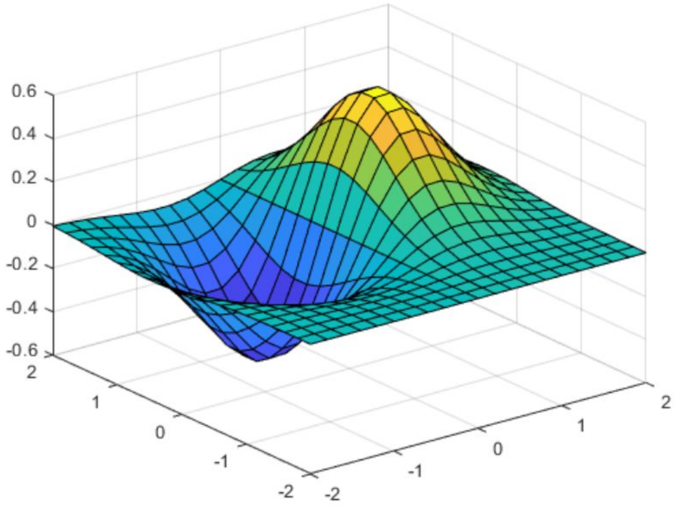
Konvoluční (neuronové) sítě

Úvod



Konvoluční sítě

- Naše síť dosahovala přesnosti > 98 %

Konvoluční sítě

- Naše síť dosahovala přesnosti > 98 %
- Přitom nebere v potaz “prostorové” struktury v datech

Konvoluční sítě

- Naše síť dosahovala přesnosti > 98 %
- Přitom nebere v potaz “prostorové” struktury v datech
- Tyto struktury se musí síť naučit z tréninkových dat

Konvoluční sítě

- Naše síť dosahovala přesnosti > 98 %
- Přitom nebere v potaz “prostorové” struktury v datech
- Tyto struktury se musí síť naučit z tréninkových dat
- Konvoluční sítě jsou navržené k tomu, aby uměly využít prostorových struktur v datech => hodí se především na rozpoznání obrazu

Konvoluční sítě

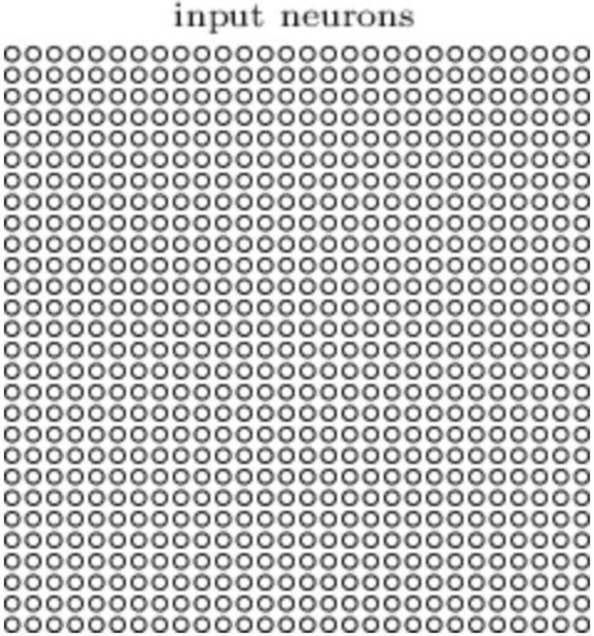
- Naše síť dosahovala přesnosti > 98 %
- Přitom nebere v potaz “prostorové” struktury v datech
- Tyto struktury se musí síť naučit z tréninkových dat
- Konvoluční sítě jsou navržené k tomu, aby uměly využít prostorových struktur v datech => hodí se především na rozpoznání obrazu
- Díky tomu se trénují rychle, což umožňuje trénink velkých hlubokých sítí

Konvoluční sítě

- Tři koncepty: lokální receptivní pole, sdílené váhy a pooling (*local receptive fields, shared weights, and pooling*)

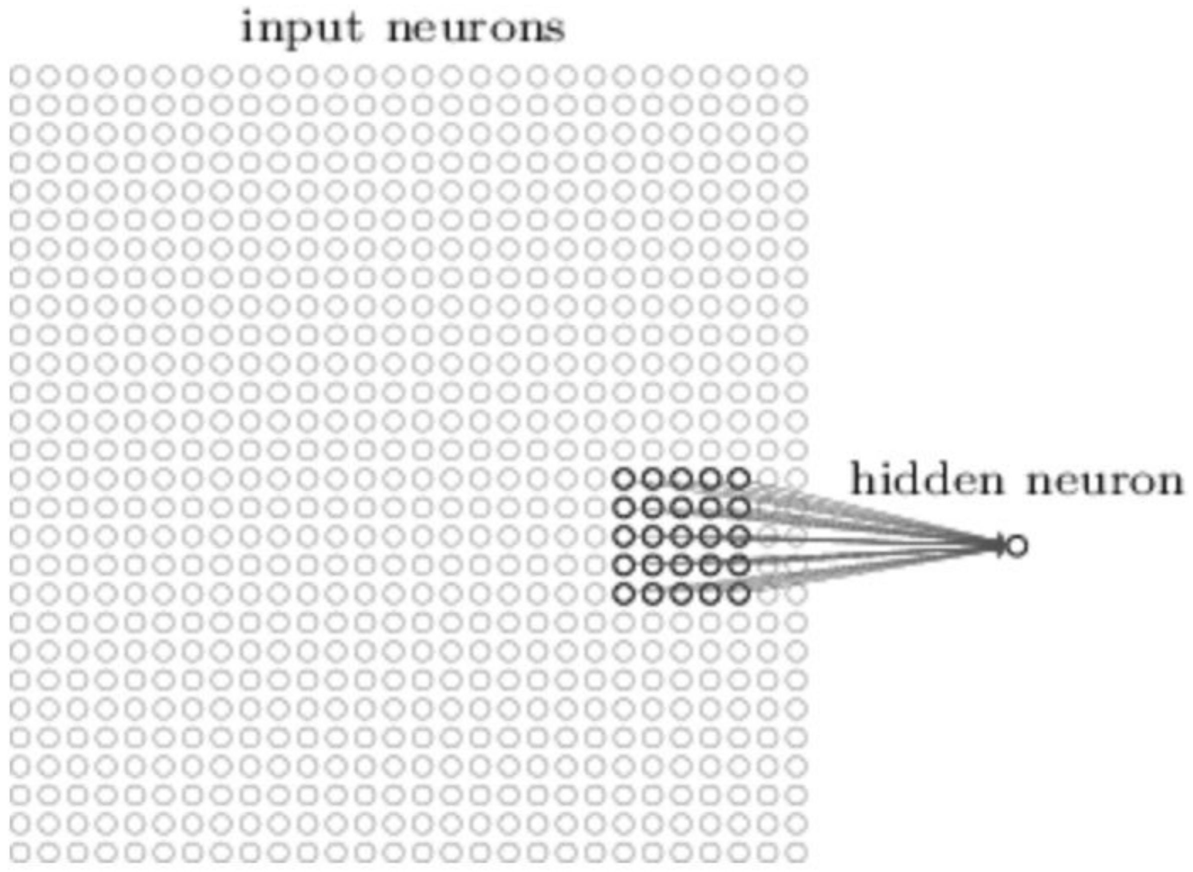
Konvoluční sítě

- Tři koncepty: lokální receptivní pole, sdílené váhy a pooling (*local receptive fields, shared weights, and pooling*)
- Začneme tím, že vstupní vrstvu budeme uvažovat jako mřížku neuronů (místo vektoru)



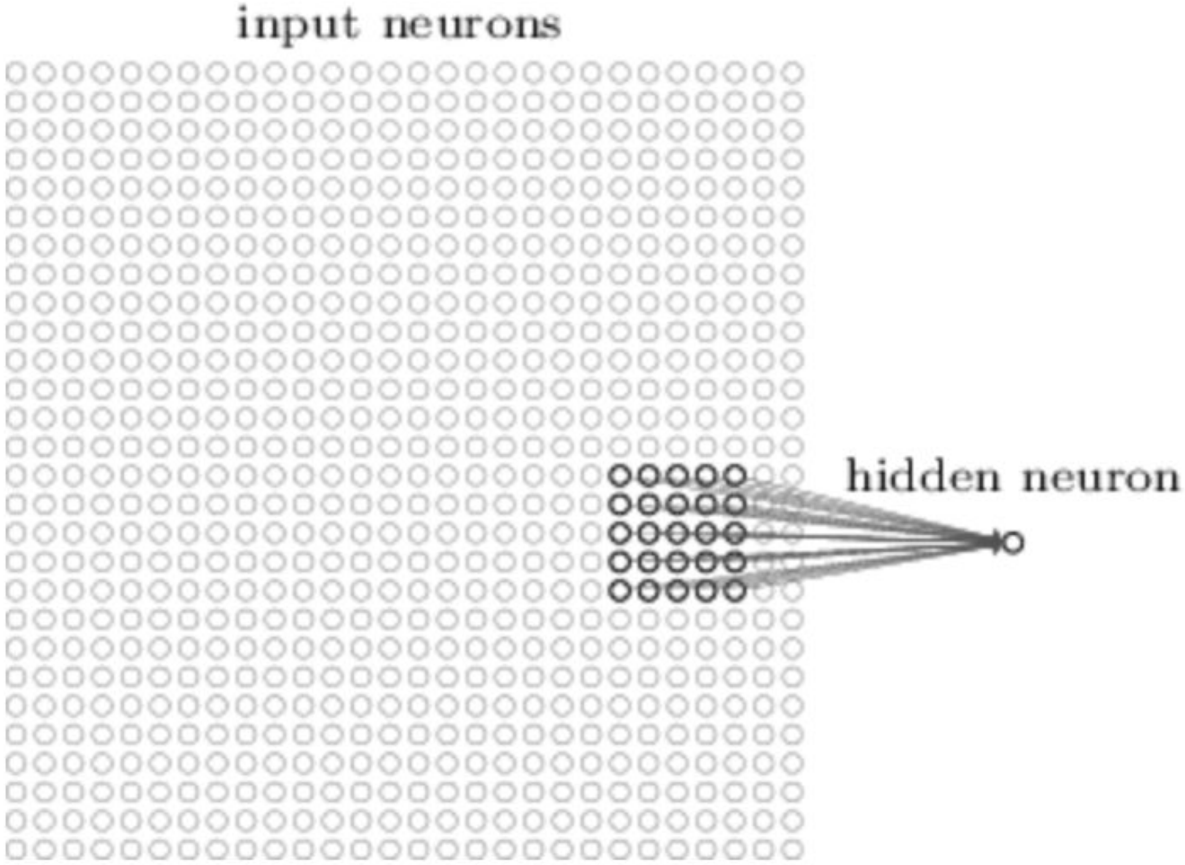
Konvoluční sítě - lokální receptivní pole

- Spojení malé lokalizované oblasti neuronů s neuronem v další vrstvě



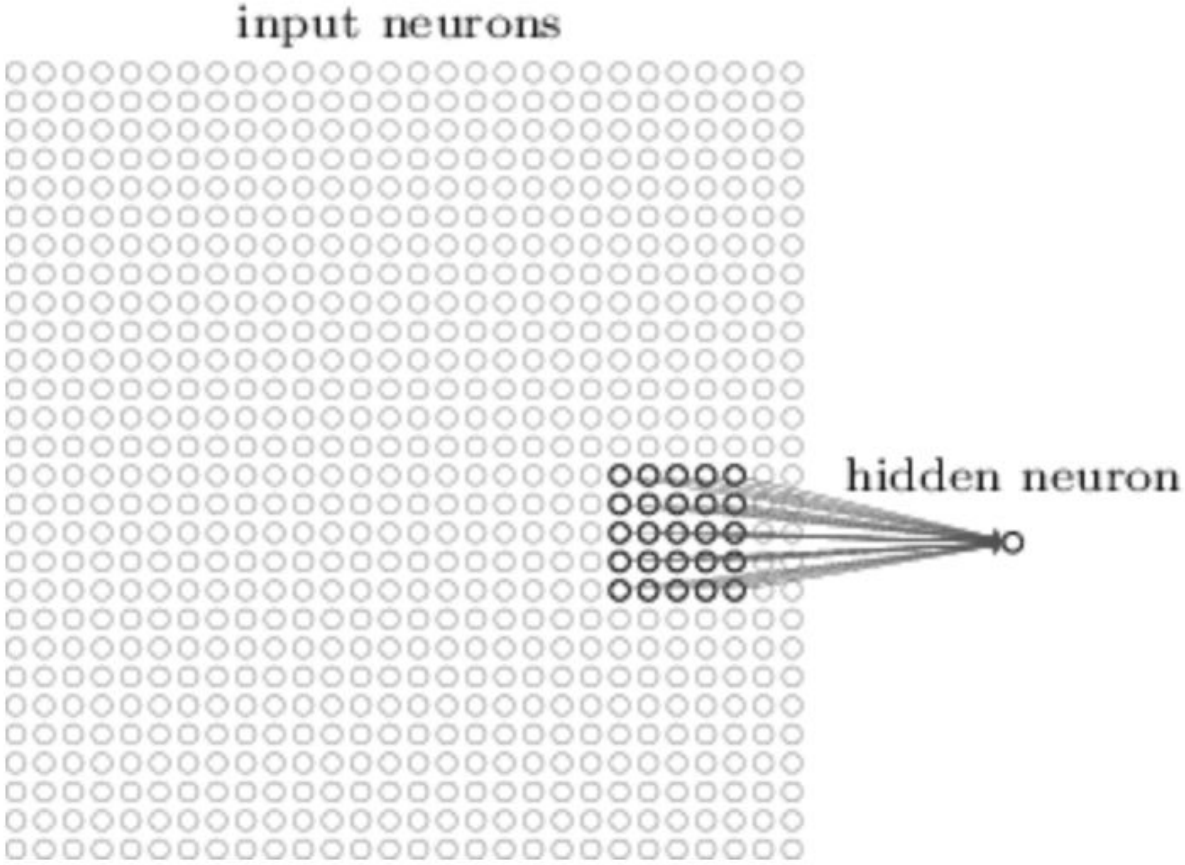
Konvoluční sítě - lokální receptivní pole

- Každému spojení přísluší váha a celému neuronu bias



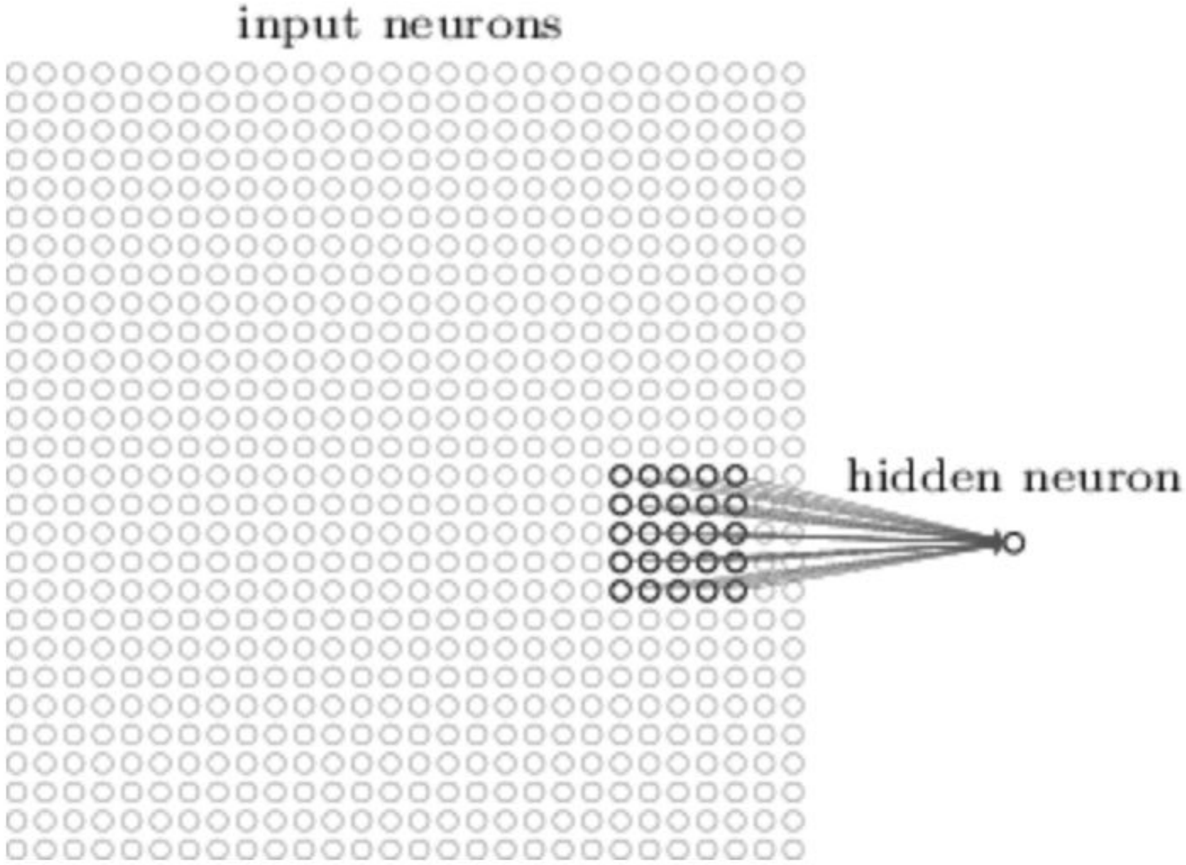
Konvoluční sítě - lokální receptivní pole

- Každému spojení přísluší váha a celému neuronu bias
- Této oblasti říkáme lokální receptivní pole



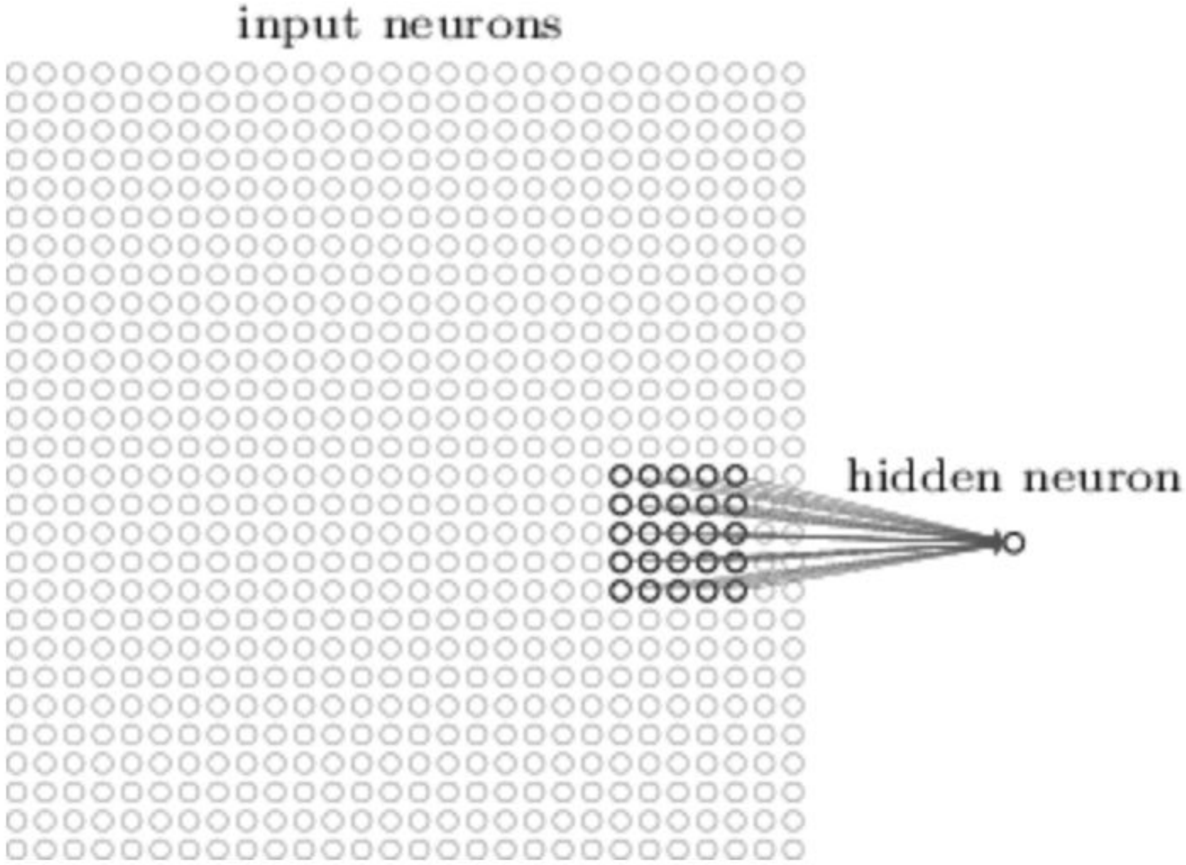
Konvoluční sítě - lokální receptivní pole

- Tento neuron tedy analyzuje “kousek” inputu



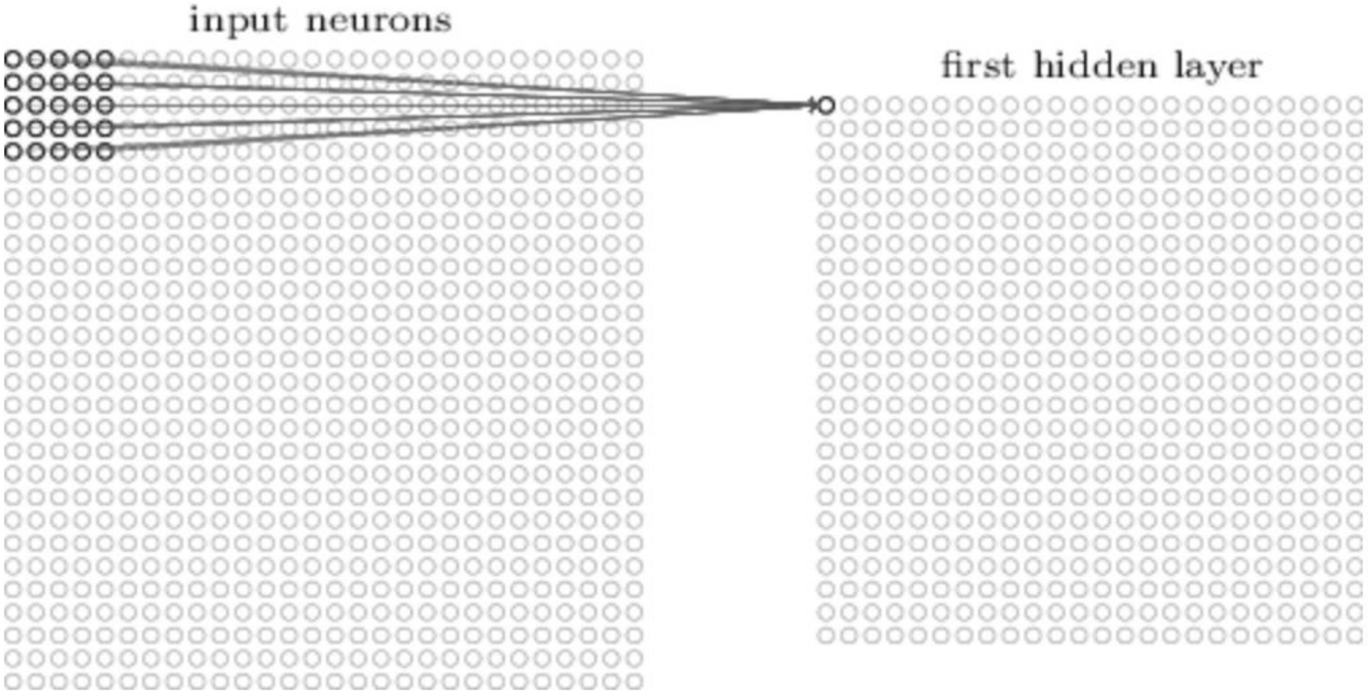
Konvoluční sítě - lokální receptivní pole

- Tento neuron tedy analyzuje “kousek” inputu
- Posouváme lokální receptivní pole přes celý input



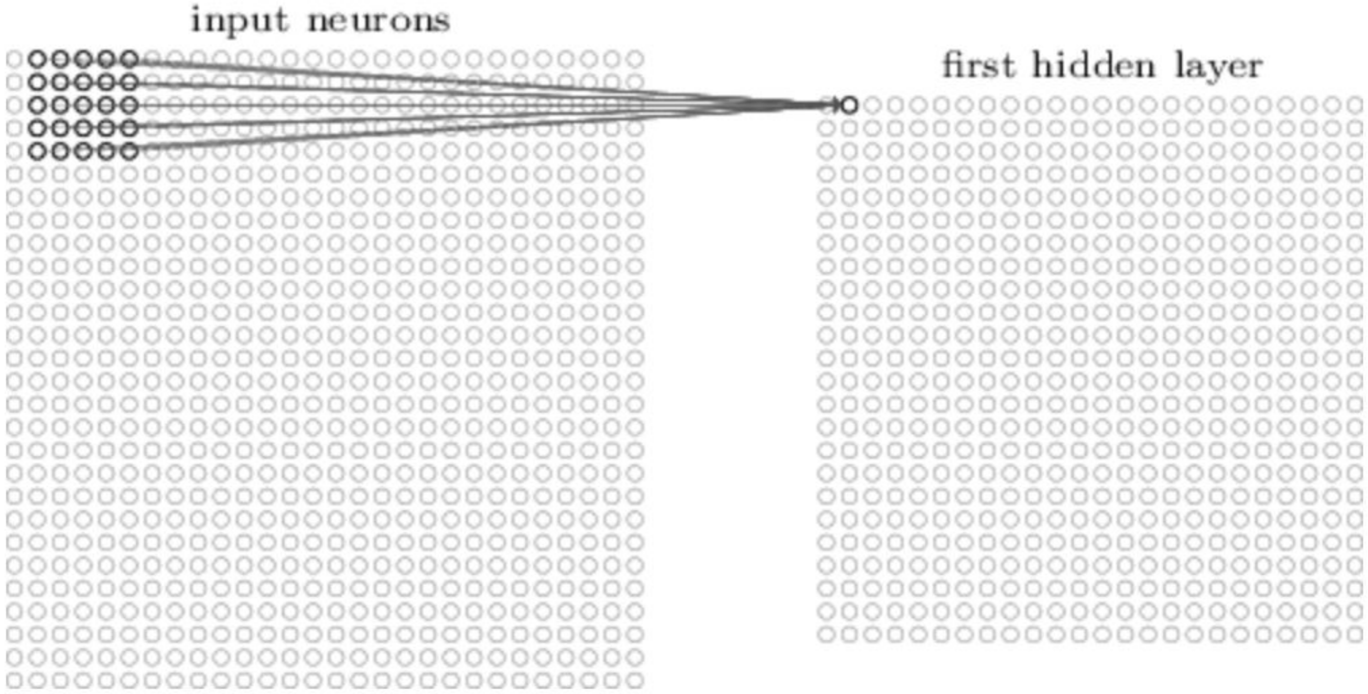
Konvoluční sítě - lokální receptivní pole

- Nový hyperparametr ***stride*** - velikost kroku (často 1)



Konvoluční sítě - lokální receptivní pole

- Nový hyperparametr ***stride*** - velikost kroku (často 1)



Konvoluční sítě - lokální receptivní pole

- Pro stride = 1:
- Pro input 28x28 a lokální receptivní pole 5x5 bude mít skrytá vrstva 24x24 neuronů (okno 5x5 můžeme posunout doprava/dolů 23x než narazíme na okraj)

Konvoluční sítě - lokální receptivní pole

- Pro stride = 1:
- Pro input 28x28 a lokální receptivní pole 5x5 bude mít skrytá vrstva 24x24 neuronů (okno 5x5 můžeme posunout doprava/dolů 23x než narazíme na okraj)
- Obecně pro input $(n_1 \times n_2)$ a lok. rec. pole $(m_1 \times m_2)$ bude mít skrytá vrstva $(n_1 - m_1 + 1 \times n_2 - m_2 + 1)$ neuronů

Konvoluční sítě - sdílené váhy

- Každý neuron v konvoluční vrstvě má $n_1 \times n_2$ vah a 1 bias

Konvoluční sítě - sdílené váhy

- Každý neuron v konvoluční vrstvě má $n_1 \times n_2$ vah a 1 bias
- Tyto parametry jsou stejné pro celou vrstvu, tj. pro j-tý neuron k-té vrstvy platí:

$$\sigma\left(b + \sum_{l=0}^{n_1} \sum_{m=0}^{n_2} w_{l,m} a_{j+l,k+m}\right)$$

kde sigma je libovolná akt. funkce, $w_{l,m}$ je matice vah $n_1 \times n_2$, b je bias a $a_{x,y}$ je aktivace inputu na pozici x,y lok. rec. pole

Konvoluční sítě - sdílené váhy

- Všechny neurony v této skryté vrstvě tedy detekují stejný příznak (feature)

Konvoluční sítě - sdílené váhy

- Všechny neurony v této skryté vrstvě tedy detekují stejný příznak (feature)
- Představte si, že váhy a bias jsou určeny tak, že neuron detekuje vertikální čáru => tato schopnost se hodí na všech částech obrázku

Konvoluční sítě - sdílené váhy

- Všechny neurony v této skryté vrstvě tedy detekují stejný příznak (feature)
- Představte si, že váhy a bias jsou určeny tak, že neuron detekuje vertikální čáru => tato schopnost se hodí na všech částech obrázku
- Translační invariance

Konvoluční sítě - sdílené váhy

- Někdy tomuto mapování mezi input a skrytou vrstvou (tj. aktivaci skryté vrstvy) říkáme ***feature map***

Konvoluční sítě - sdílené váhy

- Někdy tomuto mapování mezi input a skrytou vrstvou (tj. aktivaci skryté vrstvy) říkáme ***feature map***
- Vahám a biasu říkáme ***sdílené***

Konvoluční sítě - sdílené váhy

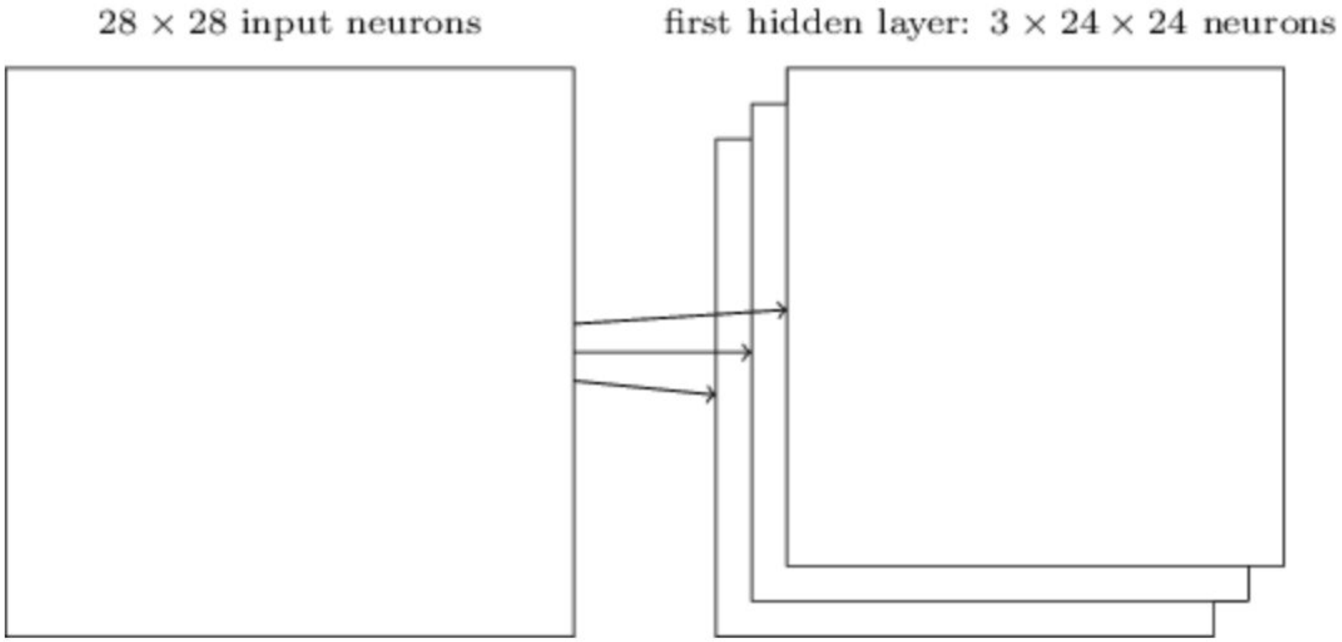
- Někdy tomuto mapování mezi input a skrytou vrstvou (tj. aktivaci skryté vrstvy) říkáme ***feature map***
- Vahám a biasu říkáme ***sdílené***
- Sdílené váhy a bias tvoří ***feature extractor***, kterému říkáme ***kernel (jádro)*** nebo ***filter (filtr)***

Konvoluční sítě - sdílené váhy

- Někdy tomuto mapování mezi input a skrytou vrstvou (tj. aktivaci skryté vrstvy) říkáme ***feature map***
- Vahám a biasu říkáme ***sdílené***
- Sdílené váhy a bias tvoří ***feature extractor***, kterému říkáme ***kernel (jádro)*** nebo ***filter (filtr)***
- Aby mohla síť detekovat více příznaků, takových filtrů definujeme několik. Ty pak tvoří ***konvoluční vrstvu***

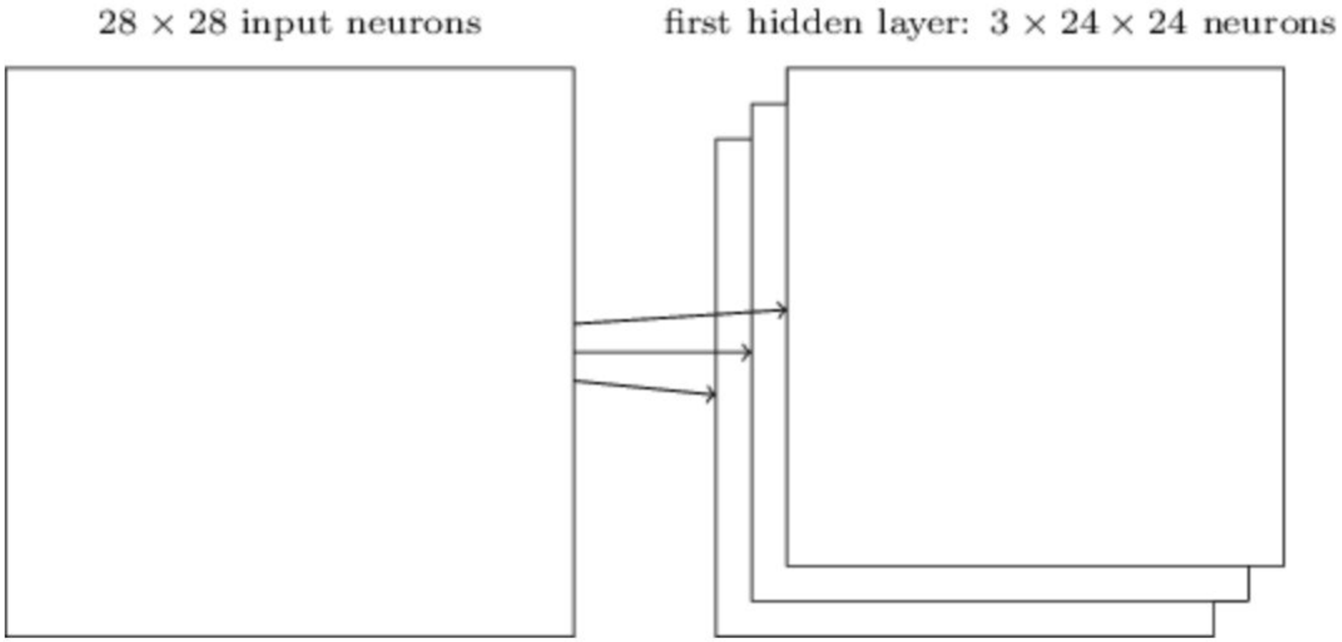
Konvoluční sítě - sdílené váhy

- Aby mohla síť detekovat více příznaků, takových filtrů definujeme několik. Ty pak tvoří ***konvoluční vrstvu***



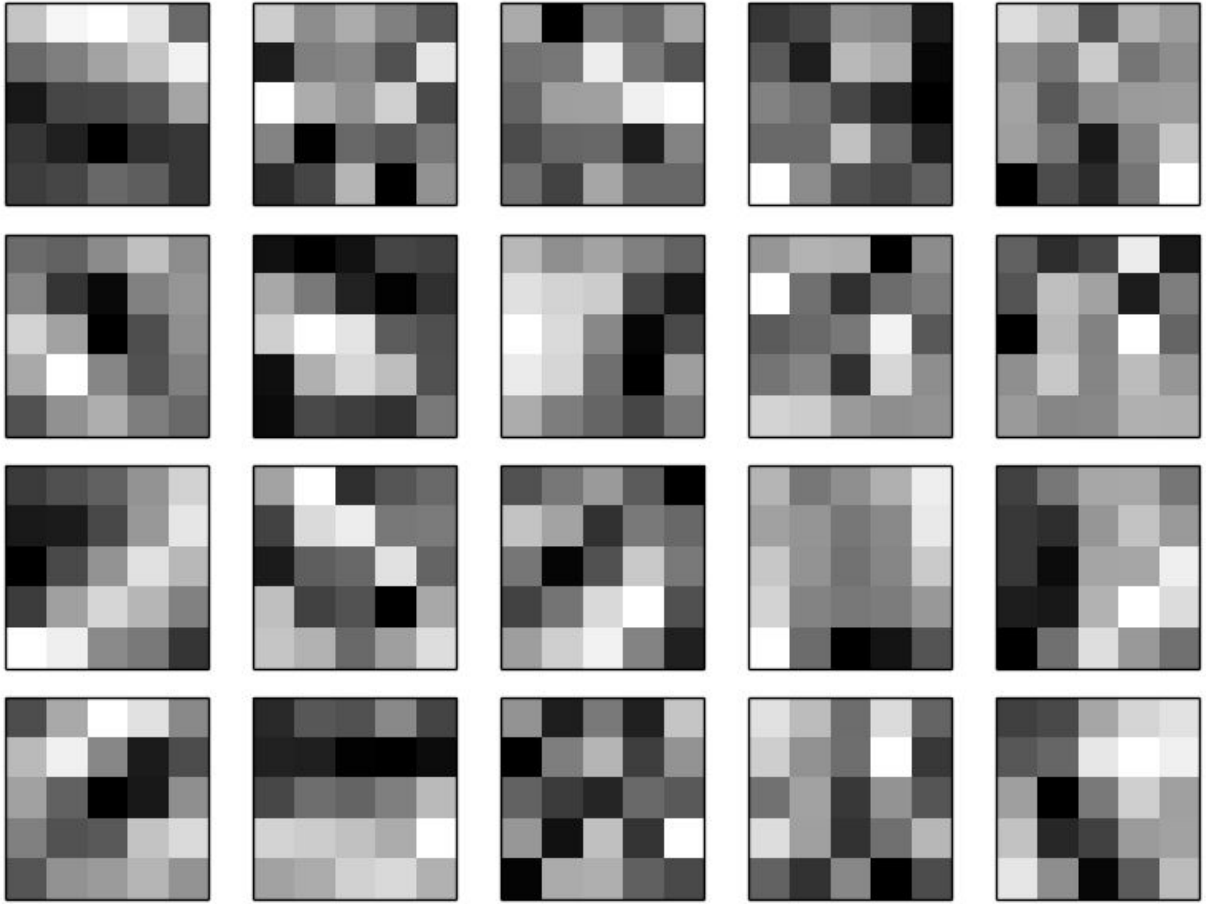
Konvoluční sítě - sdílené váhy

- Aby mohla síť detekovat více příznaků, takových filtrů definujeme několik. Ty pak tvoří ***konvoluční vrstvu***
- 3 filtry tvořící 3 feature mapy detekující 3 příznaky



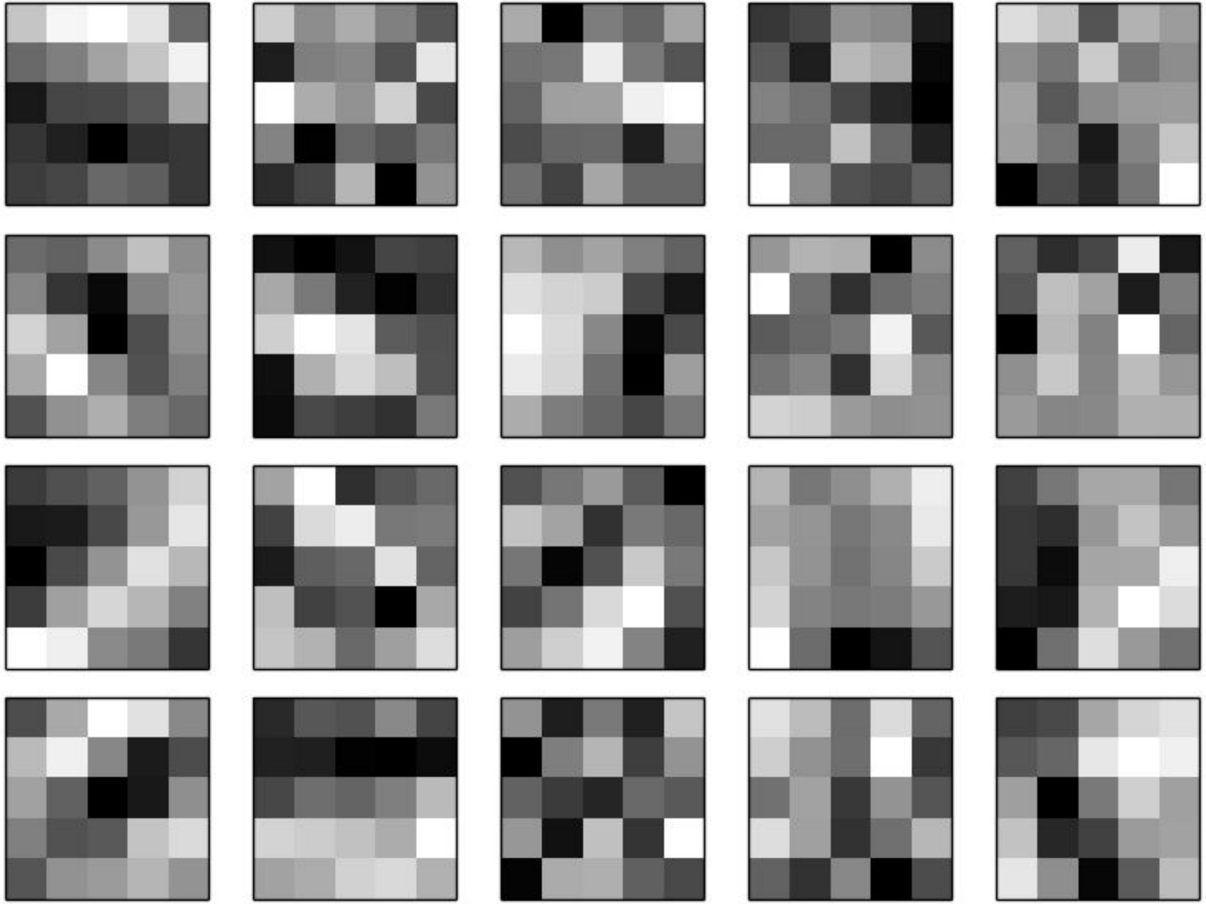
Konvoluční sítě - sdílené váhy

- Příklad filtrů pro MNIST:



Konvoluční sítě - sdílené váhy

- Příklad filtrů pro MNIST:
- Čím tmavší pixel, tím vyšší váha, tj. filter reaguje na tyto pixely silněji



Konvoluční sítě - sdílené váhy

- Filtry využívají prostorové struktury, jak jsme chtěli

Konvoluční sítě - sdílené váhy

- Filtry využívají prostorové struktury, jak jsme chtěli
- Výhoda sdílených vah a biasů je nízký počet parametrů při učení - pro filtr (5x5) potřebujeme pouze 26 parametrů

Konvoluční sítě - sdílené váhy

- Filtry využívají prostorové struktury, jak jsme chtěli
- Výhoda sdílených vah a biasů je nízký počet parametrů při učení - pro filtr (5x5) potřebujeme pouze 26 parametrů
- Pro MNIST s inputem 28x28 a konvoluční sítí se 40 filtry 5x5 máme 520 parametrů, pro fully connected se 30 neurony je to 23550 parametrů

Konvoluční sítě - pooling layers

- Po konvoluční vrstvě zpravidla následuje pooling vrstva (někdy se jejich kombinace uvažuje jako jedna vrstva)

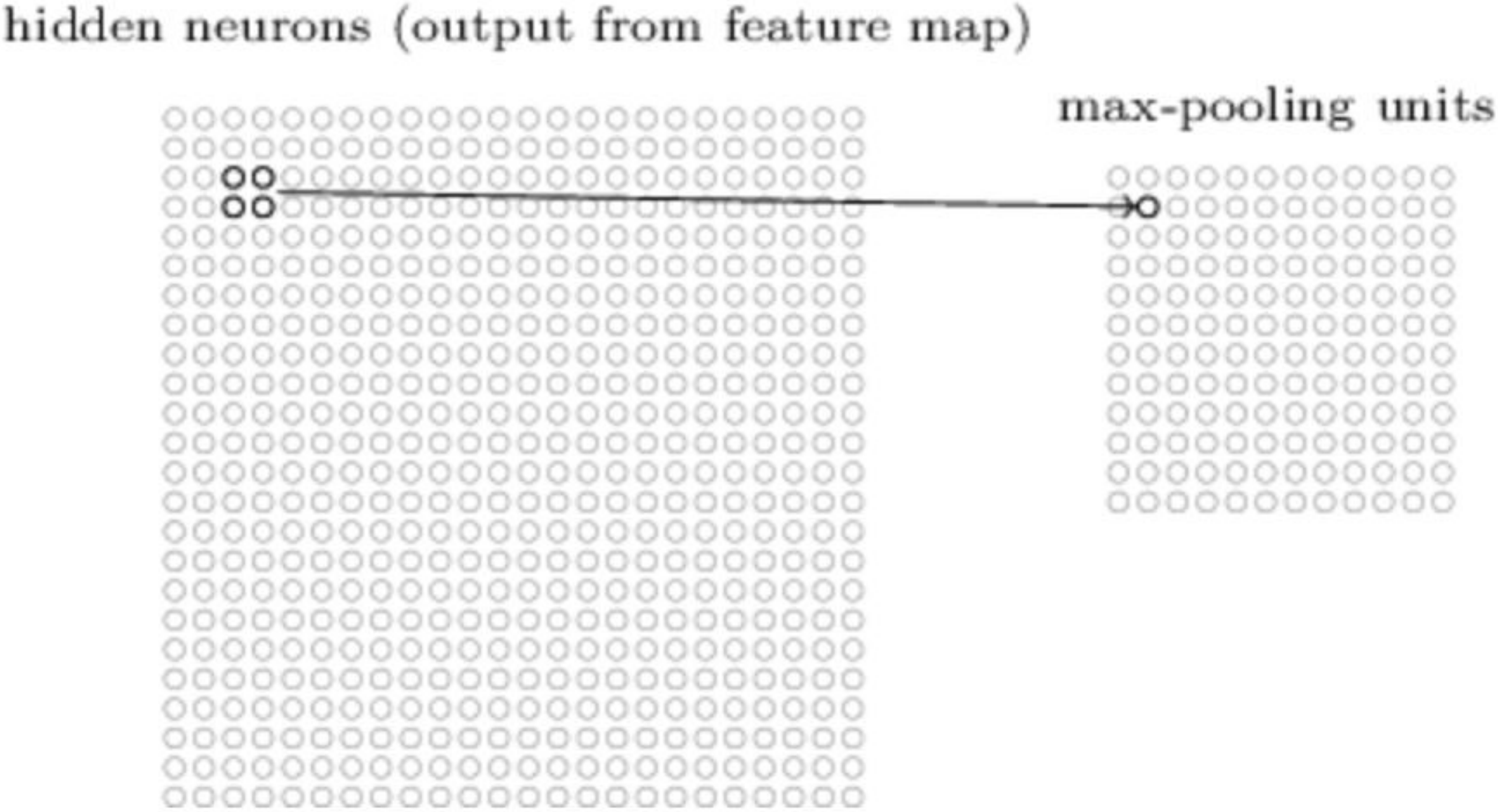
Konvoluční sítě - pooling layers

- Po konvoluční vrstvě zpravidla následuje pooling vrstva (někdy se jejich kombinace uvažuje jako jedna vrstva)
- Jejich úkolem je zjednodušit output konvoluční vrstvy agregováním několika neuronů (např. oblast (2x2) - další hyperparametr) do jednoho

Konvoluční sítě - pooling layers

- Po konvoluční vrstvě zpravidla následuje pooling vrstva (někdy se jejich kombinace uvažuje jako jedna vrstva)
- Jejich úkolem je zjednodušit output konvoluční vrstvy agregováním několika neuronů (např. oblast (2x2) - další hyperparametr) do jednoho
- Existuje několik tipů, např. max-pooling, average-pooling nebo L2 pooling

Konvoluční sítě - pooling layers



Konvoluční sítě - pooling layers

- Pooling významně snižuje počet parametrů v síti

Konvoluční sítě - pooling layers

- Pooling významně snižuje počet parametrů v síti
- Pooling nám říká, jestli je daný příznak v inputu, přičemž jeho přesnou pozici zahazuje - podstatná je přítomnost tohoto příznaku s ohledem na ostatní příznaky

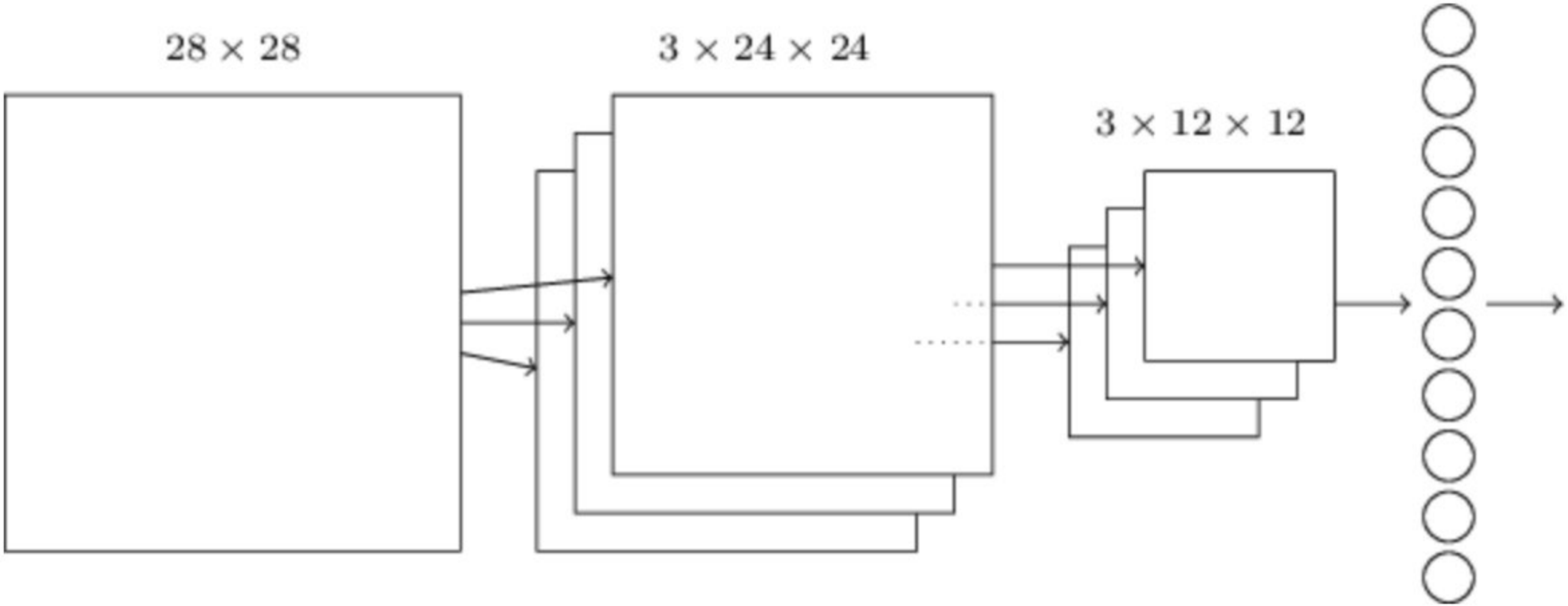
Konvoluční sítě - pooling layers

- Pooling významně snižuje počet parametrů v síti
- Pooling nám říká, jestli je daný příznak v inputu, přičemž jeho přesnou pozici zahazuje - podstatná je přítomnost tohoto příznaku s ohledem na ostatní příznaky
- Na pooling můžeme napojit fully-connected layer, jejíž úkol je na základě přitomnosti příznaků rozhodnout, jak input klasifikovat

Konvoluční sítě - pooling layers

- Pooling významně snižuje počet parametrů v síti
- Pooling nám říká, jestli je daný příznak v inputu, přičemž jeho přesnou pozici zahazuje - podstatná je přítomnost tohoto příznaku s ohledem na ostatní příznaky
- Na pooling můžeme napojit fully-connected layer, jejíž úkol je na základě přitomnosti příznaků rozhodnout, jak input klasifikovat
- Pooling vrstva nemá váhy ani bias

Konvoluční sítě - celá síť



Konvoluční sítě - proč konvoluční?

- Posouváním filtru provádíme konvoluci:

$$\sigma\left(b + \sum_{l=0}^{n_1}\sum_{m=0}^{n_2} w_{l,m}a_{j+l,k+m}\right)$$ můžeme přepsat jako

$$a_{\text{conv}} = \sigma(b + w \circledast a_{\text{input}})$$ přičemž $\circledast$ je

konvoluční operátor

Konvoluční sítě - proč konvoluční?

- Posouváním filtru provádíme konvoluci:

$$\sigma\left(b+\sum_{l=0}^{n_1}\sum_{m=0}^{n_2} w_{l,m}a_{j+l,k+m}\right)$$ můžeme přepsat jako

$$a_{\text{conv}} = \sigma(b + w \circledast a_{\text{input}})$$ přičemž $\circledast$ je

konvoluční operátor

- Často v konvoluční síti mluvíme o jednotkách (unit) místo neuronů, podobně se často říká jen “konvoluční sítě” místo “konvoluční neuronové sítě”

Konvoluční sítě - backpropagation

Backpropagation - 4 rovnice algoritmu

- **Shrnutí:**

1. $\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'\left(z_j^L\right)$ zapíšeme jako $\nabla_a C \odot \sigma'\left(z^L\right)$

2. $\delta^l = \left(\left(w^{l+1}\right)^T \delta^{l+1}\right) \odot \sigma'\left(z^l\right)$, po prvcích $\delta_j^l = \sum_{k=1}^{n_{l+1}} w_{kj}^{l+1} \delta_k^{l+1} \sigma'\left(z_j^l\right)$

3. $\frac{\partial C}{\partial b_j^l} = \delta_j^l$

4. $\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l$

Konvoluční sítě - backpropagation

- Budeme chtít tyto čtyři rovnice odvodit pro jednoduchou konvoluční síť s konvoluční,pooling vrstvou a fully connected vrstvou. Budeme uvažovat konvoluční síť s jedním filtrem a max-pooling (2x2)

Konvoluční sítě - backpropagation

- Budeme chtít tyto čtyři rovnice odvodit pro jednoduchou konvoluční síť s konvoluční,pooling vrstvou a fully connected vrstvou. Budeme uvažovat konvoluční síť s jedním filtrem a max-pooling (2x2)
- Vyjdeme z důkazů původního backpropagation algoritmu a upravíme je pro tuto síť

Konvoluční sítě - backpropagation

- Zavedeme si následující značení:

$a^0_{j,k}$ aktivace vstupní vrstvy

$w^1_{l,m}$ sdílené váhy konvoluční vrstvy

b^1 sdílený bias konvoluční vrstvy

$z^1_{j,k}$ vážený input do neuronu (j,k), $z^1_{j,k} = b^1 + \sum_l \sum_m w^1_{l,m} a^0_{j+l,k+m}$

$a^1_{j,k}$ aktivace neuronu (j,k) konvoluční vrstvy, $a^1_{j,k} = \sigma\left(z^1_{j,k}\right)$

$a^2_{j,k}$ aktivace neuronu (j,k) pooling vrstvy, $a^2_{j,k} = \max\left\{a^1_{2j\pm1,2k\pm1}\right\}$

$w^3_{l,j,k}$ váha mezi neuronem (j,k) pooling vrstvy a neuronem l output vrstvy

b^3_l bias output vrstvy

z^3_l vážený input do neuronu l output vrstvy, $z^3_l = b^3_l + \sum_j \sum_k w^3_{l,j,k} a^2_{j,k}$

a^3_l aktivace neuronu l output vrstvy, $a^3_l = \sigma\left(z^3_l\right)$

Backpropagation - 4 rovnice algoritmu: důkaz

- **Důkaz 1. :** $\delta_j^L = \frac{\partial C}{\partial a_j^L}\sigma'\left(z_j^L\right)$
- Vyjdeme z definice a aplikujeme derivaci složené funkce:

$$\delta_j^L \equiv \frac{\partial C}{\partial z_j^L} = \sum_{k=1}^{n_L} \frac{\partial C}{\partial a_k^L}\frac{\partial a_k^L}{\partial z_j^L}$$

přitom aktivace k-tého neuronu poslední vrstvy závisí pouze na $z_k^L \implies \frac{\partial a_k^L}{\partial z_{j^L}} = 0 \text{ pokud } j \neq k$

Dohromady tedy $\delta_j^L \equiv \frac{\partial C}{\partial z_j^L} = \sum_{k=1}^{n_L} \frac{\partial C}{\partial a_k^L}\frac{\partial a_k^L}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L}\frac{\partial a_j^L}{\partial z_j^L}$

Dále víme, že $a_j^L = \sigma\left(z_j^L\right)$, což můžeme dosadit do předchozího vztahu

$$\delta_j^L \equiv \frac{\partial C}{\partial z_j^L} = \sum_{k=1}^{n_L} \frac{\partial C}{\partial a_k^L}\frac{\partial a_k^L}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L}\frac{\partial a_j^L}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L}\frac{\partial \sigma}{\partial z_j^L}\left(z_j^L\right) = \frac{\partial C}{\partial a_j^L}\sigma'\left(z_j^L\right)$$

Konvoluční sítě - backpropagation

- 1. rovnice platí i pro konvoluční sítě

Backpropagation - 4 rovnice algoritmu: důkaz

- **Důkaz 2. :** $\delta_j^l = \sum_{k=1}^{n_{l+1}} w_{kj}^{l+1} \delta_k^{l+1} \sigma'\left(z_j^l\right)$

- Chceme přepsat chybu v ***l***-té vrstvě pomocí chyby v ***(l+1)*** vrstvě

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l}$$

zároveň z definice ***z***: $z_k^{l+1} = \sum_{j=1}^{n_{l+1}} w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_{j=1}^{n_{l+1}} w_{kj}^{l+1} \sigma\left(z_j^l\right) + b_k^{l+1}$

což můžeme zderivovat: $\dfrac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1} \sigma'\left(z_j^l\right)$

a dosadit do původní rovnice:

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \delta_k^{l+1} w_{kj}^{l+1} \sigma'\left(z_j^l\right) = \sum_{k=1}^{n_{l+1}} w_{kj}^{l+1} \delta_k^{l+1} \sigma'\left(z_j^l\right)$$

Konvoluční sítě - backpropagation

- 1. rovnice platí i pro konvoluční sítě $\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L)$
- Pooling vrstva nemá žádný vážený input **Z**, důkaz použít nelze. Musíme spočítat chybu neuronu v této vrstvě ručně

Konvoluční sítě - backpropagation

- 1. rovnice platí i pro konvoluční sítě $\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L)$
- Pooling vrstva nemá žádný vážený input **Z**, důkaz použít nelze. Musíme spočítat chybu neuronu v této vrstvě ručně
- Uvědomme si, že neuron (j,k) v konvoluční vrstvě se vstupuje do výpočtu maxima pro neuron $\left(\left\lfloor \frac{j}{2} \right\rfloor, \left\lfloor \frac{k}{2} \right\rfloor\right)$ v pooling vrstvě

Konvoluční sítě - backpropagation

$$\delta^1_{j,k} = \frac{\partial C}{\partial z^1_{j,k}} = \sum_l \frac{\partial C}{\partial z^3_l}\frac{\partial z^3_l}{\partial z^1_{j,k}} = \sum_l \delta^3_l \frac{\partial z^3_l}{\partial z^1_{j,k}}$$

$$= \sum_l \delta^3_l \frac{\partial z^3_l}{\partial a^2_{j',k'}}\frac{\partial a^2_{j',k'}}{\partial z^1_{j,k}}, a^2_{j',k'} \text{ je jediná aktivace v pooling vrstvě závisející na } \partial z^1_{j,k},\ \text{kde } j', k' = \left\lfloor \frac{j}{2} \right\rfloor, \left\lfloor \frac{k}{2} \right\rfloor$$

$$= \sum_l \delta^3_l w^3_{l,j,',k'} \frac{\partial a^2_{j',k'}}{\partial z^1_{j,k}}, \text{ protože } z_l^3 \text{ závisí na } a^2_{j',k'} \text{ příslušnou váhou } w^3_{l,j,',k'}$$

$$= \sum_l \delta^3_l w^3_{l,j,',k'} \frac{\partial a^2_{j',k'}}{\partial a^1_{j,k}}\frac{\partial a^1_{j,k}}{\partial z^1_{j,k}} = \sum_l \delta^3_l w^3_{l,j,',k'} \frac{\partial a^2_{j',k'}}{\partial a^1_{j,k}} \sigma\prime(z^1_{j,k})$$

Konvoluční sítě - backpropagation

$$\delta^1_{j,k} = \frac{\partial C}{\partial z^1_{j,k}} = \sum_l \frac{\partial C}{\partial z^3_l} \frac{\partial z^3_l}{\partial z^1_{j,k}} = \sum_l \delta^3_l \frac{\partial z^3_l}{\partial z^1_{j,k}}$$

$$= \sum_l \delta^3_l \frac{\partial z^3_l}{\partial a^2_{j',k'}} \frac{\partial a^2_{j',k'}}{\partial z^1_{j,k}}, a^2_{j',k'} \text{ je jediná aktivace v pooling vrstvě závisející na } \partial z^1_{j,k}, \text{ kde } j',k' = \left\lfloor \frac{j}{2} \right\rfloor, \left\lfloor \frac{k}{2} \right\rfloor$$

$$= \sum_l \delta^3_l w^3_{l,j,',k'} \frac{\partial a^2_{j',k'}}{\partial z^1_{j,k}}, \text{ protože } z_l^3 \text{ závisí na } a^2_{j',k'} \text{ příslušnou váhou } w^3_{l,j,',k'}$$

$$= \sum_l \delta^3_l w^3_{l,j,',k'} \frac{\partial a^2_{j',k'}}{\partial a^1_{j,k}} \frac{\partial a^1_{j,k}}{\partial z^1_{j,k}} = \sum_l \delta^3_l w^3_{l,j,',k'} \frac{\partial a^2_{j',k'}}{\partial a^1_{j,k}} \sigma\prime(z^1_{j,k})$$

protože $a^2_{j',k'} = \max\{a^1_{2j'\pm1,2k'\pm1}\}$, dostáváme derivaci

$$\frac{\partial a^2_{j',k'}}{\partial a^1_{j,k}} = \begin{cases} 0 & \text{pokud } a^1_{j,k} \neq \max\{a^1_{2j'\pm1,2k'\pm1}\}, \\ 1 & \text{pokud } a^1_{j,k} = \max\{a^1_{2j'\pm1,2k'\pm1}\}. \end{cases}$$

Dohromady tedy:

$$\delta^1_{j,k} = \begin{cases} 0 & \text{pokud } a^1_{j,k} \neq \max\{a^1_{2j'\pm1,2k'\pm1}\}, \\ \sum_l \delta^3_l w^3_{l,j',k'} \sigma\prime\left(z^1_{j,k}\right) & \text{pokud } a^1_{j,k} = \max\{a^1_{2j'\pm1,2k'\pm1}\}. \end{cases}$$

Konvoluční sítě - backpropagation

- 1. rovnice platí i pro konvoluční sítě $\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L)$
- 2. rovnice $\delta_{j,k}^1 = \begin{cases} 0 & \text{pokud } a_{j,k}^1 \neq \max\left\{a_{2j'\pm1,2k'\pm1}^1\right\}, \\ \sum_l \delta_l^3 w_{l,j',k'}^3 \sigma'\left(z_{j,k}^1\right) & \text{pokud } a_{j,k}^1 = \max\left\{a_{2j'\pm1,2k'\pm1}^1\right\}. \end{cases}$

Backpropagation - 4 rovnice algoritmu: důkaz (cvičení)

- **Důkaz 3. :** $\dfrac{\partial C}{\partial b_j^l} = \delta_j^l$
- Vyjdeme z definice chyby neuronu a použijeme derivaci složené funkce (rozvineme závislost na biasu)

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \frac{\partial C}{\partial b_k^l} \frac{\partial b_k^l}{\partial z_j^l} = \frac{\partial C}{\partial b_j^l} \frac{\partial b_j^l}{\partial z_j^l}, \text{protože } b_k^l \text{ závisí jen na } z_k^l, \text{ neboli } \frac{\partial b_k^l}{\partial z_j^l} = 0 \text{ když j} \neq \text{k}$$

dále z definice $z_j^l = \sum_{k=1}^{n_l} w_{kj}^l a_k^{l-1} + b_j^l \iff b_j^l = z_j^l - \sum_{k=1}^{n_l} w_{kj}^l a_k^{l-1}$, zderivováním získáme $\dfrac{\partial b_j^l}{z_j^l} = 1$

což dosadíme do předchozí rovnice: $$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_{k=1}^{n_{l+1}} \frac{\partial C}{\partial b_k^l} \frac{\partial b_k^l}{\partial z_j^l} = \frac{\partial C}{\partial b_j^l} \frac{\partial b_j^l}{\partial z_j^l} = \frac{\partial C}{\partial b_j^l}$$

Konvoluční sítě - backpropagation

- $\frac{\partial C}{\partial b_l^3} = \delta_l^3$, důkaz normálního BP zde platí

Konvoluční sítě - backpropagation

- $\frac{\partial C}{\partial b_l^3} = \delta_l^3$, důkaz normálního BP zde platí
- $\frac{\partial C}{\partial b^1} = \sum_j \sum_k \frac{\partial C}{\partial z_{j,k}^1} \frac{\partial z_{j,k}^1}{\partial b^1} = \sum_j \sum_k \delta_{j,k}^1 \frac{\partial z_{j,k}^1}{\partial b^1} = \sum_j \sum_k \delta_{j,k}^1,$

protože $z_{j,k}^1 = b^1 + \sum_j \sum_k w_{l,m}^1 a_{j+l,k+m}^0$

Konvoluční sítě - backpropagation

- 1. rovnice platí i pro konvoluční sítě $\delta_j^L = \frac{\partial C}{\partial a_j^L}\sigma'(z_j^L)$
- 2. rovnice $\delta_{j,k}^1 = \begin{cases} 0 & \text{pokud } a_{j,k}^1 \neq \max\left\{a_{2j'\pm1,2k'\pm1}^1\right\}, \\ \sum_l \delta_l^3 w_{l,j',k'}^3 \sigma'\left(z_{j,k}^1\right) & \text{pokud } a_{j,k}^1 = \max\left\{a_{2j'\pm1,2k'\pm1}^1\right\}. \end{cases}$
- 3. rovnice
  - $\frac{\partial C}{\partial b_l^3} = \delta_l^3$
  - $\frac{\partial C}{\partial b^1} = \sum_j \sum_k \delta_{j,k}^1$

Backpropagation - 4 rovnice algoritmu: důkaz (cvičení)

- **Důkaz 4. :** $\dfrac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j$

- Podobné úvahy jako předtím, použijeme derivaci složené funkce (rozvineme **z**) a uvědomíme si, že z^l_j závisí pouze na w^l_{jk}, takže se všechny členy sumy kromě i=j vynulují

$$\frac{\partial C}{\partial w^l_{jk}} = \sum_{i=1}^{n_l} \frac{\partial C}{\partial z^l_i} \frac{\partial z^l_i}{\partial w^l_{jk}} = \frac{\partial C}{\partial z^l_j} \frac{\partial z^l_j}{\partial w^l_{jk}} = \delta^l_j \frac{\partial z^l_j}{\partial w^l_{jk}} = \delta^l_j a^{l-1}_k, \text{ protože z definice } z^l_j = \sum_{k=1}^{n_l} w^l_{jk} a^{l-1}_k \odot + b^l_j,$$

$$\text{a tedy } \frac{\partial z^l_j}{\partial w^l_{jk}} = a^{l-1}_k$$

Konvoluční sítě - backpropagation

- $\dfrac{\partial C}{\partial w^3_{l,j,k}} = a^2_{j,k}\delta^3_l$, důkaz normálního BP zde platí

Konvoluční sítě - backpropagation

- $\dfrac{\partial C}{\partial w^3_{l,j,k}} = a^2_{j,k}\delta^3_l$, důkaz normálního BP zde platí

- $\dfrac{\partial C}{\partial w^1_{l,m}} = \sum_j \sum_k \dfrac{\partial C}{\partial z^1_{j,k}} \dfrac{\partial z^1_{j,k}}{\partial w^1_{l,m}} = \sum_j \sum_k \delta^1_{j,k} \dfrac{\partial z^1_{j,k}}{\partial w^1_{l,m}} = \sum_j \sum_k \delta^1_{j,k} a^0_{j+l,k+m},$

protože $z^1_{j,k} = b^1 + \sum_l \sum_m w^1_{l,m} a^0_{j+l,k+m}$

Konvoluční sítě - backpropagation

- 1. rovnice platí i pro konvoluční sítě $\delta_j^L = \frac{\partial C}{\partial a_j^L}\sigma'(z_j^L)$
- 2. rovnice $\delta_{j,k}^1 = \begin{cases} 0 & \text{pokud } a_{j,k}^1 \neq \max\left\{a_{2j'\pm1,2k'\pm1}^1\right\}, \\ \sum_l \delta_l^3 w_{l,j',k'}^3 \sigma\prime\left(z_{j,k}^1\right) & \text{pokud } a_{j,k}^1 = \max\left\{a_{2j'\pm1,2k'\pm1}^1\right\}. \end{cases}$
- 3. rovnice
  - $\frac{\partial C}{\partial b_l^3} = \delta_l^3$
  - $\frac{\partial C}{\partial b^1} = \sum_j \sum_k \delta_{j,k}^1$
- 4. rovnice
  - $\frac{\partial C}{\partial w_{l,j,k}^3} = a_{j,k}^2 \delta_l^3$
  - $\frac{\partial C}{\partial w_{l,m}^1} = \sum_j \sum_k \delta_{j,k}^1 a_{j+l,k+m}^0,$