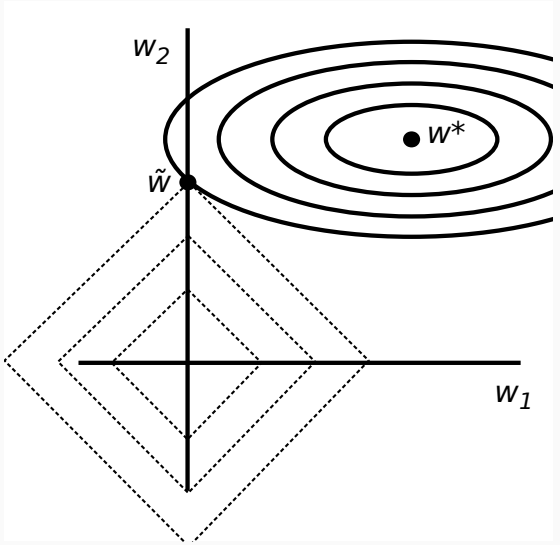
Za ovaj optimizacijski problem postoji analitičko rješenje:

$$\widetilde{w}_i = \mathrm{sgn}(w_i^*) \max \left\{ |w_i^*| - \frac{\alpha}{\lambda_i}, 0 \right\}$$

Regularizacija L_1 normom vektora parametara modela

Rješenje L1-regulariziranog kvadratnog problema:

$$\widetilde{w}_i = \text{sgn}(w_i^*) \max\left\{|w_i^*| - \frac{\alpha}{\lambda_i}, 0\right\}$$

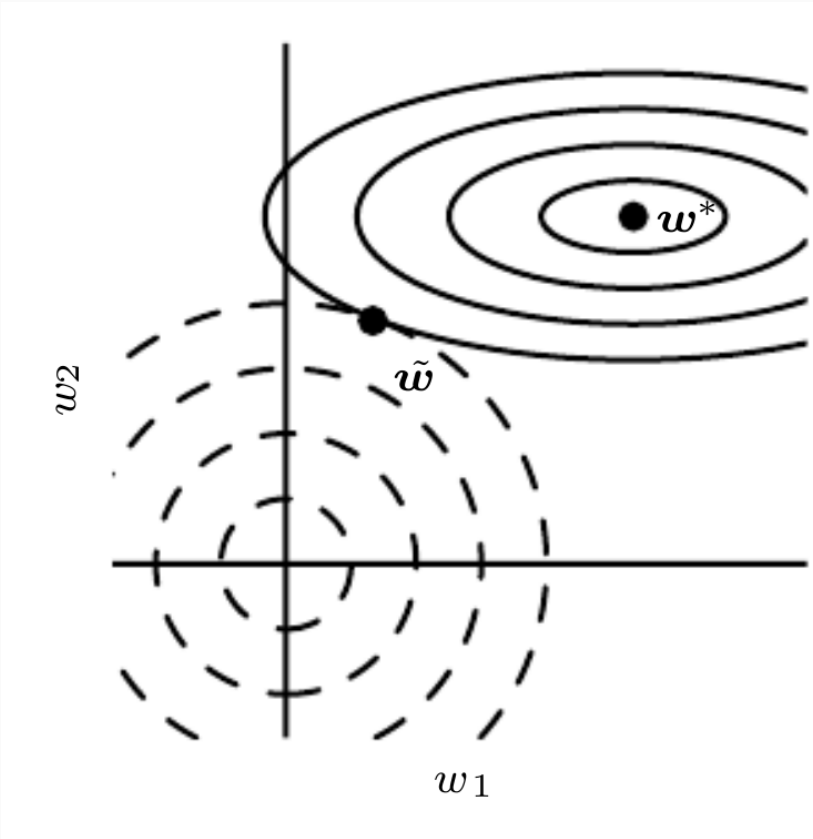


L_1 regularizacija vodi na rijetke modele: modeli za koje su neke vrijednosti parametara 0.

L_1 regularizacija istovremeno uči model i obavlja selekciju/eliminaciju značajki.

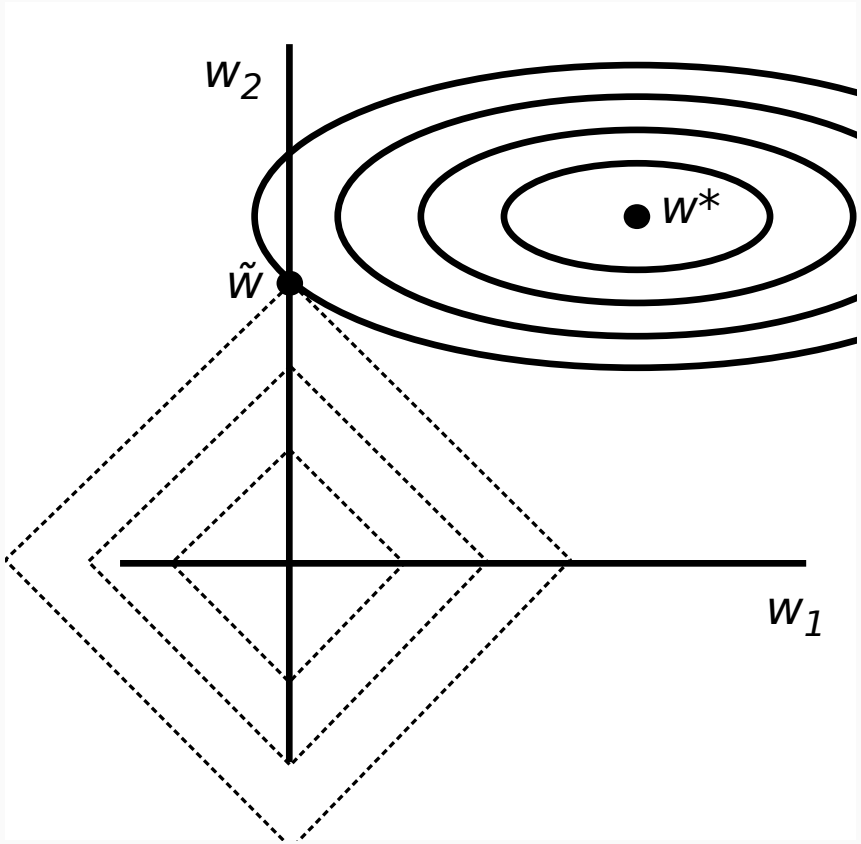
Usporedba L_1 i L_2 regularizacije

L_2



$$\widetilde{w}_i = \frac{\lambda_i}{\lambda_i + \alpha} w_i^*$$

L_1



$$\widetilde{w}_i = \text{sgn}(w_i^*) \max\left\{|w_i^*| - \frac{\alpha}{\lambda_i}, 0\right\}$$

Obje regularizacije guraju w_i prema ishodištu, i to tim jače što je w_i manje važan za podatkovni gubitak (tj. tim jače što je λ_i - manji)

Norme vektora parametara kao ograničenja optimizacijskog postupka

Alternativni pogled: regularizacija osigurava da parametri budu unutar kugle $\Omega(\boldsymbol{\Theta}) < k$.

Radijus k ovisan je o parametru α: veći α znači manji k i obratno.

Oblik kugle ovisi o korištenoj normi (npr. L_1, L_2)

Regularizaciju možemo izraziti i direktno preko k. Modifikacija optimizacijskog postupka:

- napravimo korak gradijentnog spusta i
- projiciramo ažurirani $\boldsymbol{\Theta}$ na najbližu točku koja zadovoljava ograničenje $\Omega(\boldsymbol{\Theta}) < k$.

Norme vektora parametara kao ograničenja optimizacijskog postupka

Ova formulacija ne mijenja funkciju gubitka.

- Promjena funkcije gubitka može uzrokovati da optimizacijski postupak zapne u dijelu prostora koji odgovara malim vrijednostima parametara Θ.
- U ovakvoj formulaciji to se izbjegava budući se projekcija obavlja tek kada vektor parametara naraste dovoljno da izađe iz kugle.

Ova formulacija omogućava stabilniji optimizacijski postupak:

- kod korištenja velikih koraka učenja moguće je da postupak počne divergirati (vektor parametara počinje rasti) zbog pozitivne povratne veze.
- U ovakvoj formulaciji nekontroliran rast nije moguć.

Pregled

- Regularizacija
- Regularizacija norme vektora parametara modela
- Regularizacija generiranjem podataka i unošenjem šuma
- Regularizacija ranim zaustavljanjem
- Polunadzirano i višezadaćno učenje, dijeljenje parametara
- Regularizacija baggingom i dropout

Regularizacija generiranjem podataka

Podatci za učenje mogu se promatrati kao odličan regularizator.

Neki problemi dozvoljavaju jednostavno generiranje podataka za učenje modifikacijom postojećih podataka za učenje.

Ova tehnika se pokazala jako uspješnom kod raspoznavanja slika (translatiranje, skaliranje, rotacija) i za raspoznavanje govora.

Pretpostavka: modifikacija ne utječe na ishod predikcije.

Danas je modificiranje podataka popularno za učenje reprezentacija sa zamjenskim gubitkom na neoznačenim podatcima [kolesnikov19cvpr]

- npr. model se uči da pogodi parametar modifikacije (rotaciju)...
- ...ili da razlikuje modificirani podatak od drugih podataka itd.

Regularizacija unošenjem šuma

Unošenje šuma u model također ima regularizacijski efekt.

Šum možemo primijeniti na:

- ulaz modela (podatke iz skupa za učenje),
- reprezentacije u skrivenim slojevima,
- parametre modela,
- oznake u skupu za učenje (eng. label smoothing).
  - Ako pretpostavimo da je vjerojatnost točnog označavanja $1 - \epsilon$ onda one-hot oznak transformiramo:

$$[0, 0 \cdots 1, 0, \cdots 0] \rightarrow \left[\frac{\epsilon}{k}, \frac{\epsilon}{k}, \cdots, 1 - \frac{k-1}{k}\epsilon, \frac{\epsilon}{k}, \cdots, \frac{\epsilon}{k}\right]$$

Pregled

- Regularizacija
- Regularizacija norme vektora parametara modela
- Regularizacija generiranjem podataka i unošenjem šuma
- Regularizacija ranim zaustavljanjem
- Polunadzirano i višezadaćno učenje, dijeljenje parametara
- Regularizacija baggingom i dropout

Regularizacija ranim zaustavljanjem

Broj epoha je hiper-parametar koji možemo efikasno, inkrementalno validirati u jednoj epizodi učenja

Jedino što moramo promjeniti u odnosu na osnovnu proceduru:

- povremeno evaluirati model na skupu za validaciju
- čuvati parametre koji postižu najbolju validacijsku točnost
- usporavanje učenja uslijed evaluiranja možemo ublažiti:
  - evaluacijom na drugom procesoru
  - smanjenjem skupa za evaluaciju
  - rjeđom validacijom

Nedostatak: moramo imati skup za validaciju, što znači da prije konačnog testiranja trebamo ponoviti učenje na trainval skupu

Regularizacija ranim zaustavljanjem

Algorithm 7.1 The early stopping meta-algorithm for determining the best amount of time to train. This meta-algorithm is a general strategy that works well with a variety of training algorithms and ways of quantifying error on the validation set.

Let n be the number of steps between evaluations.
Let p be the "patience," the number of times to observe worsening validation set error before giving up.
Let $\boldsymbol{\theta}_o$ be the initial parameters.
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_o$
$i \leftarrow 0$
$j \leftarrow 0$
$v \leftarrow \infty$
$\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$
$i^* \leftarrow i$
while $j < p$ **do**
  Update $\boldsymbol{\theta}$ by running the training algorithm for n steps.
  $i \leftarrow i + n$
  $v' \leftarrow \text{ValidationSetError}(\boldsymbol{\theta})$
  if $v' < v$ **then**
    $j \leftarrow 0$
    $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$
    $i^* \leftarrow i$
    $v \leftarrow v'$
  else
    $j \leftarrow j + 1$
  end if
end while
Best parameters are $\boldsymbol{\theta}^*$, best number of training steps is i^*

Zaustavljanje učenja konačnog modela prema broju iteracija

Algorithm 7.2 A meta-algorithm for using early stopping to determine how long to train, then retraining on all the data.

Let $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ be the training set.
Split $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ into $(\boldsymbol{X}^{(\text{subtrain})}, \boldsymbol{X}^{(\text{valid})})$ and $(\boldsymbol{y}^{(\text{subtrain})}, \boldsymbol{y}^{(\text{valid})})$ respectively.
Run early stopping (Algorithm 7.1) starting from random $\boldsymbol{\theta}$ using $\boldsymbol{X}^{(\text{subtrain})}$ and $\boldsymbol{y}^{(\text{subtrain})}$ for training data and $\boldsymbol{X}^{(\text{valid})}$ and $\boldsymbol{y}^{(\text{valid})}$ for validation data. This returns i^*, the optimal number of steps.
Set $\boldsymbol{\theta}$ to random values again.
Train on $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ for i^* steps.

Zaustavljanje učenja konačnog modela prema validacijskoj grešci

Algorithm 7.3 Meta-algorithm using early stopping to determine at what objective value we start to overfit, then continue training until that value is reached.

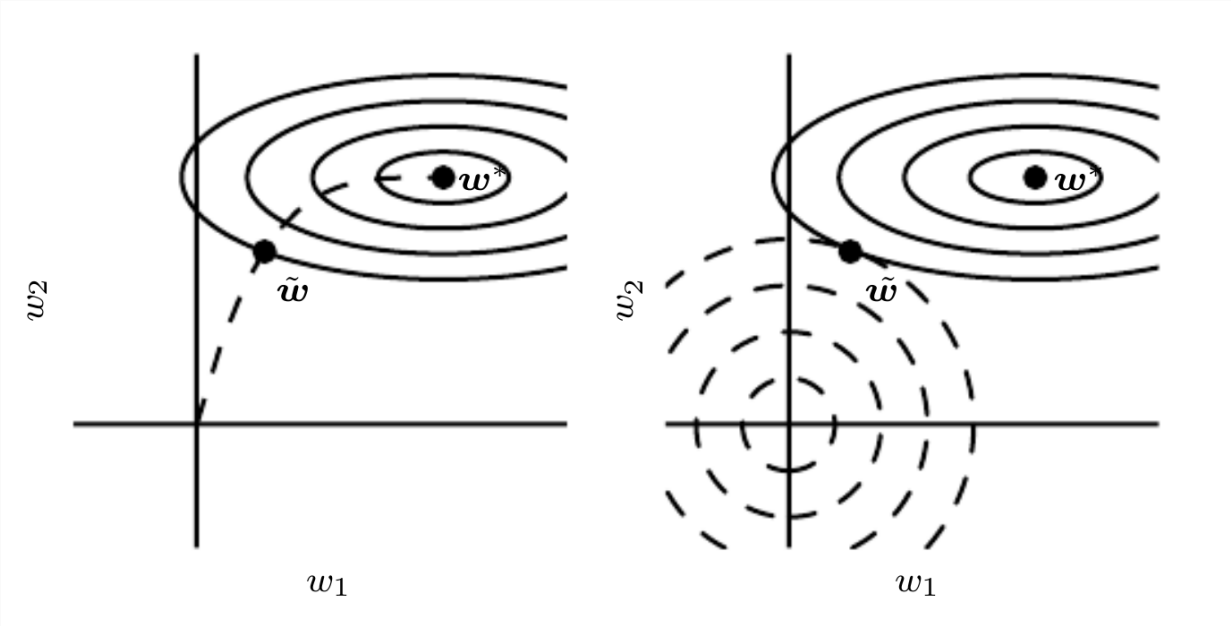
Let $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ be the training set.
Split $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ into $(\boldsymbol{X}^{(\text{subtrain})}, \boldsymbol{X}^{(\text{valid})})$ and $(\boldsymbol{y}^{(\text{subtrain})}, \boldsymbol{y}^{(\text{valid})})$ respectively.
Run early stopping (Algorithm 7.1) starting from random $\boldsymbol{\theta}$ using $\boldsymbol{X}^{(\text{subtrain})}$ and $\boldsymbol{y}^{(\text{subtrain})}$ for training data and $\boldsymbol{X}^{(\text{valid})}$ and $\boldsymbol{y}^{(\text{valid})}$ for validation data. This updates $\boldsymbol{\theta}$.
$\epsilon \leftarrow J(\boldsymbol{\theta}, \boldsymbol{X}^{(\text{subtrain})}, \boldsymbol{y}^{(\text{subtrain})})$
while $J(\boldsymbol{\theta}, \boldsymbol{X}^{(\text{valid})}, \boldsymbol{y}^{(\text{valid})}) > \epsilon$ **do**
  Train on $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ for n steps.
end while

Teoretska analiza ranog zaustavljanja

Pretpostavimo da prođemo kroz τ iteracija postupka gradijentnog spusta, s korakom ϵ, i da je gradijent u svakom koraku ograđen.

Tada je prostor parametara koji možemo dosegnuti iz početne vrijednosti paramatera Θ_0 omeđen.

Veličina kugle ovisi o $\tau\epsilon$: što je ta vrijednost veća, to je kugla veća.



Pregled

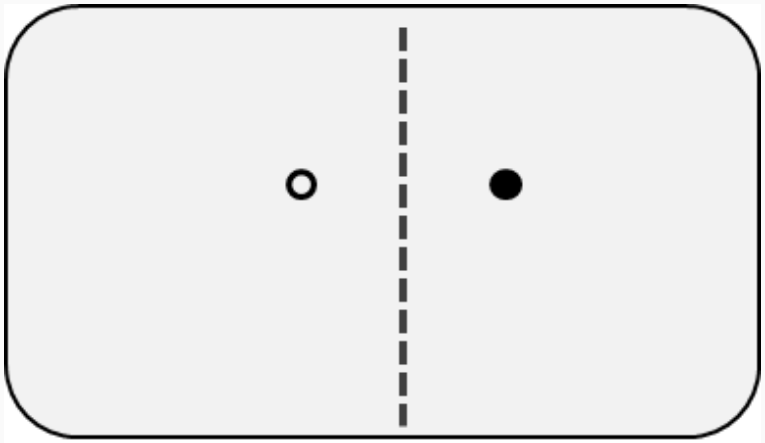
- Regularizacija
- Regularizacija norme vektora parametara modela
- Regularizacija generiranjem podataka i unošenjem šuma
- Regularizacija ranim zaustavljanjem
- Polunadzirano i višezadaćno učenje, dijeljenje parametara
- Regularizacija baggingom i dropout

polunadzirano učenje kao regularizacija

Polunadzirano učenje (eng. semi-supervised learning) je učenje $P(\boldsymbol{y}|\boldsymbol{x})$ koristeći i označene podatke (uzorke iz $P(\boldsymbol{x}, \boldsymbol{y})$) i neoznačene podatke (uzorke iz $P(\boldsymbol{x})$).

Možemo konstruirati model koji minimizira i nadzirani gubitak ($-\log P(\boldsymbol{y}|\boldsymbol{x})$) i nenadzirani gubitak (npr. $-\log P(\boldsymbol{x})$).

Dodatni signal za učenje može dovesti do boljeg semantičkog sadržaja dijeljenih značajki i poboljšati generalizaciju.

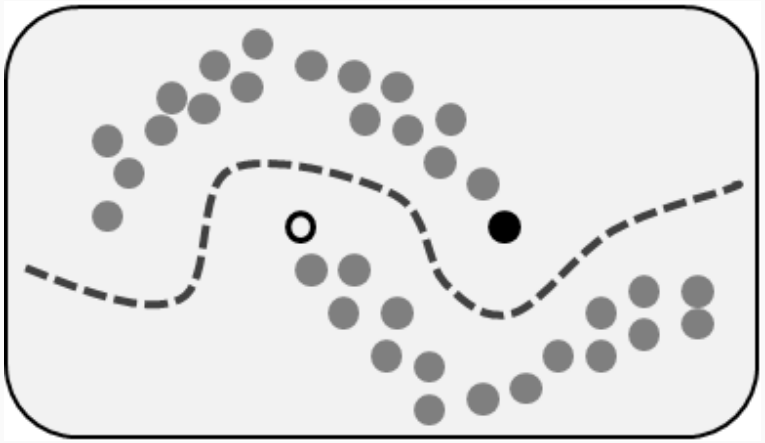


polunadzirano učenje kao regularizacija

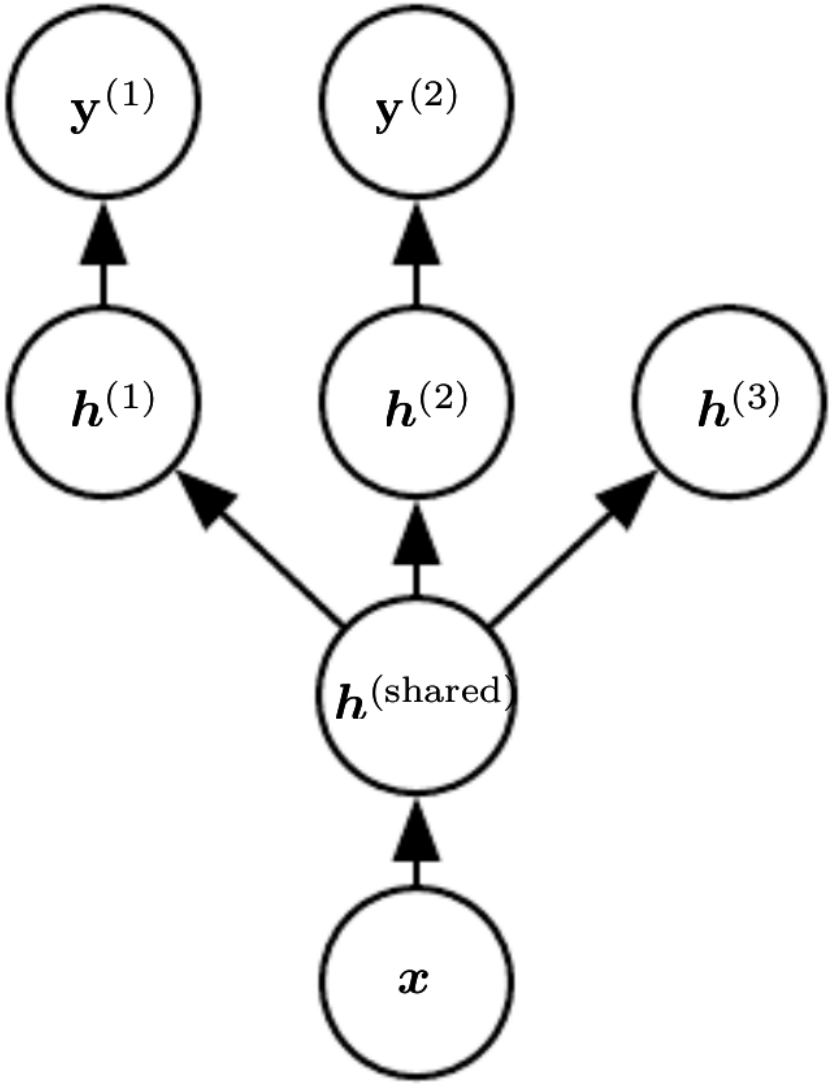
Polunadzirano učenje (eng. semi-supervised learning) je učenje $P(\boldsymbol{y}|\boldsymbol{x})$ koristeći i označene podatke (uzorke iz $P(\boldsymbol{x}, \boldsymbol{y})$) i neoznačene podatke (uzorke iz $P(\boldsymbol{x})$).

Možemo konstruirati model koji minimizira i nadzirani gubitak ($-\log P(\boldsymbol{y}|\boldsymbol{x})$) i nenadzirani gubitak (npr. $-\log P(\boldsymbol{x})$).

Dodatni signal za učenje može dovesti do boljeg semantičkog sadržaja dijeljenih značajki i poboljšati generalizaciju.



Višezadaćno učenje kao regularizacija



Npr:

$y^{(1)}$... $p(\mathsf{c}|\mathsf{x})$

$y^{(2)}$... $p(\mathsf{x})$

$h^{(3)}$ ... latentna reprezentacija (za zamjenski gubitak)

Regularizacija vezanjem i dijeljenjem parametara

Vezanje parametara: ne znamo kakve parametre trebamo, ali znamo da postoji veza između dva problema

$$\Omega(\boldsymbol{w}^{(A)}, \boldsymbol{w}^{(B)}) = \|\boldsymbol{w}^{(A)} - \boldsymbol{w}^{(B)}\|_2^2$$

Dijeljenje parametara: u slučajevima kad želimo da skupovi parametara budu isti. Prednost u odnosu na vezanje parametara: trebamo čuvati samo jedan skup parametara.

Rijetke reprezentacije kao regularizator

- L_1 regularizacija težina $\rightarrow$ rijetki modeli:

$$\begin{bmatrix} 18 \\ 5 \\ 15 \\ -9 \\ -3 \end{bmatrix} = \begin{bmatrix} 4 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & -1 & 0 & 3 & 0 \\ 0 & 5 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & -4 \\ 1 & 0 & 0 & 0 & -5 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ -2 \\ -5 \\ 1 \\ 4 \end{bmatrix}$$

$\boldsymbol{y} \in \mathbb{R}^m$ $\quad$ $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ $\quad$ $\boldsymbol{x} \in \mathbb{R}^n$

$$\widetilde{J}(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha \|\boldsymbol{\Theta}\|_1$$

Rijetke reprezentacije kao regularizator

- L_1 regularizacija aktivacija $\rightarrow$ rijetke reprezentacije:

$$\begin{bmatrix} -14 \\ 1 \\ 19 \\ 2 \\ 23 \end{bmatrix} = \begin{bmatrix} 3 & -1 & 2 & -5 & 4 & 1 \\ 4 & 2 & -3 & -1 & 1 & 3 \\ -1 & 5 & 4 & 2 & -3 & -2 \\ 3 & 1 & 2 & -3 & 0 & -3 \\ -5 & 4 & -2 & 2 & -5 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ -3 \\ 0 \end{bmatrix}$$

$\boldsymbol{y} \in \mathbb{R}^m$ $\qquad$ $\boldsymbol{B} \in \mathbb{R}^{m \times n}$ $\qquad$ $\boldsymbol{h} \in \mathbb{R}^n$

$$\widetilde{J}(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha \|\boldsymbol{h}\|_1$$

Pregled

- Regularizacija
- Regularizacija norme vektora parametara modela
- Regularizacija generiranjem podataka i unošenjem šuma
- Regularizacija ranim zaustavljanjem
- Polunadzirano i višezadaćno učenje, dijeljenje parametara
- Regularizacija baggingom i dropout

Bagging

Bagging (ili **b**ootstrap **agg**regat**ing**) [breiman96ml]: jednostavna ali efikasna metoda za poboljšanje generalizacijske točnosti

Neka je zadan skup za učenje $\boldsymbol{X} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$.

Iz $\boldsymbol{X}$ uzorkujemo M podskupova $\boldsymbol{X}_m = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N'}$

- uzorci se mogu ponavljati (uzorkovanje sa zamjenom)

Na svakom uzorku učimo jednu instancu modela: $f(\boldsymbol{x}, \boldsymbol{\Theta}_m)$, te iz dobivenih instanci formiramo **ansambl**

Zaključivanje provodimo usrednjavanjem predikcija članova ansambla:

$$\hat{\boldsymbol{y}} = \frac{1}{M} \sum_{m=1}^{M} f(\boldsymbol{x}, \boldsymbol{\Theta}_m)$$

Regularizacijski efekt bagginga

Pretpostavimo da smo naučili M skalarnih regresijskih modela:

- opišimo pogrešku i-tog modela varijablom ϵ_i
- pogreška ansambla regresora tada je: $E_a = \frac{1}{M} \sum_i \epsilon_i$

Pretpostavimo zajedničku distribuciju grešaka modela: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

- dijagonalni elementi kovarijacijske matrice: $\mathbb{E}[\epsilon_i^2] = \sigma_{i,i} = v$,
- ne-dijagonalni elementi su: $\mathbb{E}[\epsilon_i \epsilon_j] = \sigma_{i,j} = c$.

Regularizacijski efekt bagginga (2)

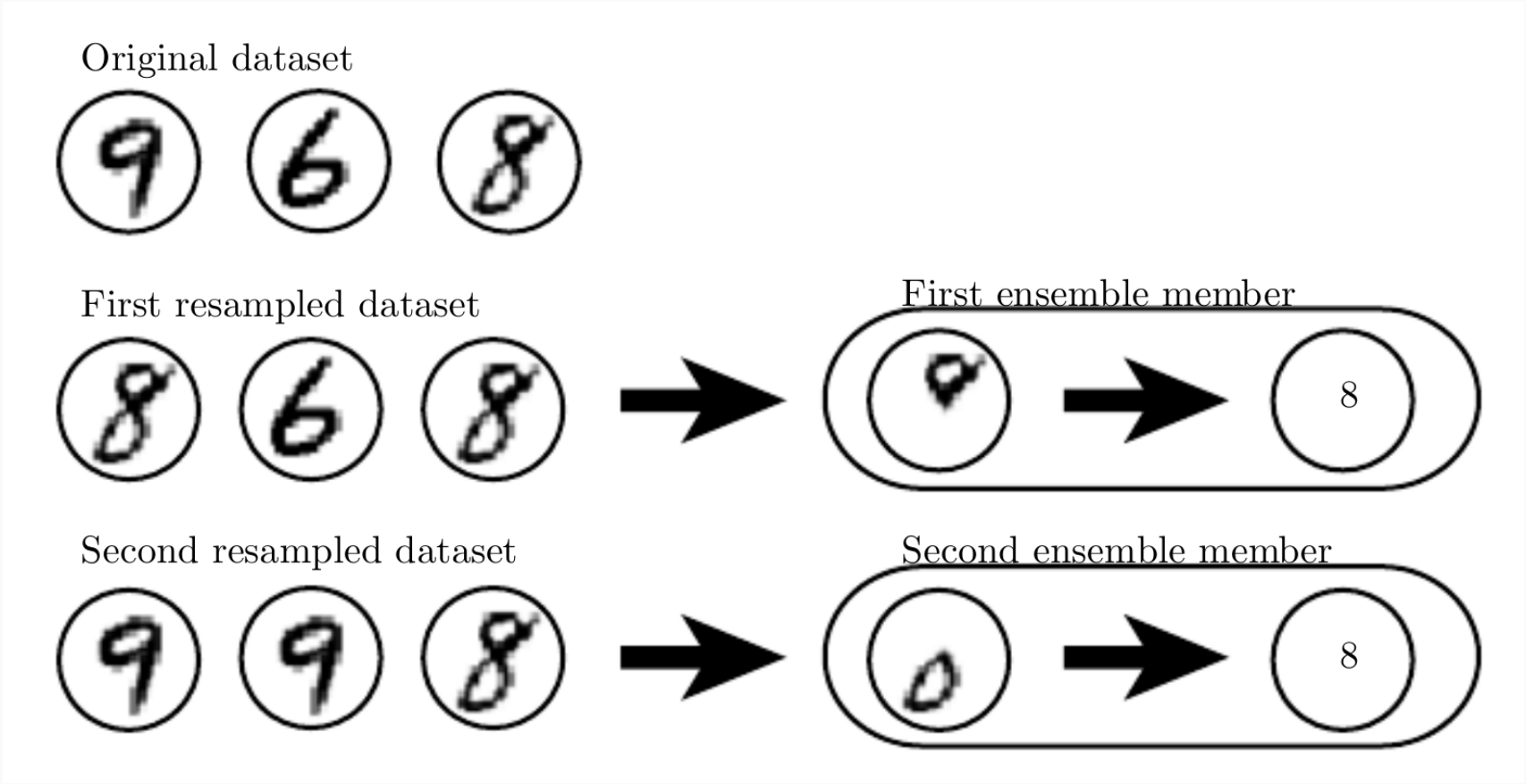
Očekivanje kvadrata pogreške ansambla E_a^2 tada je:

$$\mathbb{E}\left[\left(\frac{1}{M}\sum_i \epsilon_i\right)^2\right] = \frac{1}{M^2}\mathbb{E}\left[\sum_i\left(\epsilon_i^2 + \sum_{i\neq j}\epsilon_i\epsilon_j\right)\right]$$
$$= \frac{1}{M}v + \frac{M-1}{M}c$$

Dva ekstremna slučaja:

1. Greške pojedinih regresora su potpuno korelirane $v = c$: ansambl tada griješi kao i pojedini modeli.
2. Greške pojedinih regresora su potpuno nekorelirane $c = 0$: greška ansambla tada opada linearno s brojem regresora M.

Bagging: ilustracija



Usrednjavanje modela

Bitno je da modeli daju drugačije predikcije, učenje iz raznih podskupova je samo jedan način da se to omogući.

U dubokim modelima različitost možemo postići:

- različitim hiper-parametrima modela i postupka učenja
- različitim slučajnim inicijalizacijama
- različitim rasporedom podataka u mini-grupama

Ti efekti su dovoljni da naučeni modeli rade različite greške.

Usrednjavanje modela u praksi radi jako dobro: to je način kako se dobivaju najbolji rezultati (i pobjeđuje na natjecanjima).

Dropout [srivastava15jmlr]

Tehnika regularizacije dubokih modela:

- srodna ansambliranju, računski manje zahtjevna

Intuicija: efikasni ansambl velikog broja različitih modela.

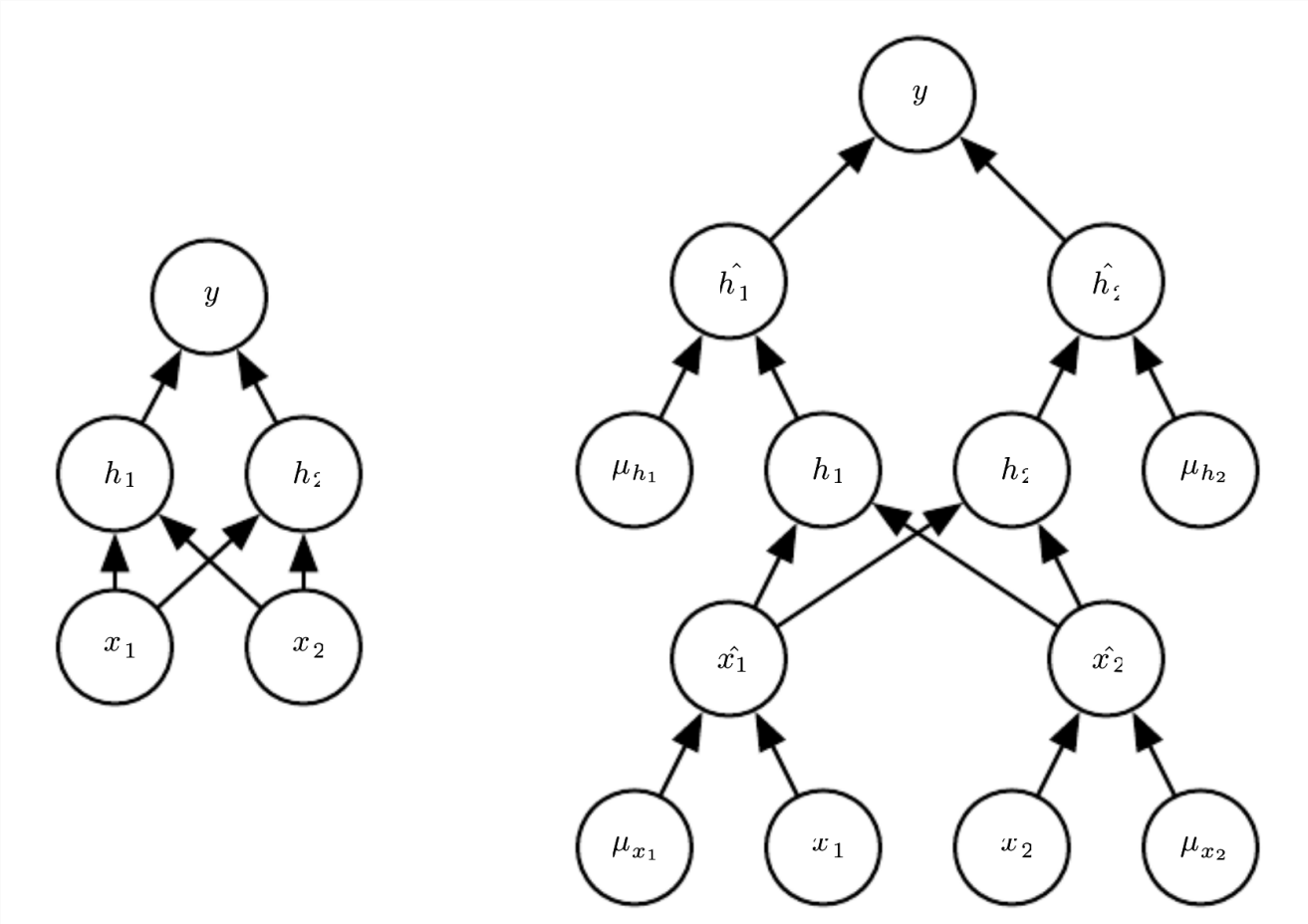
Kako dobiti veliki broj različitih modela?

- iz početnog modela formirati podmodele gašenjem/uklanjanjem aktivacija: latentnih i ulaznih značajki
- u praksi, odabir provodimo množenjem vektora aktivacija sloja $\boldsymbol{x}$ s binarnom slučajnom maskom $\boldsymbol{\mu} \sim B(1, p)^N$:

$$\boldsymbol{x}' = \boldsymbol{x} \odot \boldsymbol{\mu}$$

- tipično, za ulazne značajke $p = 0.8$, a za skrivene značajke $p = 0.5$

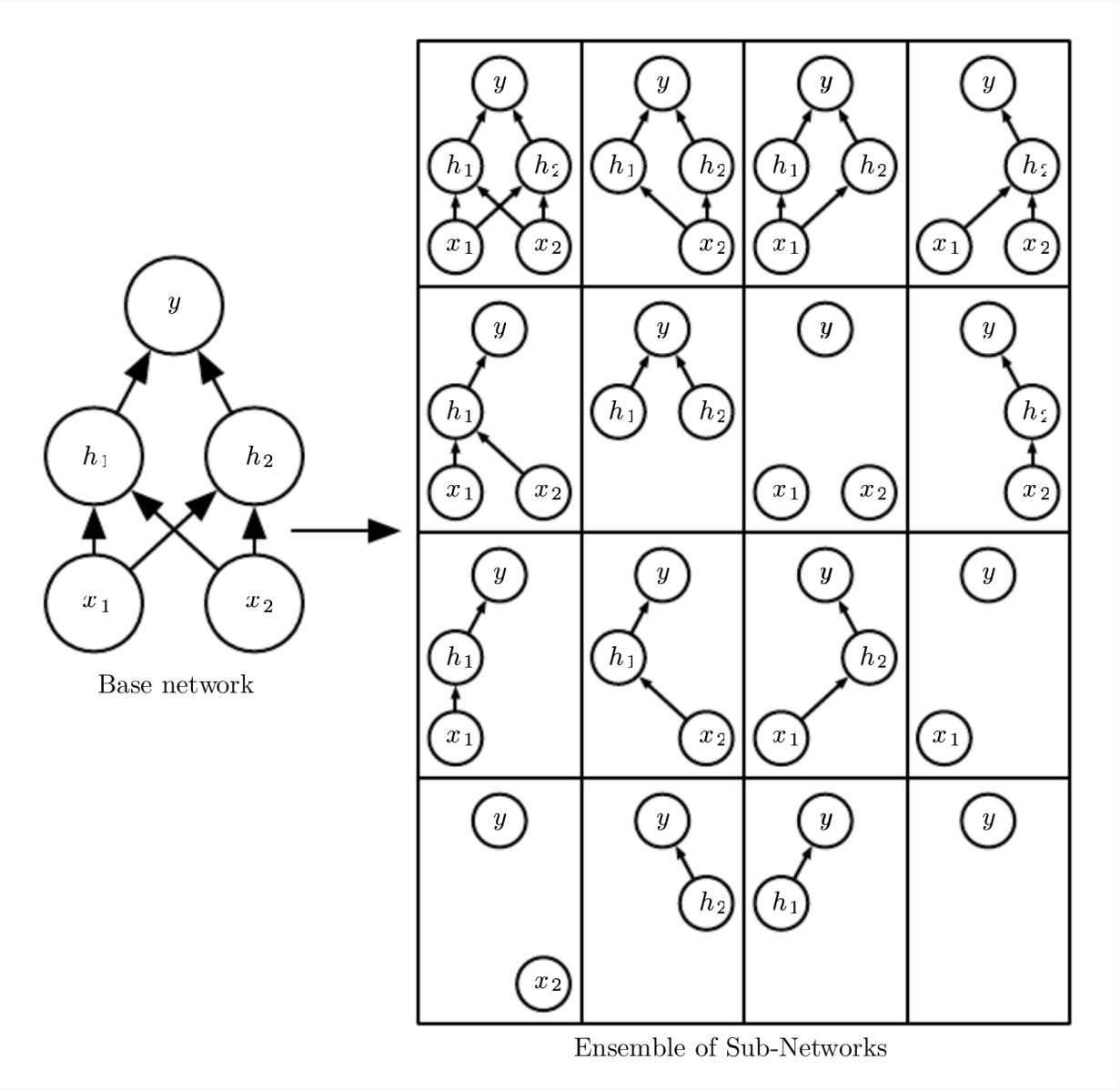
Dropout kao ansambl 2^N modela



$$\boldsymbol{\mu} = [\mu_{x_1}, \mu_{x_2}, \mu_{h_1}, \mu_{h_2}]$$

$$p(\boldsymbol{\mu}) = p(\mu_{x_1})p(\mu_{x_2})p(\mu_{h_1})p(\mu_{h_2})$$

Dropout kao ansambl 2^N modela (2)

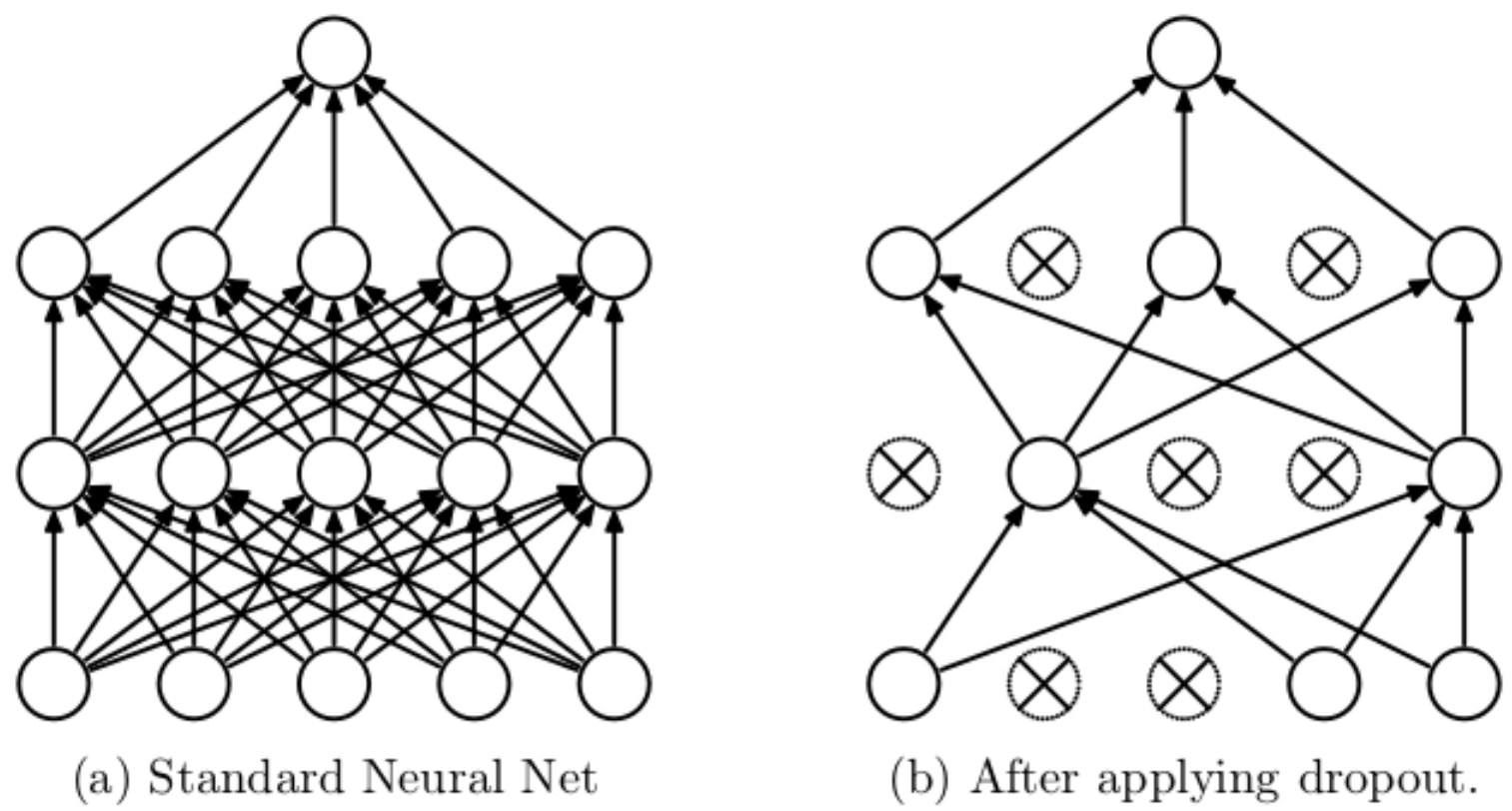


U praksi, N > 100: većina podmodela ima put od ulaza do izlaza

Dropout u fazi učenja

Dropout pristup: zasebno učiti dijelove istog modela

- formalno, minimiziramo: $\mathbb{E}_{\boldsymbol{\mu}} J(\boldsymbol{\Theta}, \boldsymbol{\mu}) = \sum_{\boldsymbol{\mu}} J(\boldsymbol{\Theta}, \boldsymbol{\mu}) p(\boldsymbol{\mu}))$
- eksponencijalnu složenost možemo primiriti uzorkovanjem $\boldsymbol{\mu}$ u svakoj iteraciji učenja (nepristrana procjena gradijenta)



[srivastava15jmlr]

Dropout u fazi ispitivanja

U slučaju bagginga, predikcija ansambla za podatak $\boldsymbol{x}$ je:

$$\frac{1}{M}\sum_{m} p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta}_m)$$

Kod dropouta ansambliramo podmodele:

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta}) = \sum_{\boldsymbol{\mu}} p(\boldsymbol{\mu}) p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\mu}) \tag{1}$$

- prikazani izraz računa **očekivanje** preko distribucije maski $p(\boldsymbol{\mu})$ koje definiraju podmodele
- broj članova u izrazu je 2^N gdje je N broj aktivacija.

Dropout u fazi ispitivanja: uzorkovanje pod-modela

U praksi možemo raditi s M podmodela ($M \ll 2^N$)

Uzorkujemo $\boldsymbol{\mu_m} \sim p(\boldsymbol{\mu})$ i određujemo izlaze podmodela $m \in 1..M$:

$$\hat{\boldsymbol{y}}_m = p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\mu}_m)$$

Konačna predikcija je srednja vrijednost izlaza podmodela:

$$\hat{\boldsymbol{y}} = \frac{1}{M} \sum_{m=1}^{M} \hat{\boldsymbol{y}}_m$$

- problem: za ovo trebamo M unaprijednih prolaza...
- prednost: možemo izračunati varijancu predikcije...

Dropout u fazi ispitivanja: skaliranje težina

Ako aritmetičku sredinu zamjenimo geometrijskom:

$$\widetilde{p}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta}) = \left(\prod_{\boldsymbol{\mu}} p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta}_{\boldsymbol{\mu}}) \right)^{2^{-n}} \quad \ldots$$

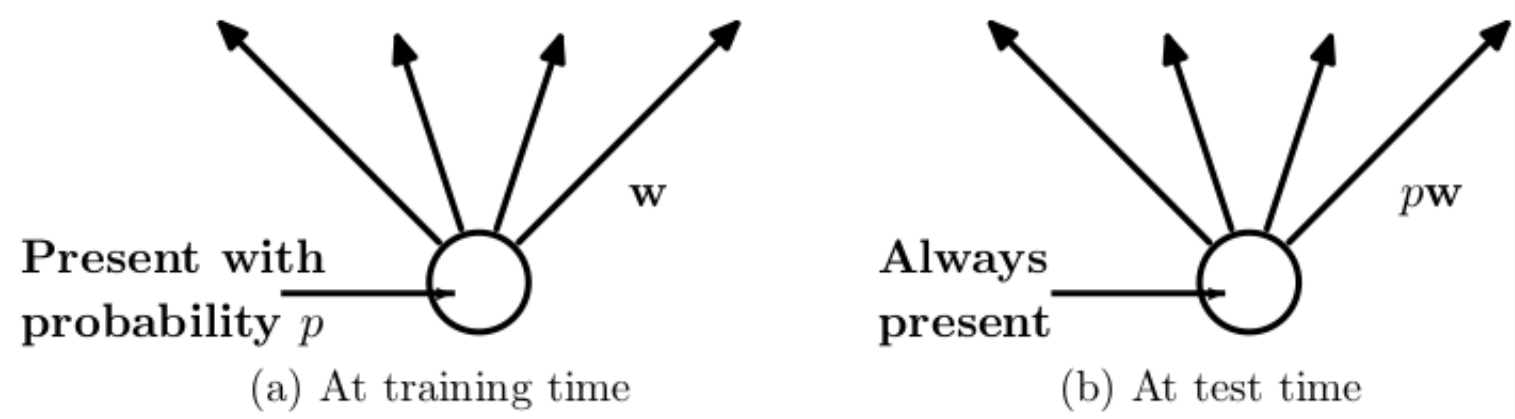
... te re-normaliziramo predikcije na distribuciju preko izlaza:

$$\hat{p}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{\widetilde{p}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta})}{\sum_{c=1}^{C} \widetilde{p}(y_c = 1|\boldsymbol{x}, \boldsymbol{\Theta})}$$

Tada dropout u fazi ispitivanja možemo aproksimirati skaliranjem težina modela.

Dropout u fazi ispitivanja

Izlazne težine aktivacije (= stupci matrice $\mathbf{W}$) množimo s vjerojatnošću njene prisutnosti tijekom učenja: $\boldsymbol{w}'_{h_i} = p(\mu_{h_i})\boldsymbol{w}_{h_i}$



[srivastava15jmlr]

- intuicija: osigurati da očekivani ulaz čvora u fazi ispitivanja bude jednak očekivanom ulazu čvora u fazi učenja
- prednost: jedan umesto M unaprijednih prolaza.

Druga mogućnost: skalirati aktivacije **prilikom učenja**: $h'_i = h_i / p(\mu_{h_i})$

Dropout: prednosti

Moguća primjena na razne tipove modela (konvolucijski, povratni) bez modifikacije funkcije cilja.

Jednostavna kombinacija s ostalim metodama regularizacije.

Računski znatno manje zahtjevan od bagginga

U usporedbi s ne-regulariziranim modelom:

- složenije učenje (više iteracija, npr. $2\times$)
- jednako brzo zaključivanje

Dropout: kada?

Dropout efektivno smanjuje kapacitet modela; najbolji rezultati postižu se povećanjem modela i duljim treniranjem. Za jako velike skupove podataka nije praktičan.

Kod konvolucijskih modela mogu se izbacivati:

- pojedinačne aktivacije
- cijele mape značajki (drop_channel)

Normalizacija nad grupom (eng. batch norm) također ima i regularizacijski efekt: dodavanje aditivnog i multiplikativnog šuma koji ovisi o podacima u mini-grupi. Popularna alternativa dropoutu.

Primjer: skaliranje težina u višerazrednoj logističkoj regresiji

Nema skrivenih slojeva $\Rightarrow$ dropout samo na ulaznim značajkama $\boldsymbol{x}$.

Za višerazrednu logističku regresiju: geometrijska sredina predikcija eksponencijalnog broja pod-modela ekvivalentna **skaliranju težina**

Višerazredna logistička regresija:

$$p(\boldsymbol{y} = y|\boldsymbol{x}, \boldsymbol{\Theta}) = \text{softmax}\left(\boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b}\right)_y$$

Razred pod-modela definiran binarnim vektorom $\boldsymbol{\mu}$:

$$p(\boldsymbol{y} = y|\boldsymbol{x}, \boldsymbol{\Theta}_{\boldsymbol{\mu}}) = \text{softmax}\left(\boldsymbol{W} \cdot (\boldsymbol{\mu} \odot \boldsymbol{x}) + \boldsymbol{b}\right)_y$$

Primjer: skaliranje težina u višerazrednoj logističkoj regresiji

Dokažimo da predikciju eksponencijalno velikog dropout-ansambla uz p=0.5 uistinu možemo dobiti skaliranjem težina!

$$
\begin{aligned}
\widetilde{p}(\boldsymbol{y} = y|\boldsymbol{x}, \boldsymbol{\Theta}) &= \left(\prod_{\boldsymbol{\mu} \in \{0,1\}^n} p(\boldsymbol{y} = y|\boldsymbol{x}, \boldsymbol{\Theta}_{\boldsymbol{\mu}}) \right)^{2^{-n}} \\
&= \frac{\left(\prod_{\boldsymbol{\mu} \in \{0,1\}^n} \exp\left(\boldsymbol{W}_{y,:}(\boldsymbol{\mu} \odot \boldsymbol{x}) + b_y \right) \right)^{2^{-n}}}{\left(\prod_{\boldsymbol{\mu} \in \{0,1\}^n} \sum_{y'} \exp\left(\boldsymbol{W}_{y',:}(\boldsymbol{\mu} \odot \boldsymbol{x}) + b_{y'} \right) \right)^{2^{-n}}} \\
&= C \cdot \left(\prod_{\boldsymbol{\mu} \in \{0,1\}^n} \exp\left(\boldsymbol{W}_{y,:}(\boldsymbol{\mu} \odot \boldsymbol{x}) + b_y \right) \right)^{2^{-n}} \\
&= C \cdot \exp\left(2^{-n} \sum_{\boldsymbol{\mu} \in \{0,1\}^n} \boldsymbol{W}_{y,:}(\boldsymbol{\mu} \odot \boldsymbol{x}) + b_y \right)
\end{aligned}
$$

Primjer: skaliranje težina u višerazrednoj logističkoj regresiji

$$
\begin{aligned}
\widetilde{p}(\boldsymbol{y} = y|\boldsymbol{x}, \boldsymbol{\Theta}) &= C \cdot \exp\left(2^{-n} \sum_{\boldsymbol{\mu} \in \{0,1\}^n} \boldsymbol{W}_{y,:}(\boldsymbol{\mu} \odot \boldsymbol{x}) + b_y\right) \\
&= C \cdot \exp\left(\boldsymbol{W}_{y,:}\left(2^{-n} \sum_{\boldsymbol{\mu} \in \{0,1\}^n} \boldsymbol{\mu}\right) \odot \boldsymbol{x} + 2^{-n} \sum_{\boldsymbol{\mu} \in \{0,1\}^n} b_y\right) \\
&= C \cdot \exp\left(\boldsymbol{W}_{y,:}(\mathbf{0.5} \odot \boldsymbol{x}) + b_y\right) \\
&= C \cdot \exp\left((\boldsymbol{W}_{y,:} \odot \mathbf{0.5}) \cdot \boldsymbol{x} + b_y\right)
\end{aligned}
$$

Lako je pokazati da bi se drugi izbori $p(\boldsymbol{\mu})$ postigli drugačijim iteriranjem kroz podmodele. Tada bismo imali:

$$
\widetilde{p}(\boldsymbol{y} = y|\boldsymbol{x}, \boldsymbol{\Theta}) = C \cdot \exp\left(\left(\boldsymbol{W}^\top \odot p(\boldsymbol{\mu})\right)^\top_{y,:} \boldsymbol{x} + b_y\right)
$$

Zadatak

Pretpostavke:

- 3-dimenzionalni ulaz: $\boldsymbol{x} \in \mathbb{R}^3$.
- dvije klase na izlazu $\boldsymbol{y} \in \mathbb{R}^2$.
- model: višerazredna logistička regresija

$$\boldsymbol{y} = \text{softmax}\left(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}\right)$$

- ulaz modela smo regularizirali dropoutom (bez mijenjanja aktivacija) uz $p(\mu_{x_1}) = 0.2, p(\mu_{x_2}) = 0.5, p(\mu_{x_3}) = 0.7$.

Zadatak: izračunajmo ulaz u softmax (eng. logits) za podatak $\boldsymbol{x} = [1, 1, 1]$, ako su parametri modela nakon učenja bili:

$$\boldsymbol{W} = \begin{bmatrix} -0.1 & 0.4 & -0.6 \\ 0.2 & -0.3 & 0.5 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 0.2 \\ -0.2 \end{bmatrix}$$