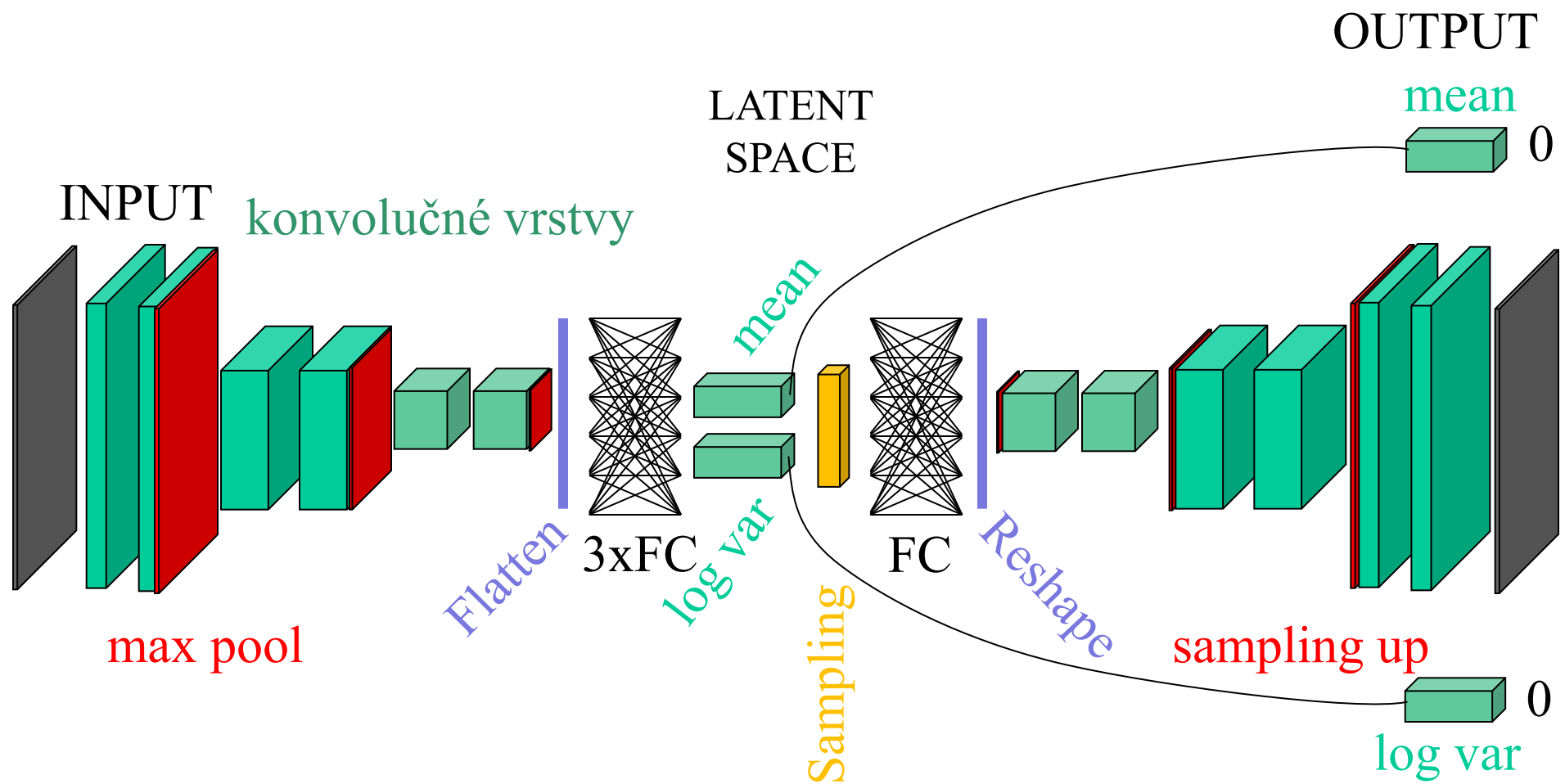
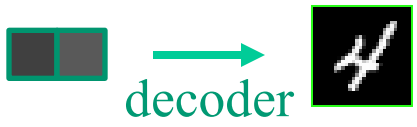
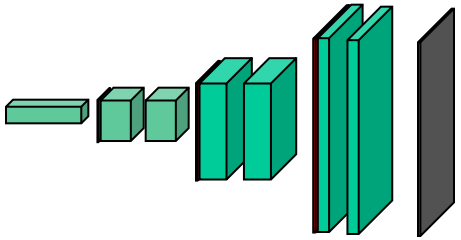
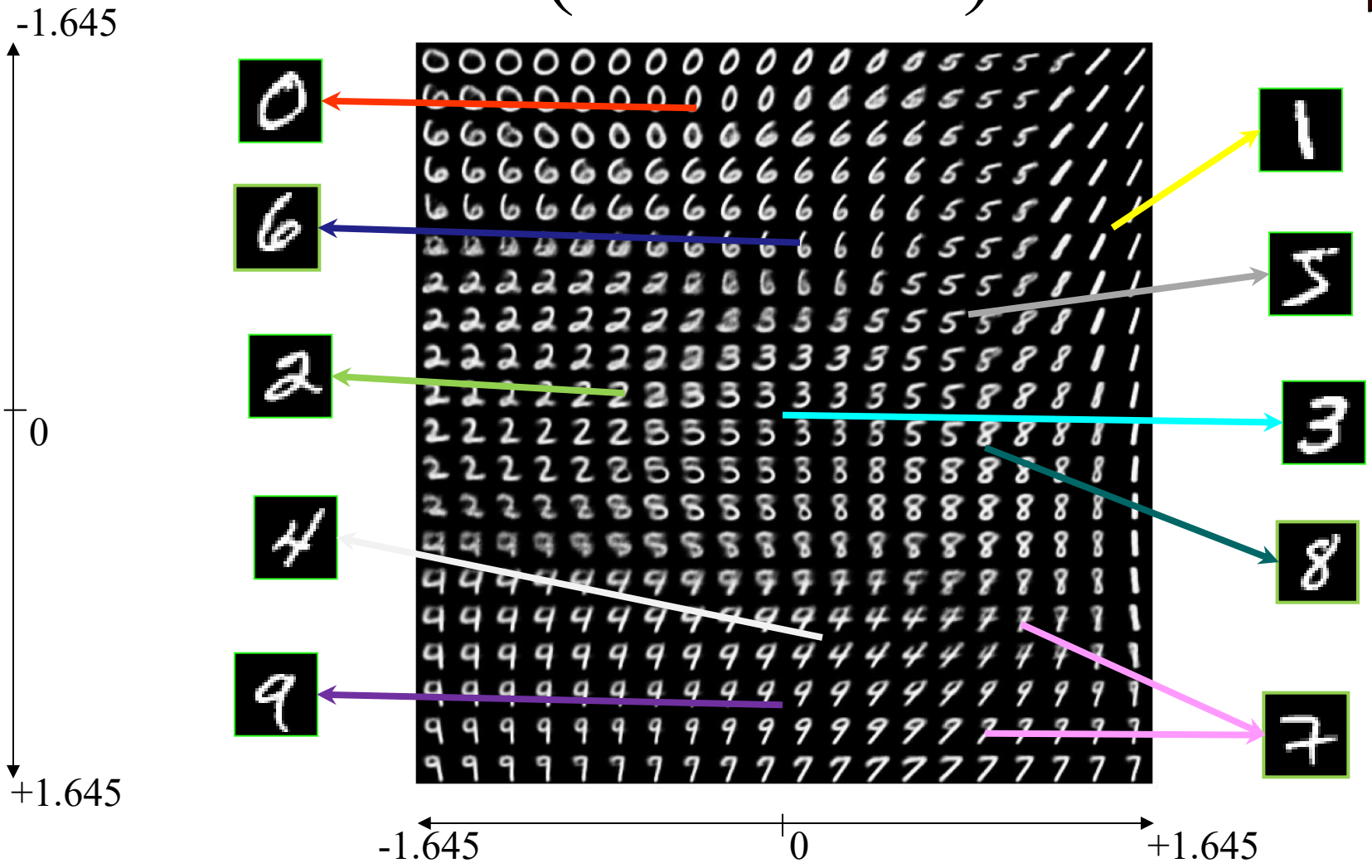
Variational autoencoder



Decoder (Generator)

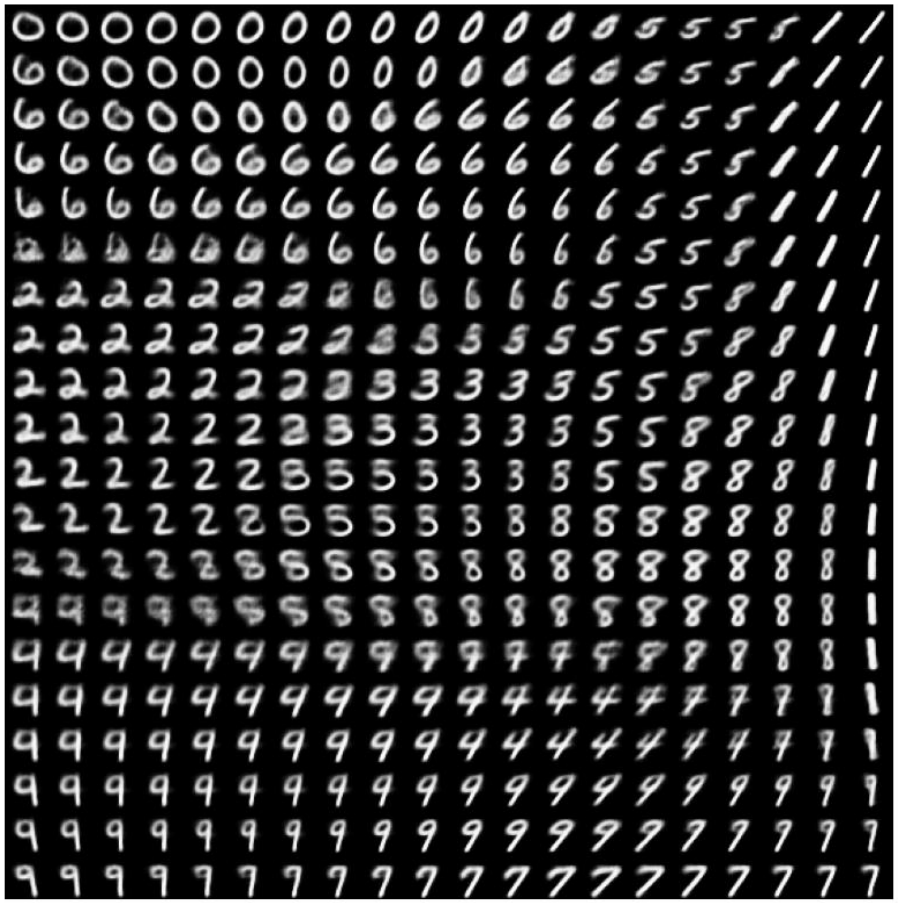






CNN

vnútro hyperkocky



Podobnosť:euklidovská vzdialenosť

Euklidovská podobnosť

vektor $(u_1, u_2, \ldots, u_N)$ má **veľkosť**

$$|u| = \sqrt{(u_1^2 + u_2^2 + \cdots + u_N^2)}$$

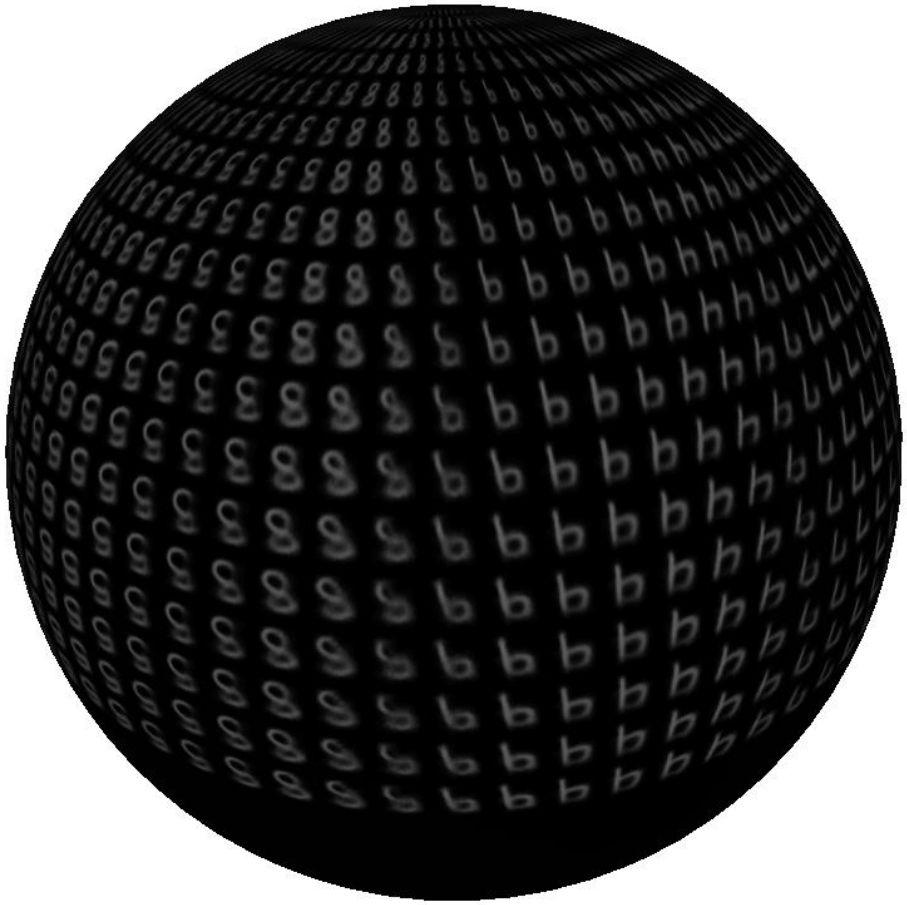
(odmocnina súčtu druhých mocnín koordinátov)

dva vektory $(u_1, u_2, \ldots, u_N)$ a $(v_1, v_2, \ldots, v_N)$ majú

podobnosť $|u - v|$

Transformery

Povrch hypergule



podobne
Zemi aj tu
sú pohoria a
priekopy

Podobnosť: cosínusová podobnosť

Kosínusová podobnosť

vektor $(u_1, u_2, \ldots, u_N)$ má **veľkosť**

$$|u| = \sqrt{(u_1^2 + u_2^2 + \cdots + u_N^2)}$$

dva vektory $(u_1, u_2, \ldots, u_N)$ a $(v_1, v_2, \ldots, v_N)$ zvierajú **uhol** ϕ

$$\cos\phi = \frac{u_1 v_1 + u_2 v_2 + \cdots + u_N v_N}{|u||v|}$$

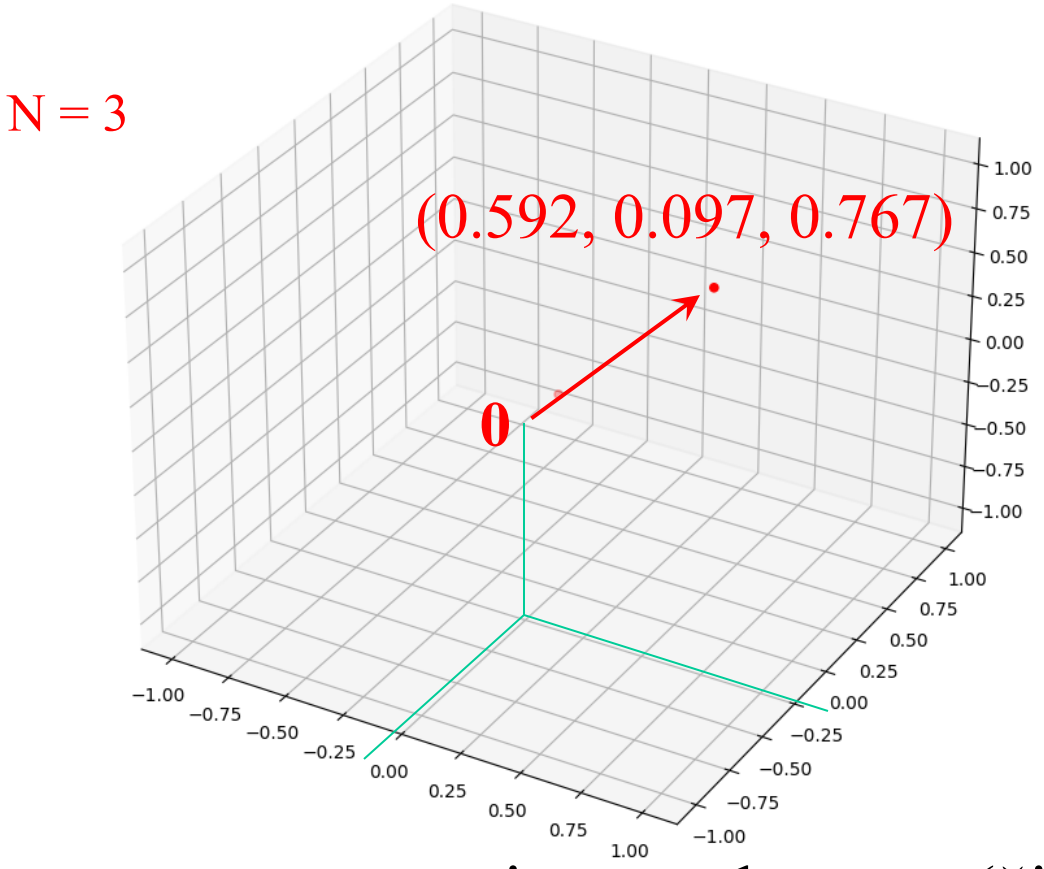
(kosínus uhla je podiel **skalárneho súčinu** koordinátov a súčinu veľkostí vektorov)

cos ϕ = 1

cos ϕ = 0

cos ϕ = -1

Embedding (Vnorenie)

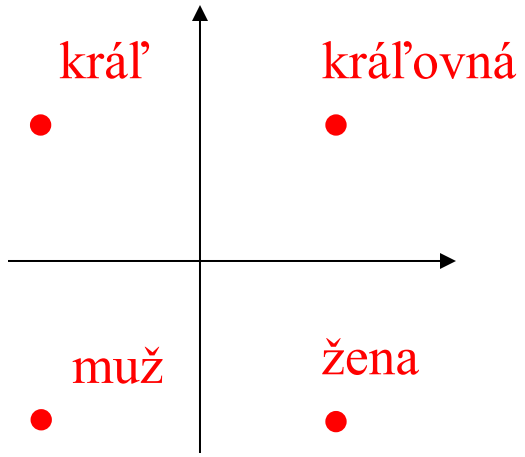


Význam reprezentujeme vektorom (či bodom) v mnohorozmernom priestore príznakov

Reprezentácia slova vektorom

slovník

kráľ	(-1,1)
kráľovná	(1,1)
muž	(-1,-1)
žena	(1,-1)



v slovníku máme 4 slová a každému priraďujeme vektor dimenzie 2
máme dva príznaky: pohlavie a vládnutie

Aritmetika významov

Kto sa má k žene tak, ako kráľ ku kráľovnej ?

Riešime rovnicu:

$$
\begin{aligned}
x - \text{žena} &= \text{kráľ} - \text{kráľovná} \\
x - (1,-1) &= (-1,1) - (1,1) \\
x - (1,-1) &= (-2,0) \\
x &= (-2,0) + (1,-1) \\
x &= (-1,-1) \\
x &= \text{muž}
\end{aligned}
$$

slovník

kráľ	(-1,1)
kráľovná	(1,1)
muž	(-1,-1)
žena	(1,-1)

Aritmetika významov

Kto sa má k žene tak, ako kráľ ku kráľovnej ?

Riešime rovnicu:

$$
\begin{aligned}
x - \text{žena} &= \text{kráľ} - \text{kráľovná} \\
x - (1,-1) &= (-1,1) - (1,1) \\
x - (1,-1) &= (-2,0) \\
x &= (-2,0) + (1,-1) \\
x &= (-1,-1) \\
x &= \text{muž}
\end{aligned}
$$

Embedding (Vnorenie)

Wipeout (Vynorenie)

slovník

kráľ	(-1,1)
kráľovná	(1,1)
muž	(-1,-1)
žena	(1,-1)

Ako nájsť správne hodnoty jednotlivých vektorov automaticky ?

Zoberieme korpus:

V kráľovskom dvore si kráľ s kráľovnou vychutnávali plesy a hudobné predstavenia. Muž a žena, ktorí boli oddaní manželským záväzkom, sa stali kráľom a kráľovnou v ich vlastnej domácnosti. Na námestí sa stretol muž so ženou a spolu snívali o tom, že by mohli byť kráľom a kráľovnou svojej vlastnej krajiny. Kráľ so svojou kráľovnou prechádzali rozkvitnutým záhradným labyrintom, objímajúc sa pod každým rozkvitnutým stromom. Muž sa rozhodol, že chce byť kráľom svojho osudu, a žena ho podporovala, aby sa stala jeho vernou kráľovnou. Na večernom bankete si kráľ so svojou kráľovnou vymieňali nežné pohľady a tajné úsmevy. V dedine si obyvatelia vybrali muža a ženu, aby sa stali kráľom a kráľovnou na miestnom jarmoku. Kráľovná pozvala mladú ženu do svojho paláca a pomohla jej pripraviť sa na večernú hostinu s kráľom. Muž a jeho milovaná žena spolu snívali o kráľovských príbehoch, ktoré by mohli prežiť. Na slávnostnom obrade bola mladá žena korunovaná za kráľovnú a jej muž sa stal kráľom. ...

a spočítame ako často sa jedno slovo vyskytuje v kontexte druhého (t.j. v texte maximálne tri slová pred ním alebo za ním)

Ako nájsť správne hodnoty jednotlivých vektorov automaticky ?

Dostaneme tabuľku početností
a v ideálnom prípade dostaneme:

	kráľ	kráľovná	muž	žena
kráľ	44	22	22	0
kráľovná	22	44	0	22
muž	22	0	44	22
žena	0	22	22	44

Ako nájsť správne hodnoty jednotlivých vektorov automaticky ?

Dostaneme tabuľku početností, ktorú môžeme znormalizovať a v ideálnom prípade dostaneme:

	kráľ	kráľovná	muž	žena
kráľ	1	0.5	0.5	0
kráľovná	0.5	1	0	0.5
muž	0.5	0	1	0.5
žena	0	0.5	0.5	1

Ako nájsť správne hodnoty jednotlivých vektorov automaticky ?

Dostaneme tabuľku početností, ktorú môžeme znormalizovať a v ideálnom prípade dostaneme:

	kráľ	kráľovná	muž	žena
kráľ	1	0.5	0.5	0
kráľovná	0.5	1	0	0.5
muž	0.5	0	1	0.5
žena	0	0.5	0.5	1

slovník

kráľ	(1,0.5,0.5,0)
kráľovná	(0.5,1,0,0.5)
muž	(0.5,0,1,0.5)
žena	(0,0.5,0.5,1)

a opäť tu platí:

$$\text{muž} - \text{žena} = \text{kráľ} - \text{kráľovná}$$

$$(0.5, 0, 1, 0.5) - (0, 0.5, 0.5, 1) = (1, 0.5, 0.5, 0) - (0.5, 1, 0, 0.5)$$

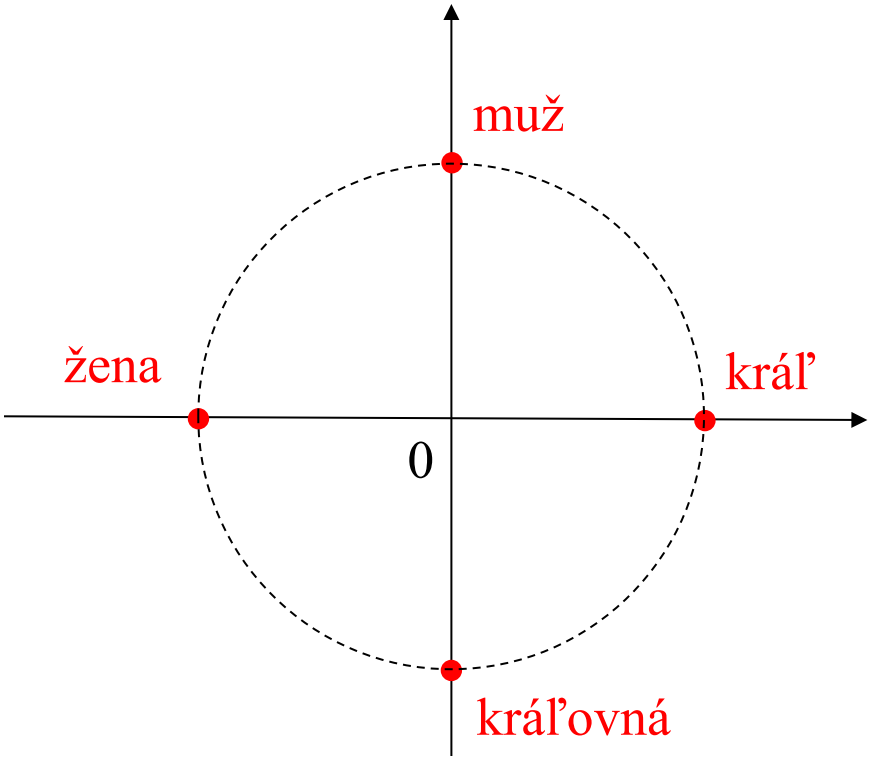
$$(0.5, -0.5, 0.5, -0.5) = (0.5, -0.5, 0.5, -0.5)$$

Je možné z automatického slovníka získať podobný tomu ručne urobenému?

Napodiv, odpoveď je kladná. Dokážeme to metódou lineárnej algebry zvanou PCA (Principal Component Analysis), keď jej dáme zredukovať dimenziu automatického slovníka (4) na 2

kráľ	(1,0.5,0.5,0)
kráľovná	(0.5,1,0,0.5)
muž	(0.5,0,1,0.5)
žena	(0,0.5,0.5,1)

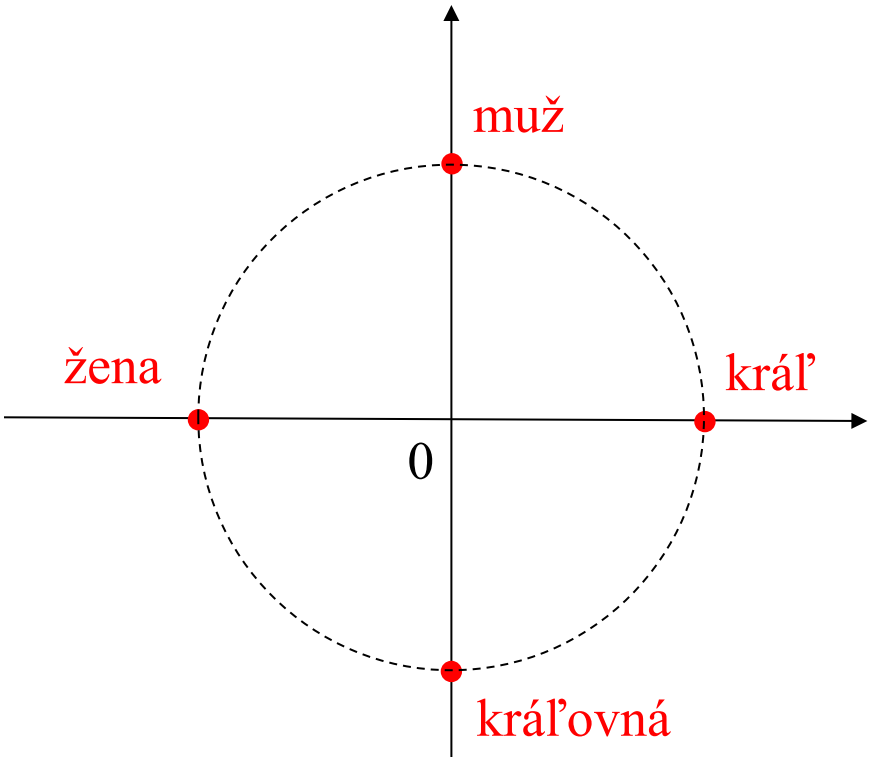
kráľ	(0.707, 0)
kráľovná	(0, -0.707)
muž	(0 ,0.707)
žena	(-0.707, 0)



Kosínusová podobnosť

muž je teraz podobnejší kráľovi (cos 90° = 0) viac než kráľovnej (cos 180° = – 1) a najpodobnejší sebe (cos 0° = 1)

(to je vďaka tomu, že PCA dáta vycentruje, t.j. priemer je v nule a v tomto prípade aj otočí o 45°)

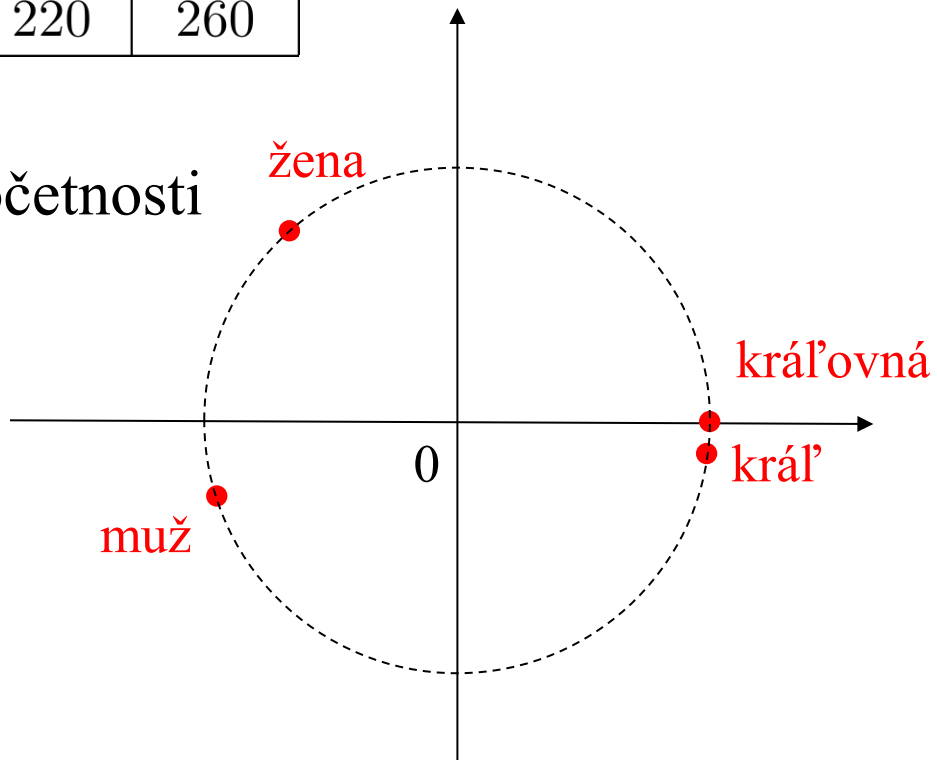


Čo by sa stalo keby početnosti v korpuse neboli optimálne?

	kráľ	kráľovná	muž	žena
kráľ	44	18	12	5
kráľovná	18	32	3	24
muž	12	3	440	220
žena	5	24	220	260

aplikujeme PCA priamo na početnosti a normalizujeme, dostaneme:

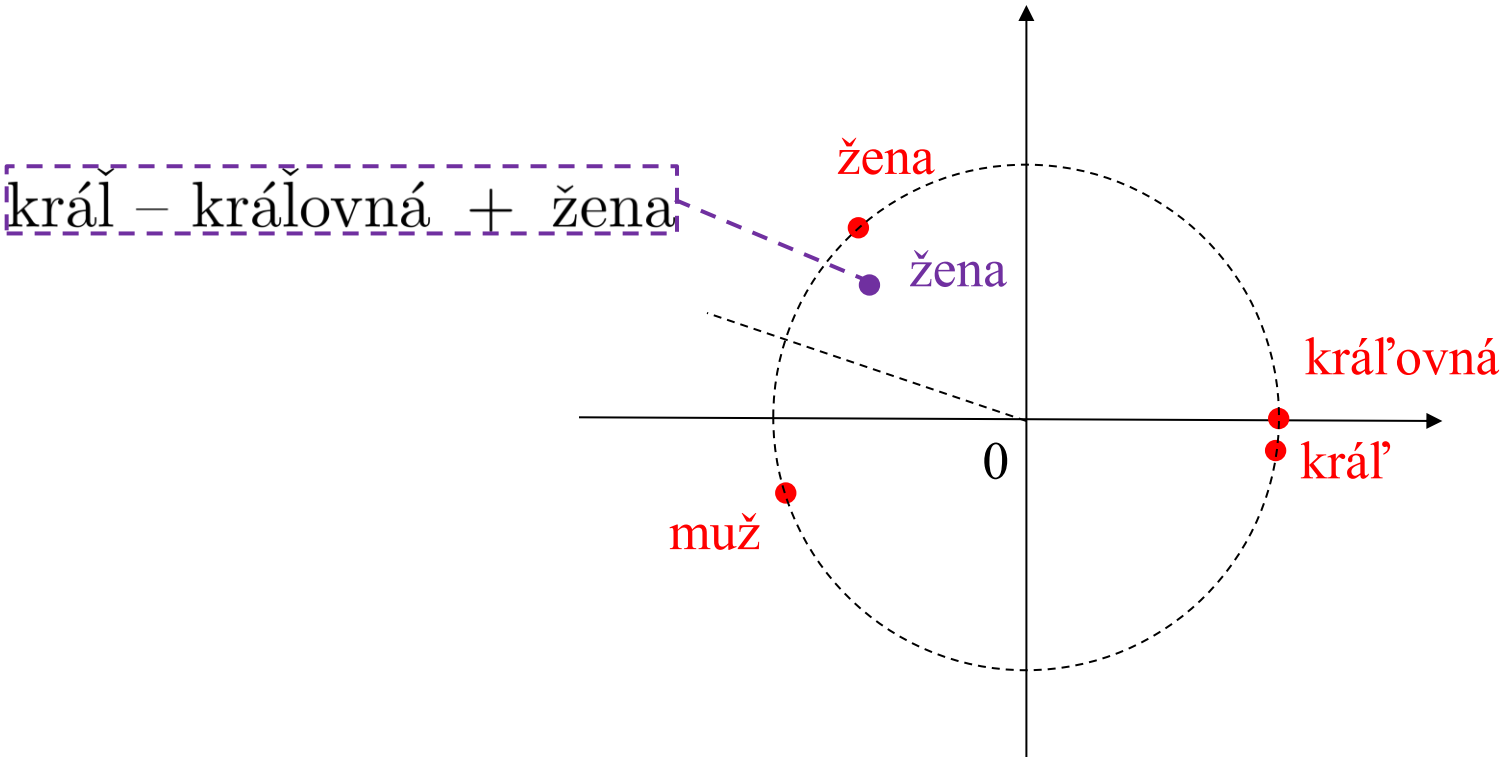
kráľ	(0.99, -0.14)
kráľovná	(1.00, 0.00)
muž	(-0.98, -0.2)
žena	(-0.78, 0.63)



Proporcie máme dobré, ale aritmetika už nefunguje

Kto sa má k žene tak, ako kráľ ku kráľovnej ?

Aritmetika odpovie: žena



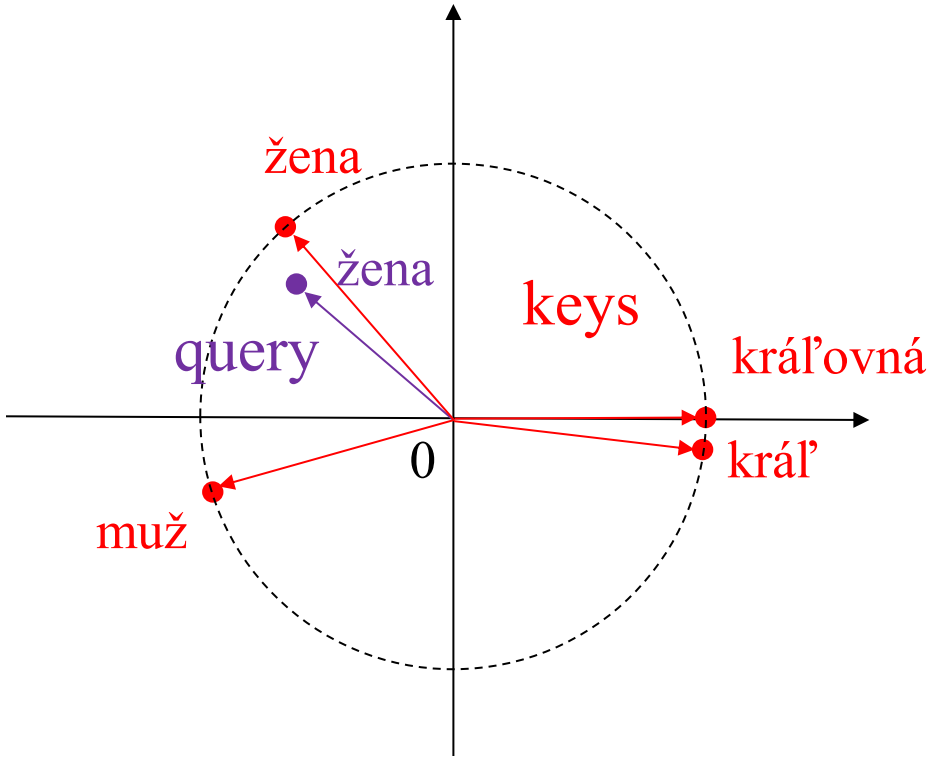
Wipeout (Vynorenie)

Výstupný embedding

Nepovie teda presne „žena“, len približne

Ako spočítame ku ktorému zo vzorov je výsledok najbližšie?

Hľadáme maximum skalárneho súčinu query so všetkými keys



Wipeout (Vynorenie)

Výstupný embedding

Máme keys $K = \begin{pmatrix} k_1 \\ k_2 \\ \dots \\ k_l \end{pmatrix}$ a query q

qK^T je vektor skalárnych súčinov q s k_i

Hľadáme teda $\arg\max_i qK^T$

Pokiaľ chceme pravdepodobnosti pre každú kategóriu: $\mathrm{softmax}(qK^T)$

Implementácia Wipeout-u

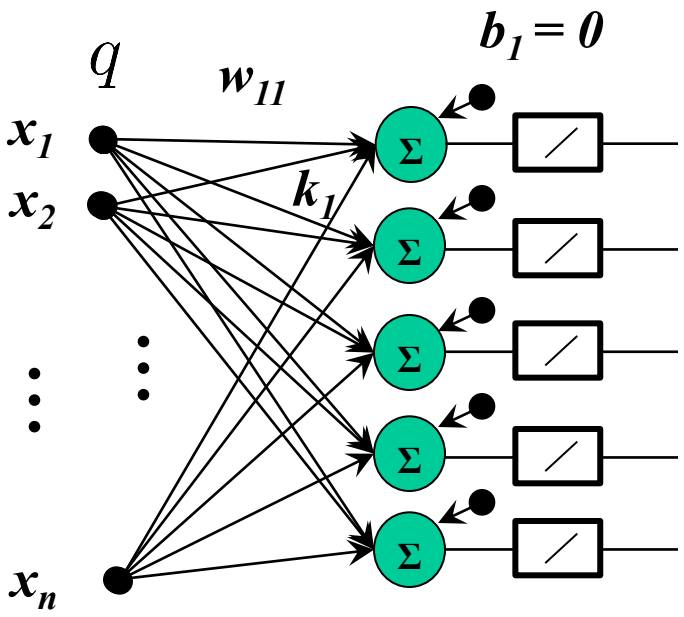
Počet vstupov x_i
= dimenzia priestoru

Počet neurónov
= počet kľúčov

Kľúče sú váhy
$k_j = w_{1j}, w_{2j}, \ldots$

Biasy sú nulové

Aktivácia lineárna



Lineárna projekcia

qK^T

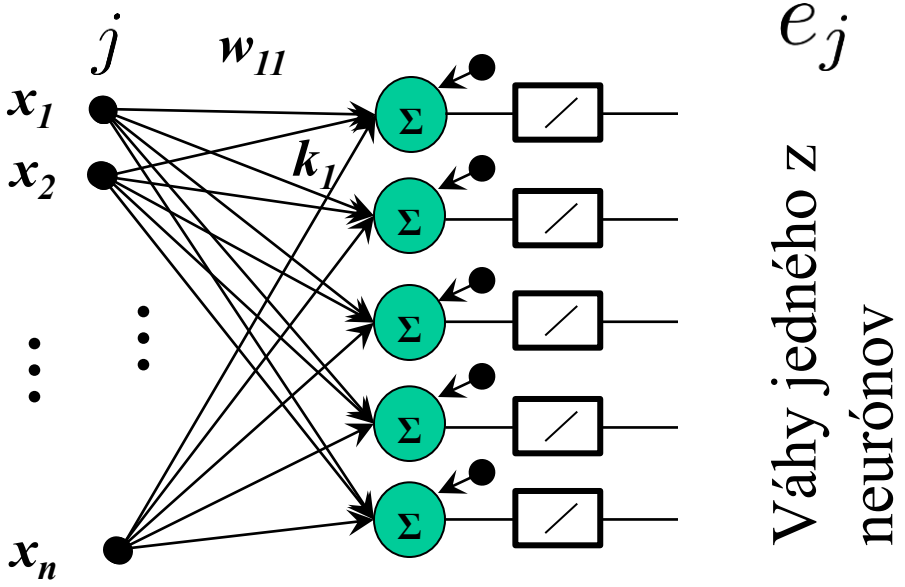
Skalárne súčiny query s kľúčmi

Implementácia Embedding-u

Vstupom je index j
= token id

Počet neurónov
= počet tokenov

Embedingy sú váhy
$e_j = w_{1j}, w_{2j}, \ldots$

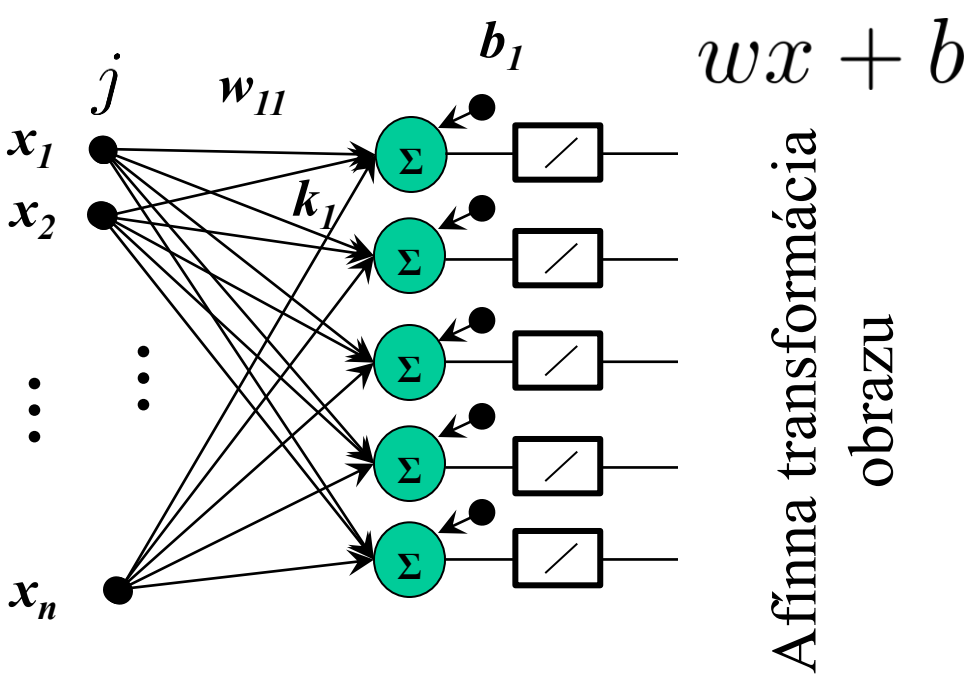


Text

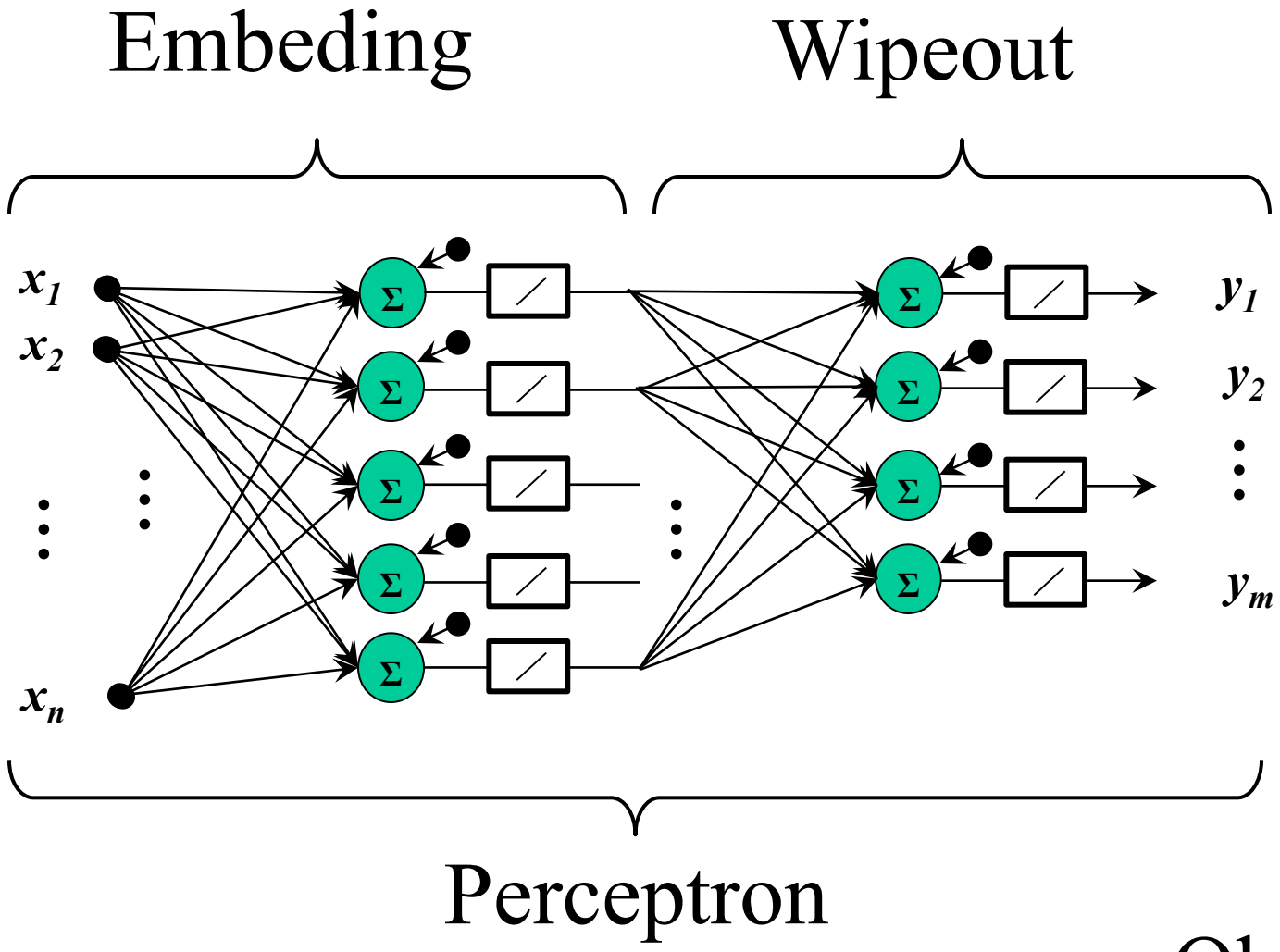
Implementácia Embedding-u

Vstupom sú farby pixelov
x = flatten(Patch)
Patch C×H×W
3×8×8 .. 3×32×32

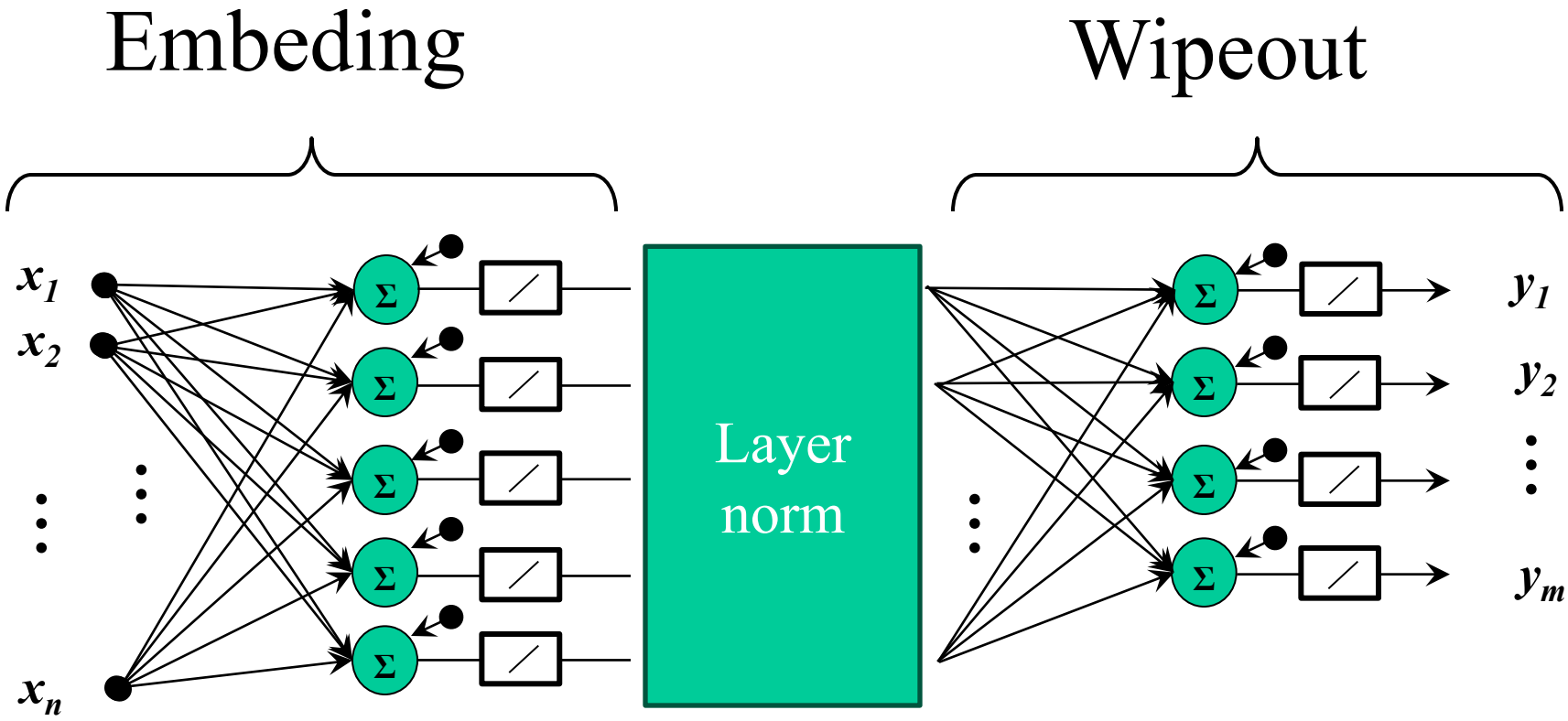
Počet neurónov je
dimenziou priestoru
embeddingov
384 .. 12288



Obraz



Obraz



Obraz

Layer Normalization

$$\text{LayerNorm}(x) = \gamma \frac{x - \mu}{\sqrt{\frac{1}{d}\sum_{i=1}^{d}(x_i - \mu)^2 + \epsilon}} + \beta$$

$$\mu = \frac{1}{d}\sum_{i=1}^{d} x_i$$

Pre jeden vstup je x výstupom z nejakej vrstvy
Z týchto výstupov odčítame priemer
Zrátame smerodajnú odchýlku a vydelíme ňou všetky príznaky
Pritom sa pre každý príznak zvlášť učíme škálu γ a biás β
(počet trénovateľných parametrov je 2 x dimenzia)
LN zabezpečuje, že príznaky budú mať zhruba rovnakú veľkosť

RMS Norm

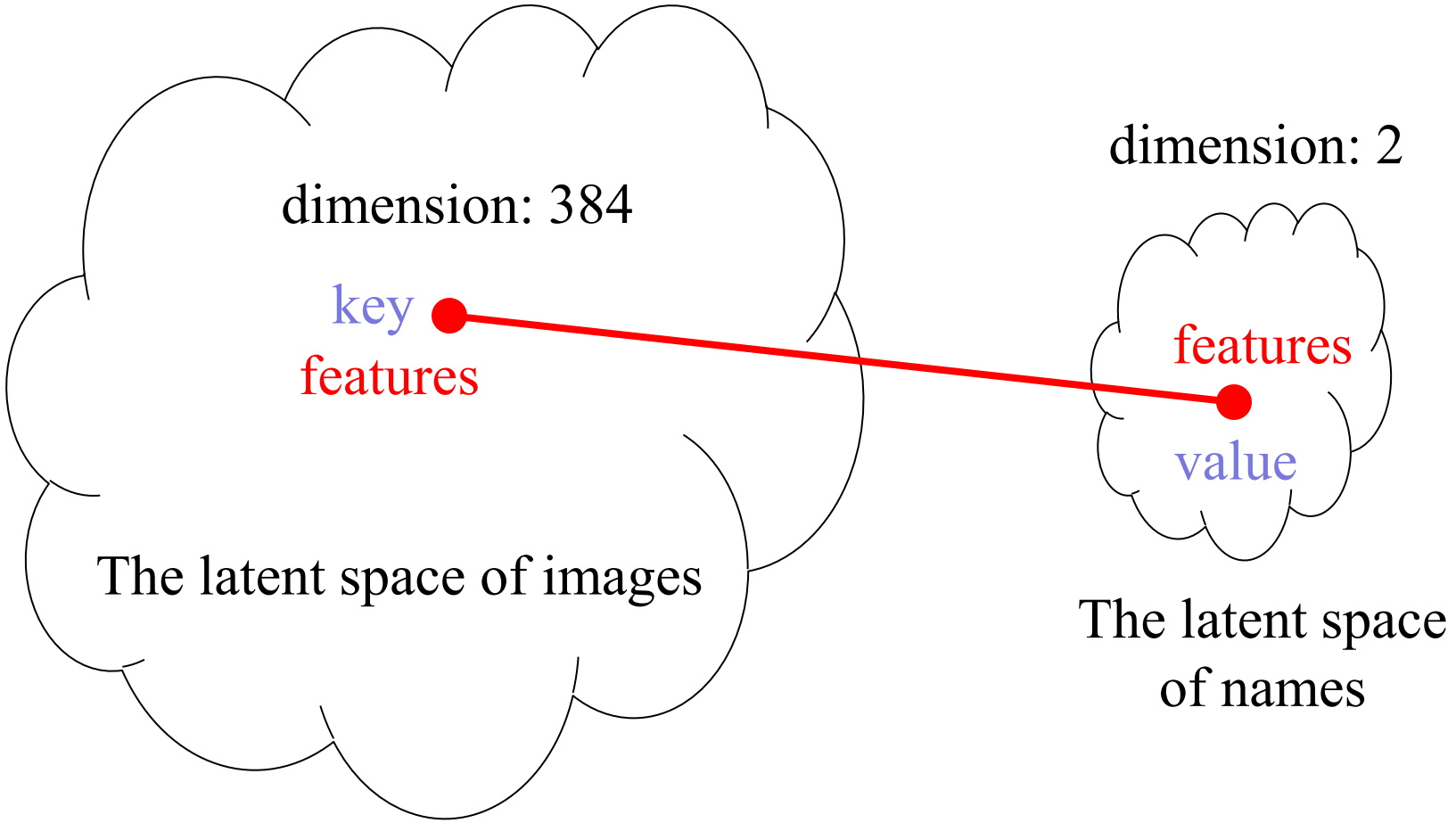
Root Mean Square Normalization

$$\text{RMSNorm}(x) = \gamma \frac{x}{\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2 + \epsilon}}$$

Pre jeden vstup je x výstupom z nejakej vrstvy
Zrátame smerodajnú odchýlku a vydelíme ňou všetky príznaky
Pritom sa pre každý príznak zvlášť učíme škálu β
(počet trénovateľných parametrov je rovný dimenzii)
RMS Norm zabezpečuje, že príznaky budú mať zhruba rovnakú veľkosť a budú vycentrované (nemáme biásy)
RMS Norm je rýchlejšia než LN

Associating

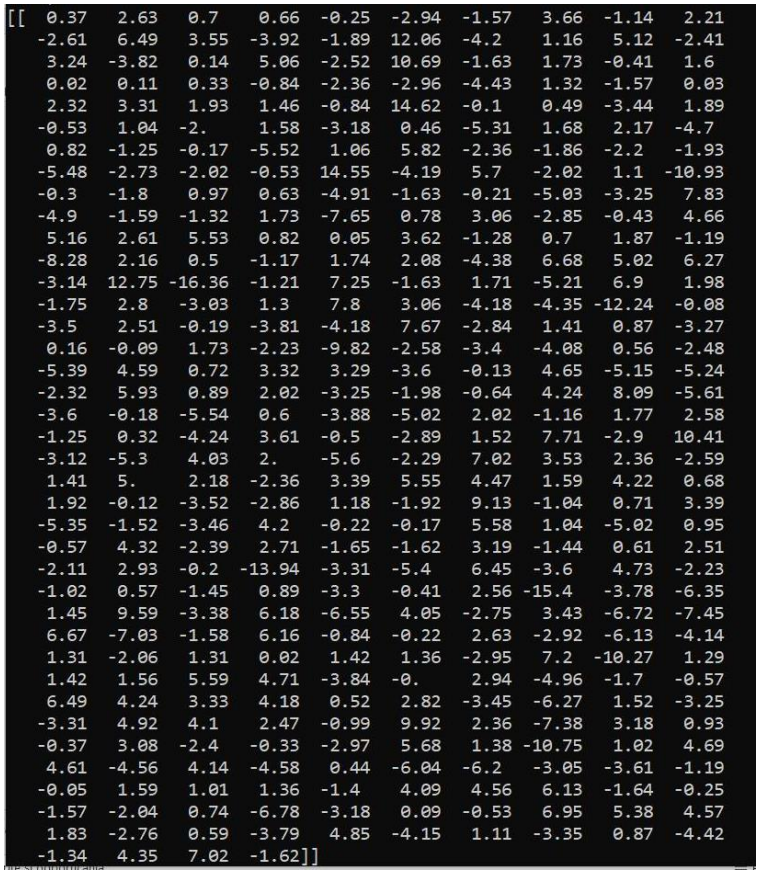
unlike the space of images, the latent space of the image features is continuous and fluent



we collect a list of the key-value pairs

Associating via Attention

1x384



```
[[ 0.37   2.63   0.7    0.66  -0.25  -2.94  -1.57   3.66  -1.14   2.21
  -2.61   6.49   3.55  -3.92  -1.89  12.06  -4.2    1.16   5.12  -2.41
   3.24  -3.82   0.14   5.06  -2.52  10.69  -1.63   1.73  -0.41   1.6
   0.02   0.11   0.33  -0.84  -2.36  -2.96  -4.43   1.32  -1.57   0.03
   2.32   3.31   1.93   1.46  -0.84  14.62  -0.1    0.49  -3.44   1.89
  -0.53   1.04  -2.     1.58  -3.18   0.46  -5.31   1.68   2.17  -4.7
   0.82  -1.25  -0.17  -5.52   1.06   5.82  -2.36  -1.86  -2.2   -1.93
  -5.48  -2.73  -2.02  -0.53  14.55  -4.19   5.7   -2.02   1.1  -10.93
  -0.3   -1.8    0.97   0.63  -4.91  -1.63  -0.21  -5.03  -3.25   7.83
  -4.9   -1.59  -1.32   1.73  -7.65   0.78   3.06  -2.85  -0.43   4.66
   5.16   2.61   5.53   0.82   0.05   3.62  -1.28   0.7    1.87  -1.19
  -8.28   2.16   0.5   -1.17   1.74   2.08  -4.38   6.68   5.02   6.27
  -3.14  12.75 -16.36  -1.21   7.25  -1.63   1.71  -5.21   6.9    1.98
  -1.75   2.8   -3.03   1.3    7.8    3.06  -4.18  -4.35 -12.24  -0.08
  -3.5    2.51  -0.19  -3.81  -4.18   7.67  -2.84   1.41   0.87  -3.27
   0.16  -0.09   1.73  -2.23  -9.82  -2.58  -3.4   -4.08   0.56  -2.48
  -5.39   4.59   0.72   3.32   3.29  -3.6   -0.13   4.65  -5.15  -5.24
  -2.32   5.93   0.89   2.02  -3.25  -1.98  -0.64   4.24   8.09  -5.61
  -3.6   -0.18  -5.54   0.6   -3.88  -5.02   2.02  -1.16   1.77   2.58
  -1.25   0.32  -4.24   3.61  -0.5   -2.89   1.52   7.71  -2.9   10.41
  -3.12  -5.3    4.03   2.    -5.6   -2.29   7.02   3.53   2.36  -2.59
   1.41   5.     2.18  -2.36   3.39   5.55   4.47   1.59   4.22   0.68
   1.92  -0.12  -3.52  -2.86   1.18  -1.92   9.13  -1.04   0.71   3.39
  -5.35  -1.52  -3.46   4.2   -0.22  -0.17   5.58   1.04  -5.02   0.95
  -0.57   4.32  -2.39   2.71  -1.65  -1.62   3.19  -1.44   0.61   2.51
  -2.11   2.93  -0.2  -13.94  -3.31  -5.4    6.45  -3.6    4.73  -2.23
  -1.02   0.57  -1.45   0.89  -3.3   -0.41   2.56 -15.4   -3.78  -6.35
   1.45   9.59  -3.38   6.18  -6.55   4.05  -2.75   3.43  -6.72  -7.45
   6.67  -7.03  -1.58   6.16  -0.84  -0.22   2.63  -2.92  -6.13  -4.14
   1.31  -2.06   1.31   0.02   1.42   1.36  -2.95   7.2  -10.27   1.29
   1.42   1.56   5.59   4.71  -3.84  -0.     2.94  -4.96  -1.7   -0.57
   6.49   4.24   3.33   4.18   0.52   2.82  -3.45  -6.27   1.52  -3.25
  -3.31   4.92   4.1    2.47  -0.99   9.92   2.36  -7.38   3.18   0.93
  -0.37   3.08  -2.4   -0.33  -2.97   5.68   1.38 -10.75   1.02   4.69
   4.61  -4.56   4.14  -4.58   0.44  -6.04  -6.2   -3.05  -3.61  -1.19
  -0.05   1.59   1.01   1.36  -1.4    4.09   4.56   6.13  -1.64  -0.25
  -1.57  -2.04   0.74  -6.78  -3.18   0.09  -0.53   6.95   5.38   4.57
   1.83  -2.76   0.59  -3.79   4.85  -4.15   1.11  -3.35   0.87  -4.42
  -1.34   4.35   7.02  -1.62]]
```

key

Somehow, we must use the collected key-value pairs to calculate a proper response when we get an unassociated query.

1x2

```
array([[ 0.64, -0.32]])
```

value

We roughly mix the query from the available keys and create the output as an analog mixture of the corresponding values.

Attention

Máme *l* kľúčov dimenzie
n a *l* hodnôt dimenzie *m*

$$K = \begin{pmatrix} k_1 \\ k_2 \\ \dots \\ k_l \end{pmatrix} \qquad V = \begin{pmatrix} v_1 \\ v_2 \\ \dots \\ v_l \end{pmatrix}$$

Pre vstupný dotaz *q*, snažíme sa namiešať z kľúčov *K* jeho nejakú projekciu do podpriestoru kľúčov:

$$proj_K(q) = cK \quad \text{kde} \quad c = softmax\left(\frac{qK^T}{d}\right)$$

Výstupom bude analogická miešanina zodpovedajúcich hodnôt *V*

$$A(q, K, V) = softmax\left(\frac{qK^T}{d}\right)V$$

Pre vhodný škáľovací faktor *d*
(pri menšom *d*, namiešavame viac z kľúčov podobných dotazu)

Attention

Máme *l* kľúčov dimenzie
n a *l* hodnôt dimenzie *m*

$$K = \begin{pmatrix} k_1 \\ k_2 \\ \dots \\ k_l \end{pmatrix} \qquad V = \begin{pmatrix} v_1 \\ v_2 \\ \dots \\ v_l \end{pmatrix}$$

Pre vstupný dotaz *q*, snažíme sa namiešať z kľúčov *K* jeho nejakú projekciu do podpriestoru kľúčov:

kosínusová podobnosť

softmax čokoľvek premení na vektor pravdepodobností

$$proj_K(q) = cK \quad \text{kde} \quad c = softmax\left(\frac{qK^T}{d}\right)$$

softmax vráti koeficienty miešaniny kľúčov

Výstupom bude analogická miešanina zodpovedajúcich hodnôt *V*

$$A(q, K, V) = softmax\left(\frac{qK^T}{d}\right)V$$

Pre vhodný škáľovací faktor *d* = odmocnina z dimenzie kľúčov
(pri menšom *d*, namiešavame viac z kľúčov podobných dotazu)

Attention

Pokiaľ je kľúčov menej ako je dimenzia priestoru, generujú v ňom podpriestor. Keď dotaz(query) q neleží v tomto podpriestore, attention dáva preň rovnakú odozvu ako pre jeho kolmý priemet do tohto podpriestoru.

$$A(q, K, V) = A(proj_K^{ort}(q), K, V)$$

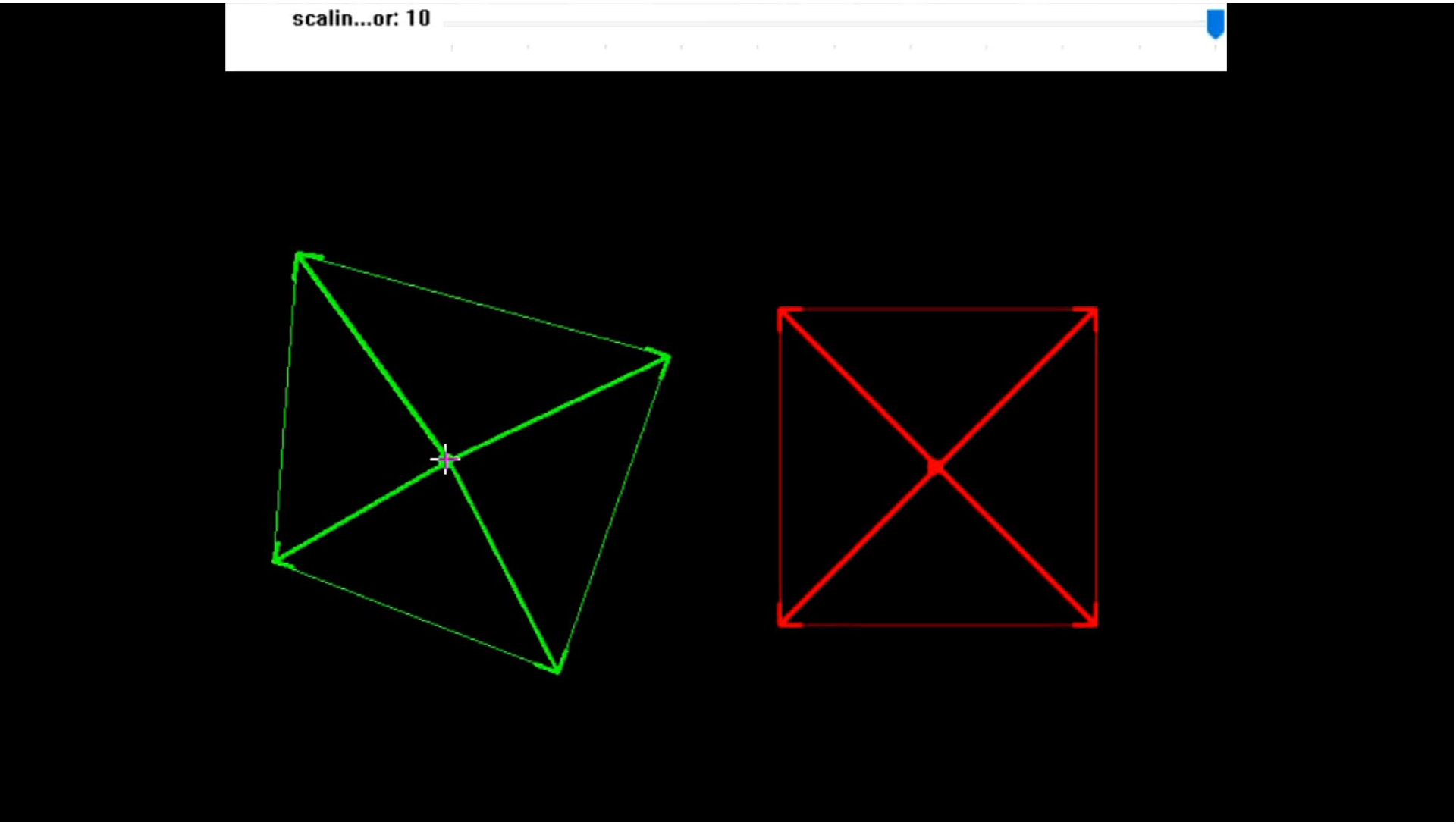
Vyplýva to z toho, že skalárny súčin vektora s vektorom generujúcim určitý podpriestor je rovnaký ako skalárny súčin jeho priemetu do tohto podpriestoru $proj_K^{ort}(q)k_i = qk_i$ pre $k_i \in K$

Týmto spôsobom attention generalizuje. Abstrahuje od určitých príznakov, napríklad od tých ktoré majú všetky kľúče rovnaké

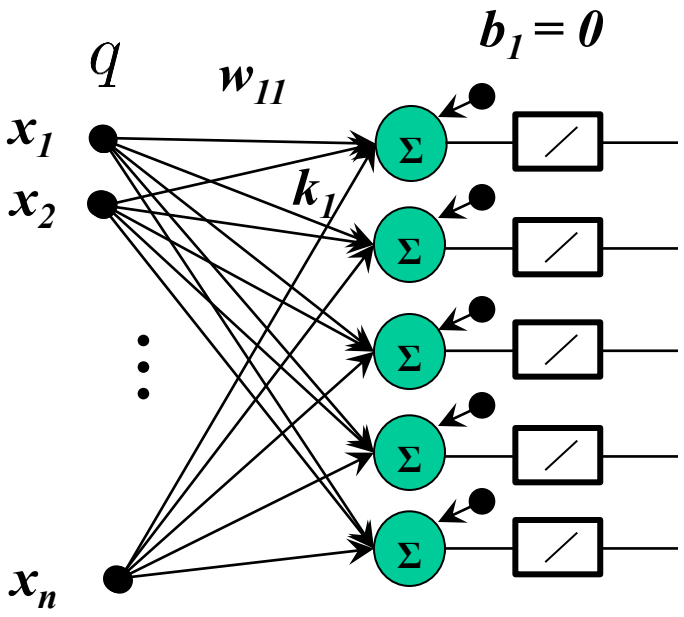
Scaling factor

$1/n^2$

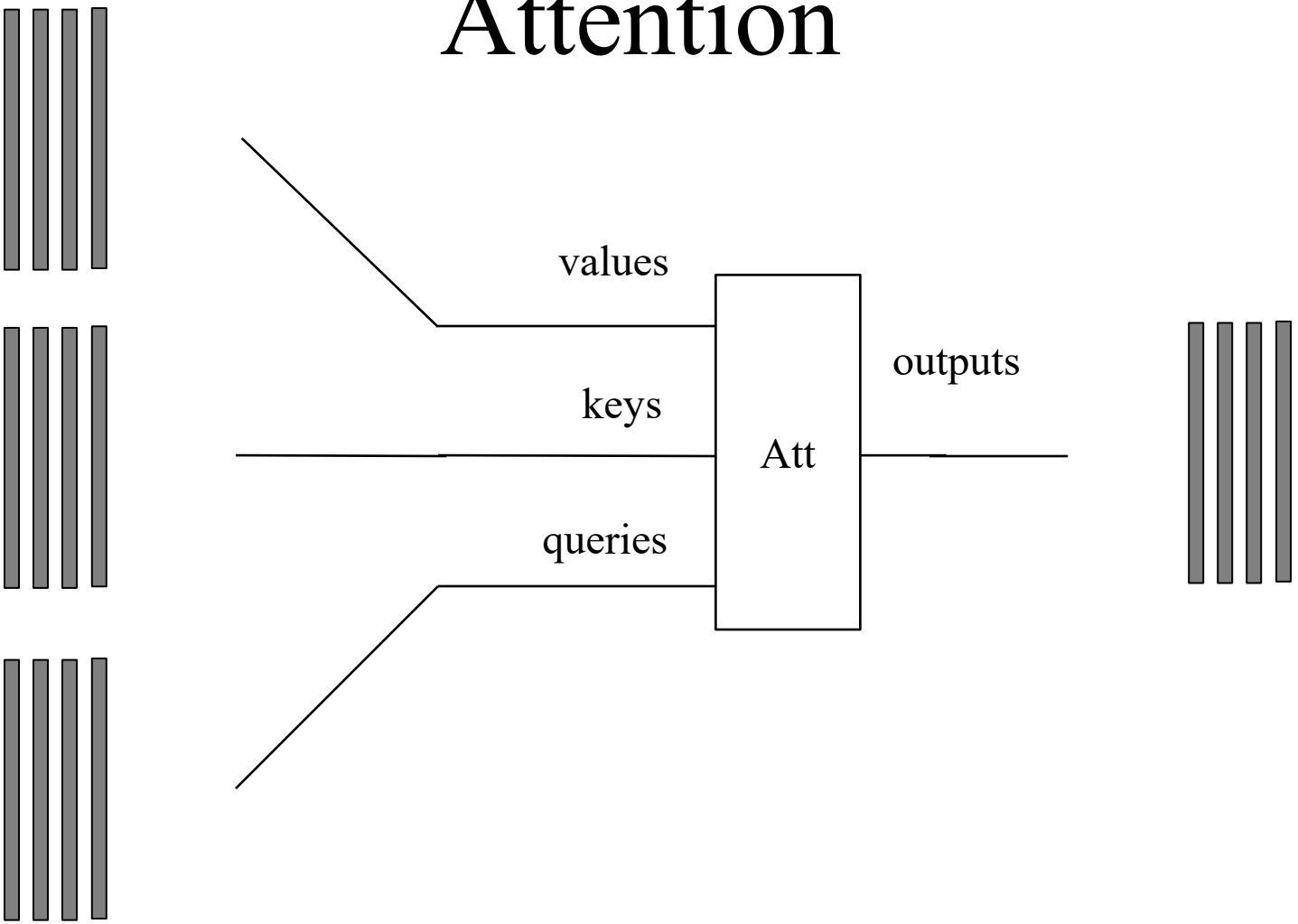
$\sqrt{(n)}$



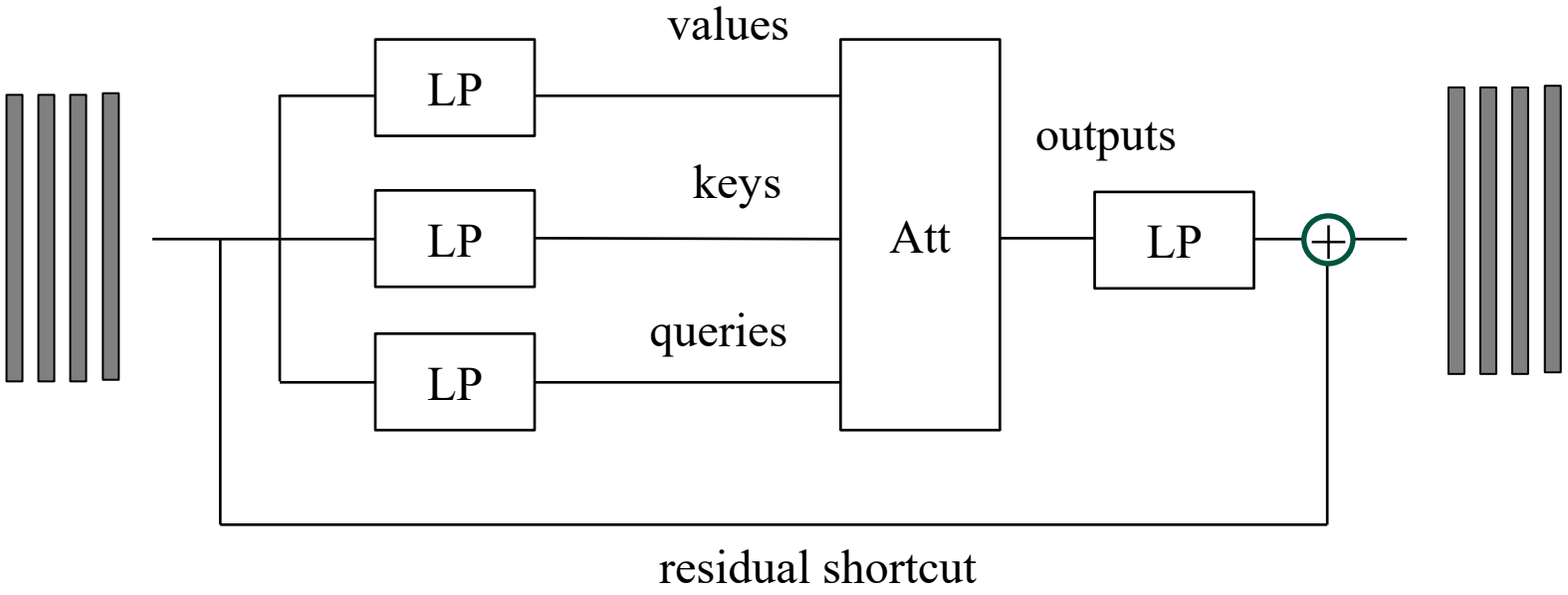
Linear projection



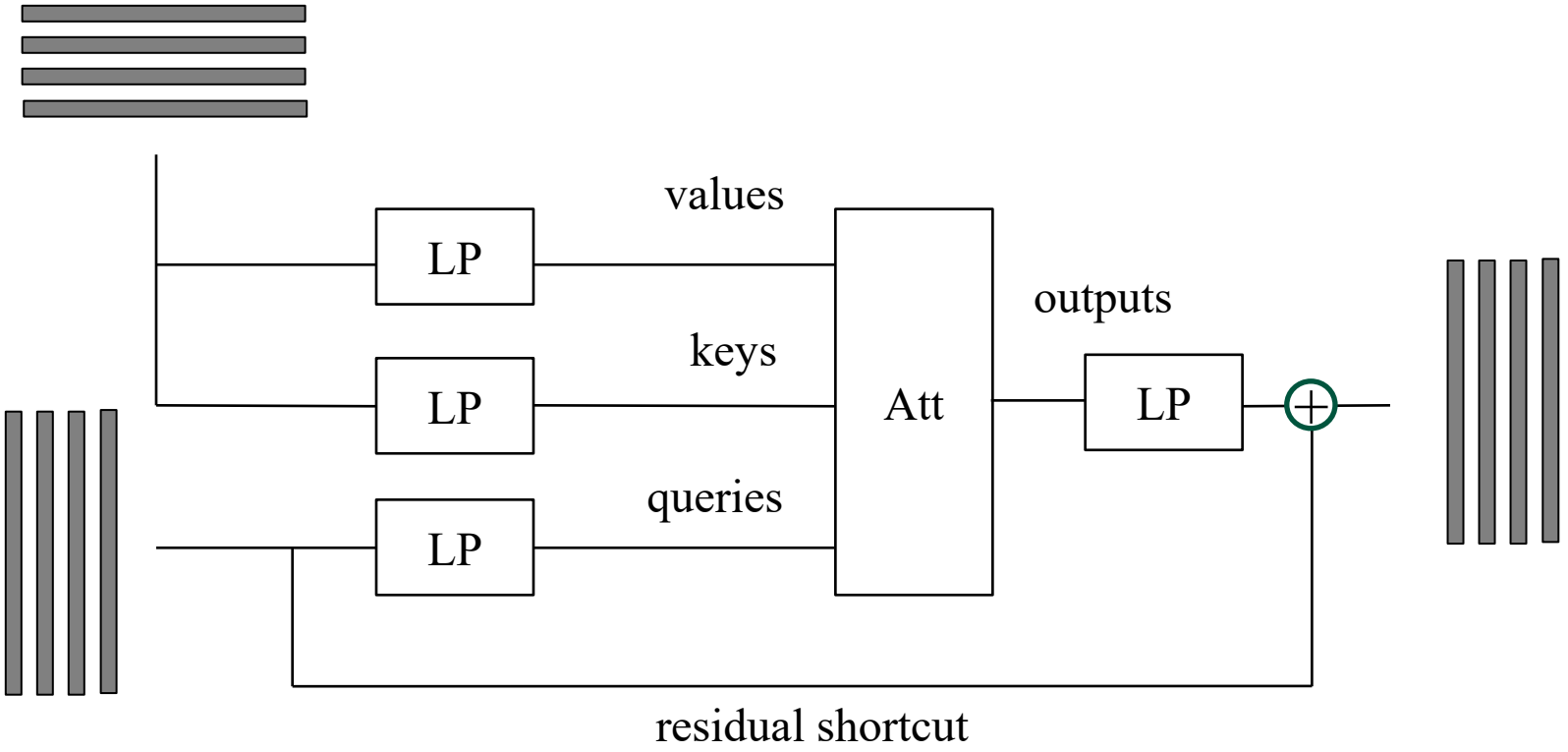
Attention



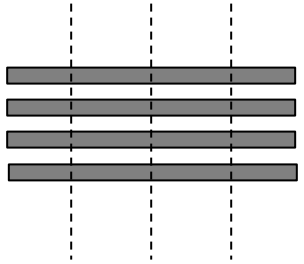
Residual Self-attention



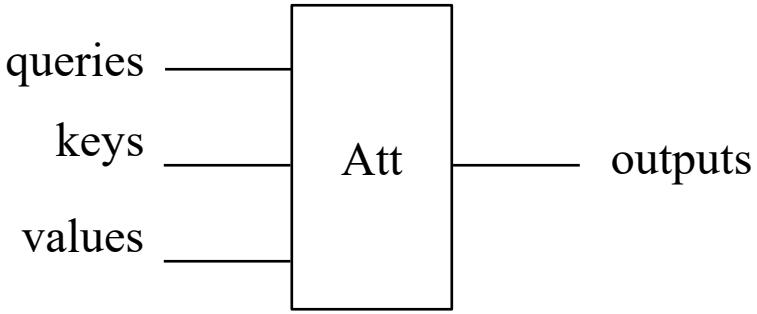
Residual Cross-attention



Multihead attention



Teplota



$$Att(Q, K, V) = \text{softmax}\left(\frac{QK^T}{t\sqrt{d}}\right)V$$

teplota

dimenzia