Distributional Semantic Models

Part 4: Elements of matrix algebra

Stefan Evert[1]
with Alessandro Lenci[2], Marco Baroni[3] and Gabriella Lapesa[4]

[1]Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
[2]University of Pisa, Italy
[3]University of Trento, Italy
[4]University of Stuttgart, Germany

http://wordspace.collocations.de/doku.php/course:start

Outline

Outline

Matrices and vectors

- $k \times n$ matrix $\mathbf{M} \in \mathbb{R}^{k \times n}$ is a rectangular array of real numbers

$$\mathbf{M} = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & & \vdots \\ m_{k1} & \cdots & m_{kn} \end{bmatrix}$$

- Each row $\mathbf{m}_i \in \mathbb{R}^n$ is an n-dimensional vector

$$\mathbf{m}_i = (m_{i1}, m_{i2}, \ldots, m_{in})$$

- Similarly, each column is a k-dimensional vector $\in \mathbb{R}^k$

```
> options(digits=3)
> M <- DSM_TermTerm$M
> M[2, ]  # row vector m2 for ‘‘dog’’
> M[, 5]  # column vector for ‘‘important’’
```

Matrices and vectors

- Vector $\mathbf{x} \in \mathbb{R}^n$ as single-row or single-column matrix
  - $\mathbf{x} = \mathbf{x}^{TT} = n \times 1$ matrix ("vertical")
  - $\mathbf{x}^T = 1 \times n$ matrix ("horizontal")
  - **transposition** operator $\cdot^T$ swaps rows & columns of matrix
- We need vectors $\mathbf{r} \in \mathbb{R}^k$ and $\mathbf{c} \in \mathbb{R}^n$ of marginal frequencies
- Notation for cell ij of co-occurrence matrix:
  - $m_{ij} = O \ldots$ observed co-occurrence frequency
  - $r_i = R \ldots$ row marginal (target)
  - $c_j = C \ldots$ column marginal (feature)
  - $N \ldots$ sample size

```
> r <- DSM_TermTerm$rows$f
> c <- DSM_TermTerm$cols$f
> N <- DSM_TermTerm$globals$N
> t(r)      # ''horizontal'' vector
> t(t(r))   # ''vertical'' vector
```


Scalar operations

- **Scalar** operations perform the same transformation on each element of a vector or matrix, e.g.
  - add / subtract fixed shift $\mu \in \mathbb{R}$
  - multiply / divide by fixed factor $\sigma \in \mathbb{R}$
  - apply function $(\log, \sqrt{\cdot}, \ldots)$ to each element
- Allows efficient processing of large sets of values
- Element-wise binary operators on matching vectors / matrices
  - $\mathbf{x} + \mathbf{y}$ = **vector addition**
  - $\mathbf{x} \odot \mathbf{y}$ = element-wise multiplication (**Hadamard product**)

```
> log(M + 1)  # discounted log frequency weighting
> (M["cause", ] + M["effect", ]) / 2 # centroid vector
```


The outer product

- Compute matrix $\mathbf{E} \in \mathbb{R}^{k \times n}$ of expected frequencies

$$e_{ij} = \frac{r_i c_j}{N}$$

  i.e. `r[i] * c[j]` for each cell ij
- This is the **outer product** of $\mathbf{r}$ and $\mathbf{c}$

$$\begin{bmatrix} r_1 \\ \vdots \\ r_k \end{bmatrix} \cdot \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix} = \begin{bmatrix} r_1 c_1 & r_1 c_2 & \cdots & r_1 c_n \\ \vdots & \vdots & & \vdots \\ r_k c_1 & r_k c_2 & \cdots & r_k c_n \end{bmatrix}$$

```
> outer(r, c) / N
```


Outline

Matrix multiplication

$$\begin{bmatrix} a_{ij} & \end{bmatrix} = \begin{bmatrix} b_{i1} & \cdots & b_{in} \end{bmatrix} \cdot \begin{bmatrix} c_{1j} & \\ \vdots & \\ \vdots & \\ c_{nj} & \end{bmatrix}$$

$$\underset{(k \times m)}{\mathbf{A}} = \underset{(k \times n)}{\mathbf{B}} \cdot \underset{(n \times m)}{\mathbf{C}}$$

- **B** and **C** must be **conformable**

☞ $\mathbf{A} \cdot \mathbf{x}$ corresponds to matrix multiplication of **A** with a single-column matrix (“vertical” vector **x**)

Some properties of matrix multiplication

- Associativity:
  $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C} =: \mathbf{ABC}$
- Distributivity:
  $\mathbf{A}(\mathbf{B} + \mathbf{B}') = \mathbf{AB} + \mathbf{AB}', \quad (\mathbf{A} + \mathbf{A}')\mathbf{B} = \mathbf{AB} + \mathbf{A}'\mathbf{B}$
- Scalar multiplication:
  $(\lambda\mathbf{A})\mathbf{B} = \mathbf{A}(\lambda\mathbf{B}) = \lambda(\mathbf{AB}) =: \lambda\mathbf{AB}$
- The square-diagonal **identity matrix**

$$\mathbf{I} := \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix} \quad \text{with} \quad \mathbf{I} \cdot \mathbf{A} = \mathbf{A} \cdot \mathbf{I} = \mathbf{A}$$

is the **neutral element** of matrix multiplication:

Transposition and multiplication

- The **transpose** $\mathbf{A}^T$ of a matrix **A** swaps rows and columns:

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \\ a_3 & b_3 \end{bmatrix}^T = \begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}$$

- Properties of the transpose:
  - $(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
  - $(\lambda\mathbf{A})^T = \lambda(\mathbf{A}^T) =: \lambda\mathbf{A}^T$
  - $(\mathbf{A} \cdot \mathbf{B})^T = \mathbf{B}^T \cdot \mathbf{A}^T$ [note the different order of **A** and **B**!]
  - $\mathbf{I}^T = \mathbf{I}$
- **A** is called **symmetric** iff $\mathbf{A}^T = \mathbf{A}$
  - symmetric matrices have many special properties that will become important later (e.g. eigenvalues)

The outer product as matrix multiplication

- The outer product is a special case of matrix multiplication

$$\mathbf{E} = \frac{1}{N}(\mathbf{r} \cdot \mathbf{c}^T)$$

- The other special case is the **inner product**

$$\mathbf{x}^T\mathbf{y} = \sum_{i=1}^{n} x_i y_i$$

- NB: $\mathbf{x} \cdot \mathbf{x}$ and $\mathbf{x}^T \cdot \mathbf{x}^T$ are not conformable

```
# three ways to compute the matrix of expected frequencies
> E <- outer(r, c) / N
> E <- (r %*% t(c)) / N
> E <- tcrossprod(r, c) / N
> E
```

Outline

Matrix algebra

Roll your own DSM

Matrix multiplication

Association scores & normalization

Computing association scores

- Association scores = element-wise combination of **M** and **E**, e.g. for (pointwise) Mutual Information

$$\mathbf{S} = \log_2(\mathbf{M} \oslash \mathbf{E})$$

  - $\oslash$ = element-wise division similar to Hadamard product $\odot$
- For sparse AMs such as PPMI, we need to compute $\max\{s_{ij}, 0\}$ for each element of the scored matrix **S**

```
> log2(M / E)
> S <- pmax(log2(M / E), 0) # not max() !
> S
```


Normalizing vectors

- Compute Euclidean norm of vector $\mathbf{x} \in \mathbb{R}^n$:

$$\|\mathbf{x}\|_2 = \sqrt{x_1^2 + \ldots + x_n^2}$$

- Normalized vector $\|\mathbf{x}_0\|_2 = 1$ by scalar multiplication

$$\mathbf{x}_0 = \frac{1}{\|\mathbf{x}\|_2}\mathbf{x}$$

```
> x <- S[2, ]
> b <- sqrt(sum(x ^ 2)) # Euclidean norm of x
> x0 <- x / b           # normalized vector
> sqrt(sum(x0 ^ 2))
```


Normalizing matrix rows

- Compute vector $\mathbf{b} \in \mathbb{R}^k$ of norms of row vectors of **S**
- Can you find an elegant way to multiply each row of **S** with the corresponding normalization factor b_i^{-1}?
- Multiplication with **diagonal matrix** $\mathbf{D_b}^{-1}$

$$\mathbf{S}_0 = \mathbf{D_b}^{-1} \cdot \mathbf{S}$$

$$\mathbf{S}_0 = \begin{bmatrix} b_1^{-1} & & \\ & \ddots & \\ & & b_k^{-1} \end{bmatrix} \cdot \begin{bmatrix} s_{11} & \cdots & s_{1n} \\ \vdots & & \vdots \\ s_{k1} & \cdots & s_{kn} \end{bmatrix}$$

Normalizing matrix rows

- Compute vector $\mathbf{b} \in \mathbb{R}^k$ of norms of row vectors of $\mathbf{S}$
- Can you find an elegant way to multiply each row of $\mathbf{S}$ with the corresponding normalization factor b_i^{-1}?
- Multiplication with **diagonal matrix** $\mathbf{D_b}^{-1}$

$$\mathbf{S}_0 = \mathbf{D_b}^{-1} \cdot \mathbf{S}$$

```
> b <- sqrt(rowSums(S^2))
> b <- rowNorms(S, method="euclidean") # more efficient

> S0 <- diag(1 / b) %*% S
> S0 <- scaleMargins(S, rows=(1 / b))   # much more efficient

> S0 <- normalize.rows(S, method="euclidean") # the easy way
```