Natural Language Processing

Narayana Santhanam

EE 645

Mar 25, 2023

University of Hawai'i Mānoa

This module

- Latent Semantic Indexing
  - SVD
- Language models (Transformers)
  - Low rank projections
  - Transfer of information

Latent Semantic Indexing

Singular value decomposition

$$M = U\Sigma V^T$$

If M is $n \times p$,

U is $n \times n$

Σ is $n \times p$

V is $p \times p$

Singular value decomposition

$$M = U\Sigma V^T$$

If M is $n \times p$,

U is $n \times n$

Σ is $n \times p$

V is $p \times p$

U, V are both orthonormal

$U^T = U^{-1}$ and $V^T = V^{-1}$

Singular value decomposition

$$M = U\Sigma V^T$$

If M is $n \times p$,

U is $n \times n$

Σ is $n \times p$

V is $p \times p$

U, V are both orthonormal

$U^T = U^{-1}$ and $V^T = V^{-1}$

Σ is diagonoal

all diagonal entries ≥ 0

(called singular values)

Singular value decomposition

M is $n \times p$,

$$M = U \Sigma V^T$$

cols of U: basis for cols of M

$U = \begin{bmatrix} \mathrm{u}_1 & \cdots & \mathrm{u}_n \end{bmatrix}$, each $\mathrm{u}_i \in \mathbb{R}^n$

u_i all have length 1, mutually perpendicular

cols of V: basis for rows of M

$V = \begin{bmatrix} \mathrm{v}_1 & \cdots & \mathrm{v}_p \end{bmatrix}$, each $\mathrm{v}_i \in \mathbb{R}^n$

v_i all have length 1, mutually perpendicular

singular values: importance of basis vectors

$\sigma_1, \ldots, \sigma_{\min(n,p)}$

Multiplying out

M is $n \times p$,

$$M = \begin{bmatrix} \mathsf{u}_1 & \dots & \mathsf{u}_n \end{bmatrix} \operatorname{diag}\left(\sigma_1, \dots, \sigma_{\min(n,p)}\right) \begin{bmatrix} \mathsf{v}_1^T \\ \vdots \\ \mathsf{v}_p^T \end{bmatrix}$$

Instructive to multiply out:

$$M = \sigma_1 \mathsf{u}_1 \mathsf{v}_1^T + \dots + \sigma_{\min(n,p)} \mathsf{u}_{\min(n,p)} \mathsf{v}_{min(n,p)}^T$$

Multiplying out

M is $n \times p$,

$$M = \begin{bmatrix} \mathrm{u}_1 & \dots & \mathrm{u}_n \end{bmatrix} \mathrm{diag}\left(\sigma_1, \dots, \sigma_{\min(n,p)}\right) \begin{bmatrix} \mathrm{v}_1^T \\ \vdots \\ \mathrm{v}_p^T \end{bmatrix}$$

Instructive to multiply out:

$$M = \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \dots + \sigma_{\min(n,p)} \mathrm{u}_{\min(n,p)} \mathrm{v}_{min(n,p)}^T$$

Each of $\mathrm{u}_i \mathrm{v}_i^T$ is a rank-1 matrix

Multiplying out

M is $n \times p$,

$$M = \begin{bmatrix} \mathrm{u}_1 & \ldots & \mathrm{u}_n \end{bmatrix} \mathrm{diag}\left(\sigma_1, \ldots, \sigma_{\min(n,p)}\right) \begin{bmatrix} \mathrm{v}_1^T \\ \vdots \\ \mathrm{v}_p^T \end{bmatrix}$$

Instructive to multiply out:

$$M = \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_{\min(n,p)} \mathrm{u}_{\min(n,p)} \mathrm{v}_{min(n,p)}^T$$

Each of $\mathrm{u}_i \mathrm{v}_i^T$ is a rank-1 matrix
Number of non-zero singular values $=$ rank of matrix

Multiplying out

M is $n \times p$,

$$M = \begin{bmatrix} \mathrm{u}_1 & \dots & \mathrm{u}_n \end{bmatrix} \mathrm{diag}\left(\sigma_1, \dots, \sigma_{\min(n,p)}\right) \begin{bmatrix} \mathrm{v}_1^T \\ \vdots \\ \mathrm{v}_p^T \end{bmatrix}$$

Instructive to multiply out:

$$M = \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \dots + \sigma_{\min(n,p)} \mathrm{u}_{\min(n,p)} \mathrm{v}_{min(n,p)}^T$$

Each of $\mathrm{u}_i \mathrm{v}_i^T$ is a rank-1 matrix
Number of non-zero singular values = rank of matrix
In fact, general definition of rank:

Rank of a matrix

M is defined rank-r if it can be written as a sum of r rank-1 matrices and no fewer.

University of Hawaiʻi Mānoa

Latent Semantic Indexing

p documents, total of n words in the documents

M is the $n \times p$ term-document matrix

Different ways to come up with M

simplest $M_{ij} = 1$ if word $i \in$ doc j

Note: M loses information about relative ordering of words

bag of words model

formally equivalent to unigram language models

Latent Semantic Indexing

Singular value decomposition of M (assume $\sigma_1 \geq \sigma_2 \geq ...$)

$$
\begin{aligned}
M &= \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_{\min(n,p)} \mathrm{u}_{\min(n,p)} \mathrm{v}_{min(n,p)}^T \\
&\approx \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_r \mathrm{u}_r \mathrm{v}_r^T \qquad (r \ll \min(n,p)) \quad = U^{(r)} {V^{(r)}}^T
\end{aligned}
$$

where $U^{(r)}$ ($V^{(r)}$) contains first r cols of U (V)

Latent Semantic Indexing

Singular value decomposition of M (assume $\sigma_1 \geq \sigma_2 \geq ...$)

$$M = \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_{\min(n,p)} \mathrm{u}_{\min(n,p)} \mathrm{v}_{min(n,p)}^T$$
$$\approx \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_r \mathrm{u}_r \mathrm{v}_r^T \qquad (r \ll \min(n,p)) \quad = U^{(r)} {V^{(r)}}^T$$

where $U^{(r)}$ ($V^{(r)}$) contains first r cols of U (V) Interpret the r vectors $\mathrm{u}_1, \ldots, \mathrm{u}_r$ as the r topics

Latent Semantic Indexing

Singular value decomposition of M (assume $\sigma_1 \geq \sigma_2 \geq ...$)

$$
\begin{aligned}
M &= \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_{\min(n,p)} \mathrm{u}_{\min(n,p)} \mathrm{v}_{min(n,p)}^T \\
&\approx \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_r \mathrm{u}_r \mathrm{v}_r^T \qquad (r \ll \min(n,p)) \quad = U^{(r)} {V^{(r)}}^T
\end{aligned}
$$

where $U^{(r)}$ ($V^{(r)}$) contains first r cols of U (V) Interpret the r vectors $\mathrm{u}_1, \ldots, \mathrm{u}_r$ as the r topics

Interpret the r vectors $\mathrm{v}_1, \ldots, \mathrm{v}_r$ as choice of topics in each doc

Demo

Pros and cons

Pros

Simple and fast

Often used to optimize search

University of Hawaiʻi Mānoa

Pros and cons

Pros

Simple and fast

Often used to optimize search

Cons

Topics orthogonal?

Negative values

- signal words absent (ok!)
- docs similar using *absence* of words, (not ok!)

Non negative matrix factorization

LSI: $M \approx U^{(r)} {V^{(r)}}^T$
How about find best A, W such that

$$M \approx AW,$$

A has r cols, W has r rows, all entries ≥ 0
Lot harder than SVD, optimization NP-hard
Approximations exist (EM, algebraic)

Language Models

Statistical models of language

Unigram, Bigram, Trigram...

Little bit of information theory (offline)

- entropy
- representation in bits
- cross entropy

Perplexity (power of a language model)

- GPT-4 2.6
- GPT-3.5 4.5

Modern Language Models

Tokenizer

Modern Language Models

Tokenizer

Brief history:

- Recurrent NN
- LSTMs
- Transformers

Modern Language Models

Tokenizer

Brief history:

- Recurrent NN
- LSTMs
- Transformers
  - only focus on this!

Transformers

What is a transformer?

Central to Transformers is the notion of *attention*

Attention in

- Linear Regression
- Kernels

Attention

$n \times p$ design matrix X, target y
Each row is an example (key)
Each target is a number (value)
Given a test example z (query), output?
Recall

$$\hat{w} = (X^T X)^{-1} X^T \mathrm{y}, \qquad \text{Prediction: } \mathrm{z}^T \hat{\mathrm{w}}$$

If $\mathrm{x}_1, \ldots, \mathrm{x}_n$ are the n examples:

$$\mathrm{z}^T \hat{\mathrm{w}} = \sum_{i=1}^{n} (\mathrm{z}^T (X^T X)^{-1} \mathrm{x}_i) y_i$$

Attention

The term $\mathrm{z}^T (X^T X)^{-1} \mathrm{x}_i$ is the attention the key x_i gets from the query z. The output is a linear combination of values y_i, with y_i weighted by the attention placed x_i.

Other algorithms

Ridge Regression

$$\mathrm{z}^T \hat{\mathrm{w}} = \sum_{i=1}^{n} (\mathrm{z}^T (X^T X + \lambda I)^{-1} \mathrm{x}_i) y_i$$

UNIVERSITY of HAWAI'I MĀNOA

Other algorithms

Ridge Regression

$$z^T\hat{w} = \sum_{i=1}^{n} (z^T(X^TX + \lambda I)^{-1} x_i) y_i$$

Support vector machines

Representer Theorem $w = \sum_{i=1}^{n} \beta_i x_i y_i$ (linear)

Soft prediction

$$z^T\hat{w} = \sum_{i=1}^{n} \beta_i (z^T x_i) y_i$$

β_i is obtained by solving the dual, most are 0

Other algorithms

Ridge Regression

$$z^T \hat{w} = \sum_{i=1}^{n} (z^T (X^T X + \lambda I)^{-1} x_i) y_i$$

Support vector machines

Representer Theorem $w = \sum_{i=1}^{n} \beta_i x_i y_i$ (linear)

Soft prediction

$$z^T \hat{w} = \sum_{i=1}^{n} \beta_i (z^T x_i) y_i$$

β_i is obtained by solving the dual, most are 0

Soft prediction (general case)

$$\sum_{i=1}^{n} \beta_i k(z, x_i) y_i$$