MALLET: MAchine Learning for Language Toolkit

http://mallet.cs.umass.edu/index.php
Agenda

- What is MALLET
- Overview of MALLET functionality
- Exercise: From pdf collection to topic model visualization
1. What is MALLET

- Java-based package for statistical natural language processing.
- MALLET routines are claimed to be highly efficient and optimized for large datasets and feature sets.
- Main developer: Andrew McCallum (University of Massachusetts Amherst)
- Open source software released under CPL: can be used for research and commercial purposes.
- According to the author (and myself): “Although quite mature in functionality, documentation is still sparse.”
- Widespread adoption by different communities. Often used in conjunction with NLTK, D3, GEPHI and TEHNE
Andrew McCallum
Professor of Computer Science, University of Massachusetts Amherst
Machine Learning - Natural Language Processing - Information Extraction - Artificial Intelligence - Social Network Analysis
Verified email at cs.umass.edu
Homepage

<table>
<thead>
<tr>
<th>Citation indices</th>
<th>All</th>
<th>Since 2009</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citations</td>
<td>36618</td>
<td>21445</td>
</tr>
<tr>
<td>h-index</td>
<td>78</td>
<td>63</td>
</tr>
<tr>
<td>i10-index</td>
<td>172</td>
<td>159</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Title / Author</th>
<th>Cited by</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conditional random fields: Probabilistic models for segmenting and labeling sequence data</td>
<td>6851</td>
<td>2001</td>
</tr>
<tr>
<td>J Lafferty, A McCallum, FCN Pereira</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A comparison of event models for naive bayes text classification</td>
<td>2261</td>
<td>1998</td>
</tr>
<tr>
<td>A McCallum, K Nigam</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AAAI-98 workshop on learning for text categorization 752, 41-48</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text classification from labeled and unlabeled documents using EM</td>
<td>2242</td>
<td>2000</td>
</tr>
<tr>
<td>K Nigam, AK McCallum, S Thrun, T Mitchell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Machine learning 39 (2-3), 103-134</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum Entropy Markov Models for Information Extraction and Segmentation.</td>
<td>1109</td>
<td>2000</td>
</tr>
<tr>
<td>A McCallum, D Freitag, FCN Pereira</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICML, 591-598</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mallet: A machine learning for language toolkit</td>
<td>1055</td>
<td>2002</td>
</tr>
<tr>
<td>AK McCallum</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MALLET tools

- Data-import (text-to-features conversion)
- Document classification: Naïve Bayes, Maximum Entropy, Decision Trees ...
- Sequence Tagging: HMM, Maximum Entropy MM, CRF
- Topic modeling: LDA, Pachinko Allocation, Hierarchical LDA
- Numerical Optimization: Limited memory BFGS
- An add-on package, GRMM, supports inference in general graphical models, and training of CRFs
According to Radim Rehurek (GENSIM developer):

MALLET’s implementation of Latent Dirichlet Allocation has lots of things going for it.

It’s based on sampling, which is a more accurate fitting method than variational Bayes. Variational methods, such as the online VB inference implemented in gensim, are easier to parallelize and guaranteed to converge... but they essentially solve an approximate, aka more inaccurate, problem.

MALLET is not “yet another midterm assignment implementation of Gibbs sampling”. It contains cleverly optimized code, is threaded to support multicore computers and, importantly, battle scarred by legions of humanity majors applying MALLET to literary studies.

Plus, written directly by David Mimno, a top expert in the field.

In fact, there are plans to incorporate a Gibbs sampling-based implementation of LDA to GENSIM; in the meantime they provide a wrapper in GENSIM that uses MALLET implementation
Using MALLLET

- There are two possible ways to use MALLLET:
  - Through the JAVA API
  - Command-line script execution
- Documentation of JAVA classes is provided, but its use is not straightforward, not even for JAVA programmers
- We will work with command-line scripts
- As said, documentation is sparse ...
2. Overview of MALLET: Data import

- Three possibilities:
  - One instance per file: Uses directory names as labels, and filenames as instance names
    ```
    $ mallet import-dir --input path_to_dir --output FILENAME
    ```
  - One file, one instance per line: File format must be the following (per line): `[inst_name] [label] [text of the page ...]`
    ```
    $ mallet import-file --input FILENAME --output FILENAME
    ```
  - SVMlight format

- Input must be text (default UTF-8); output will be binary

- Common options:
  - `--keep sequence`: For sequence tagging and topic modeling
  - `--preserve-case`
  - `--token-regex`: for non-English languages

MLG – 07/04/2014
Overview of MALLET: Data import

- **Stopwords:**
  - *--remove-stopwords:* Standard list of English stopwords. Activating this option does not read any data from disk, so you cannot modify the list without recompiling.
  - *--extra-stopwords FILENAME:* Adds additional stopwords to the standard list. Only active if *--remove-stopwords* or *--stoplist-file* is also specified.
  - *--stoplist-file FILENAME:* Provides complete control over the stopword list. This option can be used by itself, and makes MALLET override the “standard” stopwords list.
Overview of MALLET: Classification

- Command line
  
  ```
  >>mallet train-classifier --input FILENAME --output-classifier FILENAME
  ```

- Common options:
  
  - --trainer MaxEnt / NaiveBayes / C45 / DecisionTree / Adaboost
  
  We can invoke several classifiers in one command

- If we wish to perform cross-validation we have two possibilities:
  
  --training-proportion 0.9 --num-trials 10 OR --cross-validation 10

- Reporting options: standard is confusion matrix, but others could be specified (e.g., --report train:accuracy test:f1:sports)

- Applying a saved classifier to new data:
  
  ```
  >>mallet classify-file --input FILENAME --output - --classifier FILENAME
  >>mallet classify-dir --input DIRNAME --output - --classifier FILENAME
  ```

  (will provide output in stdout)
Overview of MALLET: Topic Modeling

- MALLET commands
  - train-topics: LDA / PAM training
  - infer-topics
  - evaluate-topics
  - hlda

- Topic Inference and Document Probability Estimation

  >> mallet train-topics [filename] --inference-file-name FILENAME --evaluator-file-name FILENAME

  >> mallet infer-topics [filename] --inference-file-name FILENAME

  (estimates topic probability for a new document and an already trained topic model)

  >> mallet evaluate-topics [filename] --evaluator-file-name FILENAME

  (estimates log-probability of a document being generated by an already trained topic model)
Overview of MALLET: Topic Modeling

- train-topics output files:
  - --output-model FILENAME: binary topic model
  - --input-model FILENAME: binary topic model
  - --output-state FILENAME.gz: Provides a text file containing the words in the corpus with topic assignments
  - --output-topic-keys FILENAME: Top words for each topic and Dirichlet parameter
  - --output-doc-topics FILENAME: Topical composition of documents
  - --topic-word-weights-file FILENAME: Unnormalized weights for every topic and word
  - --word-topic-counts-file FILENAME: Sparse representation of topic-word assignments
3. Exercises on Topic Modeling: Topic Modeling tool

1. Convert your pdf collection into text files:
   1. In Mac you can use PDF to Text for Batch Processing

2. Remove any .txt not containing actual text

3. Download the “topic modeling tool” from:
   • https://code.google.com/p/topic-modeling-tool/
   • https://www.dropbox.com/sh/0vepxmx2wxejawg/AAA1L4IfpN4Cv9yUPoh1Avnya

4. Run the tool on the textfiles collection, and navigate through the topics you obtain
3. Exercises on Topic Modeling: MALLET Installation

1. Download MALLET:
   - http://mallet.cs.umass.edu/download.php
   - https://www.dropbox.com/sh/0vepxmx2wxejawg/AAA1L4IfpN4Cv9yUPoh1Avnya
   (+If using the API, you also need the JDK)

2. Unzip file and move to desired folder

3. Windows users need to set environment variable MALLET_HOME with the path to the binary files

4. `./bin/mallet` should display a list of available commands

5. `./bin/mallet` command `--help` displays a list of options for the selected command

6. Browse the directories under `mallet-2.0.7.zip`
   - Explore `/class/` to see what it is implemented in MALLET
3. Exercises on Topic Modeling: Importing data

1. Set aside some of your txt files that will be used for later inference once the model is trained

2. `>>./bin/mallet import-dir --help`

3. Basic import command:
   
   `>>./bin/mallet import-dir --input PATH_TO_DIR --output FILENAME.mallet`

4. Import Training data:
   - Remove Stopwords
   - Keep Sequence information

5. Import Validation data (Use `--use-pipe-from option`)
3. Exercises on Topic Modeling: Training a First Model

1. 
   
   `>>./bin/mallet train-topics --help`

2. Basic import command:
   
   `>>./bin/mallet import-dir --input FILENAME.mallet --num-topics 10`

3. Now you got your first LDA model! Explore the most important words of each topic, and generate a text file with stopwords.

4. Import your data again. This time, make sure to remove the previous lists of stopwords, and add bigrams as well.

5. Retrain topic model as above. Check that stopwords do not appear, and check also for bigrams.
3. Exercises on Topic Modeling: Obtaining a better model

1. Train a new LDA model. This time:
   - Make sure to keep all possible (non-binary) output files
   - Save an “inference” for the validation data
   - Set the number of iterations to 1000
   - Allow for optimization of the Dirichlet distribution parameters, every 10 iterations (by default, it will not start doing so before twice that number of iterations)

2. Let us do inference on the validation data

3. See the format of output files
3. Exercises on Topic Modeling: D3 Visualization


2. D3
   - Copy folder d3project on your working directory
   - cd to that folder, and run a local webserver
     ```
     >>>twistd -no web --path=
     ```
   - Connect to localhost:8080. See what happens
   - Study the structure of flare.json
   - Generate a new .json file from MALLET output, so that each topic is a bubble, and contains 6 bubbles associated to the most important words for that topic