

Detekcija objekata u slikama pomoću dubokih neuronskih mreža

Obrada informacija

Izv.prof.dr.sc. Marko Subašić

15.12.2020.

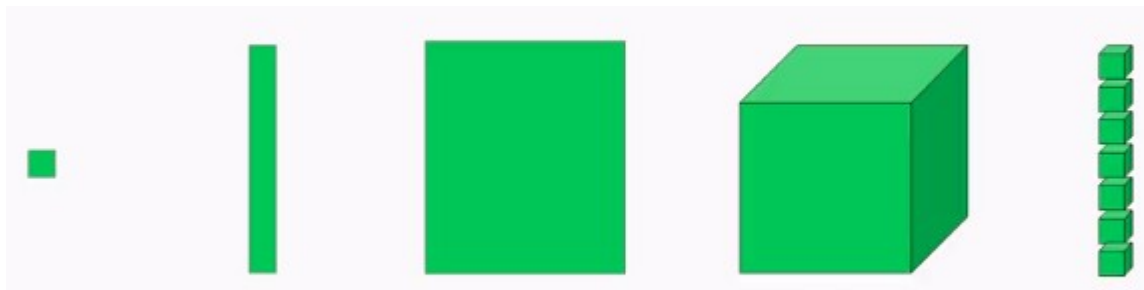
Analiza informacija u slikama

- Zbog čega su slike bogat izvor informacija?
- Gdje se u slikama nalaze korisne informacije?



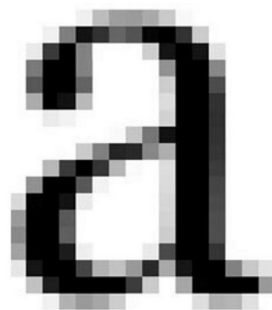
Slike kao izvor informacija

- Slika ima dvije prostorne dimenzije
 - 0D - Jedan broj nosi jednostavnu informaciju (Npr. 4 učenika u učionici)
 - Dva broja daju mogućnost pohrane dvostruko veće količine informacija
 - 1D – Vektor sadrži niz brojeva, no osim informacije sadržane u samim brojevima, bitan je i poredak brojeva
 - Položaj i međusobni **prostorni odnosi nose dio informacije**
 - 2D – Matrica sadrži niz vektora, bitan je i redoslijed vektora...
 - 3D...
 - 4D...



Gdje je korisna informacija u slici

- Sliku možemo promatrati kao matricu
- Svaki element digitalne slike – piksel – sadrži jedan komadić informacije
 - Broj piksela n i broj mogućih vrijednosti piksela m
 - Broj mogućih kombinacija (različitih slika) je m^n
 - Broj piksela možemo promatrati kao broj dimenzija
 - Svaka moguća slika tada je točka u n -dimenzionalnom prostoru



Gdje je korisna informacija u slici

- Ovisno o zadanom problemu, iz slike želimo izvući neku bitnu informaciju
- Za opis te informacije često je dovoljan ograničeni skup brojeva
 - Često samo jedan broj – npr. broj automobila na slici
- Ekstrakcija bitne informacije podrazumijeva redukciju dimenzionalnosti

$$\mathbb{R}^n \rightarrow \mathbb{R}$$

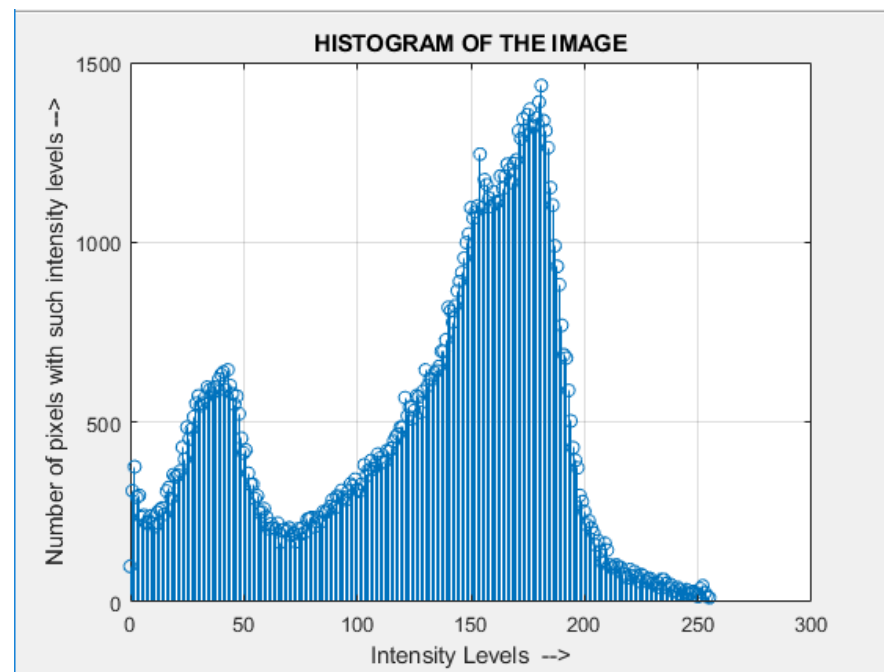
Redukcija dimenzionalnosti

- Kroz redukciju dimenzionalnosti očito odbacujemo neke nebitne informacije
 - U slikama često ima znatna količina nebitnih informacija
 - Čak i jednostavno uklanjanje nebitnih piksela može poslužiti svrsi
 - Neki dijelovi slike često nisu bitni, npr. pozadina



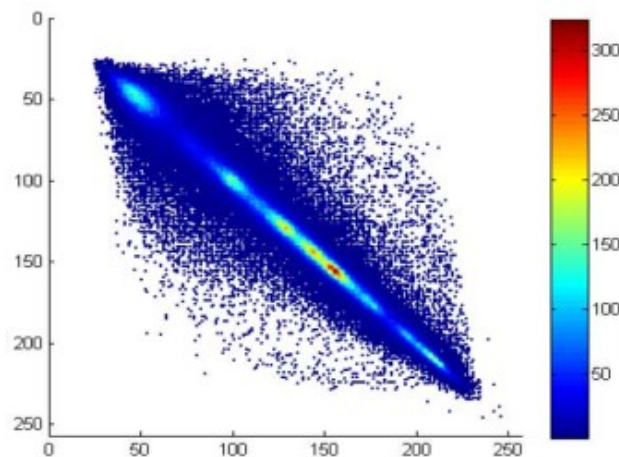
Redukcija dimenzionalnosti

- Postoji velika redundancija u slikama
 - Piksele u slici možemo promatrati kao slučajne varijable



Redukcija dimenzionalnosti

- Statistički gledano dva susjedna piksela u slici su jako korelirani
- Dva susjedna piksela vjerojatno nose istu informaciju – redundancija
 - Susjedni pikseli nose istu informaciju
 - Neke možda možemo izbaciti
 - Smanjivanje dimenzija slike



Redukcija dimenzionalnosti

- Gdje je onda bitna informacija
- U razlikama između susjednih piksela!
 - Visoka korelacija između susjednih piksela ne znači da ne mogu postojati znatne razlike
 - Razlike su rijetke, ali neke od njih su bitne

Bitne informacije u slikama

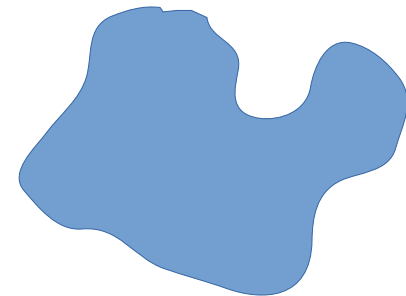
- Bitne su razlike u susjednim pikselima ili u lokalnom susjedstvu
 - Rubovi na slikama
- Potvrda da je to doista bitan izvor informacija
 - Ljudski vizualni sustav je specijaliziran za traženje lokalnih razlika – rubova
 - Isto vrijedi za mnoge životinje

Keypoints I

- Digitalne slike se sastoje od piksela
- Informaciju nose vrijednosti piksela, ali i međusobni prostorni odnosi
- Bitne informacije su lokalizirane

Detekcija objekata u slikama

- Gdje se obično javljaju lokalne promjene intenziteta u slikama?
 - Na granicama objekata
- Kako odrediti lokaciju objekta u slici?
 - Jedna mogućnost je odrediti lokacije svih točaka granice objekta



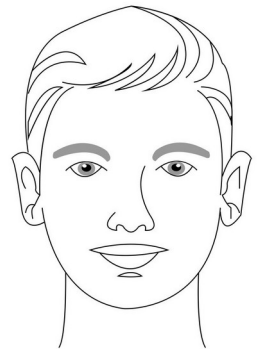
Detekcija objekata u slikama

- Granice objekta ne moraju nužno biti jedini rubovi u slici
- Bitno je odrediti koji rubovi pripadaju granici objekta
 - Prepoznati granicu objekta
- Kada možete prepoznati granicu objekta, možete ga i pronaći na slici



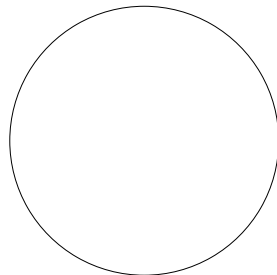
Prepoznavanje objekata u slikama

- Kako prepoznati objekt u slici
 - Po karakterističnom obliku
 - Karakterističan raspored rubnih točaka
 - Npr. oblik lica
 - Po karakterističnim detaljima
 - Opet se svodi na karakterističan raspored rubnih točaka
 - Npr. oči, usta, nos za prepoznavanje lica



Prepoznavanje objekata u slikama

- Ipak, karakterističan raspored rubnih točaka možda nije svojstven samo traženom objektu
 - Takva mogućnost ovisi o pozadini u slici – pogotovo ako u pozadini može biti bilo što
- Potrebno je dodatno razlikovati rubove



Prepoznavanje objekata u slikama

- Osim samog ruba, granicu objekta opisuju i karakteristike regija s obje strane granice
 - Opet se vraćamo na lokalna susjedstva, odnosno razlike u lokalnim susjedstvima



Prepoznavanje objekata u slikama

- Možemo zaključiti da je prepoznavanje objekta dobro raditi kroz prepoznavanje oblika, ali i unutarnjih detalja objekta
- Za obje stvari bitan izvor informacije je lokalno susjedstvo u slici
 - Upućuje da bi mogli koristiti isti alat

Keypoints II

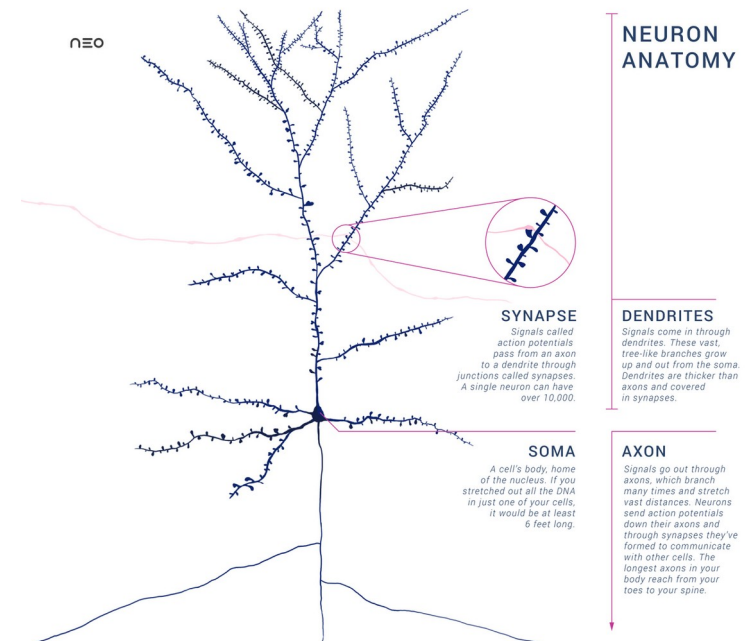
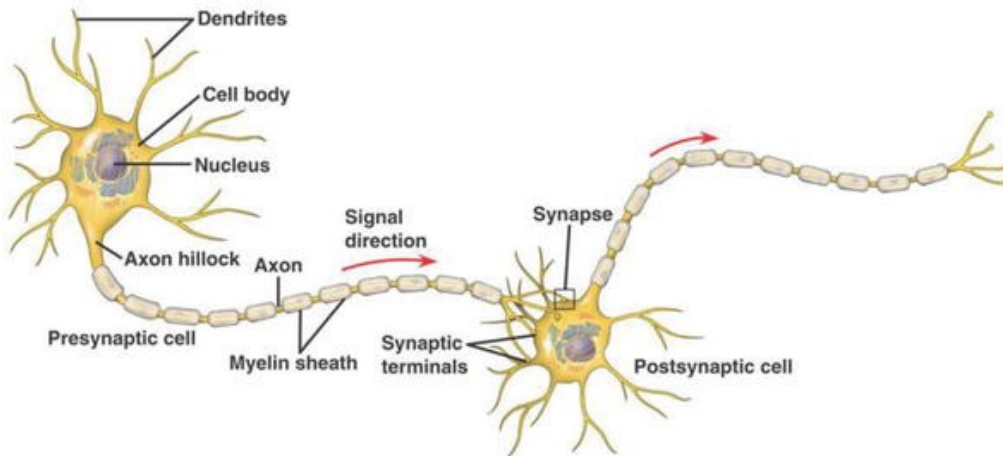
- Za detekciju objekata korisne informacije dolaze od:
 - Rubova objekta
 - Unutrašnjosti objekta
 - Okoline objekta
- Takve informacije nalaze se u lokalnim susjedstvima u slici

Neuronske mreže

- Mreže međusobno povezanih neurona
- Poznati su primjeri iz prirode
- Ljudski mozak sadrži 100 000 000 000 neurona
 - Povezanih u kompleksnu mrežu
 - Mrežu koja može učiti

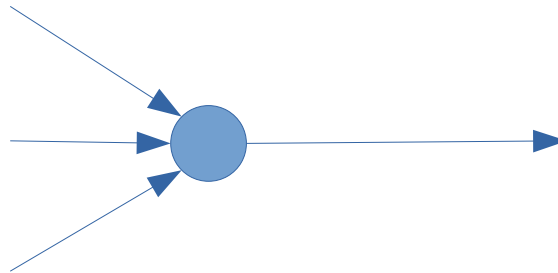
Neuron

- Prirodni neuron
 - Prima informaciju od drugih neurona
 - Prenosi informaciju drugim neuronima



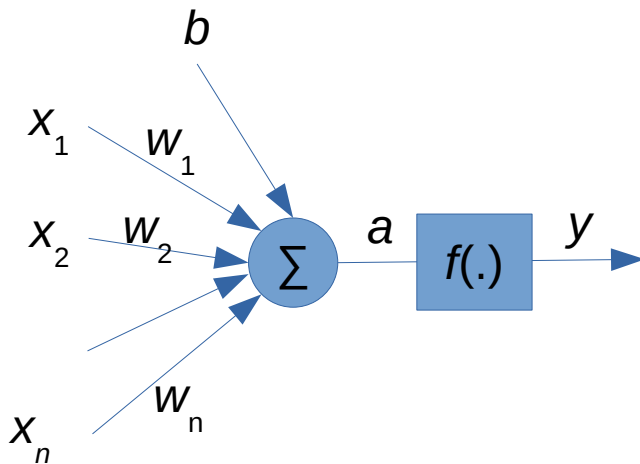
Umjetni neuron

- Pojednostavljeni **model** prirodnog neurona
 - Isto prenosi informaciju



Umjetni neuron

- Druga interpretacija je da je neuron funkcija
 - Na temelju ulaznih vrijednosti generira izlaz
 - Filtriranje ulazne informacije
 - Jednosmjernan tok informacije (*feed forward*)



$$y = f(a) = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

Matematički model umjetnog neurona

- Matematički model je relativno jednostavan
 - Matematička funkcija!
 - Uključuje jednostavne matematičke operacije
 - Zbrajanje
 - Množenje
 - Jednostavnu aktivacijsku funkciju
 - Može se izvoditi na računalu
 - Sadrži i neke parametre

Matematički model umjetnog neurona

- Parametri uključuju težine kojima se množe ulazne vrijednosti w_i
- Tu je i pomak b
- Tu je i broj pribrojnika u sumi n
- Parametri aktivacijske funkcije f

$$y = f(a) = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

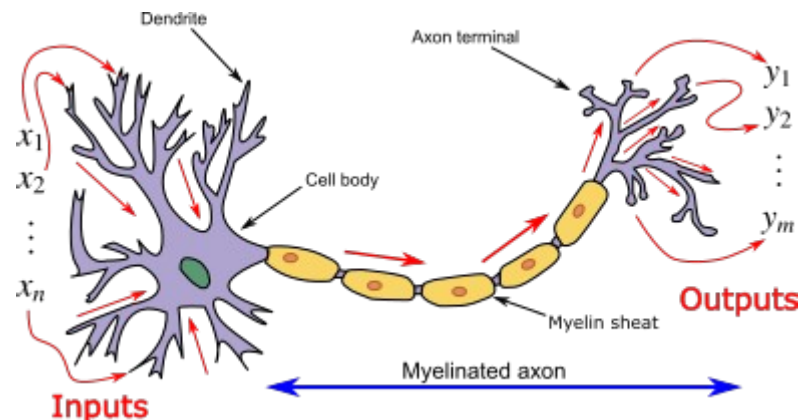
Matematički model umjetnog neurona

- Intuitivno je jasno kako se ulazna informacija transformira u izlaznu
- Ostaje pitanje učenja

$$y = f(a) = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

Učenje biološkog neurona

- Učenje bioloških mreža je kompleksno
- Neuroni se ne mijenjaju
- Učenje se odvija u sinapsama
- Učenje umjetnih neuronskih mreža daje uvid u učenje bioloških neuronskih mreža



Učenje umjetnog neurona

- Kako naučiti jedan neuron?
- Što ga naučiti?
- Što uopće može naučiti?
- Što neuron uopće radi?

Koga učimo?

- Pogledajmo opet matematički model

$$y = f(a) = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

- Prema modelu, neuron na temelju ulaznih vrijednosti računa jednu izlaznu vrijednost
 - w_i – težine
 - b – pomak
 - a – aktivacija

Jednostavniji zapis

- Suma umnožaka ulaza i težina se može prikazati ako unutarnji produkt vektora

$$\sum_{i=1}^n w_i x_i = \mathbf{w}^T \mathbf{x}$$

- Često se pomak b skriva u vektor težina

$$w_0 = b, x_0 = 1$$

$$a = \sum_{i=0}^n w_i x_i = \mathbf{w}^T \mathbf{x}$$

$$\mathbf{w} = \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}$$

Što učimo?

- Učenjem bi mogli postići da neuron „bolje” računa izlaznu vrijednost
- Učenje bi mogli provesti kroz mijenjanje parametara neurona tako da nam omoguće bolji izlaz
- Možemo odabrati koje parametre ćemo mijenjati/učiti

Što mijenjamo?

- Odabir parametara bi trebao biti usklađen sa postupkom učenja

$$y = f \left(\sum_{i=1}^n w_i x_i + b \right)$$

- Ne moramo mijenjati sve parametre
 - Za neke nije praktično izvedivo
 - Ne mogu se svi učiti na isti način

Što želimo postići?

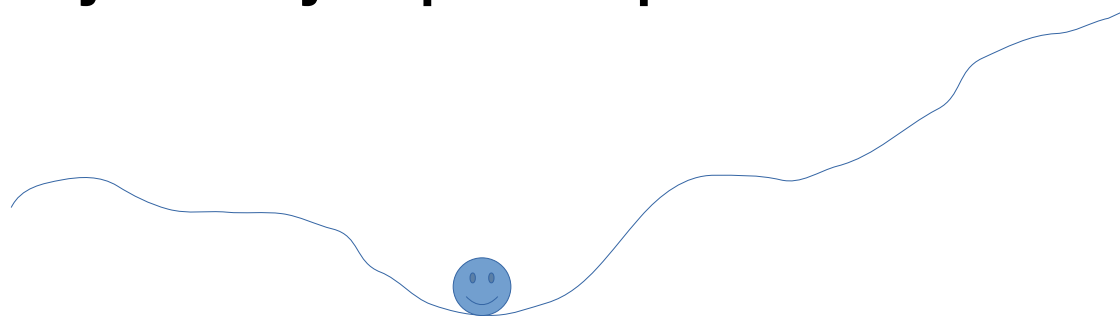
- Definirajmo konkretan cilj
 - Želimo da neuron na izlazu računa točnu vrijednost
 - Procjena izlazne vrijednosti na temelju ulaznih vrijednosti
 - Problem regresije
 - Dakle želimo da je greška minimalna
 - Dakle želimo minimizirati grešku

Minimizacija pogreške

- Minimizirajmo grešku e naše funkcije

$$e = d - y = d - f\left(\sum_{i=1}^n w_i x_i + b\right)$$

- Greška je isto funkcija
- Traženje minimuma funkcije je problem koji je u matematici jako dobro proučen
 - Postoje brojni postupci minimizacije funkcije



Učenje neurona

- Zašto ne bismo pretražili prostor parametara i pronašli globalni minimum
- S imalo većim brojem parametara problem postaje jako nepraktičan

Učenje neurona

- Matematički, postizanje našeg cilja svodi se na problem optimizacije
- Tražimo optimalno rješenje
- Optimalni neuron griješi minimalno
 - Po tome je optimalan

Učenje neurona

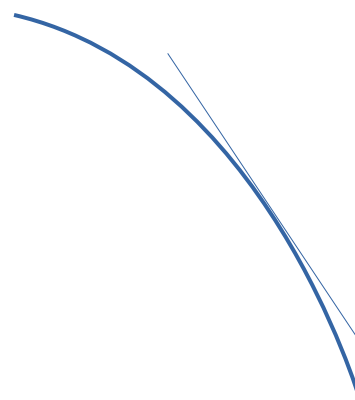
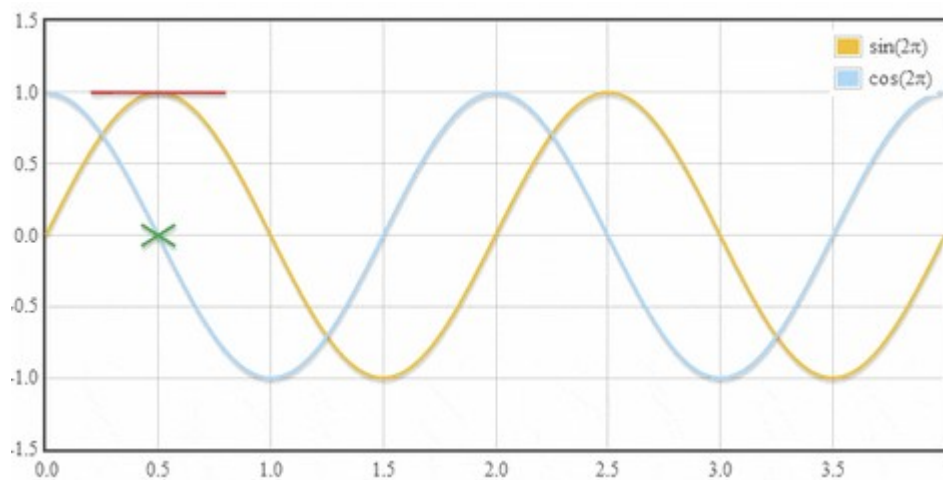
- Jedan princip optimizacije je slijeđenje gradijenta
- Gradijent u nekoj točki funkcije nam govori koji je smjer maksimalne promjene funkcije
- Jednom kada nam je taj smjer poznat, možemo krenuti u smjeru smanjenja (minimizacije)

Keypoints III

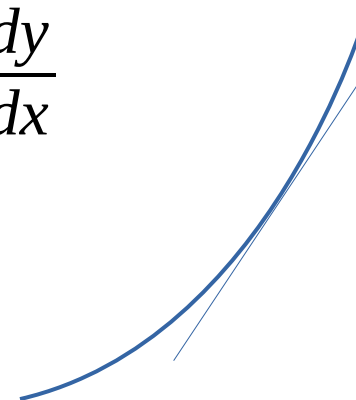
- Učenjem želimo poboljšati funkcioniranje neuronske mreže
- Poboljšanjem funkcioniranja jednog neurona, poboljšavamo funkcioniranje mreže
- Jedan princip učenja je minimizacija pogreške
 - Rezultat učenja je optimalan

Učenje neurona

- Ilustracija 1D slučaja
- Derivacije u nekoj točki nam daju gradijent



$$\frac{dy}{dx}$$



Učenje neurona

- Kako iskoristiti gradijent?
- Koristimo parcijalne derivacije funkcije greške s obzirom na odabrani parametar
- Korekcija je proporcionalna parcijalnoj derivaciji
- Suprotan smjer – negativan predznak

$$w_i = w_i + \Delta w_i \quad \Delta w_i = -\eta \frac{\partial e}{\partial w_i}$$

$$e = d - y = d - f\left(\sum_{i=1}^n w_i x_i + b\right)$$

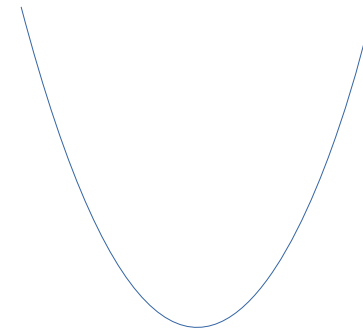
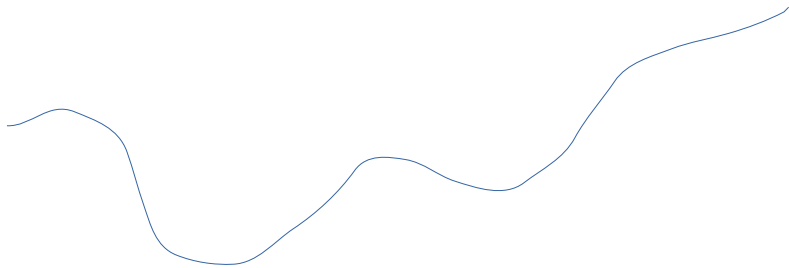
- η – faktor učenja

Koraci učenja

- Jednom kada odredimo gradijent, znamo u kojem smjeru treba ići
 - Gradijent pokazuje u smjeru povećanja pogreške
- Slijedeći gradijent se krećemo po funkciji greške i tražimo globalni minimum
 - Ne znamo unaprijed gdje je globalni minimum
- Kada je gradijent jednak nuli – stigli smo na cilj
 - Nismo ako se radi o lokalnom minimumu

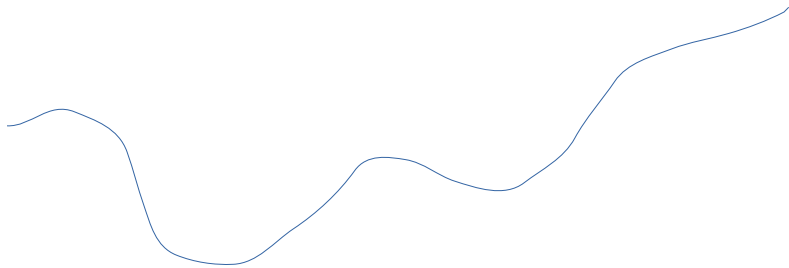
Koraci učenja

- Informacija koju nam daje gradijent je strogo lokalna
 - Smjer za smanjivanje pogreške vrijedi samo u toj točki funkcije
 - Izvan te točke ne znamo ništa o funkciji i njezinom gradijentu
 - Slijeđenje gradijenta radim uz pretpostavku da je to dobro
 - Pretpostavku možemo potvrditi kroz poznavanje karakteristika funkcije greške
 - Poželjno je da je glatka, konveksna...



Koraci učenja

- Problemi slijeđenja gradijenta
 - Ako napravimo preveliki korak, možemo preskočiti traženi minimum
 - Zbog prevelikih koraka nikada nećemo pronaći minimum
 - Premali koraci usporavaju pronalazak minimuma
 - Premalim koracima ne možemo pobjeći iz lokalnog minimuma



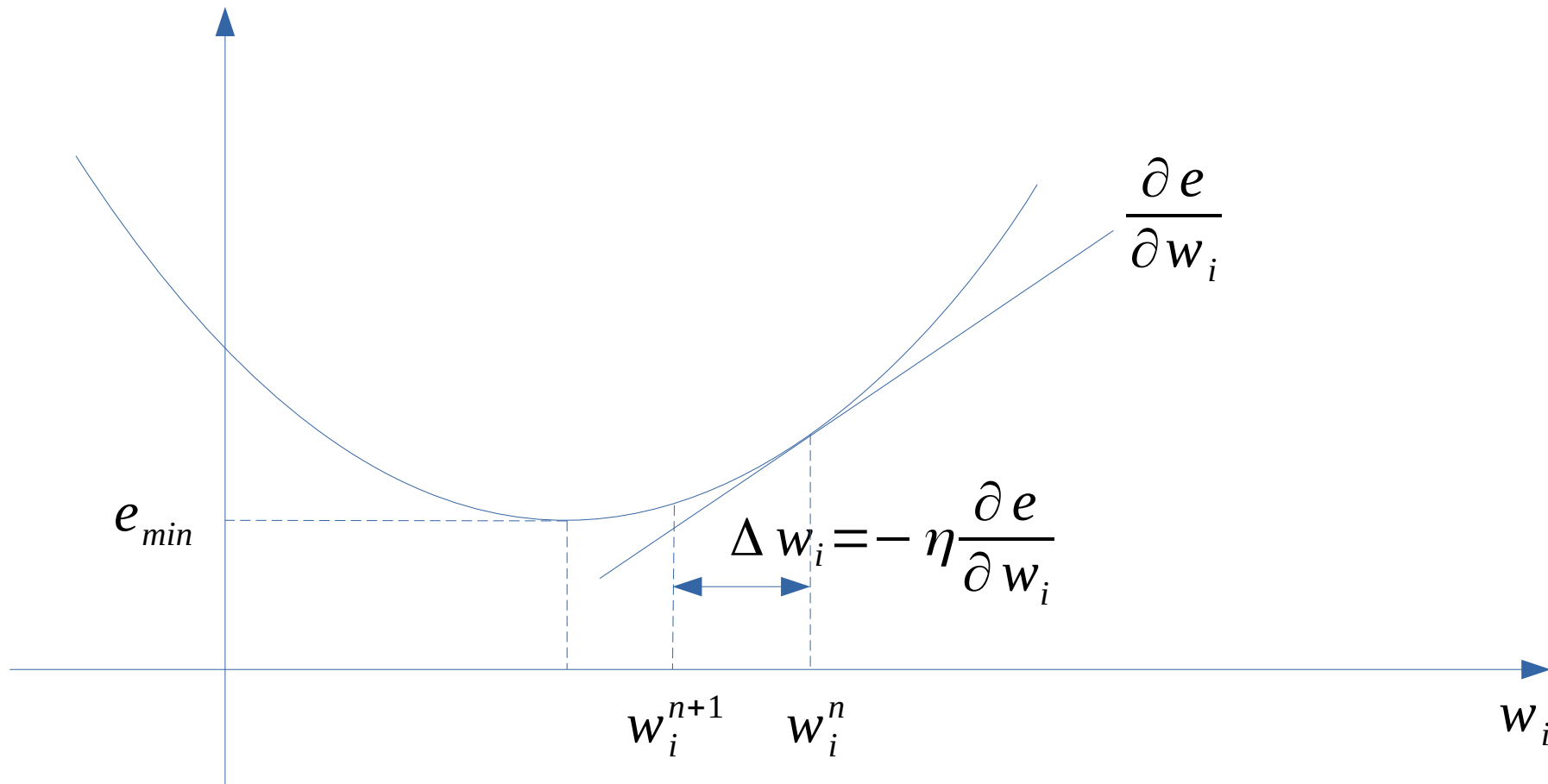
Koraci učenja

- „Veličinu koraka” učenja određuje koeficijent učenja η

$$\Delta w_i = -\eta \frac{\partial e}{\partial w_i}$$

- Potrebno je pronaći idealni kompromis
 - Dovoljno veliki korak da možemo pobjeći iz lokalnog minimuma i da konvergencija bude brza
 - Dovoljno mali korak da ne preskočimo globalni minimum
- Idealni koeficijent učenja je nemoguće odrediti u naprijed
- Postoje pokazatelji kada odabir nije dobar

Koraci učenja



Koraci učenja

- Osim odabira fiksnog η , možemo probati:
 - Krenuti sa velikim η pa ga smanjivati
 - Prvo se radi grubo pretraživanje prostora
 - Nakon što u grubo nađemo minimum, krećemo sa finijim pretraživanjem
 - Periodičke oscilacije η
 - ...

Koraci učenja

- Algoritam

- 1) Odredimo inicijalnu točku u prostoru parametara
- 2) Odredimo gradijent funkcije greške u trenutnoj točki
- 3) Ako je gradijent jednak nuli, završavamo
- 4) Pomaknemo se u novu točku prateći smjer suprotan od smjera gradijenta
- 5) Povratak na 2)

Koraci učenja

- Gradijent očekivane funkcije pogreške ponekad nije moguće odrediti
 - Trebalo bi uzeti u obzir sve uzorke
- Rješenje je u aproksimativnoj estimaciji gradijenta za jedan ili manji broj uzoraka
 - Stochastic gradient descent (SGD)

Koraci učenja

- SGD algoritam

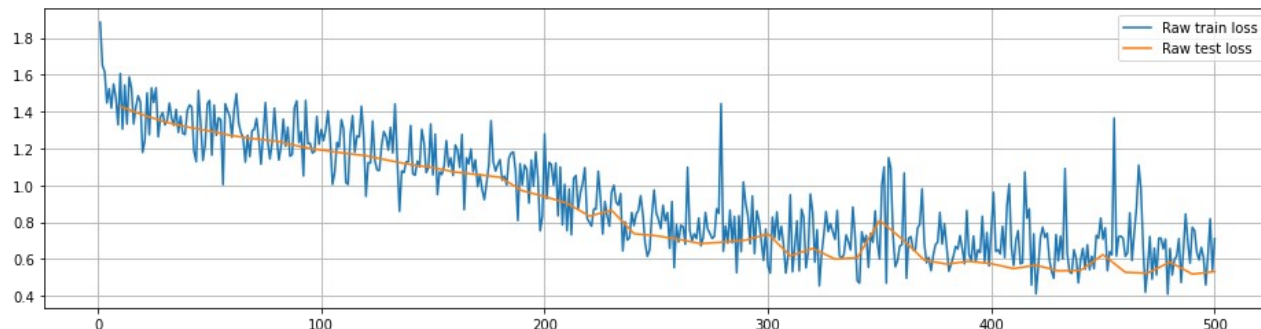
1) Odredimo inicijalnu točku u prostoru parametara

2) Odredimo jednostavnu estimaciju gradijenta funkcije greške za novi uzorak

3) Ako je gradijent jednak nuli, završavamo

4) Pomaknemo se u novu točku prateći smjer suprotan od smjera estimacije gradijenta

5) Povratak na 2)



Širi cilj

- Vratimo se sada na naš cilj
- Ne zanima nas da neuron računa točan izlaz za jedan ulaz
 - To se može trenutno postići trivijalnim postupkom
- Zanima nas da neuron radi točno za veći broj različitih ulaznih uzoraka

Zajednička greška

- Želimo da neuron dobro radi za veći broj ulaznih uzoraka

- Možemo si za cilj postaviti minimizaciju očekivane greške

$$E = \frac{1}{N} \sum e$$

$$E = \frac{1}{N} \sum |e|$$

- Otvoreno je pitanje kako definirati funkciju greške

- Možda kombinacija više kriterija

Zajednička greška

- Za kompleksnije probleme ne možemo očekivati da će greška za svaki ulazni uzorak biti 0

– Puno funkcija greške bi moglo dovesti do tog cilja

$$E = \sum |e| \quad E = \sum e^2 \quad E = (\sum e^2)^2$$

- Ako znamo da ne možemo dobiti savršeno rješenje, možemo birati kakvo će to nesavršeno rješenje biti

Dodatni ciljevi

- Odabir funkcije greške nam daje mogućnost dodatnog izbora
 - Često se radi o kompromisima
 - Brzina pronalaska minimuma – stabilnost
 - Eliminacija velikih grešaka – očekivana greška
 - ...
 - Bitno je i pitanje puta do cilja
 - Put nam određuje konačnu destinaciju

Funkcija greške

- Fokuseranje na veće pogreške

$$|e| \quad \frac{\pm \partial e}{\partial w}$$

$$e^2 \quad 2e \frac{\partial e}{\partial w}$$

$$e^4 \quad 4e^3 \frac{\partial e}{\partial w}$$

Funkcija greške

- Fokuseranje na veće pogreške

$$E = \sum |e| \qquad \frac{\partial E}{\partial w} = \sum \frac{\pm \partial e}{\partial w}$$

$$E = \sum e^2 \qquad \frac{\partial E}{\partial w} = \sum 2e \frac{\partial e}{\partial w} \qquad E = \frac{1}{2} \sum e^2$$

$$E = \sum e^4 \qquad \frac{\partial E}{\partial w} = \sum 4e^3 \frac{\partial e}{\partial w}$$

Derivacije funkcije greške

- Vratimo se na pojedine greške

$$f(a) = a \quad e = d - y = d - \sum_{i=0}^n w_i x_i \quad \frac{\partial e}{\partial w_j} = 0 - \sum_{i=0}^n \frac{\partial (w_i x_i)}{\partial w_j} = -x_j$$

$$f(a) = ? \quad a = \sum_{i=0}^n w_i x_i \quad e = d - y = d - f\left(\sum_{i=0}^n w_i x_i\right)$$

$$\frac{\partial e}{\partial w_j} = 0 - \frac{\partial f(a)}{\partial a} \frac{\partial a}{\partial w_j} = -\frac{\partial f(a)}{\partial a} \sum_{i=0}^n \frac{\partial (w_i x_i)}{\partial w_j} = -\frac{\partial f(a)}{\partial a} x_j$$

Učenje neurona

- Korekcija je proporcionalna parcijalnoj derivaciji greške
 - Korekcija je proporcionalna ulaznoj vrijednosti
 - Poveznica sa Hebovim učenjem
 - "Cells that fire together wire together."

$$\Delta w = \eta y x$$

Odabira aktivacijske funkcije

- Uobičajeni kandidati

$$f(a) = Ca$$

$$f(a) = \tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$$

$$\tanh(a) = 2\sigma(2a) - 1$$

$$f(a) = \text{sigmoid}(a) = \sigma(a) = \frac{1}{1 + e^{-a}}$$

$$f(a) = \text{ReLU}(a) = \begin{cases} a, & a > 0 \\ 0, & a \leq 0 \end{cases}$$

Odabira aktivacijske funkcije

- Ključni uvjet je da je funkcija derivabilna
 - Da postoji

$$\frac{\partial f(a)}{\partial a}$$

Odabira aktivacijske funkcije

- Derivacije uobičajenih kandidata

$$\frac{df(a)}{da}$$

$$\frac{dCa}{da} = C$$

$$\frac{d \tanh(a)}{da} = 1 - \frac{(e^x - e^{-x})^2}{(e^x + e^{-a})^2} = 1 - \tanh^2(a)$$

$$\frac{d \sigma(a)}{da} = \frac{e^{-x}}{(1 + e^{-a})^2} = \frac{1}{(1 + e^{-a})} \left(1 - \frac{1}{(1 + e^{-a})^2} \right) = \sigma(a)(1 - \sigma(a))$$

$$\frac{d \text{ReLU}(a)}{da} = \begin{cases} 1, & a > 0 \\ 0, & a \leq 0 \end{cases}$$

Odabira aktivacijske funkcije

- Korisna funkcija koja ne dolazi u obzir

$$f(a) = H(a) = \begin{cases} 1, & a > 0 \\ 0 & \end{cases}$$

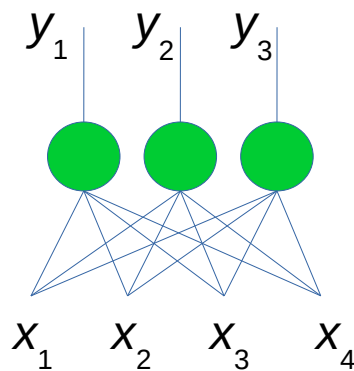
$$\frac{dH(a)}{da} = \begin{cases} \delta(a), & a = 0 \\ 0, & a \neq 0 \end{cases}$$

Keypoints IV

- Minimizaciju funkcije greške postićemo slijeđenjem gradijenta te funkcije
- Korekcija parametra je proporcionalna parcijalnoj derivaciji funkcije greške
- Izbor funkcije greške utječe na rezultat
- Izbor aktivacijskih funkcija utječe na rezultat

Više izlaza

- Ako nam treba više izlaza
- Dodamo još neurona u paralelu
- Ukupna greška tada mora biti kombinacija pojedinih grešaka



$$e_u = \sum |e|$$

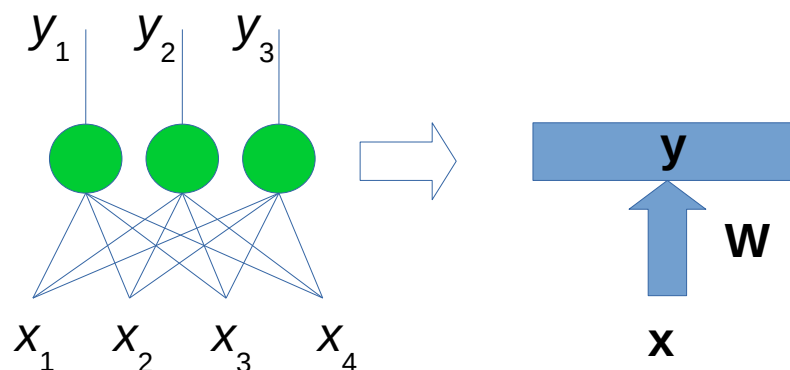
$$e_u = \frac{1}{2} \sum e^2$$

Više izlaza

- Matrična notacija

$$y_j = f(a_j) = f\left(\sum_{i=0}^n w_{ji} x_i\right) = f(\mathbf{w}_j^T \mathbf{x})$$

$$\mathbf{y} = f(\mathbf{a}) = f(\mathbf{W} \mathbf{x})$$



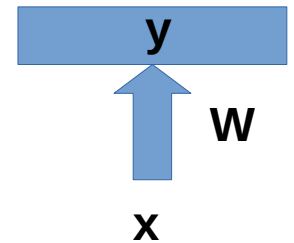
$$\mathbf{w}_j = \begin{bmatrix} w_{j0} \\ w_{j1} \\ \vdots \\ w_{jn} \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} \mathbf{w}_0^T \\ \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_m^T \end{bmatrix} = \begin{bmatrix} w_{00} & w_{01} & \cdots & w_{0n} \\ w_{10} & w_{11} & \cdots & w_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m0} & w_{m1} & \cdots & w_{mn} \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \end{bmatrix}$$

Više izlaza

- Matrična notacija

$$\Delta w_{ji} = -\eta \frac{\partial e}{\partial w_{ji}}$$

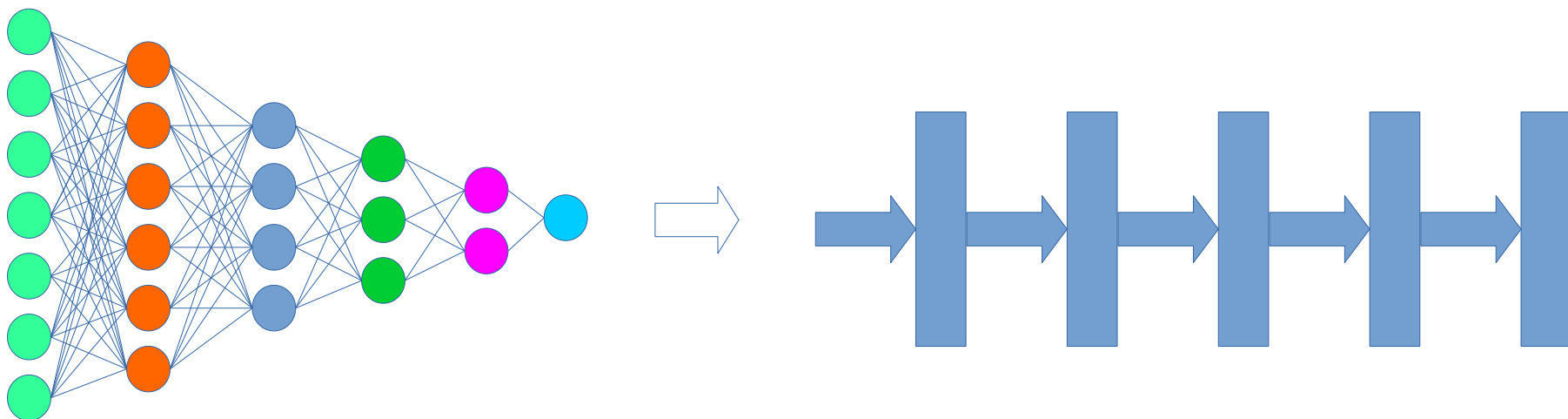
$$\frac{\partial e}{\partial \mathbf{W}} = \begin{bmatrix} \frac{\partial e}{\partial w_{00}} & \frac{\partial e}{\partial w_{01}} & \dots & \frac{\partial e}{\partial w_{0n}} \\ \frac{\partial e}{\partial w_{10}} & \frac{\partial e}{\partial w_{11}} & \dots & \frac{\partial e}{\partial w_{1n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e}{\partial w_{m0}} & \frac{\partial e}{\partial w_{m1}} & \dots & \frac{\partial e}{\partial w_{mn}} \end{bmatrix}$$



$$\mathbf{W} = \mathbf{W} - \eta \frac{\partial e}{\partial \mathbf{W}} = \begin{bmatrix} w_{00} & w_{01} & \dots & w_{0n} \\ w_{10} & w_{11} & \dots & w_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m0} & w_{m1} & \dots & w_{mn} \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial e}{\partial w_{00}} & \frac{\partial e}{\partial w_{01}} & \dots & \frac{\partial e}{\partial w_{0n}} \\ \frac{\partial e}{\partial w_{10}} & \frac{\partial e}{\partial w_{11}} & \dots & \frac{\partial e}{\partial w_{1n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e}{\partial w_{m0}} & \frac{\partial e}{\partial w_{m1}} & \dots & \frac{\partial e}{\partial w_{mn}} \end{bmatrix}$$

Višeslojne mreže

- Dodajemo slojeve neurona
- Ulaz u neke neurone je izlaz iz drugih neurona
- Izlaz na zadnjem sloju je konačan izlaz
- Između slojeva je potpuna povezanost (fully connected)



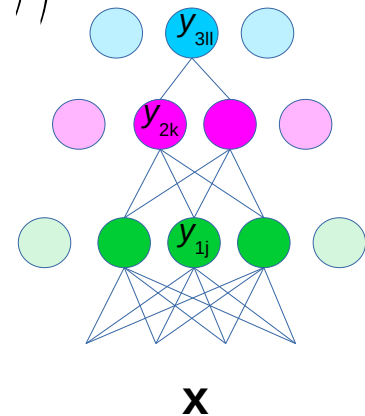
Višeslojne mreže

- Pogledajmo jednadžbe

$$y_{3l} = f_3 \left(\sum_{k=0}^r w_{lk} y_{2k} \right) = f_3 \left(\sum_{k=0}^r w_{lk} f_2 \left(\sum_{j=0}^n w_{kj} f_1 \left(\sum_{i=0}^m w_{ji} x_i \right) \right) \right)$$

$$y_{2k} = f_2 \left(\sum_{j=0}^n w_{kj} y_{1j} \right) = f_2 \left(\sum_{j=0}^n w_{kj} f_1 \left(\sum_{i=0}^m w_{ji} x_i \right) \right)$$

$$y_{1j} = f_1 \left(\sum_{i=0}^m w_{ji} x_i \right)$$



Višeslojne mreže

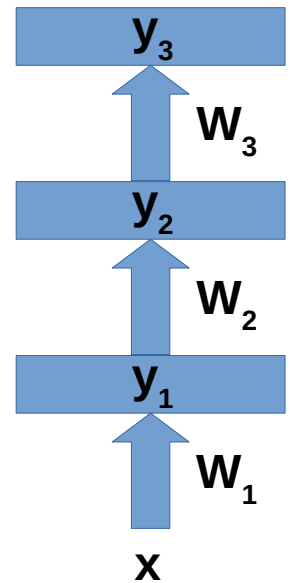
$$y = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

$$y = f(\mathbf{a}) = f(\mathbf{W} \mathbf{x})$$

$$y_3 = f(\mathbf{W}_3 y_2) = f\left(\mathbf{W}_3 f\left(\mathbf{W}_2 f\left(\mathbf{W}_1 \mathbf{x}\right)\right)\right)$$

$$y_2 = f(\mathbf{W}_2 y_1) = f\left(\mathbf{W}_2 f\left(\mathbf{W}_1 \mathbf{x}\right)\right)$$

$$y_1 = f(\mathbf{W}_1 \mathbf{x})$$



Višeslojne mreže

- Kako trenirati višeslojnu mrežu?
- Možemo opet minimizirati grešku
- Greška se računa s obzirom na izlaz iz zadnjeg sloja
- Što je sa prethodnim slojevima?

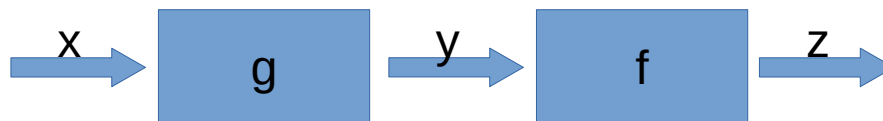
Derivacija kompozicije funkcija

$$z = f(y), \quad y = g(x), \quad z = f(g(x)) = (f \circ g)(x)$$

$$\left. \frac{dz}{dx} \right|_x = (f \circ g)'(x) = f'(g(x)) g'(x)$$

$$\left. \frac{dz}{dx} \right|_x = \left. \frac{dz}{dy} \right|_{y(x)} \left. \frac{dy}{dx} \right|_x$$

- Chain rule
- Nužno je da $f(y)$ i $g(x)$ budu derivabilne
- Ako poznajemo trenutnu brzinu promjene z u ovisnosti o y i trenutnu brzinu promjene y u ovisnosti o x tada možemo izračunati brzinu promjene z u ovisnosti o x



Višeslojne mreže

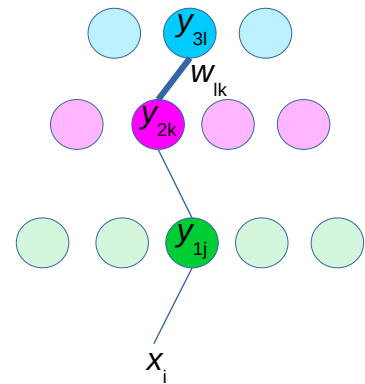
- Pogledajmo jednu putanju

$$y_{3l} = f_3(w_{lk} y_{2k})$$

$$\frac{\partial E}{\partial w_{lk}} = -e_l \frac{\partial y_{3l}}{\partial w_{lk}} = -e_l \frac{\partial f_3(a_{3l})}{\partial w_{lk}} =$$

$$= -e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} y_{2k}$$

$$E = \frac{1}{2} \sum_{l=0}^q e_l^2$$



Višeslojne mreže

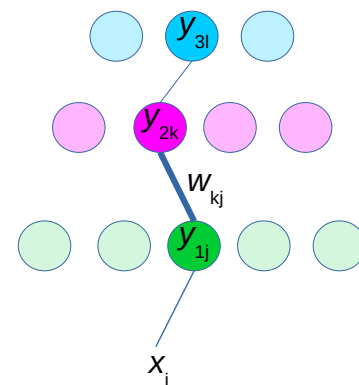
- Promotrimo težinu u sloju ispod

$$y_{3l} = f_3(w_{lk} y_{2k}) = f_3(w_{lk} f_2(w_{kj} y_{1j}))$$

$$\frac{\partial E}{\partial w_{kj}} = -e_l \frac{\partial y_{3l}}{\partial w_{kj}} = -e_l \frac{\partial f_3(a_{3l})}{\partial w_{kj}} = -e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} \frac{\partial a_{3l}}{\partial w_{kj}} =$$

$$= -e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} y_{1j}$$

$$E = \frac{1}{2} \sum_{l=0}^q e_l^2$$



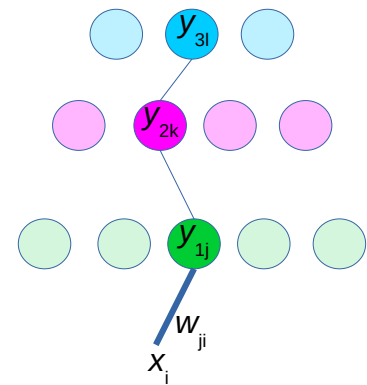
Višeslojne mreže

- Promotrimo težinu u sloju ispod

$$y_{3l} = f_3(w_{lk} y_{2k}) = f_3(w_{lk} f_2(w_{kj} y_{1j})) = f_3(w_{lk} f_2(w_{kj} f_1(w_{ji} x_i)))$$

$$E = \frac{1}{2} \sum_{l=0}^q e_l^2$$

$$\begin{aligned} \frac{\partial E}{\partial w_{ji}} &= -e_l \frac{\partial y_{3l}}{\partial w_{ji}} = -e_l \frac{\partial f_3(a_{3l})}{\partial w_{ji}} = -e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} \frac{\partial a_{3l}}{\partial w_{ji}} = \\ &= -e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial w_{ji}} = \\ &= -e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i \end{aligned}$$



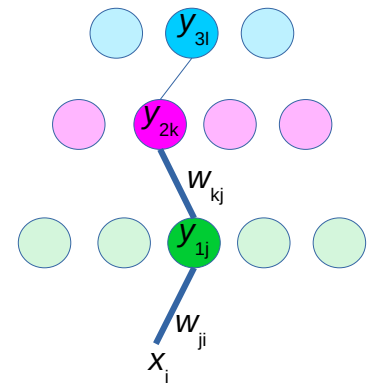
Višeslojne mreže

- Pri računanju parcijalnih derivacija u sloju niže, mogu se koristiti komponente parcijalne derivacije u sloju iznad

$$\frac{\partial E}{\partial w_{lk}} = -e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} y_{2k}$$

$$\frac{\partial E}{\partial w_{kj}} = -e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} y_{1j}$$

$$\frac{\partial E}{\partial w_{ji}} = -e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i$$



Error backpropagation

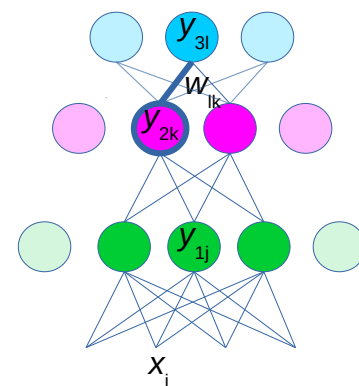
- Parcijalne derivacije funkcije greške nose informacije o grešci
 - Koriste se za učenje
- Parcijalne derivacije iz sloja iznad se propuštaju u sloj ispod
 - Unazadna propagacija pogreške
- Algoritam za učenje
 - Sloj ispod uči na temelju onoga što je učio sloj iznad

Višeslojne mreže

- Uključimo i ostale veze
- Zadnji sloj se uči na poznati način

$$y_{3l} = f_3 \left(\sum_{k=0}^r w_{lk} y_{2k} \right) \quad E = \frac{1}{2} \sum_{l=0}^q e_l^2$$

$$\frac{\partial E}{\partial w_{lk}} = - \sum_{l=0}^q e_l \frac{\partial y_{3l}}{\partial w_{lk}} = - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial w_{lk}} = - e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} y_{2k}$$



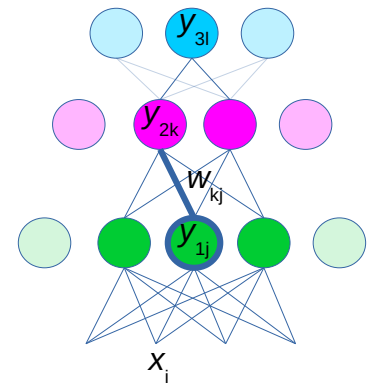
$$\frac{\partial E}{\partial y_{2k}} = - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} w_{lk} = \sum_{l=0}^q \frac{\partial E}{\partial w_{lk}} \frac{w_{lk}}{y_{2k}}$$

Višeslojne mreže

- Prethodni sloj se uči na „isti” način

$$y_{3l} = f_3 \left(\sum_{k=0}^r w_{lk} y_{2k} \right) = f_3 \left(\sum_{k=0}^r w_{lk} f_2 \left(\sum_{j=0}^n w_{kj} y_{1j} \right) \right)$$

$$\begin{aligned} \frac{\partial E}{\partial w_{kj}} &= - \sum_{l=0}^q e_l \frac{\partial y_{3l}}{\partial w_{kj}} = - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial w_{kj}} = - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} \frac{\partial (w_{lk} y_{2k})}{\partial w_{kj}} = \\ &= - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} y_{1j} = \frac{\partial E}{\partial y_{2k}} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} y_{1j} = \\ &= \frac{\partial E}{\partial y_{2k}} \frac{\partial y_{2k}}{\partial w_{kj}} \end{aligned}$$



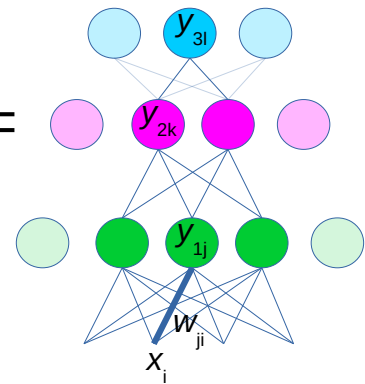
$$\frac{\partial E}{\partial y_{1j}} = \sum_{k=0}^r \frac{\partial E}{\partial w_{kj}} \frac{w_{kj}}{y_{1j}} = \sum_{k=0}^r \frac{\partial E}{\partial y_{2k}} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} = \sum_{k=0}^r \frac{\partial E}{\partial y_{2k}} \frac{\partial y_{2k}}{\partial y_{1j}}$$

Višeslojne mreže

- Pomaknimo se još jedan sloj niže

$$y_{3l} = f_3 \left(\sum_{k=0}^r w_{lk} y_{2k} \right) = f_3 \left(\sum_{k=0}^r w_{lk} f_2 \left(\sum_{j=0}^n w_{kj} f_1 \left(\sum_{i=0}^m w_{ji} x_i \right) \right) \right)$$

$$\begin{aligned} \frac{\partial E}{\partial w_{ji}} &= - \sum_{l=0}^q e_l \frac{\partial y_{3l}}{\partial w_{ji}} = - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial w_{ji}} = - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} \frac{\partial (a_{3l})}{\partial w_{ji}} = \\ &= - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial w_{ji}} = \\ &= - \sum_{l=0}^q e_l \frac{\partial f_3(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i \end{aligned}$$



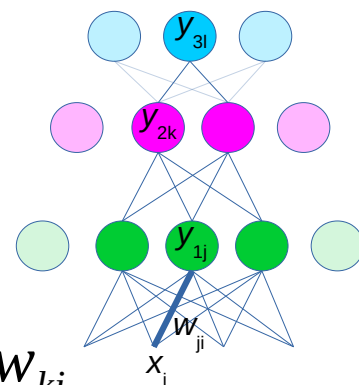
Višeslojne mreže

- Povežimo s prethodnim slojem

$$\begin{aligned}\frac{\partial E}{\partial w_{ji}} &= - \sum_{l=0}^q e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i \\ &= \left(- \sum_{l=0}^q e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \right) \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i =\end{aligned}$$

$$\frac{\partial E}{\partial y_{1j}} = - \sum_{l=0}^q e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj}$$

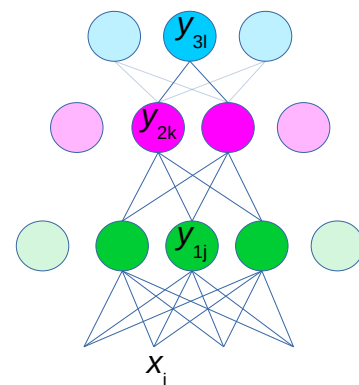
$$= \frac{\partial E}{\partial y_{1j}} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i = \frac{\partial E}{\partial y_{1j}} \frac{\partial y_{1j}}{\partial w_{ji}}$$



Višeslojne mreže

$$\frac{\partial E}{\partial y_{1j}} = \sum_{k=0}^r \frac{\partial E}{\partial y_{2k}} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj}$$

- Parcijalnu derivaciju funkcije greške po elementu jednog sloja određujemo iz
 - Parcijalnih derivacija funkcije greške po elementima iz sloja iznad
 - Derivacija aktivacijskih funkcija sloja iznad
 - Težina koje povezuju trenutni sa prethodnim slojem



Lokalni gradijent

- Lokalni gradijent δ

$$\frac{\partial E}{\partial w_{kj}} = - \sum_{l=0}^q e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} y_{1j} = - \delta_k y_{1j}$$

$$\begin{aligned} \frac{\partial E}{\partial w_{ji}} &= - \sum_{l=0}^q e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i \\ &= - \sum_{k=0}^r \delta_k w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i = - \delta_j x_i \end{aligned}$$

$$\delta_j = \sum_{k=0}^r \delta_k w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}}$$

Višeslojne mreže

- Zašto ne linearna aktivacijska funkcija $f(a) = Ca$

$$y_q = f \left(\sum_{l=1}^r w_{lj} f \left(\sum_{k=1}^n w_{jk} f \left(\sum_{i=1}^m w_{ki} x_i \right) \right) \right)$$

$$y_q = C \sum_{l=1}^r w_{lj} C \sum_{k=1}^n w_{jk} C \sum_{i=1}^m w_{ki} x_i$$

$$y_q = \sum_{l=1}^r \sum_{k=1}^n \sum_{i=1}^m C w_{lj} C w_{jk} C w_{ki} x_i$$

$$y_q = \sum_{i=1}^m \sum_{k=1}^n \sum_{l=1}^r C w_{lj} C w_{jk} C w_{ki} x_i = \sum_{i=1}^m w'_i x_i \quad w'_i = \sum_{k=1}^n \sum_{l=1}^r C w_{lj} C w_{jk} C w_{ki}$$

Keypoints V

- Učenje se odvija korekcijom parametara
 - Ostvaruje se unazadnom propagacijom informacije o pogrešci (parcijalne derivacije)
 - Parcijalne derivacije se računaju na temelju parcijalnih derivacija prethodnih slojeva
 - Bitno pojednostavljuje i ubrzava računanje
- Aktivacijske funkcije trebaju biti nelinearne

Kompleksnost mreže

- Imamo izbor
 - Broj neurona u jednom sloju
 - Broj slojeva
- Hiperparametri
 - Svi parametri koje je potrebno odrediti prije treniranja mreže
- Ukupna kompleksnost mreže mjeri se brojem neurona
 - Ista kompleksnost uz variranje broja slojeva i neurona po sloju
 - Što je bolje rješenje?

Kompleksnost mreže

- Pokazalo se da bolje funkcioniraju dublje, a uže mreže
- Plíce, a šire su lošije
- Neki autori se ne slažu
 - Za neke arhitekture bi moglo vrijediti obrnuto

Kompleksnost mreže

- Intuitivno, veća ukupna kompleksnost je bolja
 - Kompleksne probleme ne možemo rješavati jednostavnim alatima
- Može li mreža biti prekompleksna?
 - Problemi
 - Pretreniranje
 - Iščezavajući gradijent

Pretreniranje

- Prekompleksna mreža može naučiti uzorke za treniranje „na pamet”
- Na novim uzorcima mreža neće dobro raditi
- Prenuačćenost -> loša generalizacija

Pretreniranje

- Mreža je prekompleksa s obzirom na kompleksnost problema
- Premala kompleksnost problema može doći od:
 - Inherentne male kompleksnosti problema
 - Preograničenog skup za učenje
 - Ovo se možda može popraviti

Pretreniranje

- Drugo rješenje je smanjivanje kompleksnosti mreže ili drugačije modifikacije arhitekture mreže

Iščezavajući gradijent

- Gradijenti za pojedine parametre mreže jednaki su umnošku lokalnih gradijenata iznad u mreži
- Ako „samo jedan” od tih lokalnih gradijenata ode u nulu -> cijeli gradijent ide u nulu
 - Isto vrijedi za sve parametre ispod u mreži
 - Vrijednost gradijenata dovoljno blizu nule ima isti efekt

$$\frac{\partial E}{\partial w_{ji}} = - \sum_{l=0}^q e_l \frac{\partial f(a_{3l})}{\partial a_{3l}} \sum_{k=0}^r w_{lk} \frac{\partial f_2(a_{2k})}{\partial a_{2k}} w_{kj} \frac{\partial f_1(a_{1j})}{\partial a_{1j}} x_i$$

Iščezavajući gradijent

- Za parametre dubljih slojeva problem je izraženiji
 - Oni imaju više ulančanih lokalnih gradijenata
 - Raste vjerojatnost pojave množenja s nulom
- Glavni izvor problema su aktivacijske funkcije čije derivacije imaju vrijednost nula
 - Možemo promijeniti aktivacijske funkcije

Iščezavajući gradijent

- Još jedan izvor problema je inicijalizacija
 - Ako smo stigli na cilj, želimo da gradijenti budu nula
 - Gradijenti ne smiju biti nula ako još nismo stigli na cilj
 - Zbog nesretne inicijalizacije to se ipak može desiti

Eksplodirajući gradijent

- Izvor problema je sličan kao i za iščezavajući gradijent
- Množenjem više lokalnih gradijenata od kojih su neki veliki, možemo doći do velikog gradijenta
- Preveliki gradijent čini učenje nestabilnim

Eksplodirajući gradijent

- Rješenja su slična kao i kod iščezavajućeg gradijenta
- Problem je suprotan od iščezavajućeg gradijenta
 - IG: gradijent je premali
 - EG: gradijent je preveliki
- Kompromisno rješenje ili veći naglasak na posebne tehnike za svaki od tih problema

Eksplodirajući gradijent

- Ipak, u principu nikada ne želimo ekstremno velike gradijente
 - Možemo ih ograničiti
 - Pitanje je izbora praga
 - Dodatni hiperparametar

Keypoints VI

- Gradijenti određuju tijek treniranja
 - Mogu prerano zaustaviti treniranje
 - Mogu treniranje učiniti nestabilnim
- Pretreniranje je bitan problem
- Rješenja uključuju intervencije u arhitekturi mreže, postupku treniranja ili bazi za učenje

Minimizacija očekivane greške

- Cilj je smanjiti ukupnu / prosječnu grešku

$$E = \sum e^2$$

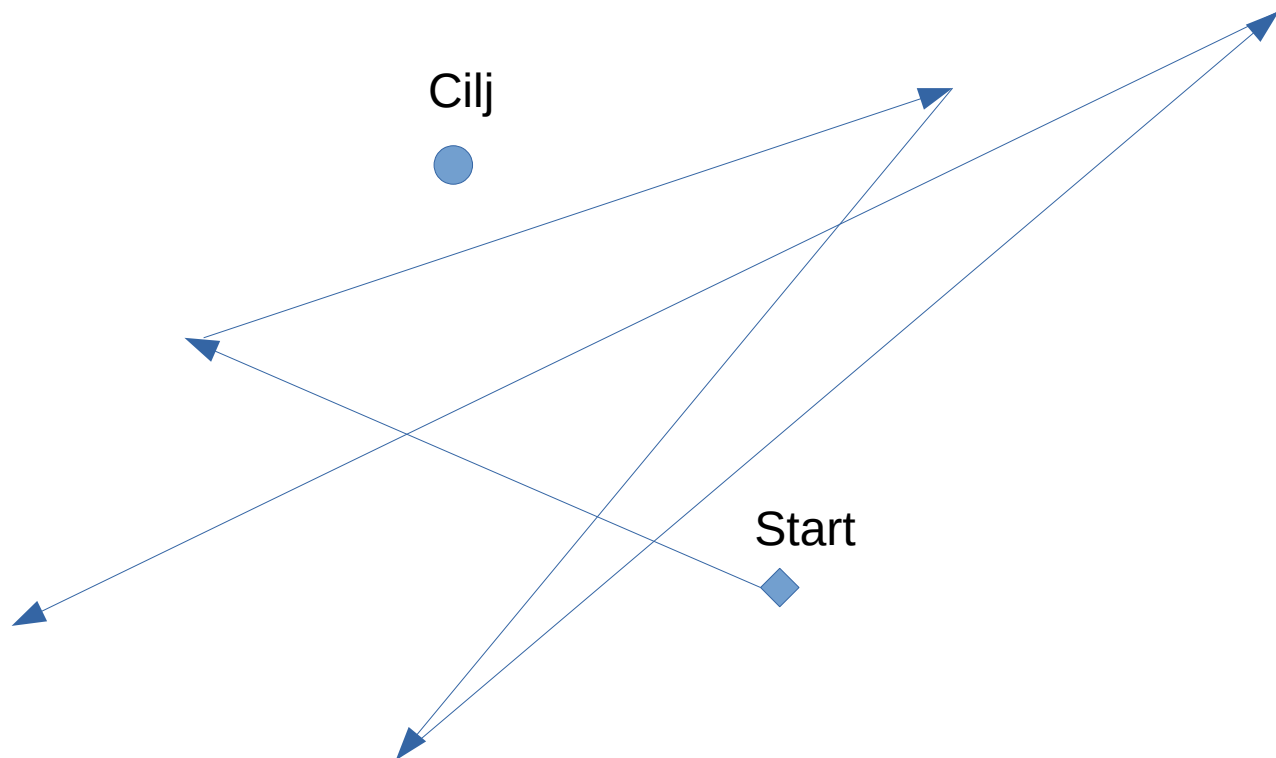
- Ta greška se sastoji od više komponenti
 - Svaka komponenta ukupne greške e odgovara grešci za jedan ulazni uzorak iz skupa za treniranje
- Smanjivanjem svake komponente e smanjujemo i njihovu kombinaciju

Minimizacija očekivane greške

- Imamo izbor
 - Svoditi redom svaku pojedinu pogrešku na nulu
 - Smanjivati postepeno svaku pojedinu pogrešku

Svođenje svake pojedine pogreške na nulu

- Možemo pratiti gradijente i svesti prvu pogrešku na nulu
 - Iterativno slijedimo gradijent do iščezavanja gradijenta
- Zatim drugu
- Zatim...
- Postoji problem

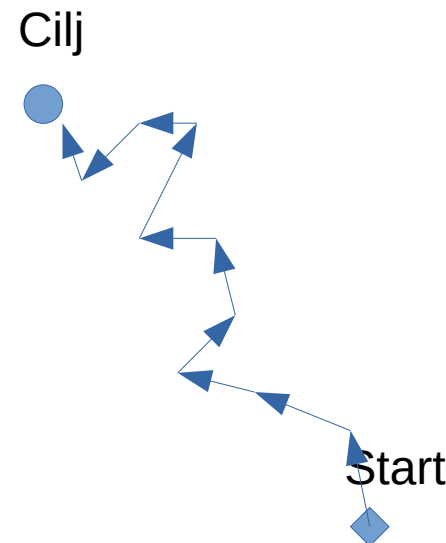


Svođenje svake pojedine pogreške na nulu

- Svaka pogreška ovisi o parametrima mreže
- Svođenjem jedne pogreške na nulu mijenjamo parametre mreže
 - Možemo očekivati da će sve ostale greške rasti!
 - Možemo očekivati i da će ukupna greška rasti!
- Ovaj pristup nije dobar

Postepeno smanjivanje svake pojedine pogreške

- Iterativno prelazimo redom sve ulazne uzorke i korigiramo parametre mreže
- Svaka korekcija se radi postepeno
 - Samo jedan korak u smjeru gradijenta za svaki ulazni uzorak
- Kada prođemo sve ulazne uzorke ponavljamo epohu
- Epohe ponavljamo dok učenje ne konvergira
 - Npr. ukupna greška se više ne smanjuje



Postepeno smanjivanje svake pojedine pogreške

- Interpretacija
 - Svaki uzorak iz skupa za učenje „vuče malo” prema smanjenu svoje greške
 - Koliko svaki uzorak vuče regulira se koeficijentom učenja η
 - Smanjenjem pojedinih grešaka smanjuje se ukupna
 - Krećemo se prema globalnom minimumu
 - Stabilnije rješenje nego da svaki uzorak „vuče jako” u svom smjeru
 - Moguća je nestabilnost
 - Možemo očekivati lošiju konačnu ukupnu pogrešku

Postepeno smanjivanje svake pojedine pogreške

- Kod kompleksnijih problema moramo imati veliki skup uzoraka za treniranje
 - Svaka epoha traje dugo
- Pomogla bi paralelizacija

Postepeno smanjivanje svake pojedine pogreške

- Određuju se gradijenti za sve uzorke u paraleli
- Određuju se korekcije parametara u paraleli
- Ukupna korekcija parametara je prosjek svih pojedinih korekcija

$$\Delta w_{uk} = \sum \Delta w_i$$

$$\Delta w_{uk} = \frac{1}{N} \sum \Delta w_i$$

Postepeno smanjivanje svake pojedine pogreške

- Za potpunu paralelizaciju problem su fizička ograničenja računala
 - Memorija mora primiti
 - Sve parametre mreže
 - Sve uzorke za treniranje
 - Sve međukorake računanja
 - Dovoljno procesorskih jedinica za paralelno računanje

Mini grupe

- Često se paralelizacije ne može provesti na čitavom skupu za treniranje
- Kompromisno rješenje su mini grupe (mini batch)
 - Slučajno se odaberu uzorci iz skupa za treniranje u broju za koji je moguće paralelno učenje

Mini grupe

- Jedan korak učenja se provede na svakoj mini grupi
- Mini grupe se uzorkuju iz skupa za treniranje dok se ne prođe cijeli skup
- Zatim se epoha ponavlja
 - Epoha ima manje koraka

Mini grupe

- Maksimalna veličina mini grupe ovisi o:
 - Kompleksnosti mreže
 - Veličini uzoraka
- Ponekad mini grupa može sadržavati tek jedan uzorak

Mini grupe

- Ulazi, izlazi i vrijednosti svih međuslojeva su drugačije za svaki uzorak grupe
- Kako ih predstaviti?
- Dodavanjem nove dimenzije
 - Mini grupa ima k uzoraka

$$\mathbf{Y} = f(\mathbf{A}) = f(\mathbf{W} \mathbf{X})$$

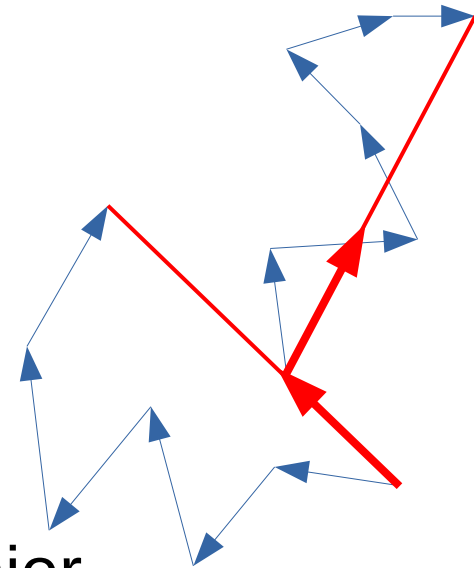
$$\mathbf{X} = \begin{bmatrix} x_{00} & x_{01} & \vdots & x_{0n} \\ x_{10} & x_{11} & \vdots & x_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k0} & x_{k1} & \vdots & x_{kn} \end{bmatrix}$$

$$\mathbf{W} = \begin{bmatrix} w_{00} & w_{01} & \cdots & w_{0n} \\ w_{10} & w_{11} & \cdots & w_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m0} & w_{m1} & \cdots & w_{mn} \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} y_{00} & y_{01} & \vdots & y_{0m} \\ y_{10} & y_{11} & \vdots & y_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ y_{k0} & y_{k1} & \vdots & y_{km} \end{bmatrix}$$

Mini grupe

- Gradijenti su drugačiji za svaki uzorak grupe
- Kako učiti?
- Udruže se svi pojedini gradijenti iz mini grupe
- Smjer minimizacije je kombinirani zajednički smjer
- Nećemo završiti na istom mjestu kao kad bi veličina mini grupe bila 1



$$\mathbf{W} = \mathbf{W} - \eta \frac{1}{k} \sum_{l=0}^k \frac{\partial e_l}{\partial \mathbf{W}} = \begin{bmatrix} w_{00} & w_{01} & \cdots & w_{0n} \\ w_{10} & w_{11} & \cdots & w_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m0} & w_{m1} & \cdots & w_{mn} \end{bmatrix} - \eta \frac{1}{k} \sum_{l=0}^k \begin{bmatrix} \frac{\partial e_l}{\partial w_{00}} & \frac{\partial e_l}{\partial w_{01}} & \cdots & \frac{\partial e_l}{\partial w_{0n}} \\ \frac{\partial e_l}{\partial w_{10}} & \frac{\partial e_l}{\partial w_{11}} & \cdots & \frac{\partial e_l}{\partial w_{1n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_l}{\partial w_{m0}} & \frac{\partial e_l}{\partial w_{m1}} & \cdots & \frac{\partial e_l}{\partial w_{mn}} \end{bmatrix}$$

Keypoints VII

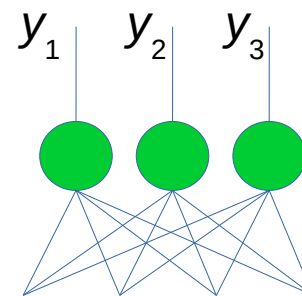
- Mini grupe omogućavaju
 - Brže učenje
 - Stabilnije učenje
 - Stabilno smanjivanje očekivane pogreške

Problem klasifikacije

- Za dvije kategorije dovoljan je jedan izlaz i prag
 - Ako je iznad praga – 1. kategorija
 - Inače 2. kategorija
- Što kada imamo više od dvije klase?
- Moguće su razne kombinacije više izlaza

Problem klasifikacije

- Za više od dvije klase potrebno je imati više izlaza
- Više izlaza omogućava i više različitih načina kodiranja kategorija
 - Npr. dva izlaza i dva praga za četiri kategorije



Problem klasifikacije

- Najčešće se koristi one-hot-encoding
 - Svaka kategorija ima svoj izlaz
 - Kategorija s najvećim izlazom je pobjednik
 - Idealno jedna 1 i sve ostale 0

$$d = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Problem klasifikacije

- Svaki izlaz predstavlja „rezultat” te kategorije
 - Najveći rezultat pobjeđuje
- Korisno bi bilo kada bi izlaze mogli interpretirati kao vjerojatnosti
 - To nam omogućuje softmax funkcija
 - Možemo je promatrati kao posebnu nelinearnu aktivacijsku funkciju sa više ulaza

$$y = \begin{bmatrix} 12 \\ 56 \\ 145 \\ \vdots \\ -28 \end{bmatrix}$$

Softmax

- Softmax je generalizacija sigmoide za više dimenzija

$$\sigma(\mathbf{y})_i = \frac{e^{y_i}}{\sum_i e^{y_i}}$$

- Svaki izlaz možemo interpretirati kao vjerojatnost / sigurnost odgovarajuće kategorije

$$0 \leq \sigma(\mathbf{y})_i \leq 1 \quad \sum_i \sigma(\mathbf{y})_i = 1$$

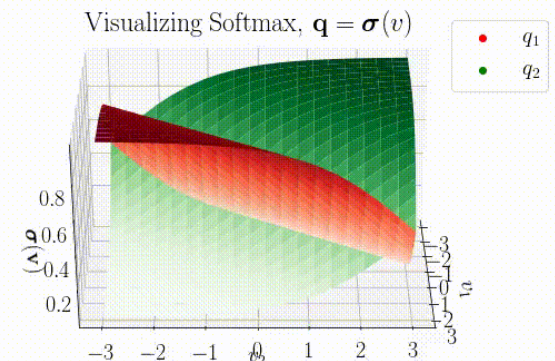
Softmax

- Što je za dvije kategorije?

$$\sigma(\mathbf{y})_1 = \frac{e^{y_1}}{\sum_i e^{y_i}} = \frac{e^{y_1}}{e^{y_1} + e^{y_2}} = \frac{1}{1 + e^{y_2 - y_1}} = \frac{1}{1 + e^{-(y_1 - y_2)}}$$

$$\sigma(\mathbf{y})_2 = \frac{1}{1 + e^{-(y_2 - y_1)}} = 1 - \sigma(\mathbf{y})_1$$

- Vjerojatnost svake kategorije određuje sigmoida
- Dvije vjerojatnosti su povezane pa je dovoljno gledati jednu
 - Ako je vjerojatnost veća od 0.5 ta kategorija je pobjednik
 - Drugu vjerojatnost ne trebamo računati
 - Ulaz u sigmoidu je razlika rezultata
 - Bitno je koji je veći – predznak razlike



Author: Charlie Lehman

Funkcija gubitka za klasifikaciju

- Izlazi se ponašaju kao vjerojatnosti
 - Na raspolaganju su nam matematički alati za vjerojatnosti
- Uzmimo onda unakrsnu entropiju za funkciju gubitka
 - Matematička interpretacija: mjera sličnosti dviju distribucija

Funkcija gubitka za klasifikaciju

- Kod klasifikacije za C kategorija izraz je sljedeći

$$CE = - \sum_i^C d_i \log y_i$$

- Za dvije klase sumu možemo raspisati

$$\begin{aligned} CE &= - \sum_i^2 d_i \log y_i = -d_1 \log(y_1) - d_2 \log(y_2) \\ &= -d_1 \log(y_1) - (1-d_1) \log(1-y_1) \end{aligned}$$

Funkcija gubitka za klasifikaciju

- Kod one-hot-encoding samo je jedna labela d_p jednaka 1, a ostale su 0

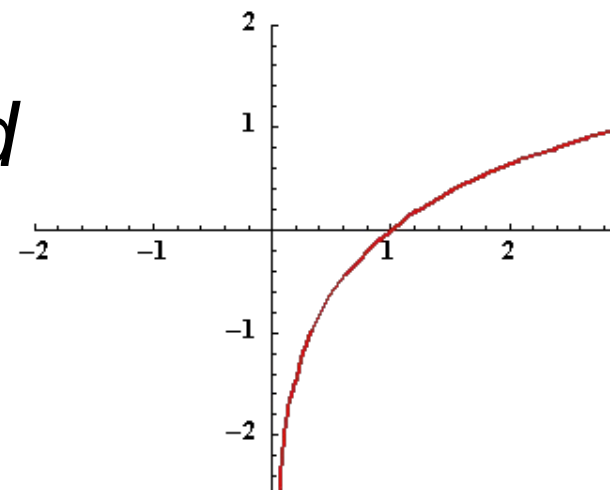
- Za binarnu klasifikaciju imamo

$$CE = - \sum_i^2 d_i \log y_i = -d_1 \log(y_1) - d_2 \log(y_2) = -d_p \log(y_p) = -\log(y_p)$$

- Za više kategorija imamo

$$CE = - \sum_i^C d_i \log y_i = -d_p \log(y_p) = -\log(y_p)$$

- Naziva se i *negative log likelihood*



Funkcija gubitka za klasifikaciju

- Prisjetimo se da u izlaznom sloju imamo softmax aktivacijsku funkciju
- Softmax + unakrsna entropija = Kategorička unakrsna entropija

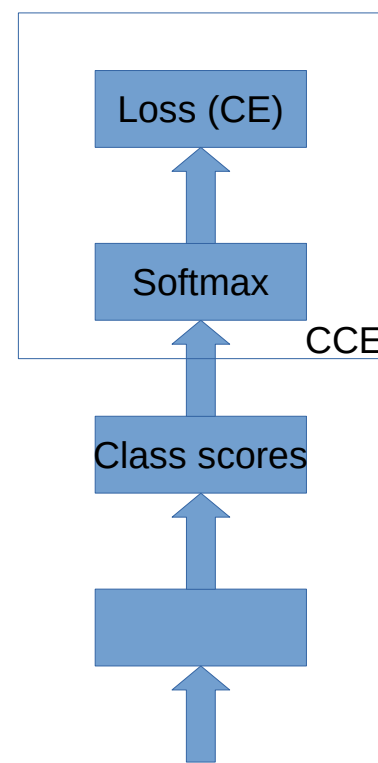
$$CCE = - \sum_i^C d_i \log \sigma(\mathbf{y})_i = -d_p \log(\sigma(\mathbf{y})_p) = -\log \left(\frac{e^{y_p}}{\sum_i^C e^{y_i}} \right)$$

- Kao i obično, interesiraju nas gradijenti

$$\frac{\partial CCE}{\partial y_p} = \frac{\partial}{\partial y_p} \left(-\log \left(\frac{e^{y_p}}{\sum_i^C e^{y_i}} \right) \right) = \frac{e^{y_p}}{\sum_i^C e^{y_i}} - 1 = \sigma(\mathbf{y})_p - 1$$

- Gradijent s obzirom na negativnu kategoriju

$$\frac{\partial CCE}{\partial y_n} = \frac{\partial}{\partial y_n} \left(-\log \left(\frac{e^{y_p}}{\sum_i^C e^{y_i}} \right) \right) = \frac{e^{y_n}}{\sum_i^C e^{y_i}} = \sigma(\mathbf{y})_n$$



Funkcija gubitka za klasifikaciju

- Prednosti CCE
 - Pokazala se boljom za problem klasifikacije
 - Jedna interpretacija vezana je za gradijente

Funkcija gubitka za klasifikaciju

- Ako na izlaz imamo softmax i sumu kvadratnih pogreški

$$E = \sum_i (\sigma(\mathbf{y})_i - d_i)^2 = F(\boldsymbol{\sigma}(\mathbf{y}))$$

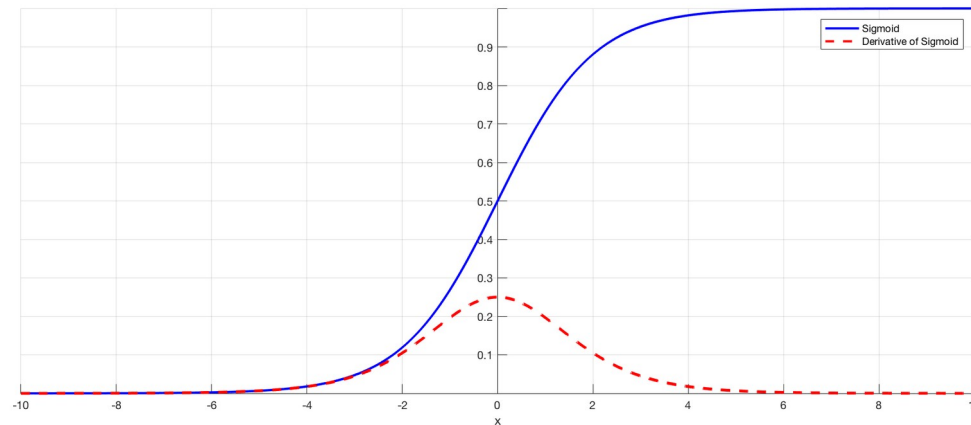
$$\frac{\partial E}{\partial y_i} = \frac{\partial F(\boldsymbol{\sigma}(\mathbf{y}))}{\partial \sigma(\mathbf{y})_i} \frac{\partial \sigma(\mathbf{y})_i}{\partial y_i}$$

$$\frac{\partial \sigma(\mathbf{y})_i}{\partial y_i} = \sigma(\mathbf{y})_i (1 - \sigma(\mathbf{y})_i)$$

Funkcija gubitka za klasifikaciju

- Za dvije kategorije imamo

$$\frac{\partial \sigma(\mathbf{y})_i}{\partial y_i} = \sigma(\mathbf{y})_i (1 - \sigma(\mathbf{y})_i)$$



Keypoints VIII

- Za problem klasifikacije obično se koriste
 - One-hot-encoding
 - Softmax
 - Kategorička unakrsna entropija
- Za problem regresije obično se koristi
 - Srednja kvadratna pogreška

Inicijalizacija

- Inicijalizacija određuje početno stanje mreže
- Postupkom učenja se postepeno iz početnog stanja kreće prema konačnom stanju
- U idealnom slučaju pronaći ćemo globalni minimum funkcije greške, bez obzira na početno stanje
 - U praksi to često nije slučaj

Inicijalizacija

- U ovisnosti o drugim parametrima mreže, moguće je napraviti pogrešnu inicijalizaciju:
 - Inicijalno stanje je pogrešno, ali s malim gradientima
 - Mreža ne može učiti
 - Iščezavajući gradient
 - Inicijalno stanje će dovesti do lokalnog minimuma koji je bitno veći od globalnog

Inicijalizacija

- Kako inicijalizirati mrežu?
- Idealno, blizu globalnog minimuma, ali ne znamo gdje je on
- Jedna mogućnost je slučajni odabir početnog stanja gdje su gradijenti veliki
 - Time sprječavamo preranu konvergenciju (iščezavajući gradijent)
- Druga mogućnost je ponoviti učenje sa slučajnom inicijalizacijom nekoliko puta

Generalizacija

- Poželjno je da mreža dobro radi na svim uzorcima
- Problem nastaje zbog toga što mrežu često nije moguće učiti na svim uzorcima
 - Trenira se na označenom podskupu svih mogućih uzoraka (skup za učenje)
- Generalizacija – karakteristika mreže da dobro radi na novim uzorcima koji nisu korišteni pri treniranju

Generalizacija

- I evaluacija naučene mreže se radi na ograničenom podskupu
 - Često skup svih mogućih uzoraka nije ograničen
- Kako možemo znati da će mreža raditi dobro na svim ostalim uzorcima?
 - Ne možemo znati!
 - Ali možemo pokušati osigurati da mreža ipak dobro generalizira

Generalizacija

- Možda i nije nužno da treniramo na svim mogućim uzorcima
 - Dovoljan bi bio reprezentativan podskup
- Reprezentativan podskup bi trebao obuhvatiti sve relevantne varijacije u ulaznim uzorcima
 - Idealno, mreža bi „vidjela” sve moguće varijacije pa novi uzorci ne bi predstavljali stvarno novu situaciju
 - I ovo je često neizvedivo
 - Ali možemo pokušati...

Generalizacija

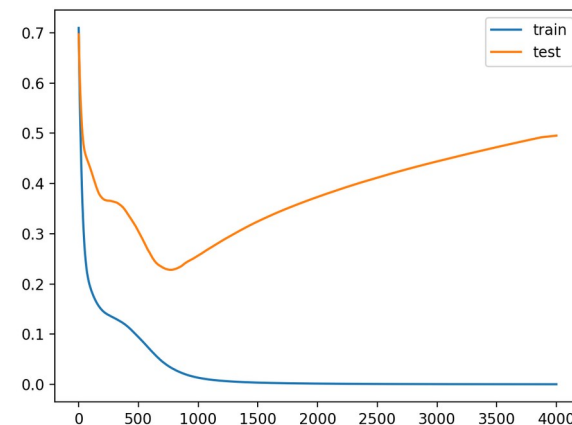
- Izuzetno je korisno da označeni skup za učenje bude što veći!
 - Oznake bi u kontekstu klasifikacije, bile poznate klase za svaki uzorak
- Skup za učenje ne može biti previše velik!
 - Ipak, veći skup za učenje obično vodi do dužeg treniranja
 - Jednokratni trošak

Generalizacija

- Druga strategija za postizanje dobre generalizacije je podjela skupa za učenje
- Obično se dijeli na tri dijela
 - Podskup za treniranje
 - Podskup za validaciju
 - Podskup za testiranje

Generalizacija

- Podskup za treniranje se koristi za učenje
 - Učenjem se minimizira prosječna greška na tom podskupu
 - On obično čini većinu skupa za učenje kako bi se postigla bolja generalizacija
- Podskup za validaciju služi kao neovisni skup novih uzoraka
 - Evaluacijom mreže na podskupu za validaciju može se procijeniti da li mreža dobro generalizira ili je pretrenirana
 - Može se iskoristiti za zaustavljanje treniranja ili za izbor hiperparametara
 - Time ipak utječe na učenje



Generalizacija

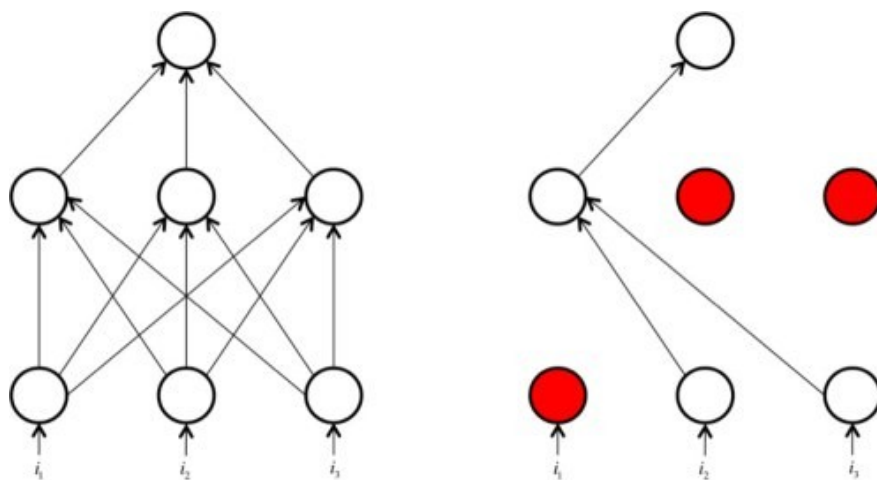
- Podskup za testiranje se koristi za konačnu evaluaciju učenja
 - Obično predstavlja manji dio skupa za učenje
 - Ključno je da je postupak učenja u potpunosti neovisan o ovom podskupu
 - Time se dobije nepristrana evaluacija
 - Skup za testiranje predstavlja nove, nikad viđene uzorke

Generalizacija

- Potrebno je pripaziti da skup za testiranje bude doista neovisan o skupovima za validaciju i treniranje
 - Ovisi o problemu
 - Ponekad je teško izvesti
- Potrebno je da sva tri podskupa budu reprezentativni predstavnici ukupnog skupa uzoraka
- Korisno je da sva tri podskupa budu što veći

Dropout

- Dodatni mehanizam regularizacije
- Primarno cilja na sprječavanje pretreniranja – postizanje bolje generalizacije
- Slučajno i privremeno „brisanje” neurona tijekom učenja



Dropout

- Efektivno učenje više različitih mreža čiji se rezultati kombiniraju na izlazu
- Neuroni moraju naučiti detektirati značajke neovisno o drugim neuronima
- Nužna je normalizacija zbog uklanjanje neurona

Augmentacija

- Često je skup za učenje neadekvatne veličine
- Skup za treniranje može se obogatiti simuliranim varijacijama kako bi se mreža naučila dobro generalizirati
 - Pretpostavka koja obično dobro funkcionira

Augmentacija

- Skup za učenje se može augmentirati dodavanjem:
 - Šuma
 - Transformacija
 - Geometrijske transformacije
 - Nerigidne transformacije
 - Parcijalnog maskiranja

Augmentacija

- Augmentacijom se ciljano dodaju efekti za koje se pretpostavlja da se mogu pojaviti u ulaznim uzorcima
 - Oni ne bi smjeli utjecati na konačni rezultat
 - Mreža uči ignorirati ih
 - Generalno želimo da mreža nauči ignorirati nebitne detalje i varijacije

Stratifikacija

- Poželjno je da skup za učenje ima urovnoteženu zastupljenost svih varijacija
 - Problem je neovisan o veličini skupa za učenje
 - Možemo očekivati da će mreža lošije raditi za one varijacije koje su slabije zastupljene u skupu za učenje
 - Posljedica samog postupka učenja koji minimizira prosječnu grešku

Stratifikacija

- Umjetno ujednačavanje distribucije ulaznih uzoraka nazivamo stratifikacija
 - Ne unosi nove informacije (nove varijacije)
 - Eventualno izostavlja neke
 - Provodi se samo na podskupu za treniranje
 - Pitanje je odabira parametara po kojima se provodi ujednačavanje

Stratifikacija

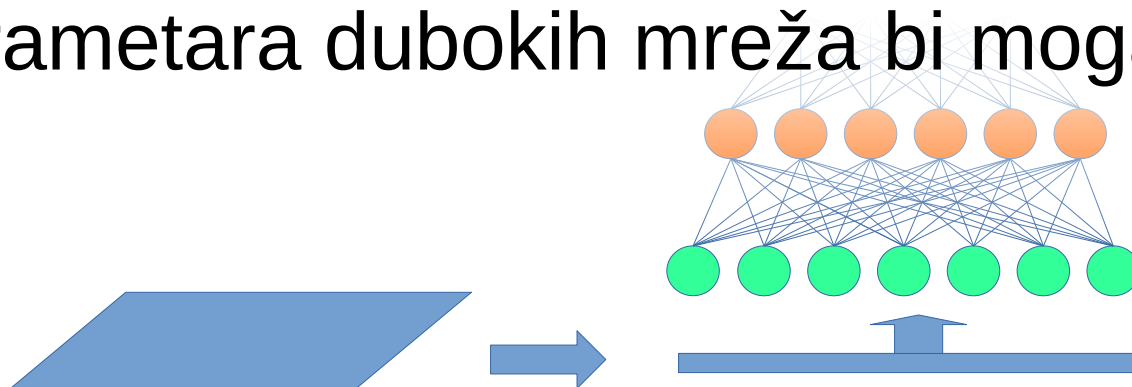
- Jedan pristup je višekratno korišćenje istih uzoraka
 - Onih koji su slabo zastupljeni
 - Npr. višestruko uključivanje u mini grupe
- Moguće je i izbacivanje jako zastupljenih uzoraka

Keypoints IX

- Inicijalizacija parametara mreže može znatno utjecati na rezultat učenja
 - Izbjegavanje nepovoljne inicijalizacije
- Podjela skupa uzoraka za učenje radi postizanja bolje generalizacije
- Dropout kao mehanizam postizanja bolje generalizacije
- Manipuliranje skupom za učenje radi postizanja bolje generalizacije

Slika kao ulaz u neuronsku mrežu

- Mogli bi primijeniti potpunu povezanost ulaza elemenata između slojeva
 - Za sliku dimenzija $M \times N$, broj težina prema prvom sloju je $M \times N \times Q$
 - Q je broj neurona u prvom sloju
 - Ne bi trebao biti puno manji od $M \times N$
 - Želimo puno slojeva kako bi ostvarili beneficije dubokih modela...
- Broj parametara dubokih mreža bi mogao biti velik

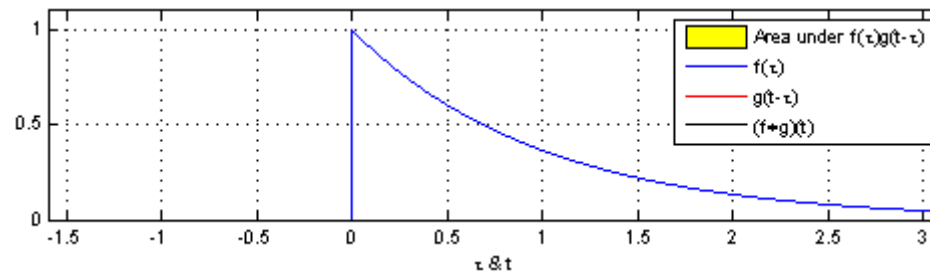
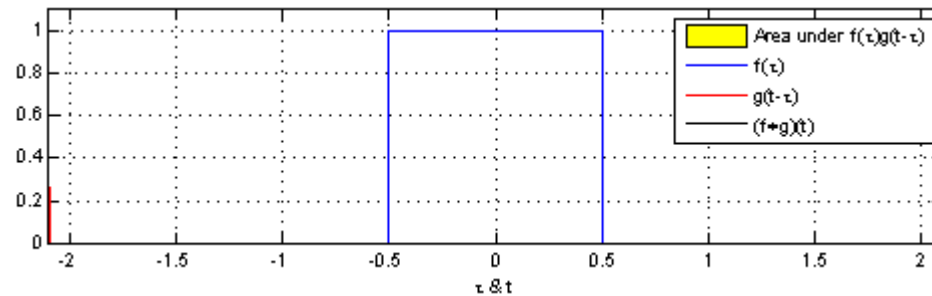


Konvolucija

- Lat. convolvere: kotrljati / odvijati zajedno
- Konvolucija je matematička operacija nad dvije funkcije
- Rezultat je nova funkcija

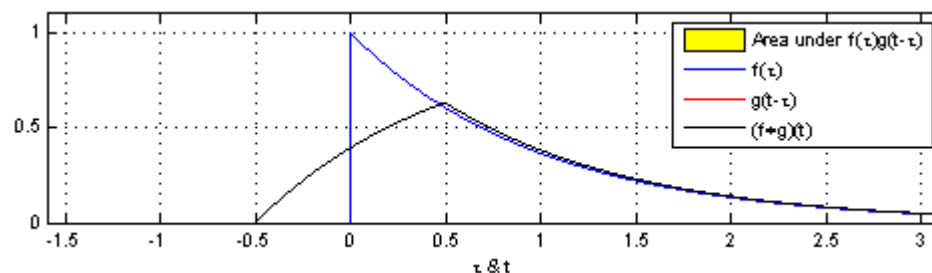
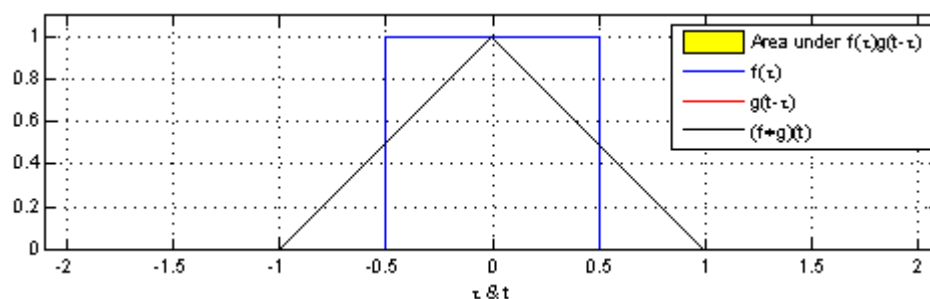
Konvolucija

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(t - \tau) d\tau$$



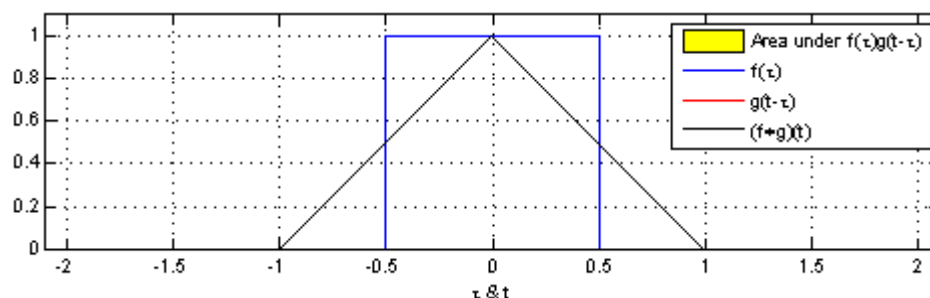
Konvolucija

- Ključni momenti:
 - Maksimalni odziv tamo gdje se dvije funkcije najviše poklapaju
 - Postoje i druge vrijednosti
 - Za druge funkcije isto postoji manji maksimum!!



Konvolucija

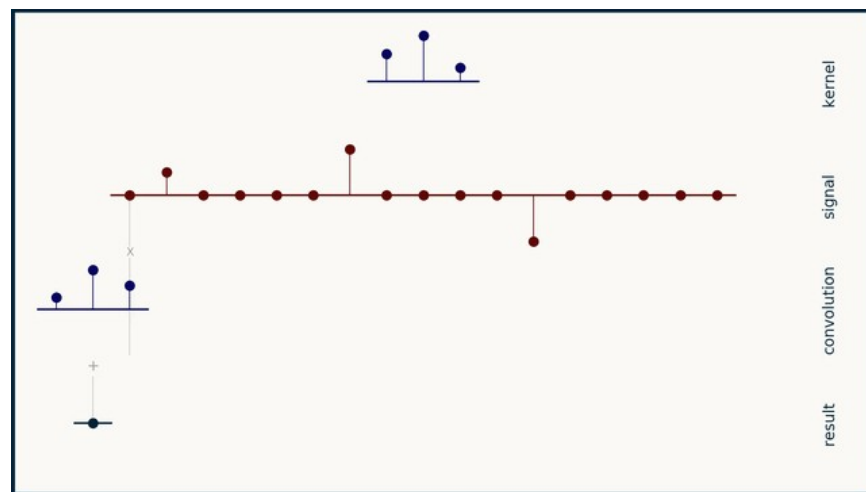
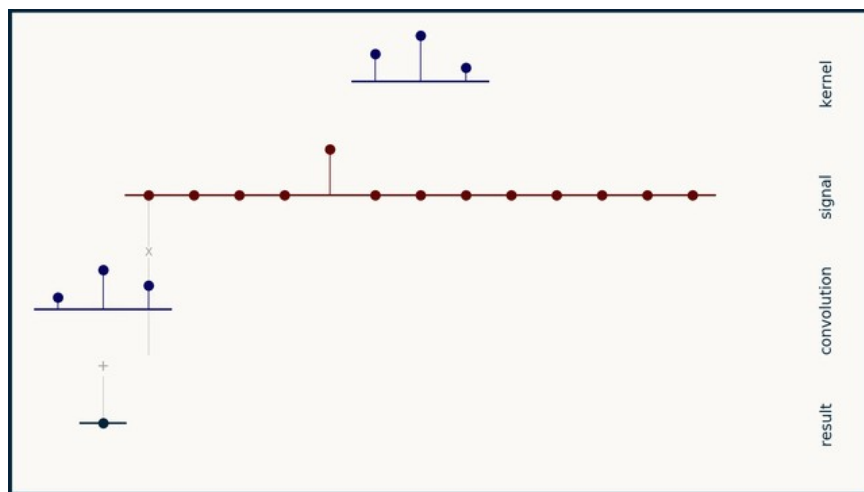
- Ključni momenti:
 - Traženjem najveće vrijednosti novonastale funkcije možemo pronaći kopiju/sliku tražene funkcije
 - U ovom slučaju pravokutnog impulsa



Konvolucija

- Zanima nas konvolucija digitalnih signala / slika
 - Sve želimo raditi na računalu

$$(f * g)(n) = \sum_{-\infty}^{+\infty} f(m) g(n - m)$$



Korelacija

- Vrlo slično kao konvolucija
 - Za realne funkcije, razlika je samo u zrcaljenju oko ishodišta
- Preciznije - unakrsna korelacija između funkcija f i h

$$(f \star h)(n) = \sum_{-\infty}^{+\infty} f(m)h(m-n)$$

Korelacija

- Ako je

$$h(n) = g(-n)$$

tada je

$$\begin{aligned}(f \star h)(n) &= \sum_{-\infty}^{+\infty} f(m) h(m-n) = \sum_{-\infty}^{+\infty} f(m) g(-(m-n)) = \\ &= \sum_{-\infty}^{+\infty} f(m) g(n-m) = (f * g)(n)\end{aligned}$$

Korelacija

- Unakrsna korelacija je mjera sličnosti dviju funkcija za različite međusobne posmake n
- Korelacija (lat. con = sa, relatio = odnos) predstavlja suodnos ili međusobnu povezanost između različitih funkcija
 - Povezanost znači da je vrijednost jedne varijable moguće s određenom vjerojatnošću predvidjeti na osnovi saznanja o vrijednosti druge varijable

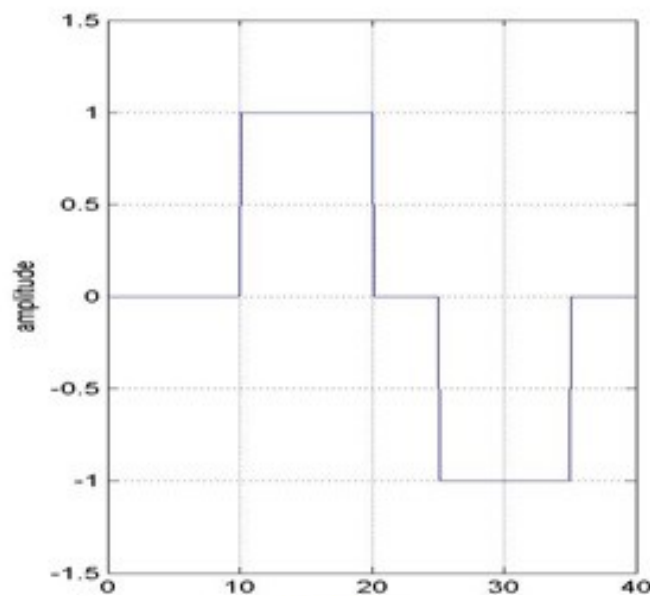
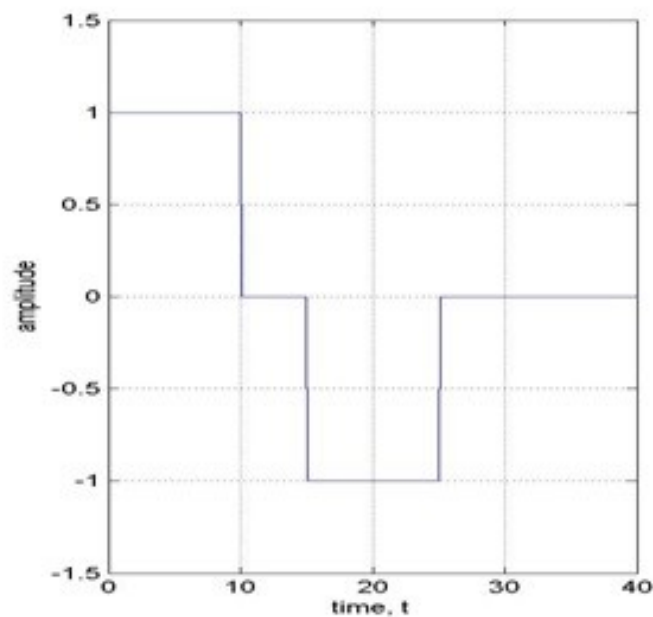
$$(f \star h)(n) = \sum_{-\infty}^{+\infty} f(m)h(m-n)$$

Korelacija

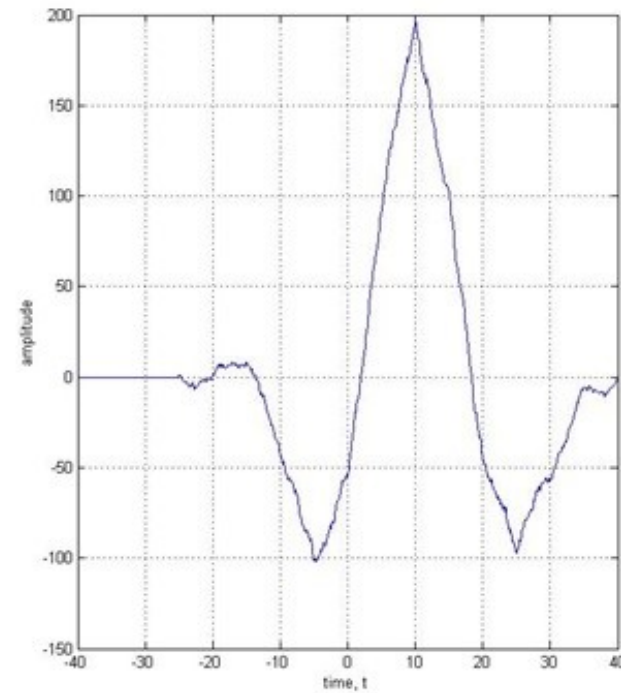
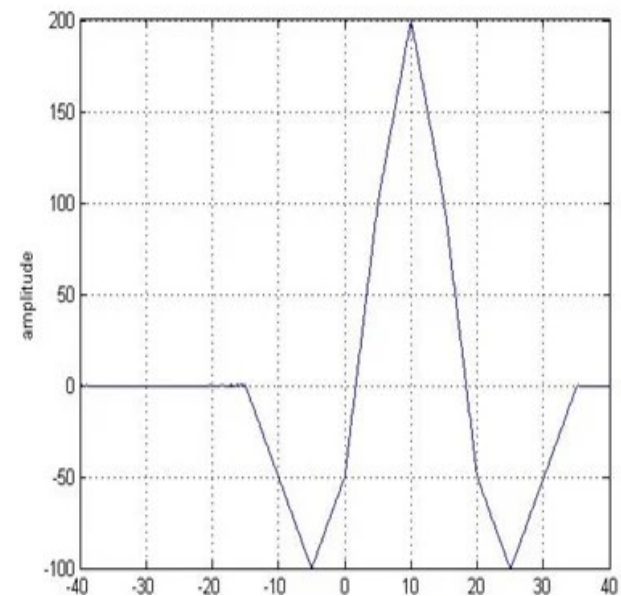
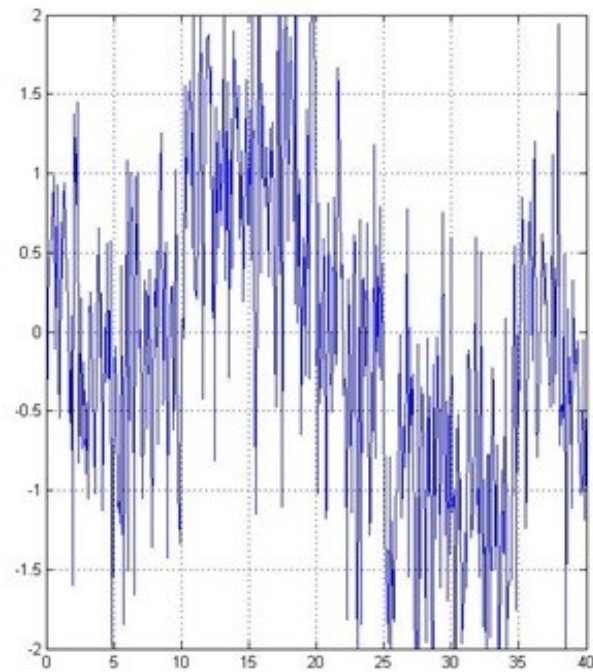
- Dvije funkcije su slične ako im vrijednosti prate isti trend
 - Kada jedna raste i druga raste
 - Ne moraju nužno imati iste vrijednosti

$$(f \star h)(n) = \sum_{-\infty}^{+\infty} f(m) h(m-n)$$

Korelacija



(a)



Keypoints X

- Konvolucija/korelacija bi nam mogla pomoći u traženju objekata (funkcija) u ulaznim uzorcima
- Pretraživanje bi moglo biti robustno na manje varijacije u izgledu objekta

2D Konvolucija

- Prelaskom u dvije dimenzije, radi se sa dvodimenzionalnim funkcijama

$$(f * g)(x, y) = \sum_{-\infty}^{+\infty} f(k, l) g(x - k, y - l)$$

- Primjena je ista kao i u 1D slučaju
 - Možemo pronalaziti razne funkcije
- Rezultat je isto 2D funkcija

2D Konvolucija

- Slike i dijelove slika isto možemo predstaviti kao 2D funkcije

$$I(x, y)$$

- Vrijednost funkcije / određuje svjetlinu na poziciji (x, y)

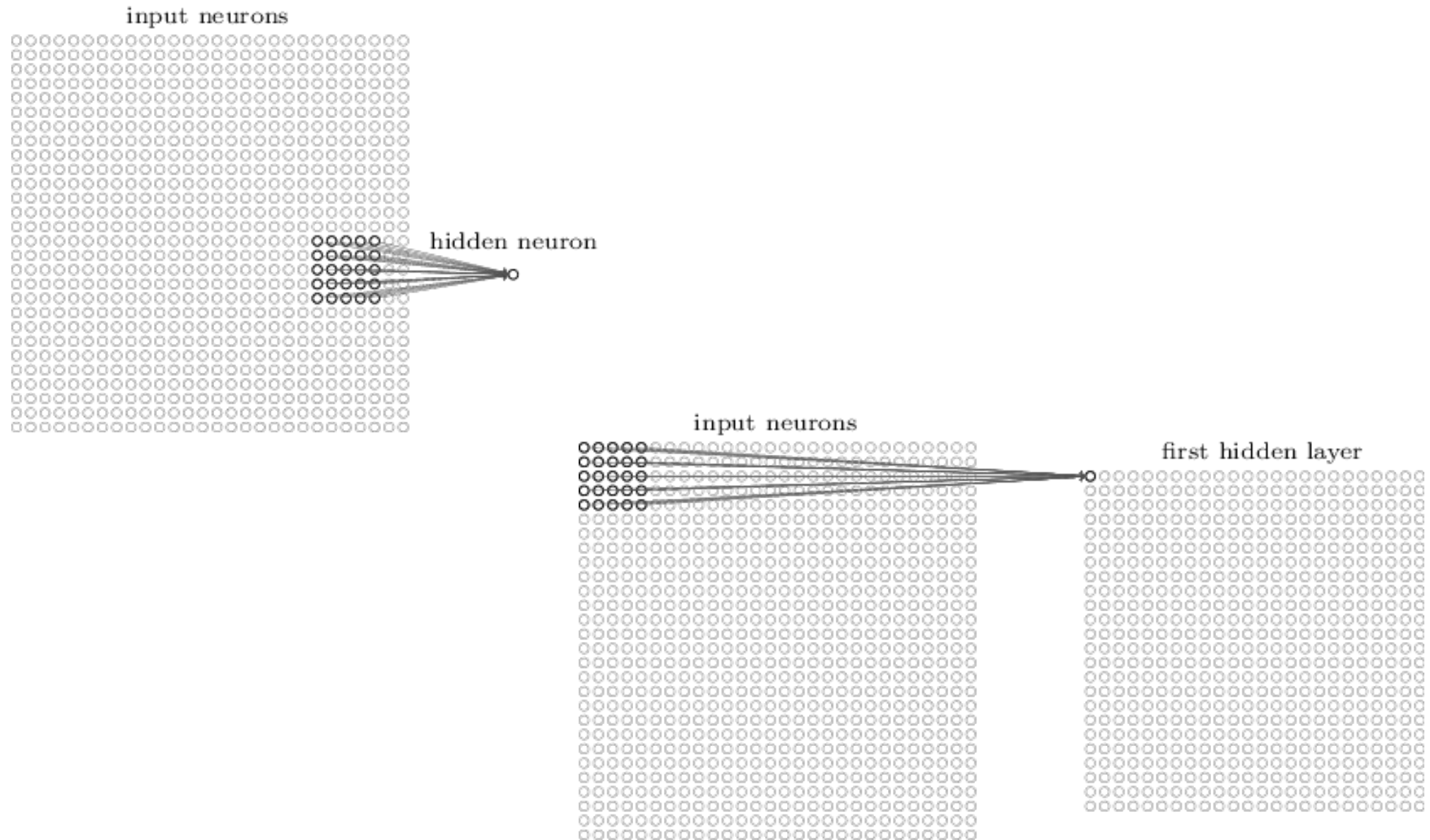
2D Konvolucija

- Ako želimo potražiti (detektirati) objekt na slici, konvolucija bi nam mogla pomoći
 - Zapravo želimo pronaći sliku objekta G u većoj slici I

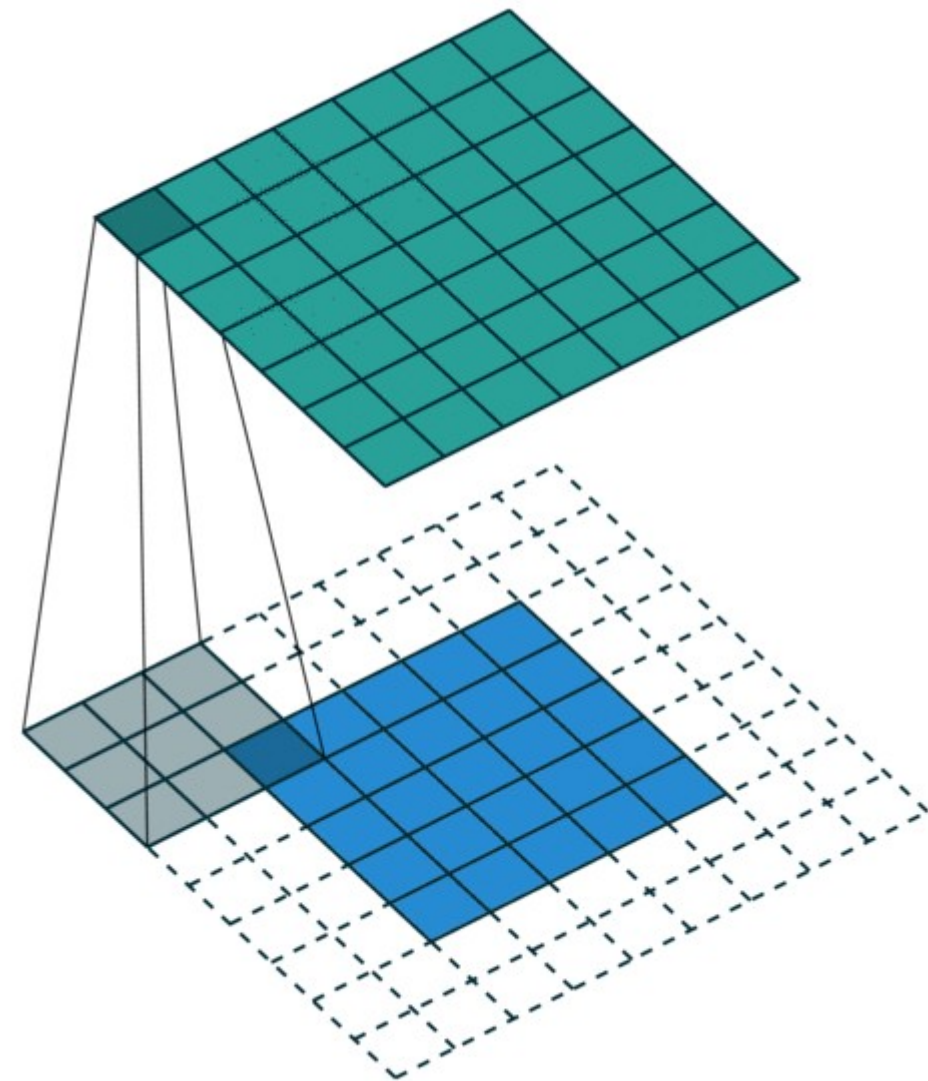
$$(I * G)(x, y) = \sum_{-\infty}^{+\infty} I(k, l) G(x - k, y - l)$$

- Dovoljno velika vrijednost u odzivu indicira lokaciju objekta

2D Konvolucija



2D Konvolucija



1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

Autor: Divyanshu Mishra

2D Konvolucija

- 2D konvolucija se već dugo koristi u analizi slike
- Ukoliko je izgled objekta poznat, lako je konstruirati pretraživačku funkciju $G(x,y)$ – maska, operator

$$(I * G)(x, y) = \sum_{-\infty}^{+\infty} I(k, l) G(x - k, y - l)$$

Detekcija rubova

- Najpoznatiji primjeri su detekcija rubova (edge detection)
 - Postoje brojne maske za traženje rubova (Sobel, Previtt, Roberts,...)
 - Rubovi su najjednostavnije pojave koje se javljaju u slikama
 - Često nose bitne informacije - značajke
 - Spomenuli smo ih u uvodnom dijelu o analizi slike

$$(I * G)(x, y) = \sum_{-\infty}^{+\infty} I(k, l) G(x - k, y - l)$$

Detekcija rubova

- Sobel

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Prewitt

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

- Roberts

$$\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

- Za razumijevanje funkcioniranja pogledajte prethodni opis korelacije
- S obzirom da orijentacija ruba nije ograničena, potrebne su barem dvije maske

Detekcija rubova

7	2	3	3	8
4	5	3	8	4
3	3	2	8	4
2	8	7	2	7
5	4	4	5	4

*

1	0	-1
1	0	-1
1	0	-1

=

6		

$$\begin{aligned} &7 \times 1 + 4 \times 1 + 3 \times 1 + \\ &2 \times 0 + 5 \times 0 + 3 \times 0 + \\ &3 \times -1 + 3 \times -1 + 2 \times -1 \\ &= 6 \end{aligned}$$

Autor: Faruk Kaledibi

Detekcija značajki

- Postoje operatori za traženje složenijih značajki slike poput kutova (sjecišta dva ruba)
- Postoje i složenije značajke na slici
- Postavlja se pitanje kako formirati maske za složenije značajke

Detekcija značajki

- Objekti se obično sastoje od nekih podobjekata
 - Ti podobjekti su značajke objekta
 - Često objekti i njihovi podobjekti znatno variraju u izgledu
- Kako formirati maske za detekciju podobjekata ili cijelih objekata?
- Koliko takvih maski treba biti da bi detekcija uvijek bila moguća?

Detekcija objekata

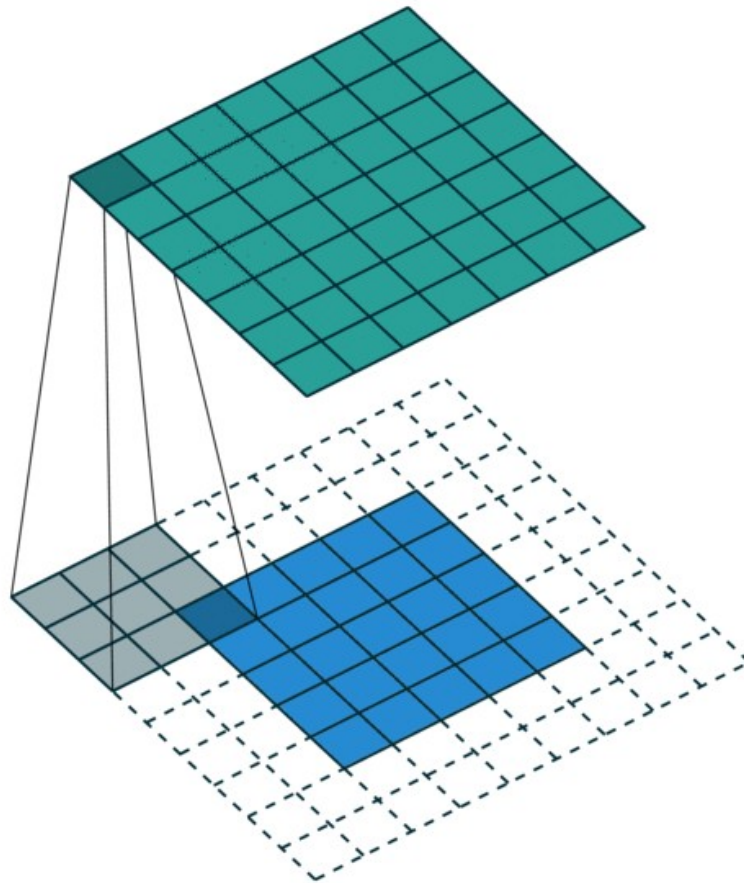
- Čini se da problem nije univerzalno rješiv
- Ipak, u ograničenim uvjetima bi možda bio rješiv
 - Ograničimo se na jedan problem
 - Traženje ograničenog broja objekata
 - Možda bi mogle pomoći neuronske mreže

Detekcija objekata

- Koje karakteristike bi detektor objekata u slici trebao imati?
 - Invarijantnost na translataciju
 - Invarijantnost na skaliranje
 - Invarijantnost na rotaciju
 - Invarijantnost na druge geometrijske transformacije
 - Ako je objekt manji od cijele slike, za detekciju na nekoj poziciji trebamo gledati samo lokalno susjedstvo

Invarijantnost 2D konvolucije

- Konvolucija omogućava invarijantnost na pomak



Invarijantnost 2D konvolucije

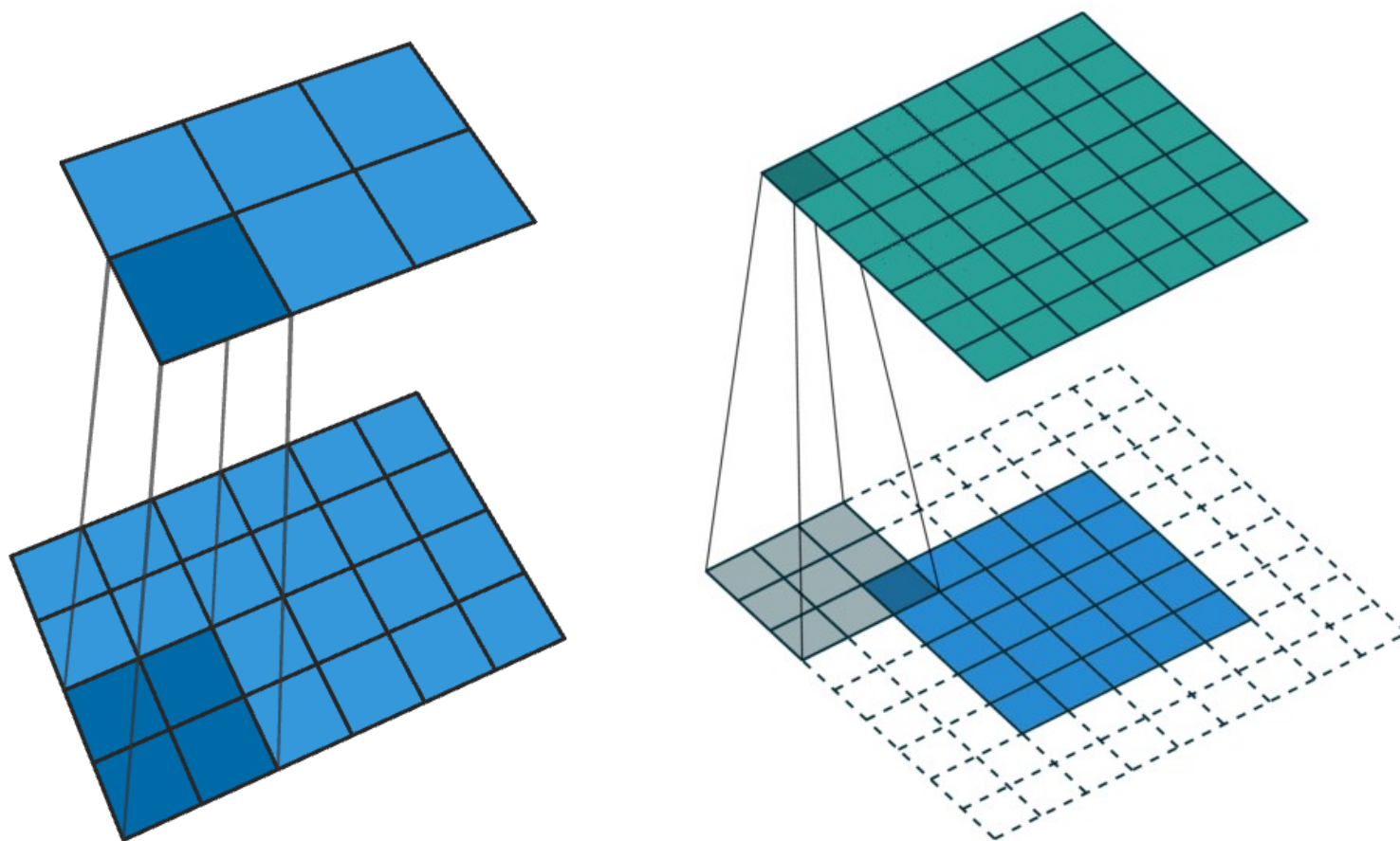
- Invarijantnost na druge geometrijske transformacije može se postići specijaliziranim konvolucijskim maskama
 - Za razne orijentacije rubova su nam trebale dvije
- Isto vrijedi i za sve ostale varijacije koje se mogu naći u objektima
 - Što je objekt složeniji, potrebno je više maski
 - U kaskadi konvolucija, viši slojevi trebaju više maski

Konvolucijska mreža

- Svaka maska proizvodi novu „sliku” koju nazivamo mapa značajki
- Svaka mapa značajki zadržava informaciju o lokaciji u 2D prostoru
- Konvolucija se može raditi na više mapa značajki istovremeno
 - Ideja je da se za sljedeći sloj traži pojava koja se može javiti u bilo kojoj mapi značajki

Konvolucijska mreža

- Veličina maske određuje veličinu lokalnog susjedstva



Konvolucijska mreža

- Vrijednosti maske možemo predstaviti težinama na ulazu u neuron
- Dodaje se i aktivacijska funkcija f
- Fokusiranjem na lokalno susjedstvo eliminiramo potpunu povezanost
 - Time smanjujemo broj težina – broj parametara mreže

$$Y(x, y) = f((W * I)(x, y)) = f\left(\sum_{-\infty}^{+\infty} W(k, l) I(x - k, y - l)\right)$$

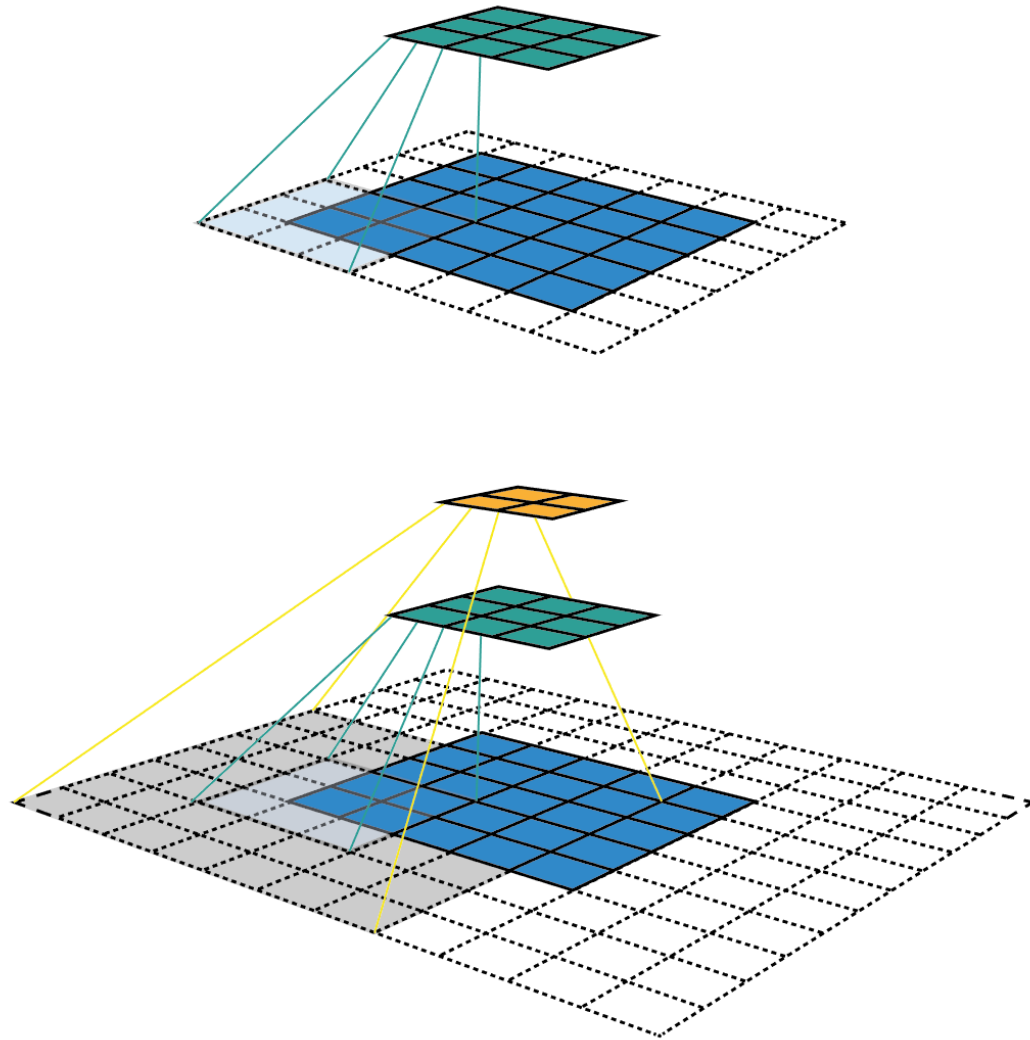
Konvolucijska mreža

- Svaki neuron traži isti objekt
 - Maska/težine trebaju biti isti za sve neurone
 - Svi dijele isti skup težina
 - Drastično smanjenje broja parametara

Hijerarhijska konvolucija

- Dodavanje više uzastopnih konvolucijskih slojeva u kaskadu
 - Omogućuje hijerarhijsku detekciju značajki
 - Kaskada efektivno čini konvoluciju sa maskom čija veličina odgovara konvoluciji svih maski u kaskadi
 - Svaka maska može biti manja
 - Dodatno smanjenje broja parametara

Hijerarhijska konvolucija



Učenje konvolucijskog sloja

- Može li konvolucijski sloj biti učen backpropagation algoritmom?

$$Y(x, y) = f((W * I)(x, y)) = f\left(\sum_{-\infty}^{+\infty} W(k, l) I(x - k, y - l)\right)$$

- Ne bi trebalo biti problema
 - Aktivacijska funkcija je prihvatljiva
 - Množenje i zbrajanje je prihvatljivo

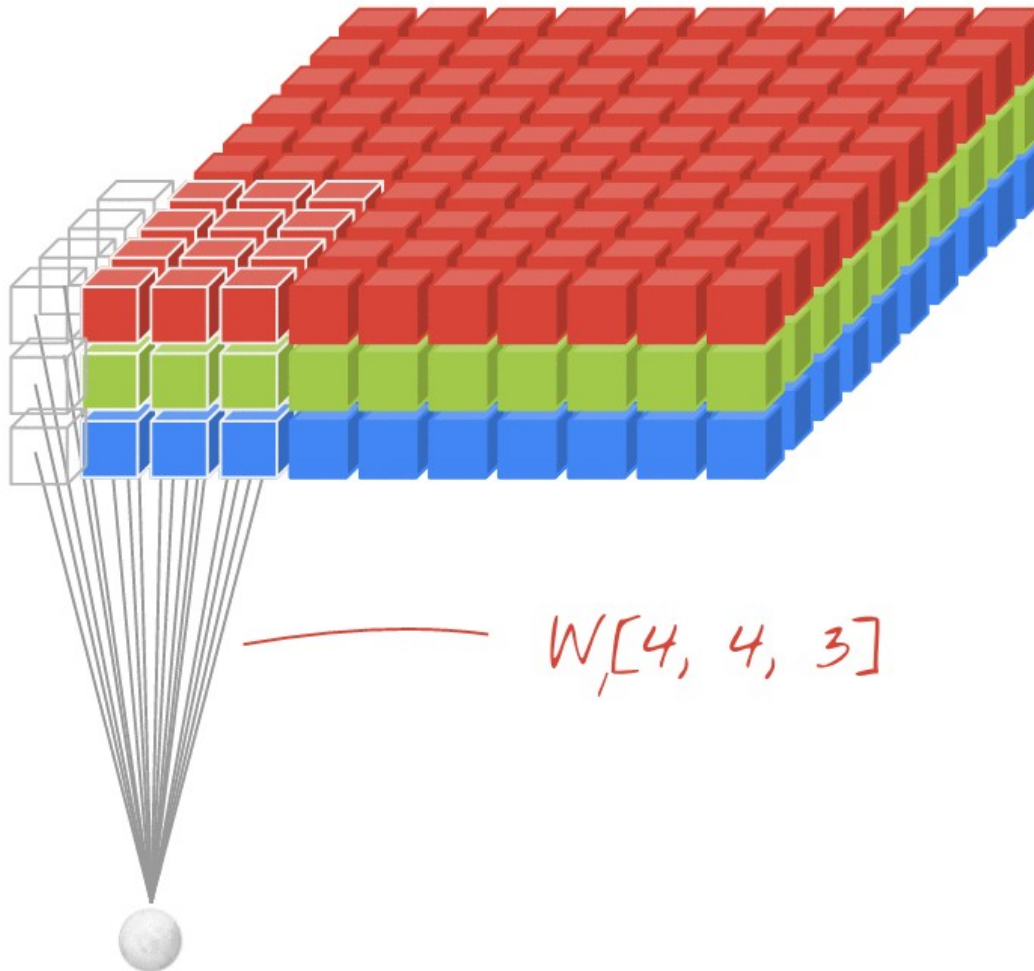
Učenje konvolucijskog sloja

- Svaki neuron u sloju iznad vidi samo svoje ograničeno receptivno polje
- Učenje se isto odvija na lokalnoj razini
- Učenje na jednoj slici je efektivno učenje na puno malih sličica

2.5D Konvolucija

- Ulaz može činiti nekoliko slika iste veličine
- Za svaku poziciju imamo vektor vrijednosti
- I dalje konvolucijom pretražujemo 2D prostor, ali se efektivno rade zasebne konvolucije na svakoj slici te se njihov rezultat zbraja
- Primjer je slika u boji
 - Po jedna slika za R, G i B kanal
- Isti princip vrijedi i za konvolucije u višim slojevima čiji ulaz čine sve mape značajki iz prethodnog sloja
 - Npr. 1024 mapi značajki iz prethodnog sloja

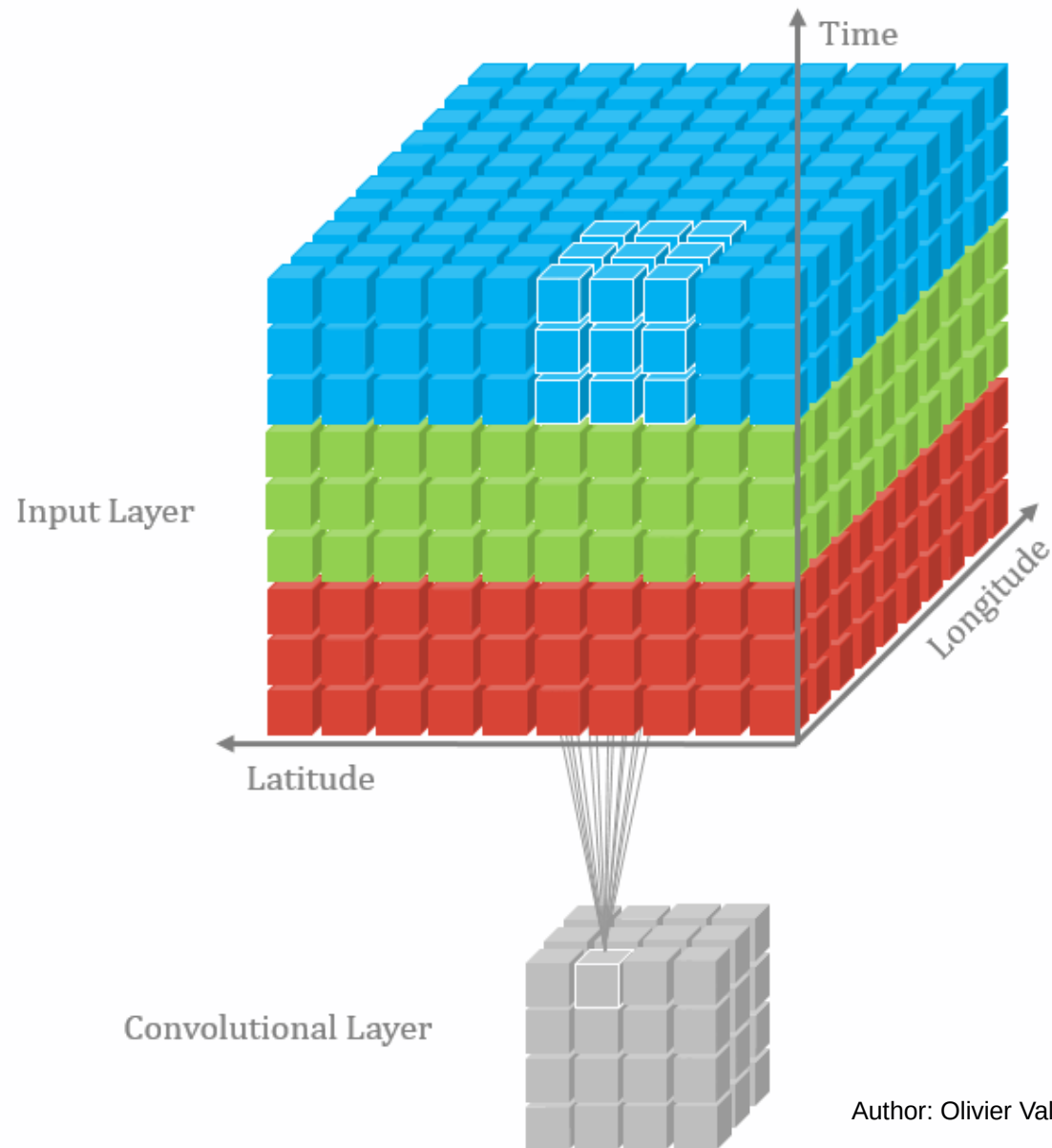
2.5D Konvolucija



3D Konvolucija

- Konvolucija se može raditi i na višedimenzionalnim podacima
 - Posmak u sve tri dimenzije
- Npr. neki medicinski uređaju snimaju volumen
- Neki medicinski uređaji snimaju i 3D video -> 4D ulaz

3D Konvolucija

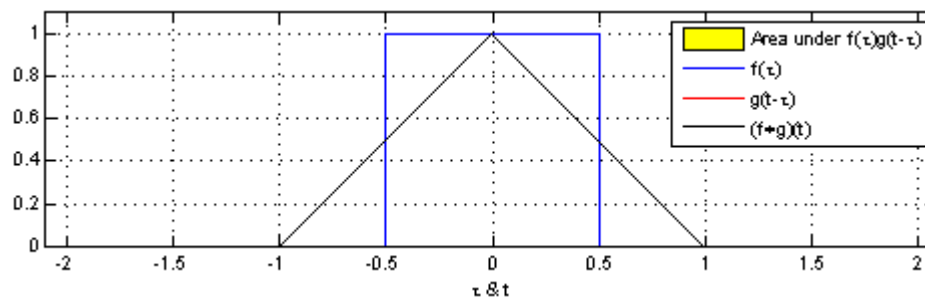


Konvolucijska mreža

- Kakva je situacija sa dimenzijama slojeva mreže i težina
 - Za slojeve mreže:
 - Širina x visina (prostorne dimenzije ulazne slike)
 - Broj kanala
 - Broj elemenata u mini grupi
 - Jedan sloj je 4D tenzor
 - Težine se mogu predstaviti 3D tenzorom
 - Širina x visina (prostorne dimenzije ulazne slike)
 - Broj kanala

Pooling

- Dodatni mehanizam za postizanje invarijantnosti na geometrijske transformacije
- Idealni detektor bi trebao imati približno isti odziv u susjednim točkama
 - Translacija ne utječe puno na detektor



Pooling

- U sljedeći sloj bi možda trebalo prenijeti informaciju o samo jednom objektu u istom susjedstvu
- Pooling sloj obavlja funkciju združivanja lokalnih informacija
 - Smanjenje prostorne rezolucije
- Postoje dva pristupa
 - Usrednjavanje
 - Maximum – često korišten
- Ne unose dodatne hiperparametre

Max-pooling

- Prosljeđuje se samo maksimalna vrijednost
 - Kao skretnica
- Gradijenti se prosljeđuju u niže slojeve samo preko tog neurona koji je imao maksimalnu vrijednost
 - Kao skretnica

7	3	5	2
8	7	1	6
4	9	3	9
0	8	4	5

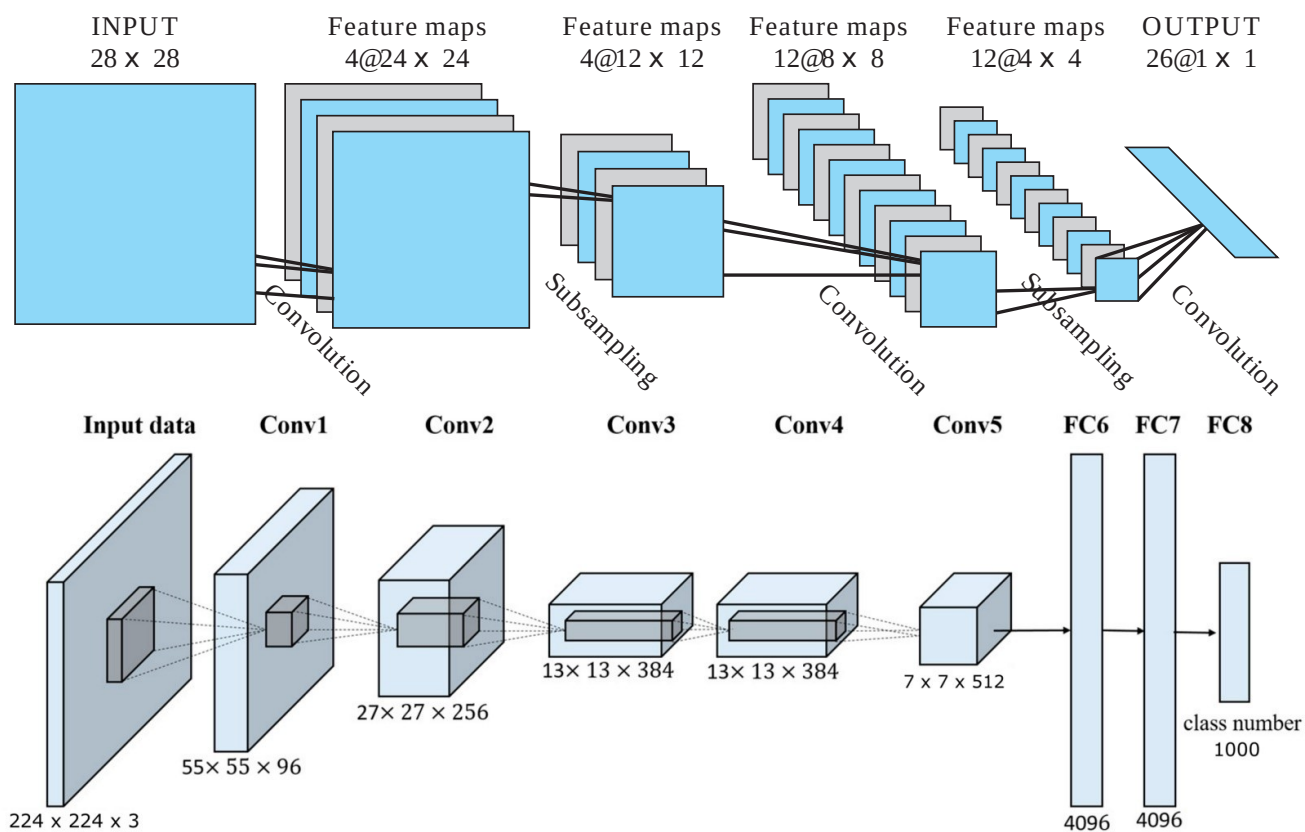
maxpool →

Output

8	6
9	9

Konvolucijske mreže

- Feed-forward mreža
- Konvolucijski slojevi
- Pooling slojevi
- Potpuno povezani slojevi



Keypoints XI

- Konvolucijski slojevi
 - Prilagođeni su analizi slika
 - Efektivno smanjuju kompleksnost mreže
 - Sprječavaju pretreniranje
 - Omogućuju dublje mreže

Mreža za detekciju objekata u slikama

- Na kraju, kako detektirati objekte
 - Odrediti klasu objekta
 - One-hot-encoding
 - Klasifikacija
 - Odrediti opisni pravokutnik (bounding box)
 - Kordinate dijagonalnih kutova (4 broja)
 - Regresija

Mreža za detekciju objekata u slikama

- Funkcija gubitka je kombinacija gubitka klasifikacije i gubitka regresije
 - Dodatni koeficijent λ uređuje međusobni odnos dviju funkcija gubitaka

$$E_{uk} = E_{class} + \lambda E_{regress}$$

Mreža za detekciju objekata u slikama

