Introduction to Mallet

LING 570

Fei Xia

Week 8: 11/18/2009

Mallet basics

- A package developed by McCallum’s group at UMass.
- It is written in Java.
- It includes most ML algorithms that we will cover in LING572.
- The package has been used by researchers from all over the world.
- It is still under development:
  - Some functions are missing
  - Some code has bugs

On Patas

- Mallet package:
  /NLP_TOOLS/tool_sets/mallet/latest

- Fei's classes:
  ~/dropbox/07-08/572/fei_mallet

- In each directory, there are several subdirectories:
  - bin/: shell script.
  - class/: the Java classes.
  - src/: the Java source code
  - lib/: (only for the Mallet dir), the jar files
  - doc/: some documents that explain the usage of main commands

Check the env !!

- If the following is already NOT in the default setup for all patas users, you need to add the following to your ~/.bash_profile, then start a new terminal

```
PATH=$PATH:$HOME/dropbox/07-08/572/fei_mallet/bin

 export  PATH

 CLASSPATH=$CLASSPATH:$HOME/dropbox/07-08/572/fei_mallet/class:/NLP_TOOLS/tool_sets/mallet/latest/lib/mallet.jar:/NLP_TOOLS/tool_sets/mallet/latest/lib/mallet-deps.jar

export   CLASSPATH
```

To test the env

- Type “which classify”:
  /opt/dropbox/07-08/572/fei_mallet/bin/classify

- Type “which vectors2info”
  /NLP_TOOLS/tool_sets/mallet/latest/bin/vectors2info

- If they do not work,

  echo $PATH
  ➔ /opt/dropbox/07-08/572/fei_mallet/bin should be there.

  echo $CLASSPATH
  ➔ /opt/dropbox/07-08/572/fei_mallet/class,
  /NLP_TOOLS/tool_sets/mallet/latest/lib/mallet.jar,
  /NLP_TOOLS/tool_sets/mallet/latest/lib/mallet-deps.jar should be there

Mallet commands

- Types:
  - Data preparation
  - Format conversation: text <-> binary
  - Training
  - Decoding
- All the commands are actually shell scripts that will call java.

An example: classifier2info

```
#!/bin/sh
malletdir=`dirname $0`
malletdir=`dirname $malletdir`
cp=$malletdir/lib/mallet.jar:$malletdir/class:$malletdir/lib/mallet-
    deps.jar:$CLASSPATH

mem=200m
arg=`echo "$1" | sed -e 's/-Xmx//'`
if test $1 != $arg ; then
   mem=$arg
   shift
fi

java -Xmx$mem-classpath   $cp
    edu.umass.cs.mallet.base.classify.tui.Classifier2Info"$@"
```

Data preparation

The format of the feature vectors

- Text format:

  instanceName   targetLabel   f1   v1   f2   v2 ….

- Binary format:
  - It stores the mapping from featName to featIdx, from targetLabel to targetIdx, etc.

- The learner/decoder uses only the binary format.

➔ We need to convert the text format to the binary format before training/decoding with the info2vectors command.

Data preparation

- info2vectors: convert vectors from the text format to the binary format
- vectors2info: convert vectors from the binary format to the text format
- vectors2vectors: split the binary vectors into training vectors and test vectors (all in the binary format)

- info2vectors --input news.vectors.txt --output news.vectors

- vectors2info --input news.vectors --print-matrix siw | remove_blank_line.exec > news.vectors.new_txt

- diff   news.vectors.txt   news.vectors.new_txt

  ➔ they are the same except that the (feat, val) pairs might be in different order.

- vectors2vectors --input news.vectors --training-portion 0.9 --training-file train.vectors --testing-file test.vectors

  The split uses a random function inside.

When training data and test data are prepared separately

info2vectors --input train.vectors.txt --output train.vectors

=> create train.vectors

info2vectors --input test.vectors.txt --output test.vectors --use-pipe-from   train.vectors

=> create test.vectors, which contains the same mapping

Training

Training

```
vectors2train  --training-file  train.vectors   --trainer  MaxEnt
--output-classifier   foo_model   --report  train:accuracy
    train:confusion > foo.stdout   2>foo.stderr
```

It will create

foo_model (the model): features and their weights

foo_stdout: the report, including training acc, confusion matrix

foo_stderr (the training info): iteration values, etc.

The name of trainer: MaxEnt, C45, DecisionTree, NaiveBayes, …

Viewing the model

```
classifier2info   --classifer    me_model   > me_model.txt
```

There is a typo in the Java code, so the option is misspelled.

- An example model:

```
FEATURES FOR CLASS guns
<default> 0.1298
fire 0.3934
firearms 0.4221
government 0.3721
arabic -0.0204
```

Accuracy and confusion matrix

- Confusion Matrix, row=true, column=predicted accuracy=0.9711111111111111

```
    label        0     1     2  | total
0   misc       846   27    23  | 896
1   mideast  12   899      2  | 913
2   guns       12      2  877  | 891
```

Train accuracy mean = 0.9711

Testing

Testing and evaluation

```
classify  --testing-file  test.vectors  --classifier  foo_model
   --report   test:accuracy   test:confusion  test:raw
    >foo_res.stdout   2> foo_res.stderr
```

In foo_res.stdout:

```
  instName  tgtLable  c1: score1 c2:score2 …

  talk.politics.guns/54600 guns guns:0.999 misc:9.24E-4
    mideast:1.42E-5
```

Test data accuracy= 0.87

Training, testing and eval

```
vectors2classify --training-file train.vectors --
   testing-file test.vectors --trainer MaxEnt >
   foo.stdout  2>foo.stderr
```

It is the same as vectors2train followed by classify.

The training and test accuracies are at the end of foo.stdout.

The error message in stderr

```
Logging configuration class
  "edu.umass.cs.mallet.base.util.Logger.Default
  Configurator" failed
  java.lang.ClassNotFoundException:
  edu.umass.cs.mallet.base.util.Logger.DefaultC
  onfigurator
```

➔ Please ignore this message.

Summary

Main commands

- Data preparation: info2vectors
  => create vectors in the binary format
  => use –use-pipe-from option when the training and test data are created separately.

- Training: vectors2train
  => create a model from the training data

- Testing and evaluation: classify
  => create classification results

- All the three are Fei's classes.

- Both vectors2train and classify have the --report option.

Other commands

- Split vectors into training and testing: vectors2vectors
  => It uses a random function.

- Viewing the vectors: vectors2info
  => use remove_blank_line.exec to remove the final blank line.

- Viewing the model: classifier2info
  => the –classifer option is misspelled.

- vectors2classify: training, test and eval
  – It is the same as vectors2train + classify

- All of these are Mallet's classes.

Other commands (cont)

- csv2vectors:
  - Convert a text file into the vectors in the binary format.
  - The text file has the format:

    InstName classLabel f1 f2 f3 …
  - Similar to info2vectors, but it does not allow feature values

Naming convention

- *.vectors: feature vectors in binary format
- *.vectors.txt: feature vectors in text format

- *_model: models in binary format
- *_model.txt: models in text format

File format

- Vectors in the text format:
  - InstName classLabel   fn1   fv1   fn2   fv2 ....
  - The order of the (featName, val) pairs does not matter.

- Classification results:
  - InstName   classLabel c1 score1 c2 score2 ....
  - (class, score) pairs are ordered by the score.

More information

- Mallet url (optional): for version 2.0.5
  http://mallet.cs.umass.edu/index.php

- A tutorial that I wrote for Mallet two years ago (optional):

  http://courses.washington.edu/ling572/winter07/homework/mallet_guide.pdf

  It discusses the main classes in Mallet.

Mallet version

- Latest version and tutorials from UMass' site: version 2.0.5

- Version on Patas: version 0.4

➔ There could be a mismatch between the two versions.

Hw8

Hw8

- Purpose:
  - Learn to use Mallet package
  - Learn to create feature vectors

- Text classification task

- Three categories: guns, mideast, and misc

- Each category has 1000 files under
  ~/dropbox/09-10/570/20_newsgroups/talk.politics.*/

- Q1: use text2vectors to create feature vectors, and make sure that Mallet works for you.

```
text2vectors –input  20_newsgroups/talk.politics.*
 --skip-header –output  news3.vectors
 => create news3.vectors
```

- Q2: the same task, but you need to prepare the vectors yourself.

Features in Hw8

Given a document

- Skip the header: use the text after the first blank lines.

- Replace any char that is not [a-zA-Z] with whitespace, lowercase everything, and break the lines into tokens by whitespace. These tokens are features.

  => This is different from the typical tokenization

- Feature values are the frequencies of the tokens in the document.

An example: talk.politics.guns/53293

Xref: cantaloupe.srv.cs.cmu.edu misc.headlines:41568 talk.politics.guns:53293

…

Lines: 38

hambidge@bms.com wrote:

: In article <C4psoG.C6@magpie.linknet.com>, manes@magpie.linknet.com (Steve Manes) writes:

After “tokenization”

hambidge@bms.comwrote:

:In article<C4psoG.C6@magpie.linknet.com>,
manes@magpie.linknet.com(SteveManes) writes:

hambidge bms comwrote

In articlec psog c magpie linknet commanes
magpie linknet com stevemanes writes

After lowercasing, counting and sorting

- talk.politics.guns/53293 guns a 11 about 2 absurd 1 again 1 an 1 and 5 any 2 approaching 1 are 5 argument 1 art icle 1 as 5 associates 1 at 1 average 2 bait 1 be 6 being 1 betraying 1 better 1 bms 1 by 5 c 2 calculator 1 capita 1 choice 1 chrissakes 1 citizen 1 com 4 crow 1 dangerous 1 deaths 2 die 1 easier 1 eigth 1 enuff 1 …

Coming up

- Please try the Mallet commands ASAP to ensure it runs for you. Do not wait until Wed.