Evgenia-Maria Kontopoulou
Dimitrios Zeimpekis
Efstratios Gallopoulos

Department of Computer Engineering and Informatics
University of Patras

Athens, November 10, 2011
What is TMG:

- MATLAB toolbox for text mining over ASCII-document collections
- Educational and Research tool

Why MATLAB?

- MATLAB is a popular Problem Solving Environment
- Providing implemented Linear Algebra Algorithms
- Linear Algebra basic kernel for many IR tasks
- Easy interface with PERL
- Sparse Matrix Infrastructure

Implementation

- over 16,000 lines of `matlab` and `perl` code
- takes advantage from sparse technology provided by MATLAB (`svds`, `sparse`)
- `perl` for parsing input documents
Available Operations

- Generate, Update and Downdate Term-by-Document Matrices (TDM)
- Dimensionality Reduction of TDMs
- Retrieval
- Clustering
- Classification

Six basic Modules

- Indexing
- Dimensionality Reduction
- Non-Negative Matrix Factorizations
- Retrieval
- Clustering
- Classification
## Supported non-ASCII formats

<table>
<thead>
<tr>
<th>Type</th>
<th>ver.5.0R6</th>
<th>Filter ver.5.0R6</th>
<th>ver. 6.0R7</th>
<th>Filter ver. 6.0R7</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>TIKA</td>
</tr>
<tr>
<td>docx</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>TIKA</td>
</tr>
<tr>
<td>htm</td>
<td>√</td>
<td>strip_html</td>
<td>√</td>
<td>strip_html</td>
</tr>
<tr>
<td>html</td>
<td>√</td>
<td>strip_html</td>
<td>√</td>
<td>TIKA</td>
</tr>
<tr>
<td>odt</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>TIKA</td>
</tr>
<tr>
<td>pdf</td>
<td>√</td>
<td>ps2ascii</td>
<td>√</td>
<td>ps2ascii</td>
</tr>
<tr>
<td>ps</td>
<td>√</td>
<td>ps2ascii</td>
<td>√</td>
<td>ps2ascii</td>
</tr>
<tr>
<td>rtf</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>TIKA</td>
</tr>
<tr>
<td>tex</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>Untex</td>
</tr>
</tbody>
</table>
### Generate, Update and Downdate Term-by-Document Matrices II

**Steps**

1. **parse input file**
2. **read stoplist**
3. **construct dictionary**
   - for each document construct local dictionary
   - merge each local dictionary with a global dictionary
4. **dictionary normalization**
   - remove common words
   - remove alphanumerics (optional)
   - remove numerics (optional)
   - remove short and long terms
   - apply stemming (optional) (Por80)
5. **create tdm**
6. **remove terms due to frequency parameters**
7. **apply global and local weights**
8. **final tdm**
Text to Term-Document Matrix (tdm) Generator

Input File/Directory: [Browse]

- Create New tdm
- Create Query Matrix
- Update tdm
- Downote tdm

Dictionary: [Browse]
Global Weights: [Browse]
Update Struct: [Browse]
Document Indices: [Browse]

OPTIONS

Delimiter: [emptyline]  Line Delimiter
Stoplist: [Browse]

Min Local Frequency: [1]  Max Local Frequency: [inf]
Min Global Frequency: [1]  Max Global Frequency: [inf]
Local Term Weighting: [Term Frequency]  Global Term Weighting: [None]

Database name: [ ]  Store in MySQL

- use Normalization
- use Stemming
- Remove Alpharumics
- Display Results
- Parse All Subdirectories

Continue  Reset  Exit
Dimensionality Reduction Module

**Purpose:**
- Handling high dimensional data
- More economical representation
- Better semantic representation

**Supported Algorithms**

1. **Singular Value Decomposition (SVD)**
   - MATLAB `svds`
   - PROPACK `svd(Lar)`

2. **Centroids Method (CM) (PJR03)**

3. **Semidiscrete Decomposition (SDD) (KO00)**

4. **Clustered LSI (CLSI) (ZG06a; ZG05)**

5. **SPQR Decomposition (BPS05)**

6. **Principal Component Analysis (PCA)**

*SDD and SPQR call routines available from Netlib (TOMS)*
Dimensionality Reduction GUI

Text to Term-Document Matrix (tdm) Generator

Select Dataset

Method
- Singular Value Decomposition (SVD)
- Principal Component Analysis (PCA)
- Clustered Latent Semantic Indexing (CLSI)
- Centroid Method (CM)
- Semidiscrete Decomposition (SDD)
- SPQR

Compute SVD with
- MATLAB (svds)
- Propack

Clustering Algorithm
- Euclidean k-means
- Spherical k-means
- PDDP

Initialize Centroids: At random
Termination Criterion: Epsilon
Principal Directions: Basic
Variant: Basic
Maximum num. of PCs: [ ]
Automatic Determination of Num. of factors for each cluster: [ ]
Number of Clusters: [ ]
Display Results: [ ]
Select at least one factor from each cluster (Recommended for classification): [ ]

Number of Factors: [ ]
Store Results: [ ]
Nonnegative Matrix Factorizations (NMF) Module

**What is NMF:**
- Iterative techniques
- Final result depends on initialization
- Resulting factors can be refined

<table>
<thead>
<tr>
<th>Initialization Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Random Initialization</td>
</tr>
<tr>
<td>2. <strong>Nonnegative Double SVD</strong> NNDSVD (BG08)</td>
</tr>
<tr>
<td>3. <strong>Block Nonnegative Double SVD</strong> (ZG08)</td>
</tr>
<tr>
<td>4. <strong>Bisecting Nonnegative Double SVD</strong> (ZG08)</td>
</tr>
<tr>
<td>5. By Clustering (WCD04)</td>
</tr>
</tbody>
</table>

NNDSVD uses prepared implementation

<table>
<thead>
<tr>
<th>Factors Refinement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Multiple Update Algorithms (LS01)</td>
</tr>
<tr>
<td>2. Alternating Non-Negative-Constrained Least Squares (NMF/ANLS) (KH08)</td>
</tr>
</tbody>
</table>

NMF/ANLS uses prepared implementation
Non-Negative Matrix Factorization GUI

Text to Term-Document Matrix (tdm) Generator

Select Dataset
- Random
- Nonnegative Double SVD (NNDSVD)
- Block NNDSVD
- Bisecting NNDSVD
- By Clustering

Initialization
- Refine Factors
- Method: Multiplicative Update
- Number of Iterations
- Display Results

Compute SVD with
- MATLAB (svds)
- Propack

Clustering Algorithm
- Euclidean k-means
  - Initialize Centroids: At random
  - Termination Criterion: Epsilon
- Spherical k-means
- PODP
  - Principal Directions
  - Variant: Basic
  - Maximum num. of PCs

Number of Clusters
- Display Results

Number of Factors
- Store Results

Continue | Reset | Exit
**Procedure:**
- Queries over a dataset
- HTML Response

**Information Retrieval Techniques**

1. Vector Space Model (VSM) (SWY75)
2. Latent Semantic Analysis (LSA) (Dee+90; BDJ99)

*LSA can be combined with any DR or NMF technique*
Text to Term-Document Matrix (tdm) Generator

- Select Dataset
- Insert query
- Alternative Global Weights
- Stoplist
- Local Term Weighting
- Vector Space Model
- Latent Semantic Analysis by:
- Number of Factors
- Similarity Measure
- Retrieve Documents: Number of most relevant:
- Similarity measure exceeds:

Continue  Reset  Exit
### Clustering Module

**Procedure:**
- Collection of documents as a TDM
- Create clusters of related documents

**Supported Algorithms**
1. Euclidean k-means
2. Spherical k-means (DM01)
3. Principal Direction Divisive Partitioning (PDDP) (Boi97)
   - PDDP (1) (ZG03) with some hybrid variants of PDDP and kmeans
Clustering GUI

Text to Term-Document Matrix (tdm) Generator

Select Dataset

- Euclidean k-means
  - Initialize Centroids: At random
  - Termination Criterion: Epsilon
- Spherical k-means
- FDDP
  - Principal Directions
  - Variant: Basic
  - Maximum num. of PCs

Compute SVD with
- MATLAB (svds)
- Propack

Number of Clusters
- Display Results
- Store Results

Continue  Reset  Exit
Procedure:

- Collection of documents as training set
- List of training labels
- Classify new documents to related classes (labels)

Supported Algorithms

- k Nearest Neighbors (knn)
- Rocchio
- Linear Least Squares Fit (LLSF) (YC92)

- Combination with CLSI, CM and SVD DR techniques
- Implementations for multilabel and singlelabel collections
TMG: A MATLAB tool for text mining

Evgenia-Maria Kontopoulou Dimitrios Zeimpekis Efstratios Gallopoulos

Users: More than 2500 requests worldwide and coming

Caltech, Cambridge, CMU, Colorado, Columbia, Florida, Renault, Leuven, Max-Planck, Michigan, Oxford, Philips, Princeton, Purdue, Los Alamos, Stanford, Toronto, Livermore, ...
Thank you!


(Bol97) D. Boley. ``Principal Direction Divisive Partitioning’’. In: Data Mining and Knowledge Discovery 2 (1997), pp. 325{344.


References II


