User Manual
For Graphical Interface of MALLET

Contents

1. Introduction

1.1 Background

MALLET is a Java-based package for statistical natural language processing, including document classification and other machine learning applications to text. However, MALLET can only be run in cmd. That means users have to enter all complete and correct commands in cmd without any instructions, which is a really abstract way to use MALLET.

1.2 Purpose

For more convenience, the main purpose of this project is to design and build a graphical interface of MALLET, which is easy to input commands and select options. The most important thing is that the graphical interface is really practical for both common users and beginners.

2. Software Overview

2.1 Target

This document enables users to master the installation and use of this software.

2.2 Functions

2.2.1 Importing data

Before implementing some functions, the first step is to import data into unique MALLET format. There are two primary methods for importing data, the first is importing a directory, and the second is importing a single file.
Then the new generated MALLET file, such as web.mallet can be used in document classification.

2.2.2 Classification

Document classification means distinguishing between a fixed set of classes based on labeled training examples. MALLET includes several classification algorithms, such as Naïve Bayes, Maximum Entropy and Decision Tree, to train a classifier. Based on the classifier, users can apply a saved classifier to test new unlabeled data.

2.2.3 Sequence tagging

In natural language processing, sequence tagging means extracting information from the text, which aims to mark a sequence of words, such as a none or a non-none. Through analyzing the context, this function is used to judge the part of speech.

2.2.4 Topic modeling

Topic models provide a simple way to analyze large volumes of unlabeled text. That is, this function is used to extract topics from a number of articles through the analysis of probability distribution of words. A 'topic' consists of a cluster of words that frequently occur together. Using contextual clues, topic models can connect words with similar meanings and distinguish between uses of words with multiple meanings. For example, in the article which is related to medical care, the presence of the probability of the words, such as medicine, cure, are very high. After importing text files into unique MALLET format, the topic models can be built. Besides, a topic inference tool can also be used based on the current, trained model.

2.2.5 Optimization

MALLET includes methods for numerical optimizing functions. The primary use for numerical optimization in machine learning is to find parameters that maximize a log-likelihood function given observed data. This software gives a demo of putting two parameters to achieve the function.

2.2.6 Graphical models

GRMM is short for Graphical Models in MALLET which is an add-on package to MALLET and contains support for performing inference and learning in graphical models of arbitrary structure. GRMM supports arbitrary factor graphs and it includes efficient implementations of several inference algorithms. In this software, GRMM can be achieved by selecting model path, train path and test path. All the paths should comply with specific format.

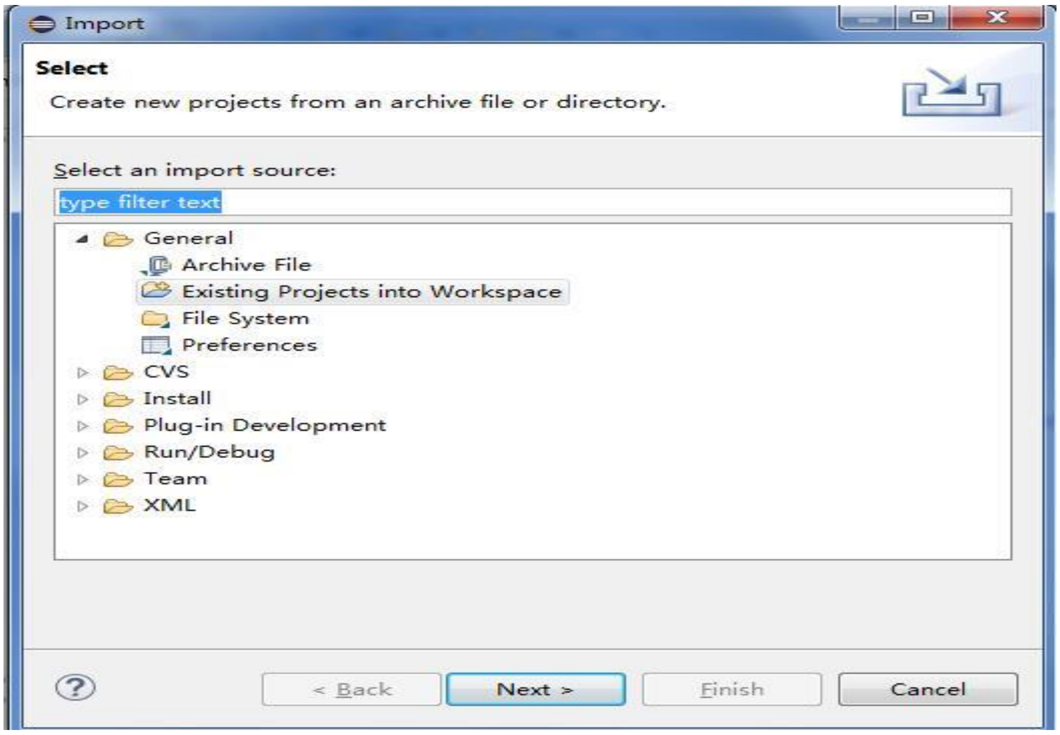
3. Operating Environment

This software is written in JAVA, which can be run in the eclipse.

4. Instructions

4.1 Importing the folder into eclipse

First, put the 'malletGUI2.3' folder into the workspace, and import in the eclipse by clicking 'File', 'Import'. Then click 'Existing Projects into Workspace' ➔ 'Select root directory' can be chosen by the button 'Browse'. Select 'malletGUI2.3' in the folder 'workspace. Finally, click the button 'confirm', then the project is imported. In 'src', choose the package which is called 'application.ui', and then run the 'MainWindow.java'. After that, a graphical interface will be displayed.



Import

Select

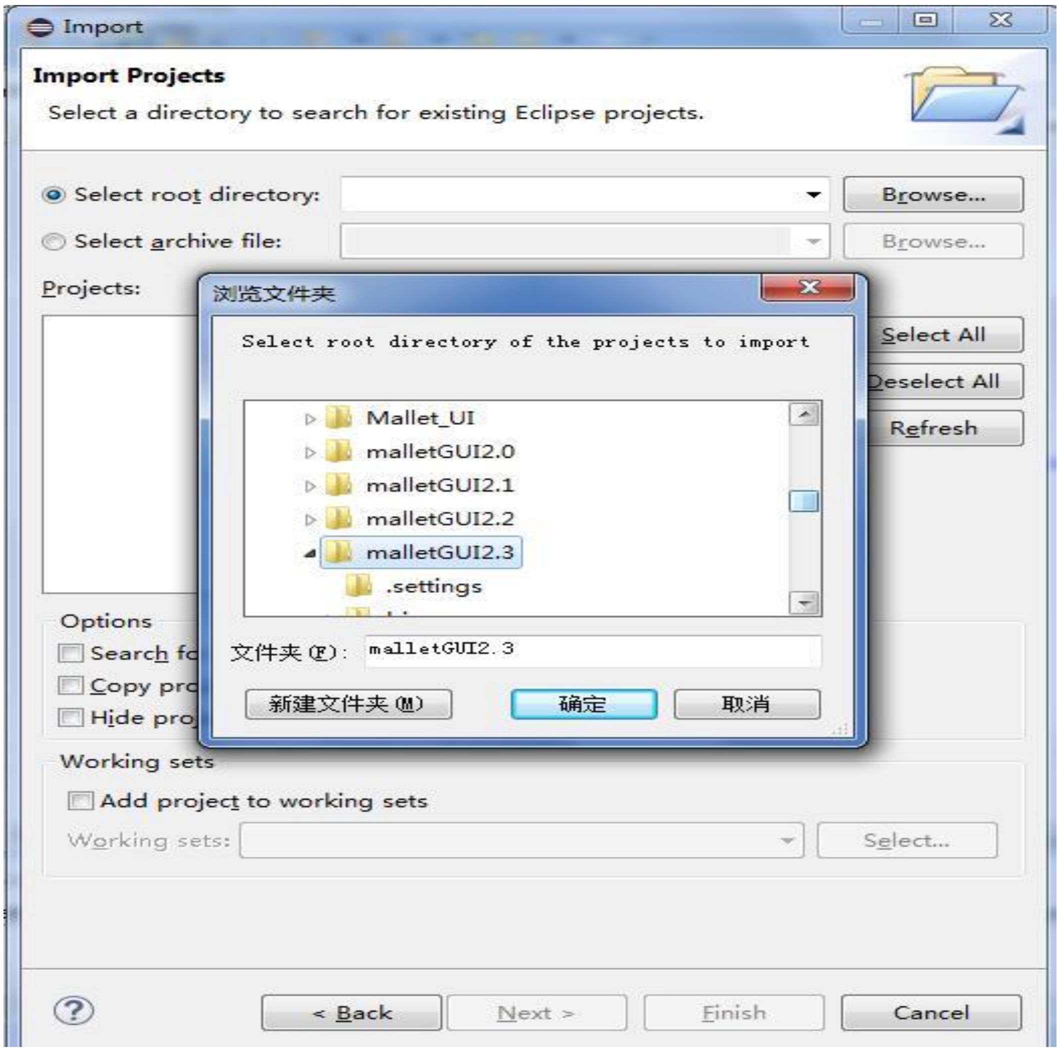
Create new projects from an archive file or directory.

Select an import source:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences
- CVS
- Install
- Plug-in Development
- Run/Debug
- Team
- XML

< Back | Next > | Finish | Cancel



Import

Import Projects

Select a directory to search for existing Eclipse projects.

◉ Select root directory: | Browse...

○ Select archive file: | Browse...

Projects:

Select All | Deselect All | Refresh

Options

☐ Search fo

☐ Copy pro

☐ Hide proj

浏览文件夹

Select root directory of the projects to import

- Mallet_UI
- malletGUI2.0
- malletGUI2.1
- malletGUI2.2
- malletGUI2.3
  - .settings

文件夹(F): malletGUI2.3

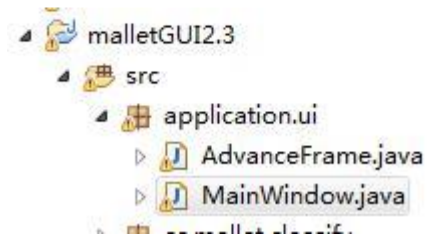
新建文件夹(M) | 确定 | 取消

Working sets

☐ Add project to working sets

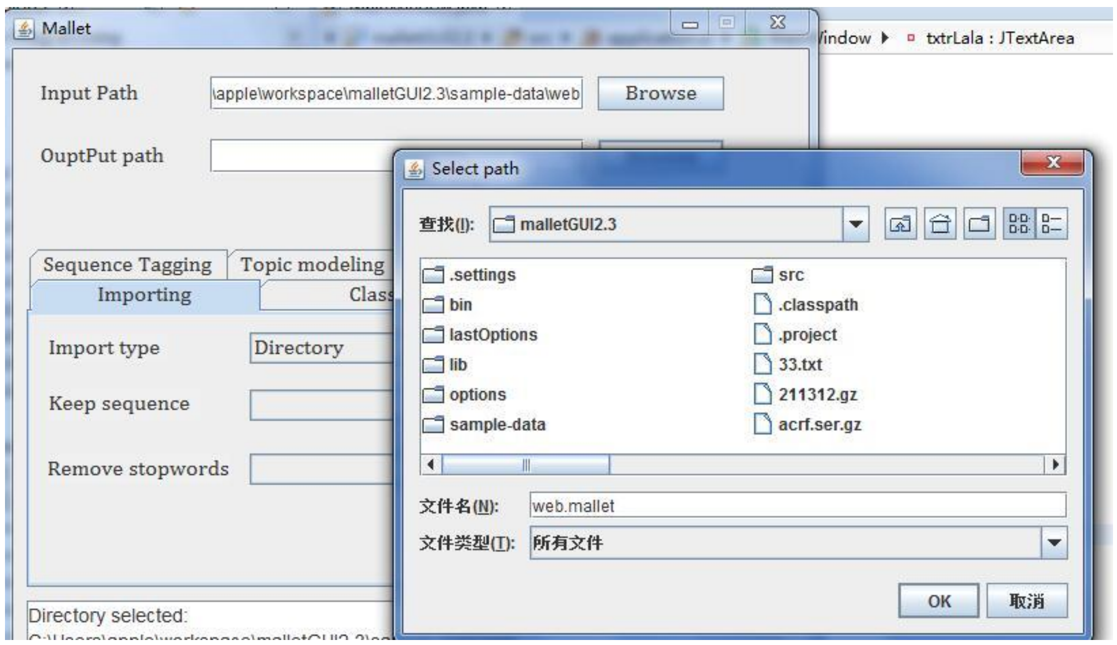
Working sets: | Select...

< Back | Next > | Finish | Cancel

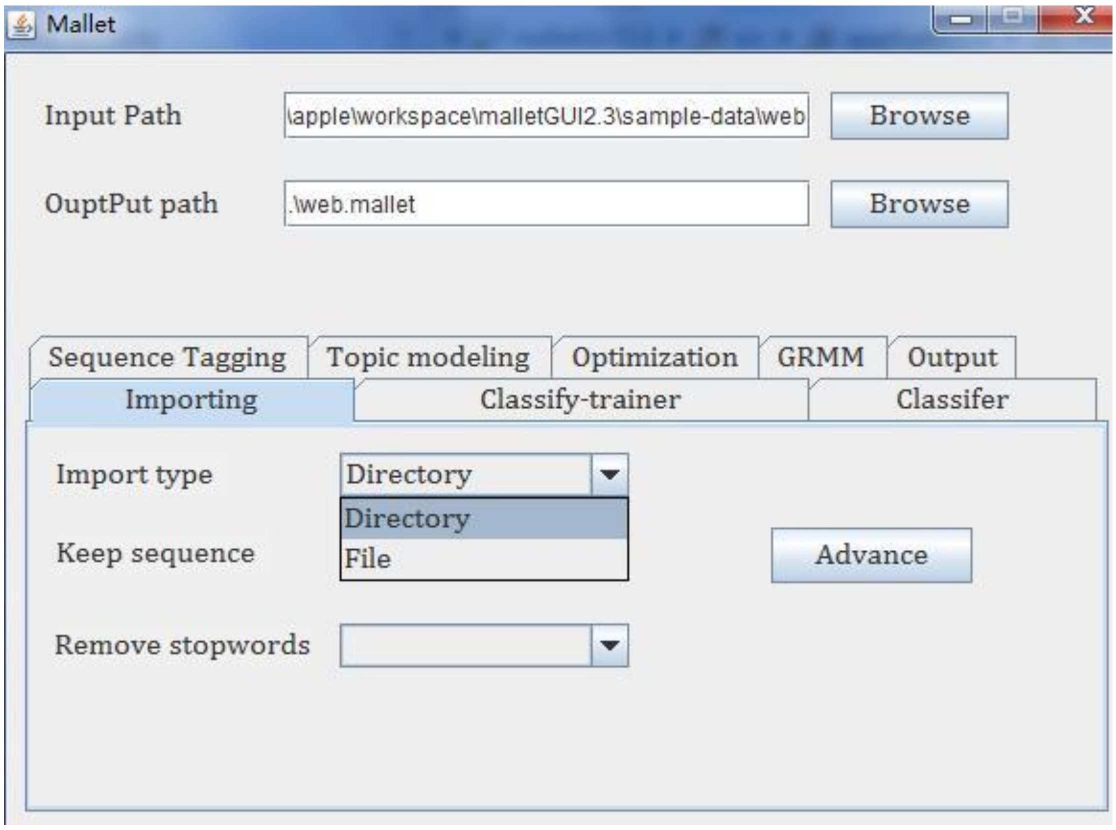


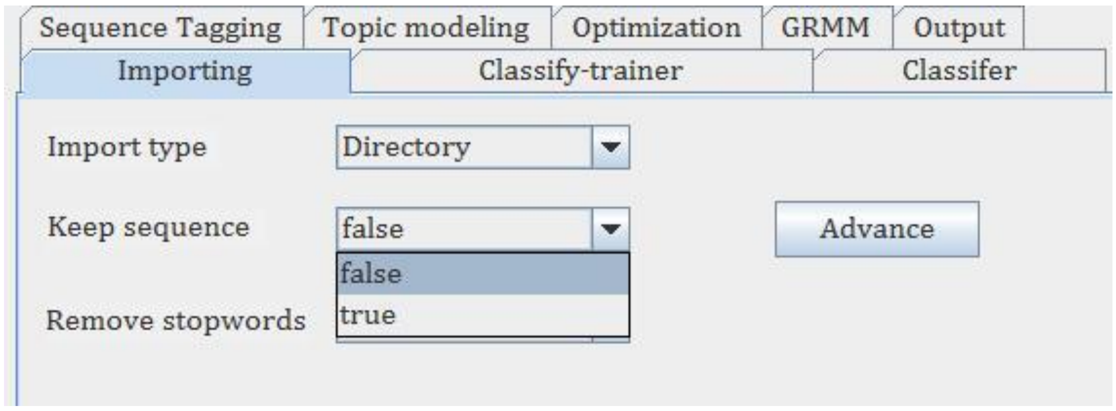
4.2 Importing data

At the top of the graphical interface, there are *input path* and *output path* that can be chosen by clicking the button '*Browse*' or entered by the users. *Output path* should be ended in .mallet.

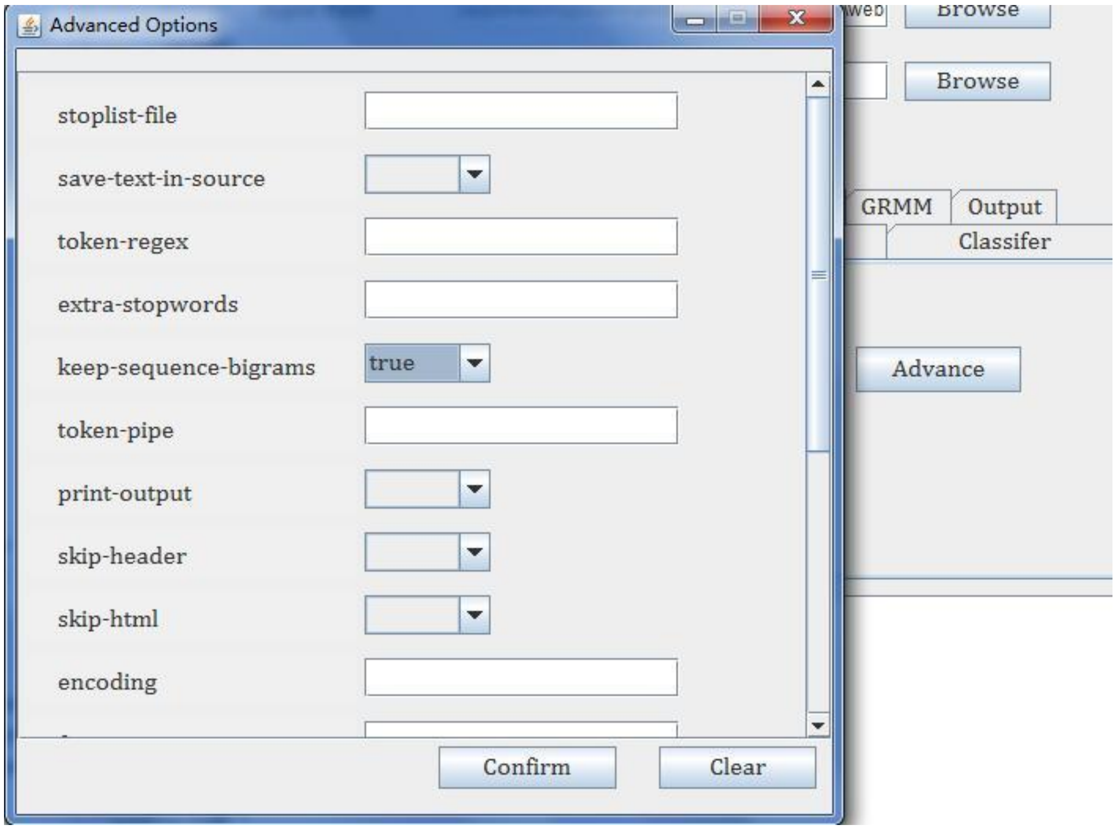


A JTabbedPane can be used to select functions, such as '*Importing*'. *Import type* must be selected depending on what users have input in the above, a *file* or a *directory*. *Keep sequence* and *Remove stopwords* are optional which can be selected through JComboBox or just leave them empty.



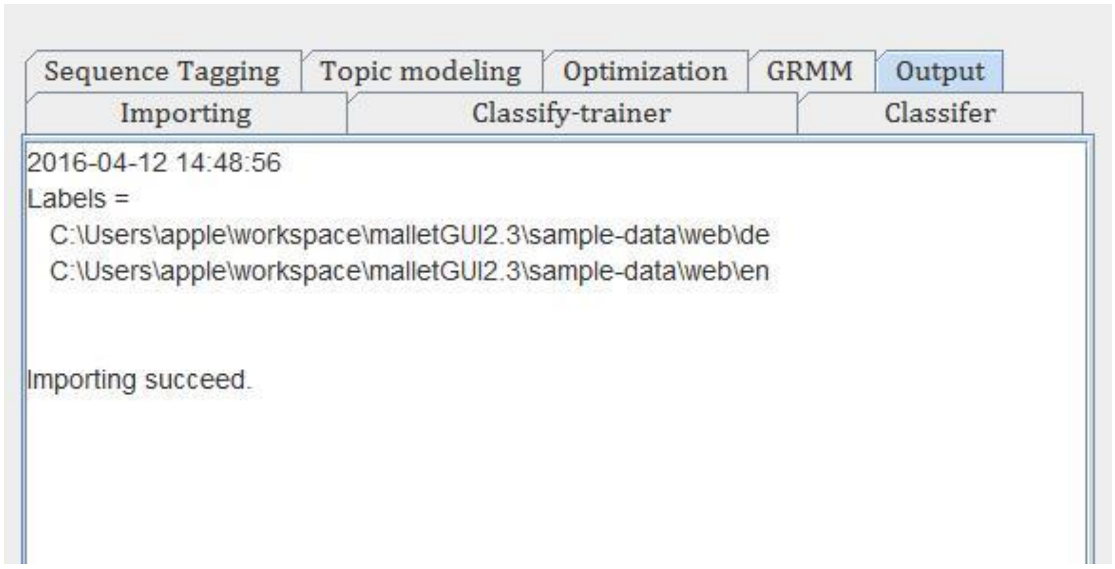


There are also other options can be seen by clicking the button ‘*Advance*’.



stoplist-file
save-text-in-source
token-regex
extra-stopwords
keep-sequence-bigrams true
token-pipe
print-output
skip-header
skip-html
encoding

At the end, clicking the button '*Run*', then the data is imported and the result can be displayed in the '*output*'.

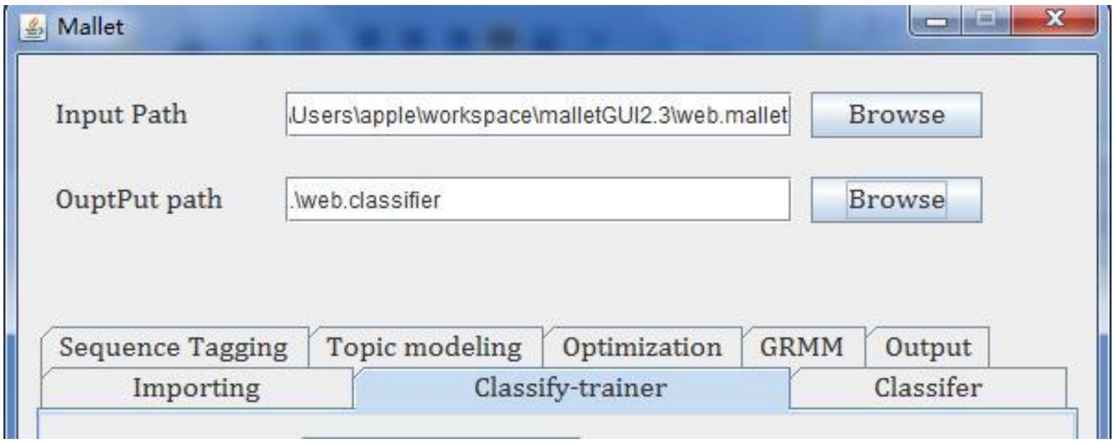


```
2016-04-12 14:48:56
Labels =
  C:\Users\apple\workspace\malletGUI2.3\sample-data\web\de
  C:\Users\apple\workspace\malletGUI2.3\sample-data\web\en


Importing succeed.
```

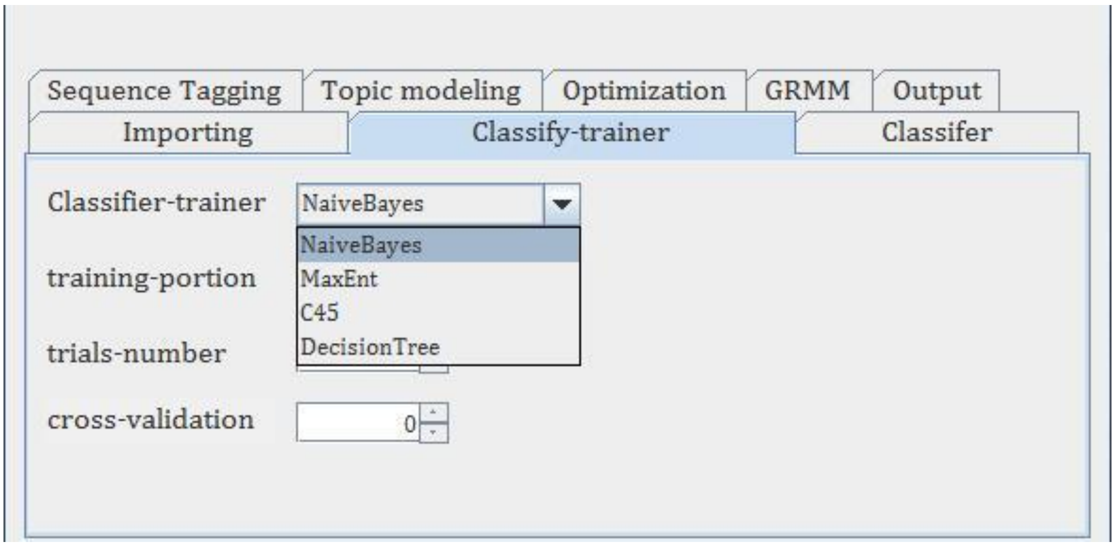
4.3 Classification

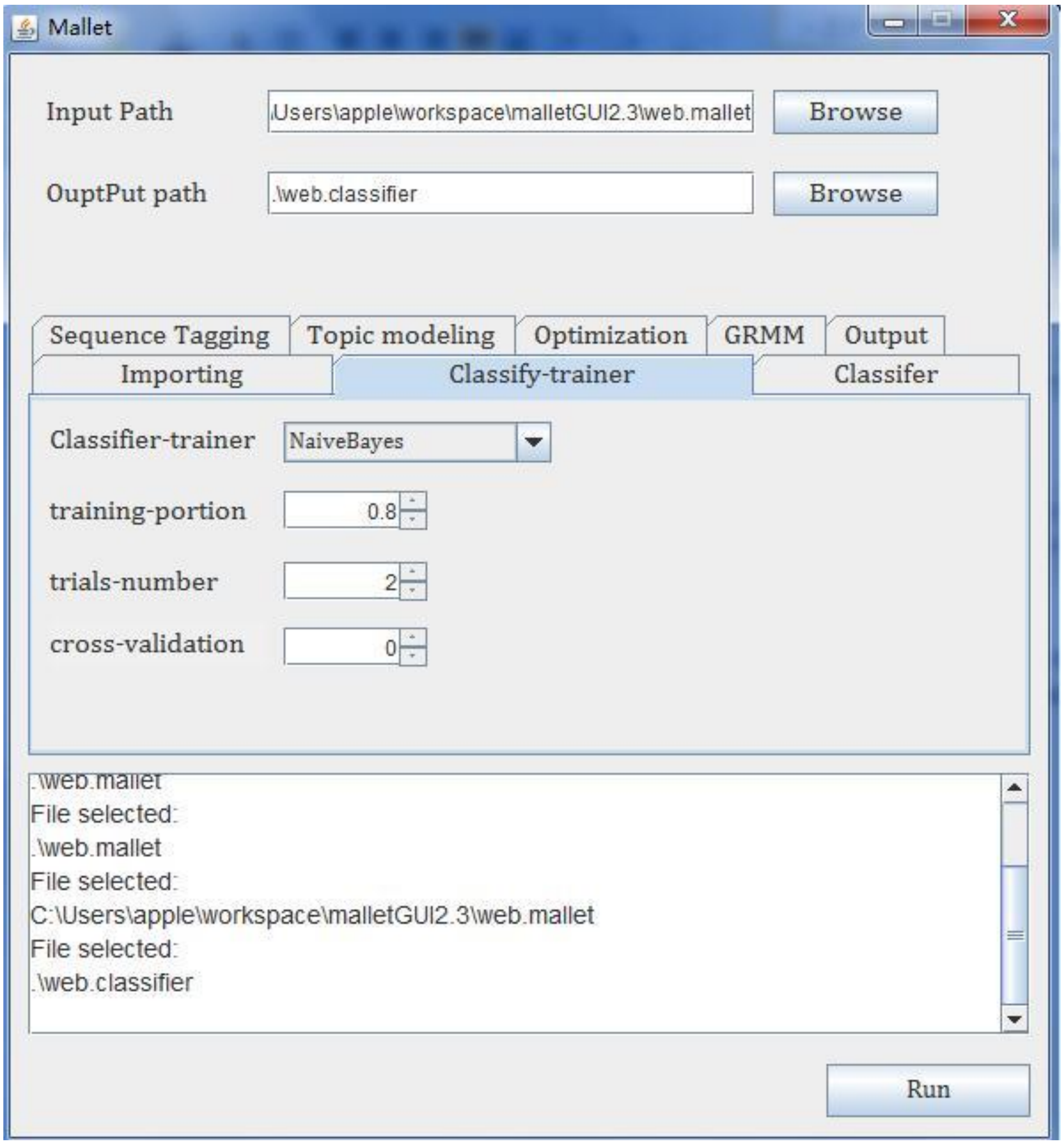
4.3.1 Train a classifier

As for the tag 'Classify-trainer', users can input data which is ended in .mallet and output data which is ended in .classifier.



The classification algorithm (such as Na ïve Bayes, Maximum Entropy, C45 and Decision Tree), *training-portion*, *trials number* and *cross-validation* are optional. Users can also leave them empty. The button '*Run*' is used to start the function and the output can be observed in '*output*' panel.





Input Path: Users\apple\workspace\malletGUI2.3\web.mallet

OuptPut path: .\web.classifier

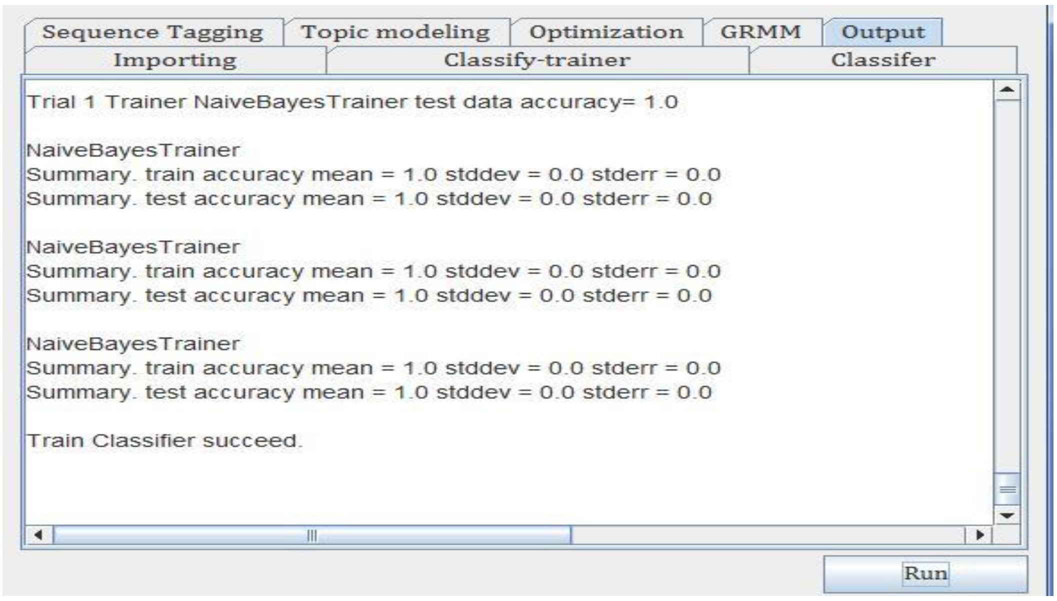
Classifier-trainer: NaiveBayes

training-portion: 0.8

trials-number: 2

cross-validation: 0

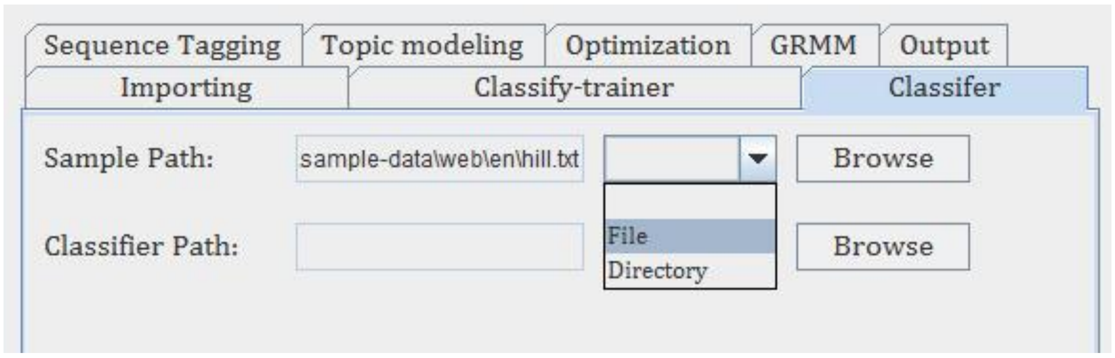
```
.\web.mallet
File selected:
.\web.mallet
File selected:
C:\Users\apple\workspace\malletGUI2.3\web.mallet
File selected:
.\web.classifier
```



```
Trial 1 Trainer NaiveBayesTrainer test data accuracy= 1.0

NaiveBayesTrainer
Summary. train accuracy mean = 1.0 stddev = 0.0 stderr = 0.0
Summary. test accuracy mean = 1.0 stddev = 0.0 stderr = 0.0

NaiveBayesTrainer
Summary. train accuracy mean = 1.0 stddev = 0.0 stderr = 0.0
Summary. test accuracy mean = 1.0 stddev = 0.0 stderr = 0.0

NaiveBayesTrainer
Summary. train accuracy mean = 1.0 stddev = 0.0 stderr = 0.0
Summary. test accuracy mean = 1.0 stddev = 0.0 stderr = 0.0

Train Classifier succeed.
```

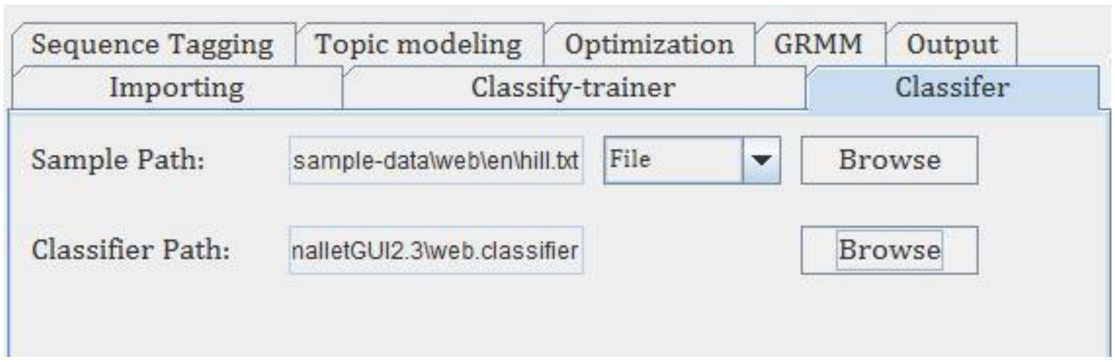
4.3.2 Forecast new data

The tag '*classifier*' is related to apply the saved classifier to test new data. Users only need to choose '*Sample Path'* and '*Classifier Path'*. Choose *file* or *directory* according to the type of new data, then clicking '*Run*'.



Sample Path: sample-data\web\en\hill.txt

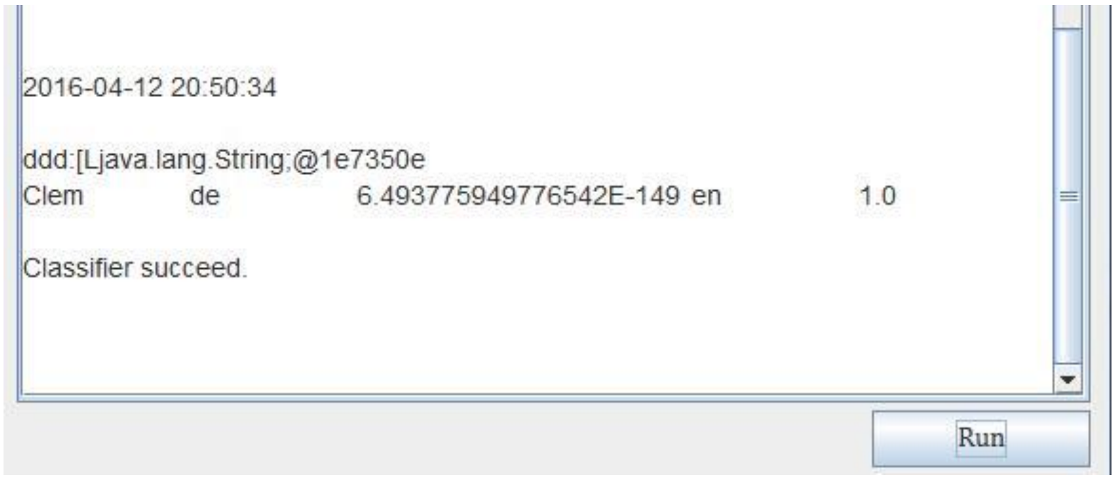
Classifier Path:



Sample Path: sample-data\web\en\hill.txt File

Classifier Path: nalletGUI2.3\web.classifier

The output can be seen.

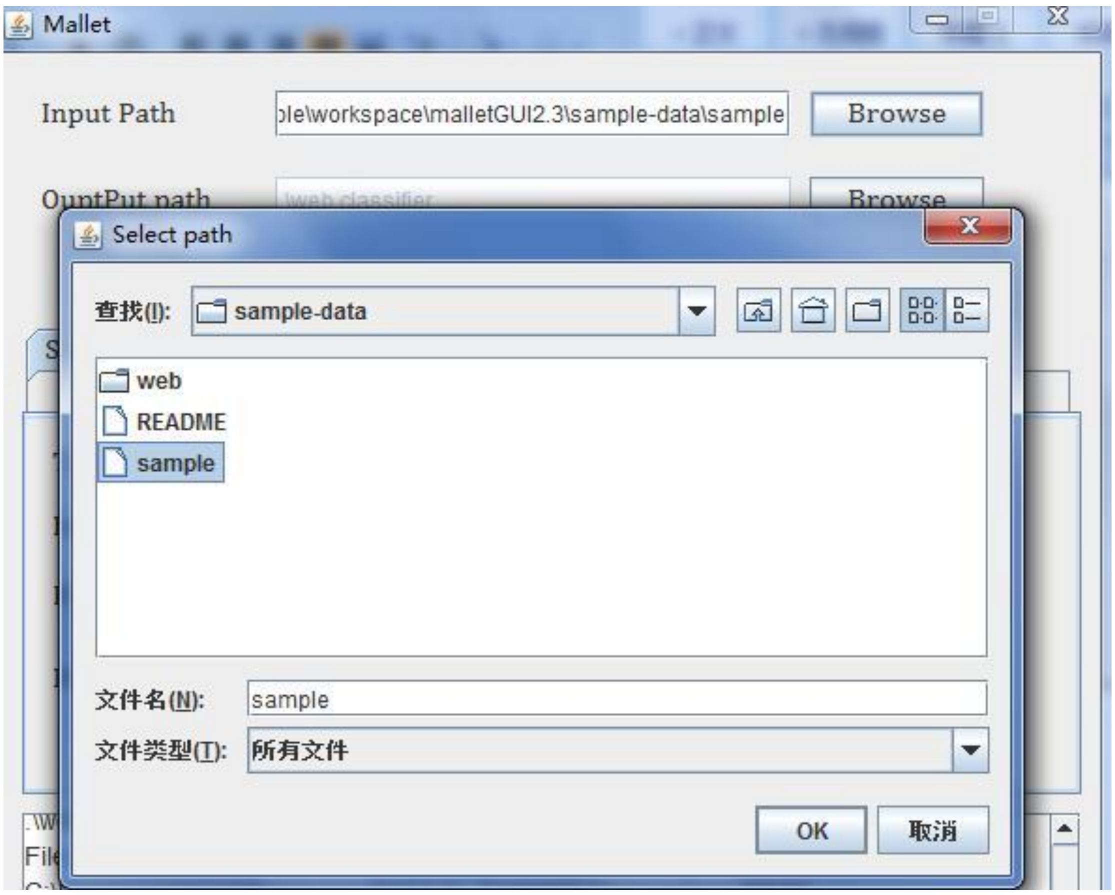


```
2016-04-12 20:50:34

ddd:[Ljava.lang.String;@1e7350e
Clem        de          6.493775949776542E-149 en         1.0

Classifier succeed.
```

4.4 Sequence tagging

4.4.1 Train a model

The file '*sample*' which is in a unique train format is selected by clicking the button '*Browse*' to fill in the '*Input Pat*h', while the 'Output Path' does not need to be filled.



Mallet

Input Path: ɔle\workspace\malletGUI2.3\sample-data\sample [Browse]

OuptPut path: \web classifier [Browse]

Select path

查找(I): sample-data

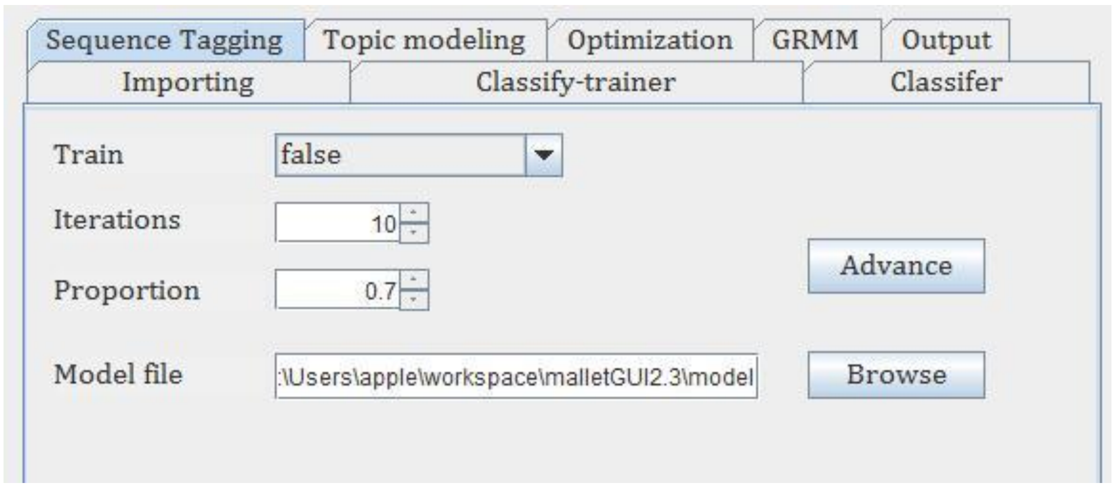
web
README
sample

文件名(N): sample

文件类型(T): 所有文件

[OK] [取消]

To train a sequence tagging model, the ‘*Train*’ must be *true*. An existing model can be selected or create a new model in ‘*Model file’*. *Iterations* and *Proportion* are optional and there are also other options can be seen by clicking the button ‘*Advance*’.



Sequence Tagging | Topic modeling | Optimization | GRMM | Output
Importing | Classify-trainer | Classifer

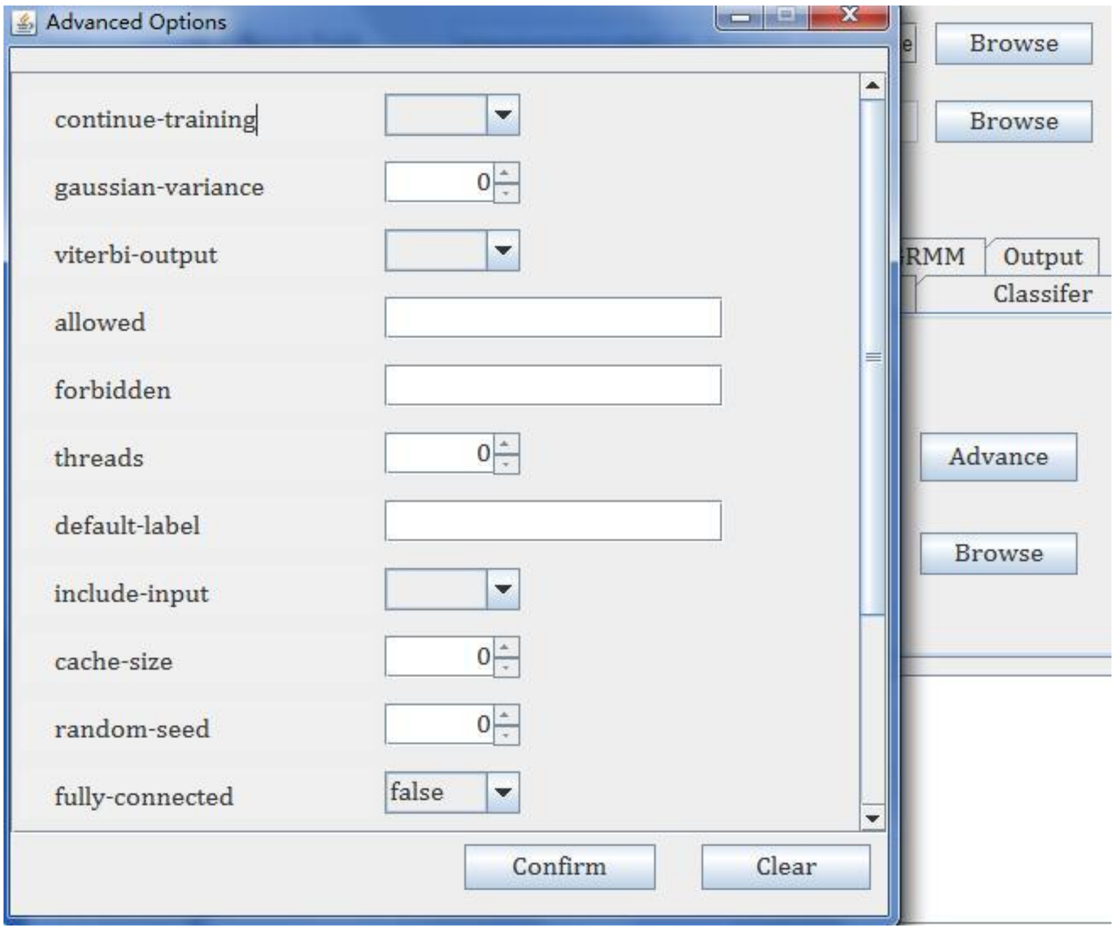
Train: false

Iterations: 10

Proportion: 0.7

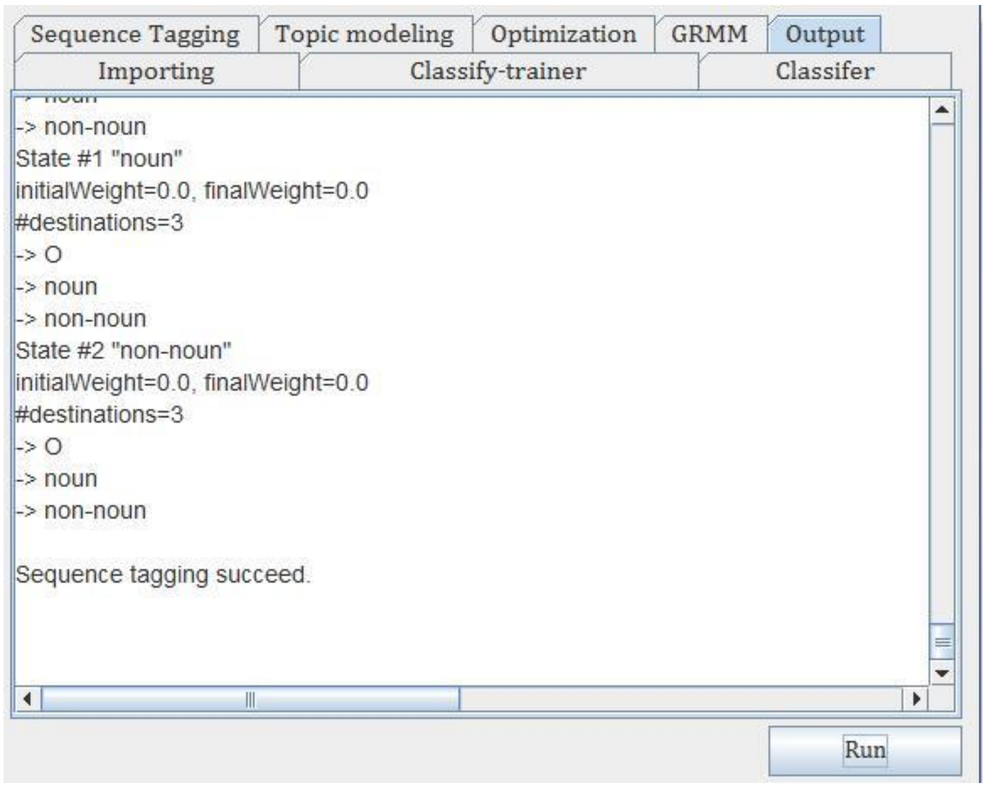
Model file: :\Users\apple\workspace\malletGUI2.3\model [Browse]

[Advance]



continue-training
gaussian-variance 0
viterbi-output
allowed
forbidden
threads 0
default-label
include-input
cache-size 0
random-seed 0
fully-connected false

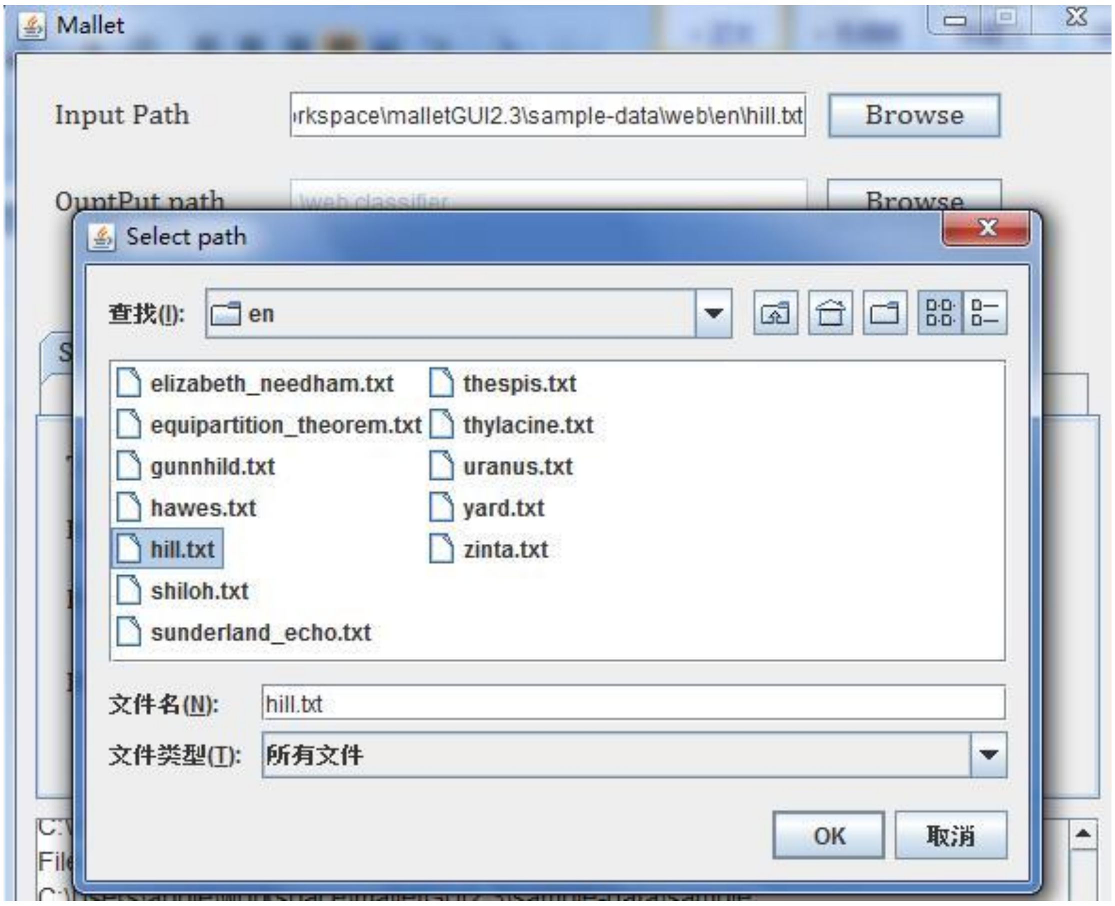
After clicking the button ‘*Run*’, the function can be implemented and the output can be seen.



```
-> noun
-> non-noun
State #1 "noun"
initialWeight=0.0, finalWeight=0.0
#destinations=3
-> O
-> noun
-> non-noun
State #2 "non-noun"
initialWeight=0.0, finalWeight=0.0
#destinations=3
-> O
-> noun
-> non-noun

Sequence tagging succeed.
```

4.4.2 Forecast text

Users can use the trained model to forecast the new text. The new text, such as ‘hill.txt’, should be selected and put in the ‘*Input Path*’.



Input Path: rkspace\malletGUI2.3\sample-data\web\en\hill.txt

Select path

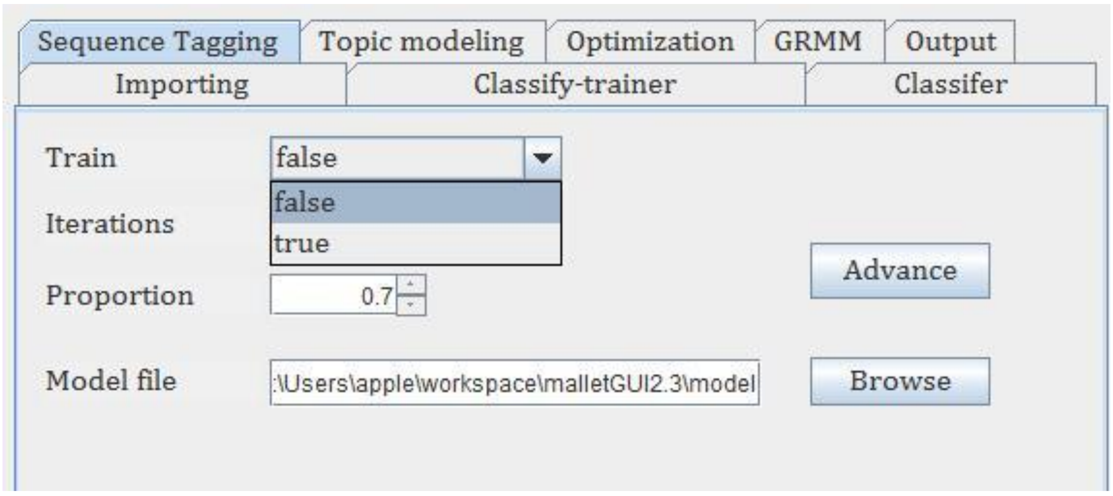
查找(I): en

elizabeth_needham.txt
equipartition_theorem.txt
gunnhild.txt
hawes.txt
hill.txt
shiloh.txt
sunderland_echo.txt
thespis.txt
thylacine.txt
uranus.txt
yard.txt
zinta.txt

文件名(N): hill.txt

文件类型(T): 所有文件

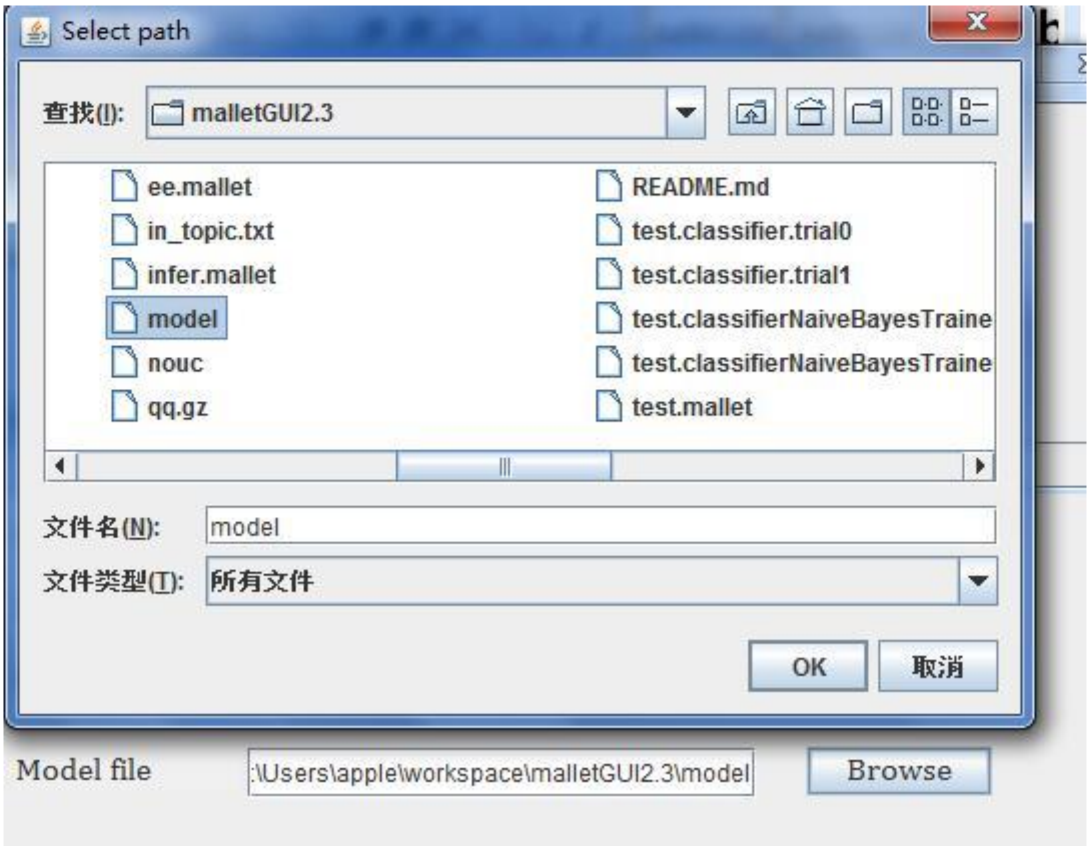
To forecast text, the ‘*Train*’ must be *false*. ‘*Model file’* should be a trained sequence tagging model which has already existed.



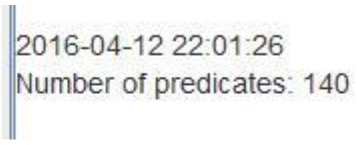
Train: false

Proportion: 0.7

Model file: :\Users\apple\workspace\malletGUI2.3\model



The output can be seen.

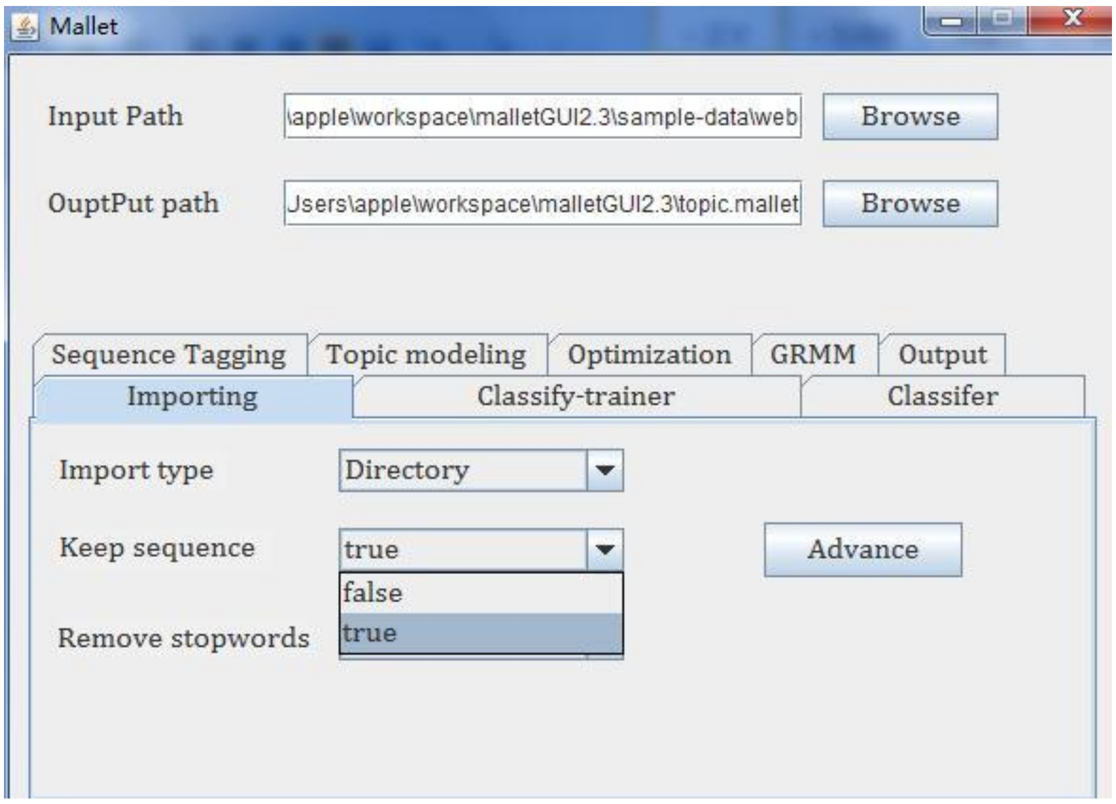


```
2016-04-12 22:01:26
Number of predicates: 140
```

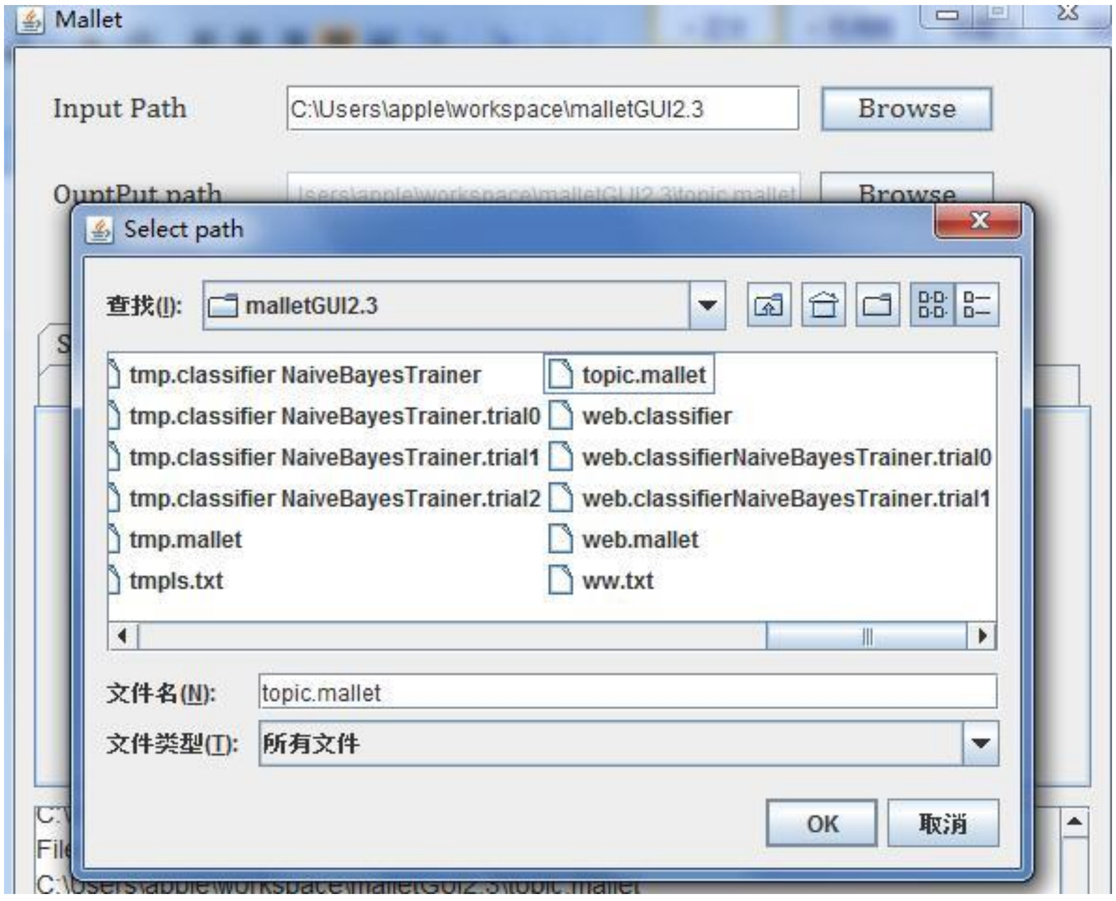
4.5 Topic modeling

4.5.1 Build the topic model

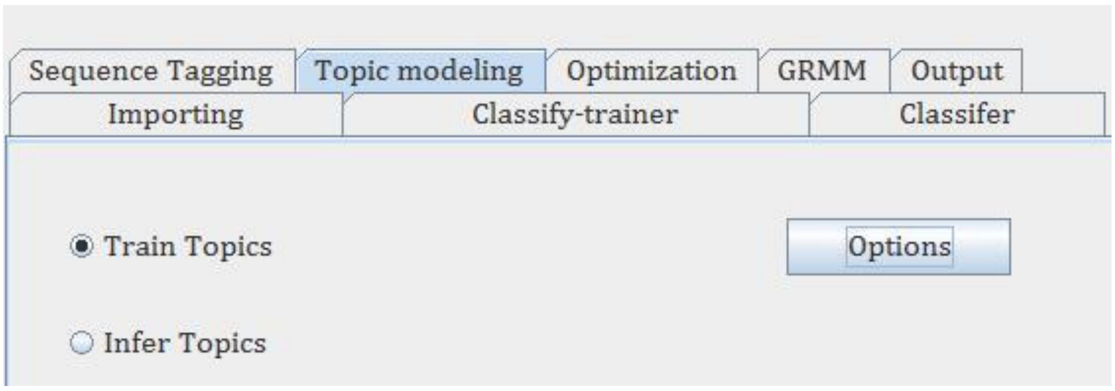
The first step is to import data and turn it into unique mallet format (.mallet), which is the same as the above. However, in order to build a topic model, one option '*Keep sequence*' must be *true*.

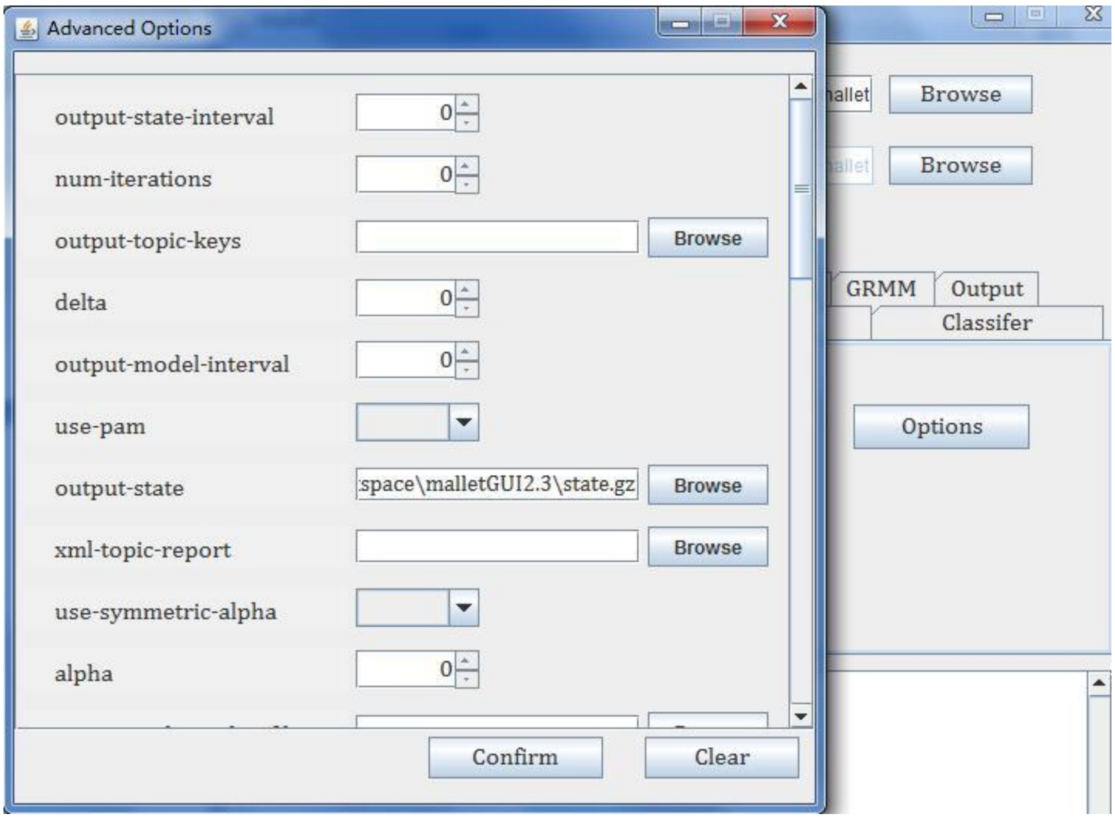


Then in '*Input Path*', the unique mallet format file should be input.

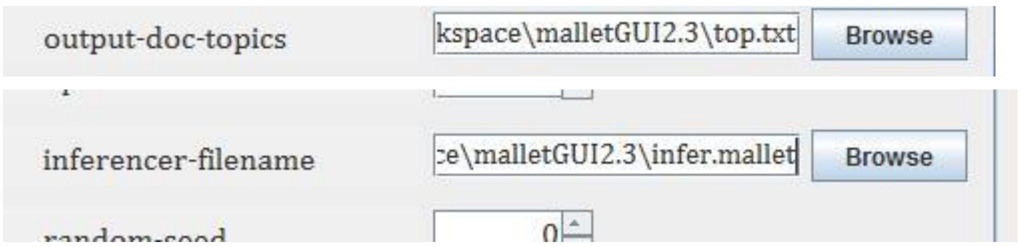

Mallet
Input Path
C:\Users\apple\workspace\malletGUI2.3
Browse
OupPut path
Browse
Select path
查找(I):
malletGUI2.3
tmp.classifier NaiveBayesTrainer
tmp.classifier NaiveBayesTrainer.trial0
tmp.classifier NaiveBayesTrainer.trial1
tmp.classifier NaiveBayesTrainer.trial2
tmp.mallet
tmpls.txt
topic.mallet
web.classifier
web.classifierNaiveBayesTrainer.trial0
web.classifierNaiveBayesTrainer.trial1
web.mallet
ww.txt
文件名(N):
topic.mallet
文件类型(T):
所有文件
OK
取消

Select ‘*Train Topics*’ to build the topic model, and then click the button ‘*Options*’ to complete the output, such as *output-state*, *output-doc-topics*, *inference-filename* and etc.



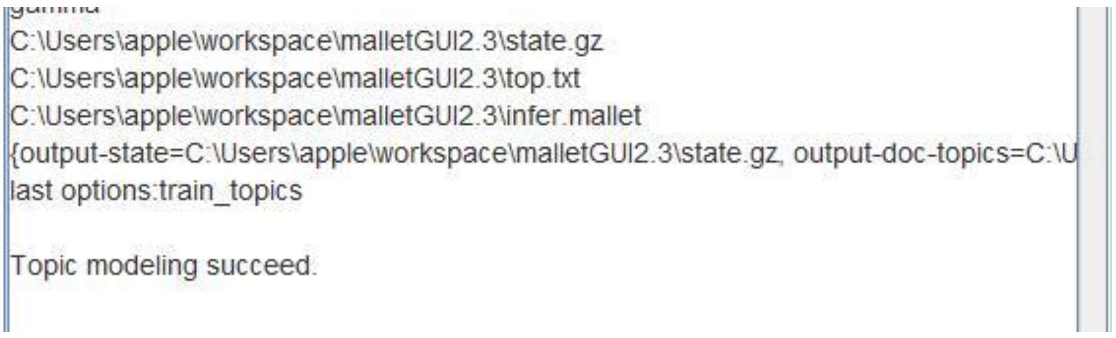


output-state-interval	0
num-iterations	0
output-topic-keys	
delta	0
output-model-interval	0
use-pam	
output-state	space\malletGUI2.3\state.gz
xml-topic-report	
use-symmetric-alpha	
alpha	0



output-doc-topics	kspace\malletGUI2.3\top.txt
inferencer-filename	ce\malletGUI2.3\infer.mallet

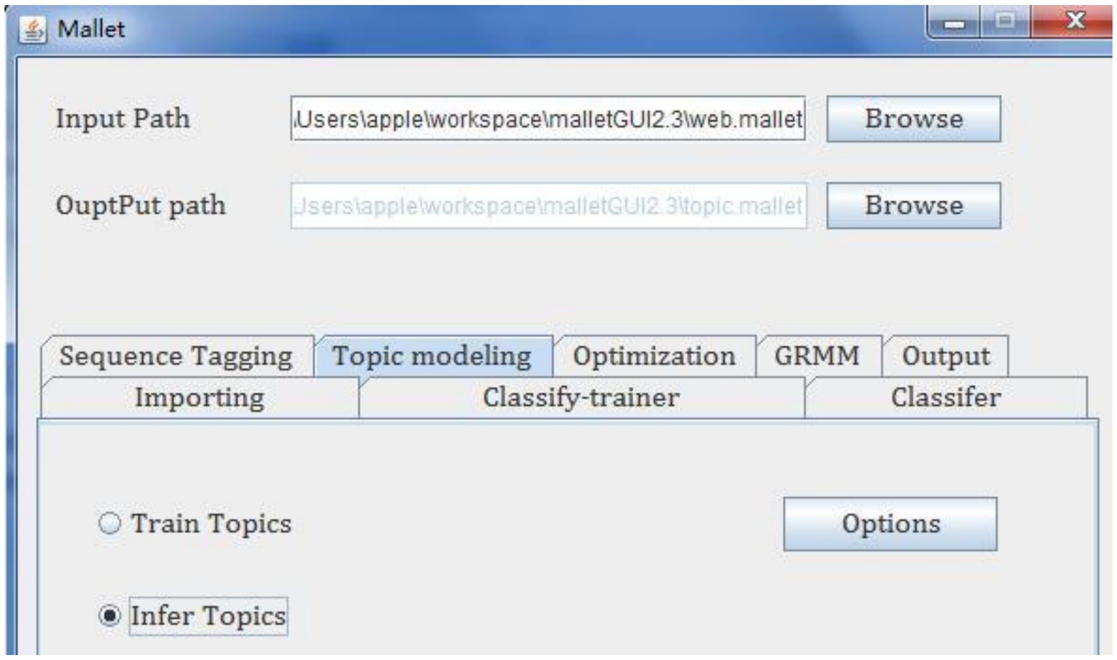
The output can be seen.



```
gamma
C:\Users\apple\workspace\malletGUI2.3\state.gz
C:\Users\apple\workspace\malletGUI2.3\top.txt
C:\Users\apple\workspace\malletGUI2.3\infer.mallet
{output-state=C:\Users\apple\workspace\malletGUI2.3\state.gz, output-doc-topics=C:\U
last options:train_topics

Topic modeling succeed.
```

4.5.2 Model inference

Predict the unknown text topics through the established model. The unknown text should be in the unique mallet format. Select '*Infer Topics*' to implement model inference and then click the button '*Options*' to complete the output including selecting which model to be used to predict the text.



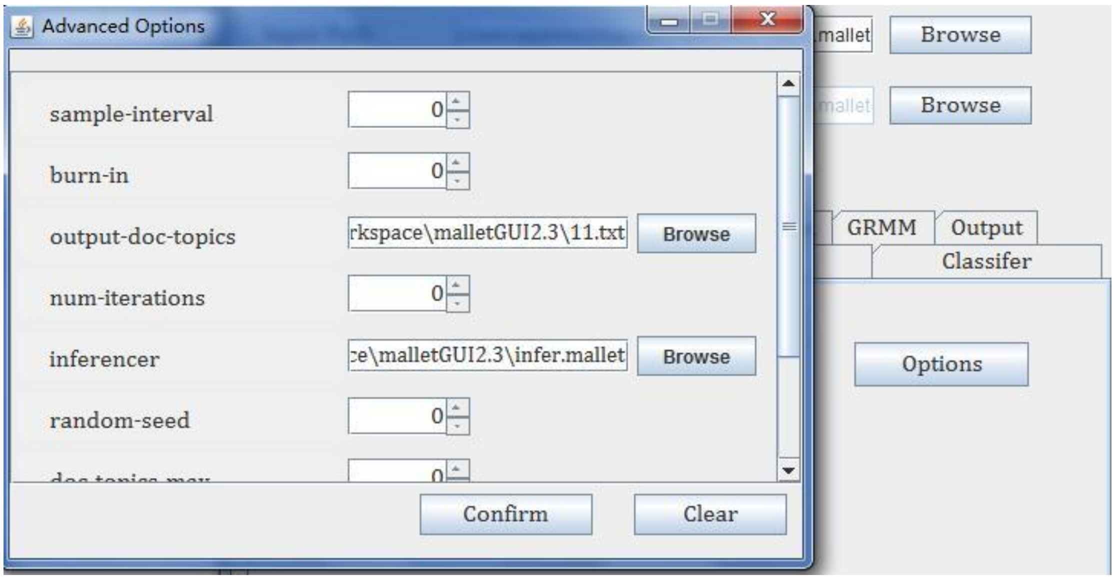
Input Path: \Users\apple\workspace\malletGUI2.3\web.mallet

OuptPut path: Jsers\apple\workspace\malletGUI2.3\topic.mallet

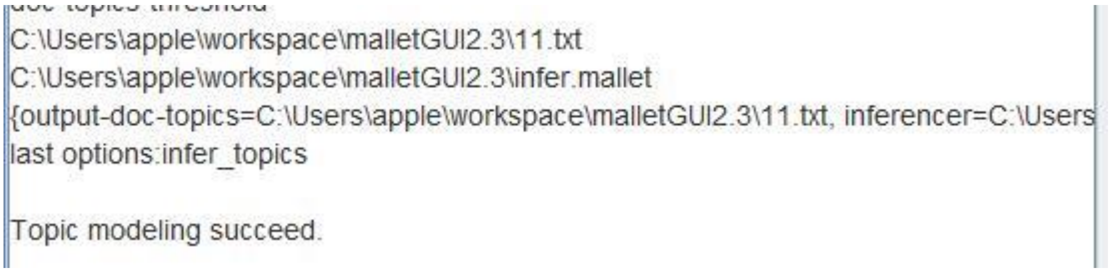
Selected tab: Topic modeling

( ) Train Topics

(●) Infer Topics



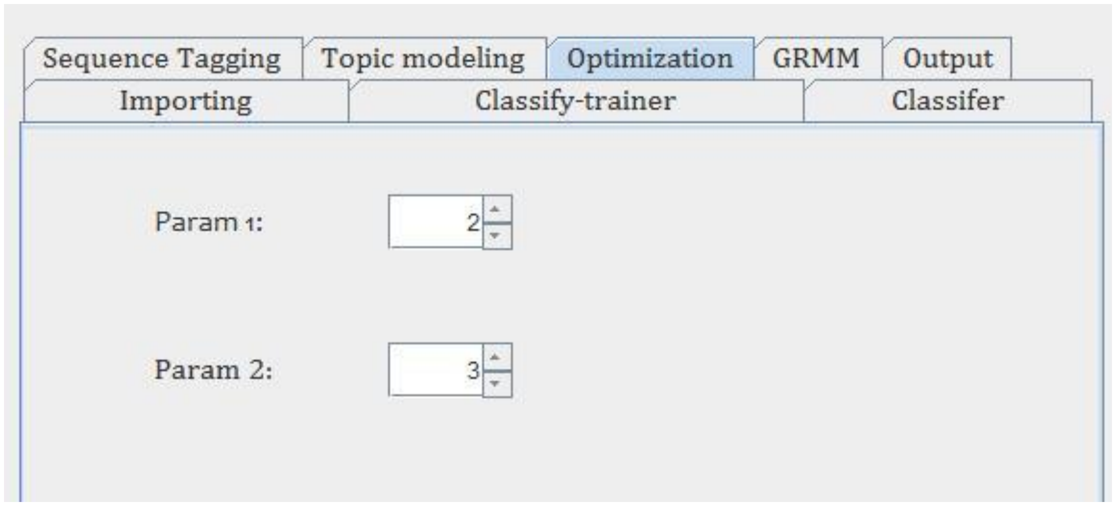
The output can be seen.



```
C:\Users\apple\workspace\malletGUI2.3\11.txt
C:\Users\apple\workspace\malletGUI2.3\infer.mallet
{output-doc-topics=C:\Users\apple\workspace\malletGUI2.3\11.txt, inferencer=C:\Users
last options:infer_topics

Topic modeling succeed.
```

4.6 Optimization

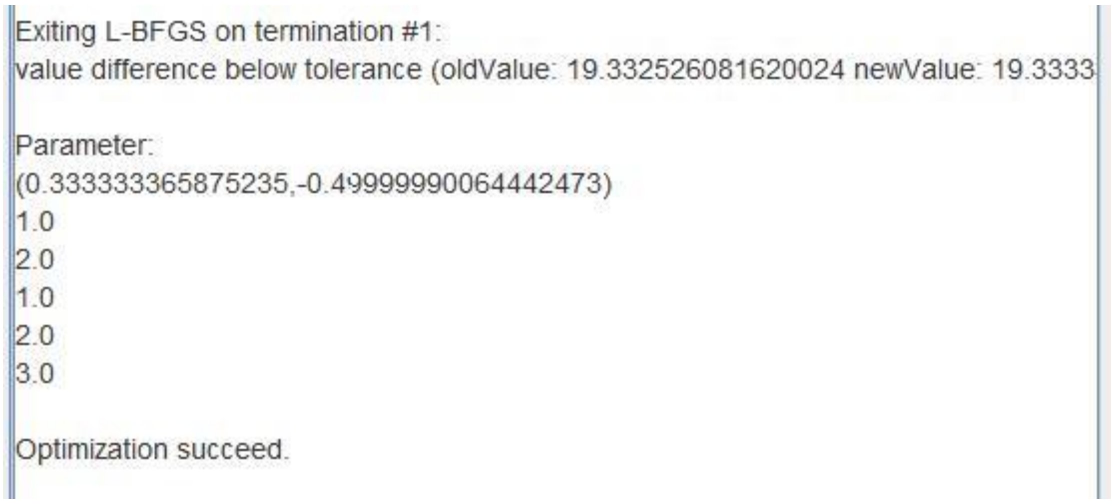
Optimization needs a specific environment which is used in the above functions. In this section, optimization is a demo rather than a solution. Users only need to input the two parameters (*Param1* and *Param2*) to implement the function.



Param 1: 2

Param 2: 3

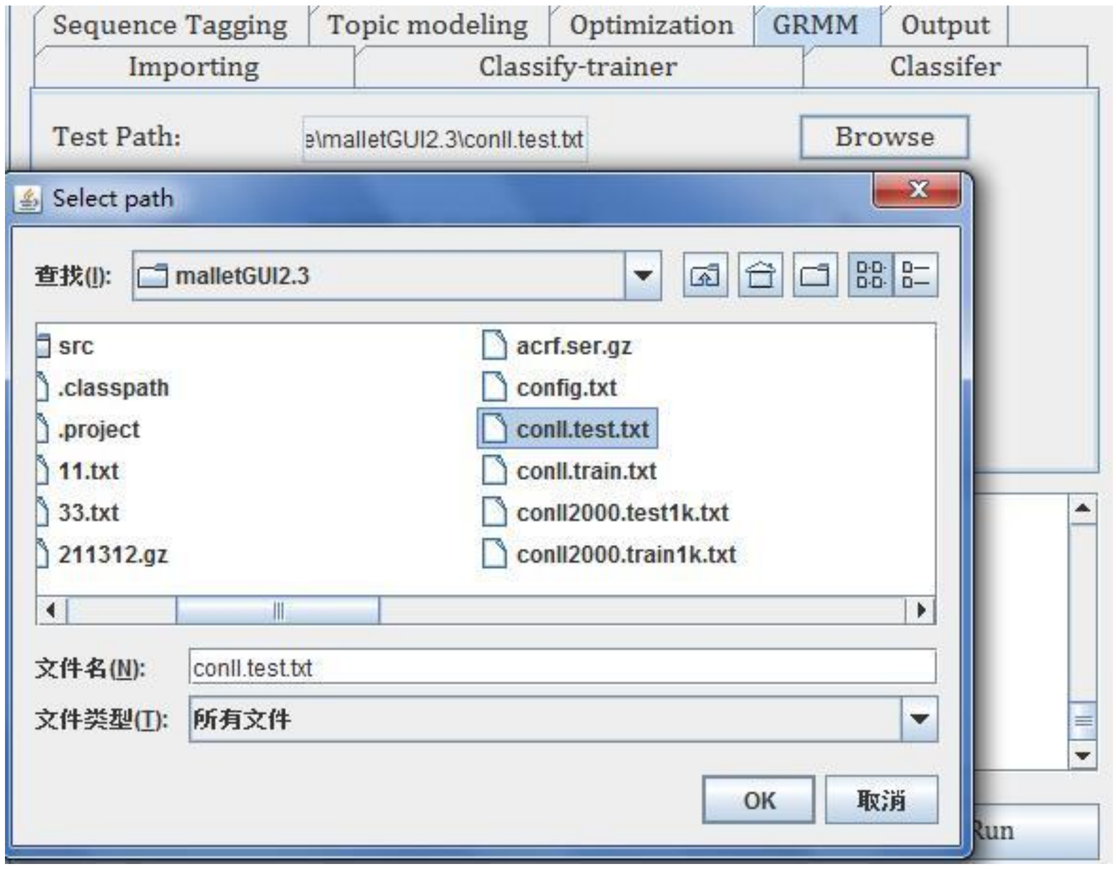
The output can be seen.



```
Exiting L-BFGS on termination #1:
value difference below tolerance (oldValue: 19.332526081620024 newValue: 19.3333

Parameter:
(0.333333365875235,-0.49999990064442473)
1.0
2.0
1.0
2.0
3.0

Optimization succeed.
```

4.7 Graphical Models in MALLET

Test Path, *Train Path* and *Model Path* should be selected by clicking the button ‘*Browse*’. The format for all files must conform to provisions. And the official website has the examples of the files.



Test Path: e\malletGUI2.3\conll.test.txt

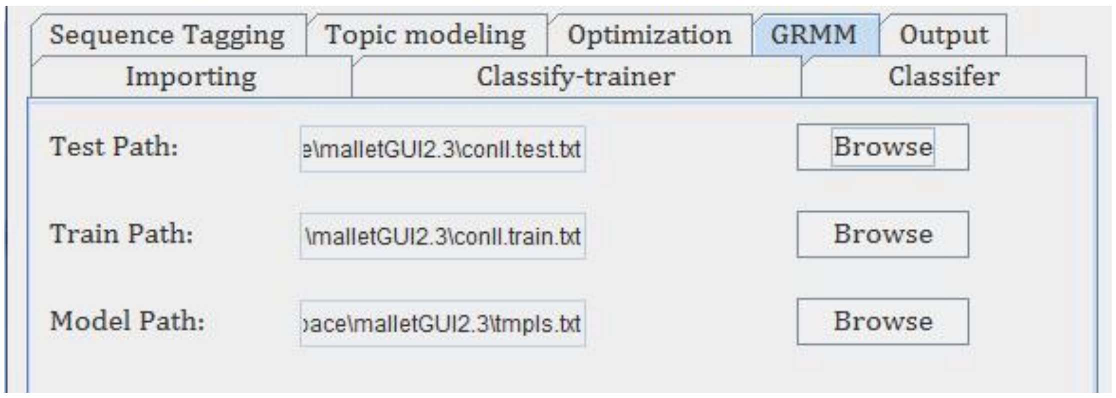
Select path

查找(I): malletGUI2.3

src
.classpath
.project
11.txt
33.txt
211312.gz
acrf.ser.gz
config.txt
conll.test.txt
conll.train.txt
conll2000.test1k.txt
conll2000.train1k.txt

文件名(N): conll.test.txt

文件类型(T): 所有文件

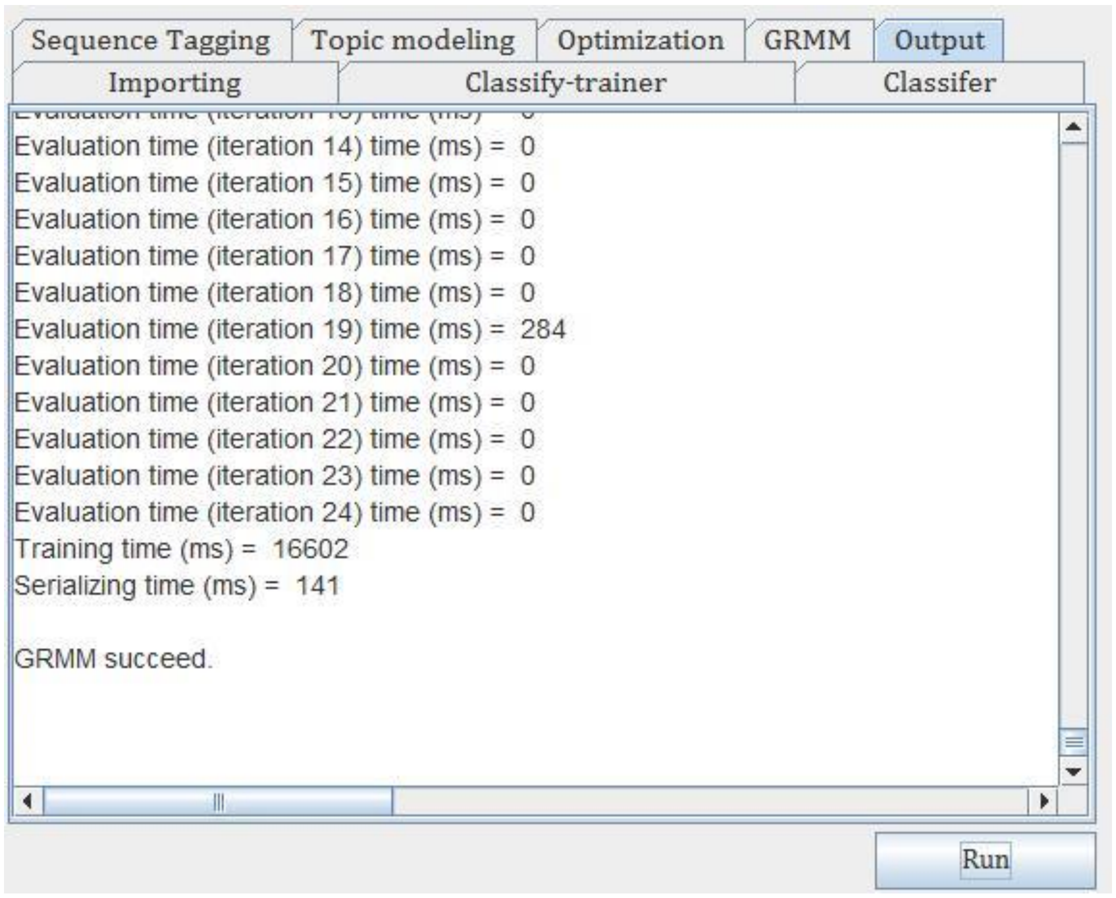


Test Path: e\malletGUI2.3\conll.test.txt

Train Path: \malletGUI2.3\conll.train.txt

Model Path: pace\malletGUI2.3\tmpls.txt

The output can be seen.



```
Evaluation time (iteration 14) time (ms) =  0
Evaluation time (iteration 15) time (ms) =  0
Evaluation time (iteration 16) time (ms) =  0
Evaluation time (iteration 17) time (ms) =  0
Evaluation time (iteration 18) time (ms) =  0
Evaluation time (iteration 19) time (ms) =  284
Evaluation time (iteration 20) time (ms) =  0
Evaluation time (iteration 21) time (ms) =  0
Evaluation time (iteration 22) time (ms) =  0
Evaluation time (iteration 23) time (ms) =  0
Evaluation time (iteration 24) time (ms) =  0
Training time (ms) =  16602
Serializing time (ms) =  141

GRMM succeed.
```