

TMG: A MATLAB tool for text mining

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

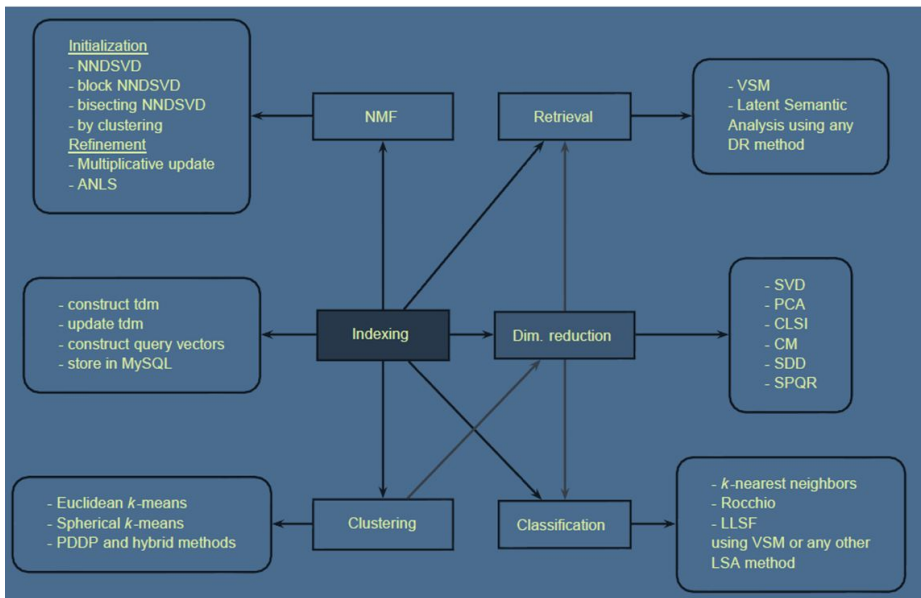
Available Operations

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

Six basic Modules

- 1 Indexing
- 2 Dimensionality Reduction
- 3 Non-Negative Matrix Factorizations
- 4 Retrieval
- 5 Clustering
- 6 Classification

Text to Matrix Generator



Supported non-ASCII formats

Type	ver.5.0R6	Filter ver.5.0R6	ver. 6.0R7	Filter ver. 6.0R7
doc	×	×	✓	TIKA
docx	×	×	✓	TIKA
htm	✓	strip_html	✓	strip_html
html	✓	strip_html	✓	TIKA
odt	×	×	✓	TIKA
pdf	✓	ps2ascii	✓	ps2ascii
ps	✓	ps2ascii	✓	ps2ascii
rft	×	×	✓	TIKA
tex	×	×	✓	Untex

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 Matrix Generator - Indexing

Window Help

Text to Term-Document Matrix (tdm) Generator

Input File/Directory

Create New tdm
 Create Query Matrix
 Update tdm
 Downdate tdm

Dictionary
 Global Weights
 Update Struct
 Document Indices

OPTIONS

Delimiter Line Delimiter
 Stoplist

Min Length Max Length
 Min Local Frequency Max Local Frequency
 Min Global Frequency Max Global Frequency
 Local Term Weighting Global Term Weighting

Database name Store in MySQL

use Normalization use Stemming Remove Alphanumerics Remove Numbers
 Display Results Parse All Subdirectories

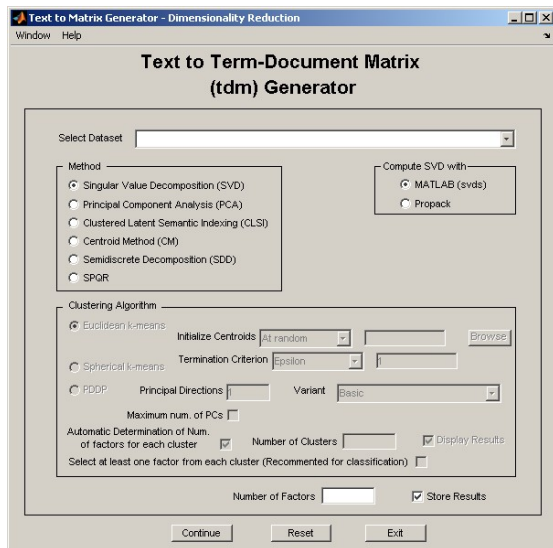
Purpose:

- 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) (K000)
- 4 **Clustered LSI (CLSI)** (ZG06a; ZG05)
- 5 SPQR Decomposition (BPS05)
- 6 Principal Component Analysis (PCA)

SDD and SPQR call routines available from `Netlib` (TOMS)



What is NMF:

- Iterative techniques
- Final result depends on initialization
- Resulting factors can be refined

Initialization Techniques

- 1 Random Initialization
- 2 **Nonnegative Double SVD NNDSVD** (BG08)
- 3 **Block Nonnegative Double SVD** (ZG08)
- 4 **Bisecting Nonnegative Double SVD** (ZG08)
- 5 By Clustering (WCD04)

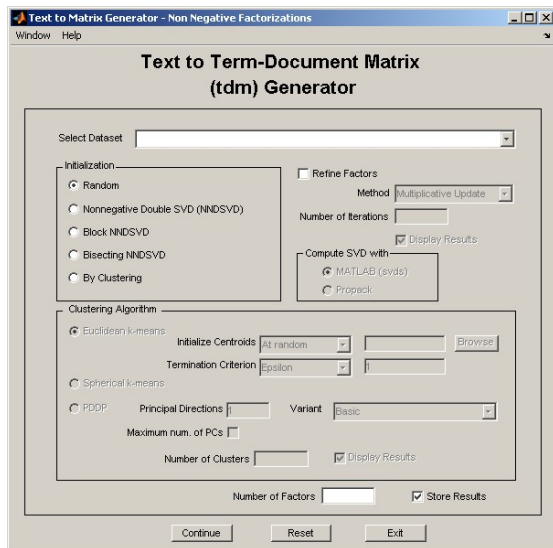
NNDSVD uses prepared implementation

Factors Refinement

- 1 Multiple Update Algorithms (LS01)
- 2 Alternating Non-Negative-Constrained Least Squares (NMF / ANLS) (KH08)

NMF / ANLS uses prepared implementation

Non-Negative Matrix Factorization GUI



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 Matrix Generator - Retrieval

Window Help

Text to Term-Document Matrix (tdm) Generator

Select Dataset

Insert query

Alternative Global Weights Use Stored Global Weights

Stoplist

Local Term Weighting

Vector Space Model

Latent Semantic Analysis by:

Number of Factors

Similarity Measure

Retrieve Documents: Number of most relevant:

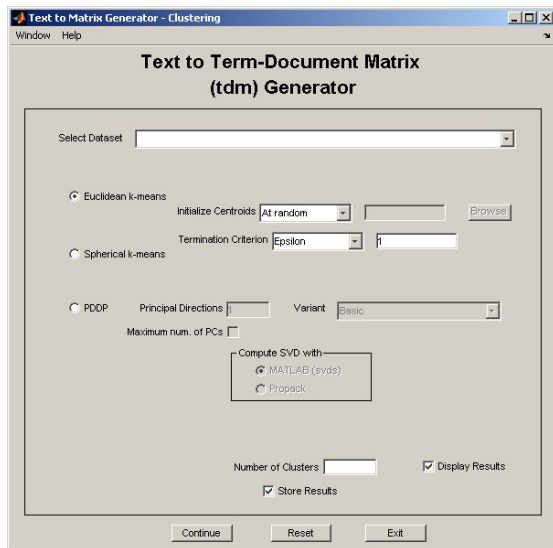
Similarity measure exceeds:

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) (Bol97)
 - **PDDP (1)** (ZG03) with some hybrid variants of PDDP and kmeans



Procedure:

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

Supported Algorithms

- 1 k Nearest Neighbors (knn)
 - 2 Rocchio
 - 3 Linear Least Squares Fit (LLSF) (YC92)
- Combination with CLSI, CM and SVD DR techniques
 - Implementations for multilabel and singlelabel collections

Text to Matrix Generator - Classification

Window Help

Text to Term-Document Matrix (tdm) Generator

Training Dataset

Training Labels Browse Use Stored Labels

Insert query

Single doc. (string)
 Multiple docs (file)

Alternative Global Weights Browse Use Stored Global Weights

Stoplist Browse

Local Term Weighting Term Frequency

Classification Method

k Nearest Neighbors (kNN)
Num. of NNs

Rocchio
Weight of Positive Examples
Weight of Negative Examples

Linear Least Squares Fit (LLSF)
Number of Factors

Collection Type

Multi-Label
 Single-Label

Use Thresholds Compute Thresholds
Thresholds Browse

Min. F-value

Vector Space Model
 Preprocessing by:

Number of Factors

Similarity Measure Cosine


Continue Reset Exit

TEXT TO MATRIX GENERATOR

Homepage | About TMG | Contact Us

Menu

- o Home
- o Documentation
- o Demo
- o Creations
- o Download
- o Contact

Designed by
 web hosting

TMG

Text to Matrix Generator (TMG) is a [MATLAB](#) toolbox that can be used for various tasks in text mining (TM). Most of TMG (version 5.0) is written in MATLAB, though a large segment of the indexing phase of the current version of TMG is written in [C++](#). Previous versions that were strictly MATLAB are also available. If [MySQL](#) and the [MATLAB Database Toolbox](#) are available, TMG exploits their functionality for additional flexibility.

TMG is especially suited for TM applications where data is high-dimensional but extremely sparse as it uses the sparse matrix infrastructure of MATLAB. Originally built as a preprocessing tool for creating term-document matrices (tdm's) from unstructured text, the new version of TMG (December'08) offers a wide range of tools for the following tasks:

1. Indexing
2. Retrieval
3. Dimensionality Reduction
4. Non-negative Matrix Factorizations
5. Clustering
6. Classification

TMG functionality is accessible to users in several ways: If MATLAB (version 7.0 or higher) is available then it is anticipated that most users will either use TMG's MATLAB-based GUI's, or will invoke the relevant functions directly from the MATLAB command line interface. Otherwise, users have available the following options (forthcoming): a) To download and use TMG's precompiled executable; b) to deploy TMG by means of a web interface to a server local to the TMG home. This latter option, however, does not allow access to the entire spectrum of TMG capabilities. c) The command line components of TMG have also been reported to run from [ODJave](#).

The indexing process constructs new and updates existing term-document matrices from documents, in the form of MATLAB sparse arrays. It may include various steps, such as removal of common words, such as articles and conjunctions, removal of very short or very long terms, removal of very frequent or infrequent terms, and the application of stemming. TMG applies common filtering techniques (removal of common words, removal of words that are too infrequent or frequent, removal of words that are too short or too long) to reduce the size of the term dictionary. TMG accepts as input files or directories consisting of ASCII text. In most cases it also processes with reasonable accuracy html and many PostScript and PDF files. TMG allows as option a variety of term-weighting and normalization schemes as well as stemming.

Users:(?) More than 2500 requests worldwide and coming

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

Thank you!

- (BDJ99) M.W. Berry, Z. DrmaĀ, and E. R. Jessup. "Matrices, vector spaces, and Information Retrieval". In: *SIAM Rev.* 41 (1999), pp. 335{362.
- (BG08) C. Boutsidis and E. Gallopoulos. "SVD based initialization: A head start for nonnegative matrix factorization". In: *Pattern Recognition* 41 (Apr. 2008), pp. 1350{1362.
- (BoI97) D. Boley. "Principal Direction Divisive Partitioning". In: *Data Mining and Knowledge Discovery* 2 (1997), pp. 325{344.
- (BPS05) M. W. Berry, S. A. Pulatova, and G. W. Stewart. "Algorithm 844: Computing sparse reduced-rank approximations to sparse matrices". In: *ACM Trans. Math. Softw.* 31.2 (2005), pp. 252{269.
- (Dee+90) S. Deerwester et al. "Indexing by Latent Semantic Analysis". In: *Journal of the American Society for Information Science* 41.6 (1990), pp. 391{407.
- (DM01) I. S. Dhillon and D. S. Modha. "Concept decompositions for large sparse text data using clustering". In: *Machine Learning* 42.1 (2001), pp. 143{175. URL: citeseer.nj.nec.com/article/dhillon01concept.html.
- (KH08) H. Kim and H.Park. In: *SIAM J. M. Anal. and Appl.* (2008).

- (KO00) T. G. Kolda and D. P. O'Leary. "Algorithm 805: computation and uses of the semidiscrete matrix decomposition". In: *ACM TOMS* 26.3 (2000), pp. 415{435. doi: 10.1145/358407.358424.
- (Lar) R.M. Larsen. *PROPACK: A software package for the symmetric eigenvalue problem and singular value problems on Lanczos and Lanczos bidiagonalization with partial reorthogonalization*. URL: <http://soi.stanford.edu/~rmunk/PROPACK/>.
- (LS01) D. D. Lee and H. S. Seung. "Algorithms for Non-negative Matrix Factorization". In: *NIPS*. MIT Press, 2001, pp. 556{562.
- (PJR03) H. Park, M. Jeon, and J.B. Rosen. "Lower Dimensional Representation of Text Data Based on Centroids and Least Squares". In: *BIT Numerical Mathematics* 43.2 (2003), pp. 427{448.
- (Por80) M.F. Porter. "An algorithm for suffix stripping". In: *Program* 3 (1980), pp. 130{137.
- (SWY75) G. Salton, A. Wong, and C. S. Yang. "A vector space model for automatic indexing". In: *Comm. ACM* 18.11 (1975), pp. 613{620.
- (WCD04) S. Wild, J. Curry, and A. Dougherty. "Improving non-negative matrix factorizations through structured initialization." In: *Pattern Recognition* 37.11 (2004), pp. 2217{2232.

- (YC92) Y. Yang and C. G. Chute. "A linear least squares fit mapping method for information retrieval from natural language texts". In: (1992), pp. 447{453.
- (ZG03) D. Zeimpekis and E. Gallopoulos. "PDDP(*l*): Towards a flexible principal direction divisive partitioning clustering algorithm". In: *Proc. Workshop on Clustering Large Data Sets (held in conjunction with the Third IEEE Int'l. Conf. Data Min.)* Ed. by D. Boley et al. Melbourne, FL, 2003, pp. 26{35.
- (ZG05) D. Zeimpekis and E. Gallopoulos. "CLSI: A Flexible Approximation Scheme from Clustered Term-Document Matrices". In: *Proc. 5th SIAM Int'l Conf. Data Mining*. Ed. by H. Kargupta et al. SIAM. Philadelphia, 2005, pp. 631{635.
- (ZG06a) D. Zeimpekis and E. Gallopoulos. "Linear and Non-Linear Dimensional Reduction via Class Representatives for Text Classification". In: *6th Int'l. Conf. Data Mining (ICDM'06)*. Los Alamitos, CA, USA: IEEE Computer Society, 2006, pp. 1172{1177.
- (ZG06b) D. Zeimpekis and E. Gallopoulos. "TMG: A MATLAB toolbox for generating term document matrices from text collections". In: *Grouping Multidimensional Data: Recent Advances in Clustering*. Ed. by J. Kogan, C. Nicholas, and M. Teboulle. Berlin: Springer, 2006, pp. 187{210.
- (ZG08) D. Zeimpekis and E. Gallopoulos. "Document clustering using nmf based on spectral information". In: *Text Mining Workshop (Atlanta)*. 2008.