Textual Lie Detection
User Manual

Your guide to their lies.

Team EPO

Contents

1 Introduction

Lawyered, a textual lie detection system, aims to classify the truth of a statement based around properties of text. The system learns to identify the truthfulness of statements based on labelled training data. A variety of machine learning algorithms can be used to perform classification. These include Decision Tree, Random Forest and Naive Bayes. Once the system is trained, statements can be classified easily. This manual will guide you through the installation and use of Lawyered.

2 Installation Guide

Lawyered is written using Python, a cross platform interpreted language. As a result, Lawyered will run on Windows, OSX and Linux based operating systems. Instructions for Linxu will be presented below, but other operating systems should be similar.

As a reference, the following packages are required:

- Python 2.7 - http://www.python.org/download/releases/2.7/
- Natural Language Toolkit (NLTK) - http://nltk.org/
- Scikit-Learn - http://scikit-learn.org/stable/install.html

On a Debian based Linux operating system, the following commands can be used to install the above packages (you will require sudo access).

```
sudo apt-get install python2.7 python-scikits-learn python-nltk
```

Once these packages have been installed, the only other requirement is the installation of text processing libraries. This is necessary in order for NLTK to be able to process text. From a command line interface run the following:

```
$ python2.7
>>> import nltk
>>> nltk.download()
```

This final command will bring up a graphical display that allows for the download of language packages. Select all packages and choose download. The files are not particularly large, but the download may take some time.

Once this is completed you should be ready to run Lawyered.

3 Software Usage

The software is run through a command line interface. The following operations can be performed:

- Train a classifier on a supplied dataset
- Load a previously trained classifier
- Save a classifier that has been trained
- Classify a set of instances as "Truth" or "Lie"
- Display help information

Help information can be viewed by running the Lawyered program with a -h or –help argument:

```
$ python2.7 lawyered.py -h
```

This will display the following information which be used to perform training and tests:

```
usage:  lawyered.py [-h] [-t trainFiles [trainFiles ...]]
                    [-f fileList [fileList ...]]  [-s classifierSaveFile]
                    [-l classifierLoadFile] [-c learningAlgorithm]
                    [fileList [fileList ...]]

Run lie detection on some files

Positional arguments:
  fileList          the list of files with statements to classify

optional arguments:
  -h, --help        show this help message and exit
  -t trainFiles [trainFiles ...]
                    A list of files upon which to train classifiers
  -f fileList [fileList ...]
                    the list of files with statements to classify
  -s classifierSaveFile
                     Where to save the produced classifiers
  -l classifierLoadFile
                        Where to load the classifiers (overrides -t argument)
                        [WARNING: NEVER LOAD UNTRUSTED CLASSIFIERS]
  -c learningAlgorithm  Select the type of learning algorithm
                        {NaiveBayes,RandomForest, ClassificationTree}
```

3.1 Examples

Training a RandomForest classifier on a labelled dataset "examples.txt" and saving this classifier model to "classifier.model".

```
$ python2.7 lawyered.py -t examples.txt -s classifier.model
```

Training a NaiveBayes classifier on a labelled dataset "examples.txt" and save this classifier model to "classifier.model".

```
$ python2.7 lawyered.py -t examples.txt -s classifier.model -c NaiveBayes
```

Load an existing classifier "classifier.model" and classify examples found in "unlabelled.txt" using a RandomForest classifier.

```
$ python2.7 lawyered.py -l classifier.model -f unlabelled.txt
```

4 File Format

Labelled and unlabelled examples must follow a strict format in order for the data to be read and classified. This should be as follows:

```
"Text", Label = {Truth, Lie, Unknown}
```

- Text must be surrounded by quotation marks, e.g. "text to classify"
- The label can be either Truth, Lie or Unknown