Contents

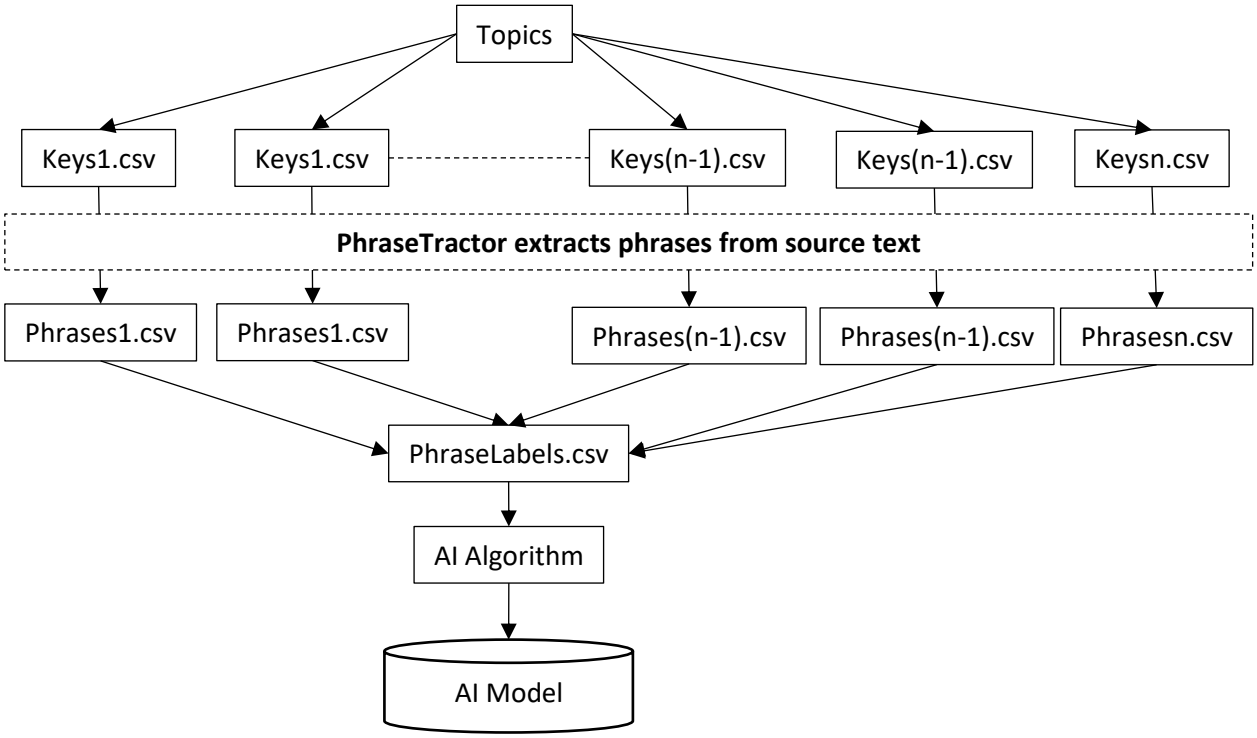
How to PhraseTractor

The Workflow in a Nutshell

PhraseTractor is a Java based command line tool that implements Information Retrieval queries, such as:

- phrases
- Regular Expression or
- Relaxed Query

to label text data for machine learning algorithms. Consider the below graphic as an overview on the involved workflow for extracting relevant phrases using the PhraseTractor tool.



The PhraseTractor tool consumes:

- one key file per topic and
- a collection of source text documents

to extract phrases that are relevant for each topic from the source collection and stores the result in each **phrase.csv** file. Each separate **phrase.csv** file can be used to train an AI model for:

- **binary** text classification, or

a union of all files (using for example Pandas or Spark) can be used to train an AI model for:

- Multiclass classification
- Multi-Label Multiclass classification

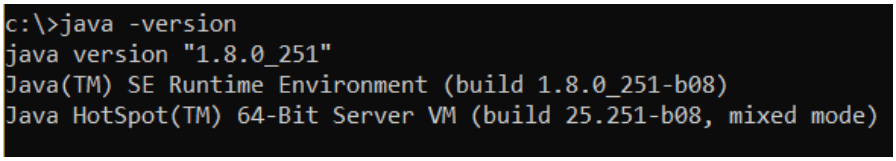
The AI model in turn can then be used to make predictions on previously unseen text.

Tool Requirements

Java Runtime

The PhraseTractor tool is written in Java and requires, therefore, an installed version of Java on Windows, Mac, or Linux. Whether Java is installed can be checked on Windows with an MS-DOS command line prompt:

- Click Windows Start Button, type CMD, and click on the CMD item to start a CMD prompt
- Type '`java -version`' (without quotes)
- You are good to go if your output is like the below screenshot



JAR File

The Java tool is distributed within a **`PhraseTractor.jar`** file. Please make sure this file is present on your file system before you continue further below.

Sample Data

Directory File Structure

The table below shows the directory file structure that is assumed to follow this guide.

The directory cell is left empty for those 2 files which are at the top level of the project directory (eg.: 'C:\tmp\test\Sentiment\'). All other directories, for example 'data', are sub-directories (e.g.: 'C:\tmp\test\Sentiment\data\') of the previously assumed root directory.

Directory	File Content (if any)
data	Sentences.csv
keys	NegativeSentiment.config NegativeSentiment.csv PositiveSentiment.config PositiveSentiment.csv
topics	
	GenTopicsFromKeys.bat PhraseTractor.jar

You can copy the sample files to the stated root directory, or you can copy them to any other directory. Please be sure to review and adjust the **Set Path** statements (as needed) in the **GenTopicsFromKeys.bat** file in the latter case.

Sample Text Collection – Sentences.csv

The content of a sample text collection file **Sentences.csv** is shown in the below cell. Lets assume for the purpose of this guide, that we want to do a sentiment analysis in which we want to identify sentences with a positive or negative sentiment. For this purpose, we assume the below text collection of around 40 sentences and all other files listed further below.

```
RowId|text
    1|Good case, Excellent value.
    2|Great for the jawbone.
    3|The mic is great.
    4|Needless to say, I wasted my money.
    5|What a waste of money and time!.
    6|And the sound quality is great.
    7|Very good quality though
    8|I advise EVERYONE DO NOT BE FOOLED!
    9|So Far So Good!.
   10|Works great!.
   11|The commercials are the most misleading.
   12|Great Pocket PC / phone combination.
   13|Doesn't hold charge.
   14|It has kept up very well.
   15|Poor Talk Time Performance.
   16|The case is great and works fine with the 680.
   17|worthless product.
   18|I was not impressed by this product.
   19|Nice headset priced right.
   20|I only hear garbage for audio.
   21|Excellent bluetooth headset.
   22|It has all the features I want
   23|This case seems well made.
   24|Disappointed with battery.
   25|I love this thing!
   26|VERY DISAPPOINTED.
   27|The buttons for on and off are bad.
   28|Buy a different phone - but not this.
   29|Great for iPODs too.
   30|This device is great in several situations
   31|Mic Doesn't work.
   32|Great choice!
   33|Nice docking station for home or work.
   34|This is a beautiful phone.
   35|Love this product.
   36|The battery runs down quickly.
   37|This phone works great.
   38|The phone loads super!
   39|Made very sturdy.
   40|Worked great!.
```

Sample Keys for Positive Sentiment – PositiveSentiment.csv

This sample key file shows key words that are relevant for sentences with a positive sentiment. That is, the PhraseTractor tool searches the listed keyword definitions and extracts phrases with a given pattern as sample phrases with a positive sentiment. The definitions in the regex column can include the following values:

regex column value	Description
0	This will extract a given phrase if it contains the exact case-insensitive keyword. That is, a definition like: **0|excellent** will match and extract the phrases: - 'This was an excellent choice.' - 'I FELT SO EXCELLENT' but it will not match the following phrase (since 'excellent' is misspelled): - 'An excelent product it is.'
1	This will extract a given phrase if it contains the exact case-insensitive keyword matched with a regular expression. That is, a definition like: **1|no problem[a-z]** will be matched on phrases like: - 'This was no problem.' - 'We had no problems following this guide.'
2	This loose matching function will extract a given phrase if it contains the exact case-insensitive keywords in any order. That is, a definition like: **2|great work** will be matched on phrases like: - 'It is **great** to **work** with PhraseTractor.' - 'This **work** is **great**.'

```
regex|text
0|excellent
0|great
0|very good
0|So Far So Good
0|good case
0|Works great
0|very well
0|Nice
0|Excellent
0|well made
1|I love[a-z]
0|beautiful
0|loads super
0|very sturdy
```

Sample Configuration file - PositiveSentiment.config

This file contains parameter settings that tell PhraseTractor how to process each key file for each topic.

Parameter	Description
TextFile	Is the complete or relative path from the config file to the text file which contains phrases that should be extracted based on the definitions in the key file.
RowIDColumnName	Is the column that should be used to Id a given text. The tool can either us an ID column that is already present in the text file, or generate a RowId on the fly, in which case this setting can be left empty. 'RowIDColumnName='
TextColumnName	Is the name of the column in the text file which contains the text that should be extracted based on key file definitions.
KeyFile	Is the complete or relative path from the config file to the key file.
WordPairFrequency	This can either be 'true' or 'false' depending on whether a list of word pairs (sorted by frequency) should be generated to support text analysis based on word pair frequency.
RegexMask	This is a regular expression mask that defines the letters that are part of a word. The below definitions includes German words including the '!' character to support words like 'very!cool' (in case this occurs in the source text and is needed for extraction).

```
TextFile          = ..\data\Sentences.csv
RowIDColumnName   = RowId
TextColumnName    = text
KeyFile           = .\PositiveSentiment.csv
OutputDir         = ..\Topics\
WordPairFrequency = false
RegexMask         = [^a-zA-Z0-9-äüöÄÜÖß_#]+
```

Sample Keys for Negative Sentiment – NegativeSentiment.csv

This sample key file shows key words that are relevant for sentences with a negative sentiment. Details are the same as mentioned above for the positive sentiments key file.

```
regex|text
2|wasted money
2|waste time
0|NOT BE FOOLED
0| misleading
0|poor
0|worthless
0|not impressed
0|garbage
0|disappointed
0|bad
0|doesn't work
```

Sample Configuration file - NegativeSentiment.config

This file contains parameter settings that tell PhraseTractor how to process each key file for each topic. Details are as explained for the positive sentiment file.

```
TextFile          = ..\data\Sentences.csv
RowIDColumnName   = RowId
TextColumnName    = text
KeyFile           = .\NegativeSentiment.csv
OutputDir         = ..\Topics\
WordPairFrequency = true
RegexMask         = [^a-zA-Z0-9-äüöÄÜÖß_#]+
```

Sample Batch File - GenTopicsFromKeys.bat

This is a sample batch file that shows a simple way of automating the processing of each topic in order to output the files in the topic definition files into the **topics** folder (but you can use any other form of suitable batch automation (eg.: Unix shell or PowerShell) ,if you prefer something else).

```
@ECHO OFF
ECHO.
SET TOOLPATH=C:\tmp\test\Sentiment\
SET DATAPATH=C:\tmp\test\Sentiment\
ECHO.
ECHO Running PhraseTractor
ECHO.
REM
REM Generating Sample Texts from Key Phrases via config for each topic
REM
java -cp %TOOLPATH%PhraseTractor.jar PhraseTractor -config %DATAPATH%Keys\NegativeSentiment.config
java -cp %TOOLPATH%PhraseTractor.jar PhraseTractor -config %DATAPATH%Keys\PositiveSentiment.config
```

The above code allows us to freely move folders around if paths in the config file are relative to the config file and paths in the above SET statements are correct.

Expected Output

Executing this batch file should produce a similar output like this:

```
C:\tmp\test\Sentiment>GenTopicsFromKeys.bat


Running PhraseTractor



    PhraseTractor: 1.0.0.0
  Properties from: 'C:\tmp\test\Sentiment\Keys\PositiveSentiment.config'
         TextFile: 'C:\tmp\test\Sentiment\data\Sentences.csv'
  RowIDColumnName: 'RowId'
   TextColumnName: 'text'
          KeyFile: 'C:\tmp\test\Sentiment\keys\PositiveSentiment.csv'
        OutputDir: 'C:\tmp\test\Sentiment\topics\'
WordPairFrequency: 'false'
        RegexMask: '[^a-zA-Z0-9-äüöÄÜÖß_#]+'
       Reading Key File: 'C:\tmp\test\Sentiment\keys\PositiveSentiment.csv'
             keys found: 13

             Processing 40 data rows from 'Sentences.csv'
           Index build: Successful
      Documents parsed: 40
    Unique words found: 111
 Regex Document Matches: 0
Writting 22 documents retrieved via keyword queries into 'C:\tmp\test\Sentiment\topics\PositiveSentiment_DocsWithKeywords.csv' file.
Writting all other documents without match via keyword queries into 'C:\tmp\test\Sentiment\topics\PositiveSentiment_NoMatchDocsWithKeywords.csv' file.
Elapsed processing time was: 0:00:00


    PhraseTractor: 1.0.0.0
  Properties from: 'C:\tmp\test\Sentiment\Keys\NegativeSentiment.config'
         TextFile: 'C:\tmp\test\Sentiment\data\Sentences.csv'
  RowIDColumnName: 'RowId'
   TextColumnName: 'text'
          KeyFile: 'C:\tmp\test\Sentiment\keys\NegativeSentiment.csv'
        OutputDir: 'C:\tmp\test\Sentiment\topics\'
WordPairFrequency: 'true'
        RegexMask: '[^a-zA-Z0-9-äüöÄÜÖß_#]+'
       Reading Key File: 'C:\tmp\test\Sentiment\keys\NegativeSentiment.csv'
             keys found: 11

             Processing 40 data rows from 'Sentences.csv'
           Index build: Successful
      Documents parsed: 40
    Unique words found: 111
Unique word-pairs found: 140
 Regex Document Matches: 0
Writting 12 documents retrieved via keyword queries into 'C:\tmp\test\Sentiment\topics\NegativeSentiment_DocsWithKeywords.csv' file.
Writting all other documents without match via keyword queries into 'C:\tmp\test\Sentiment\topics\NegativeSentiment_NoMatchDocsWithKeywords.csv' file.
Elapsed processing time was: 0:00:00

C:\tmp\test\Sentiment>
```

…and there should be 4 topic files per Topic. This guide lists and describes the output files for the positive sentiment definitions while the tool should produce similar files for the negative sentiment text definitions as well.

Expected contents in the **Sentiment\topics** folder:

File	Description
NegativeSentiment**_AllWord_Remaining_Words.csv**	Contains a list of unique words in the original text collection (Sentences.csv) and whether they are: - 1 part of this topic or - 0 not as indicated by the value in the first column. This file is useful to include key words including frequent typos since the list of words is sorted alphabetically.
NegativeSentiment**_DocsWithKeywords.csv**	Contains all sentences that were successfully extracted based on the given topic definition. This file is result file that should be used for training in a machine learning algorithm.

NegativeSentiment**_Keyword_Frequs.csv**	Contains the frequency for each key word definition that was matched for a given topic.
NegativeSentiment**_NoMatchDocsWithKeywords.csv**	Contains all documents that were **not matched** for a given topic. This file can typically be used to explore more topic relevant phrases when a given definition is already available.
PositiveSentiment**_AllWord_Remaining_Words.csv**	See detailed explanation for similar negative sentiment files given above.
PositiveSentiment**_DocsWithKeywords.csv**	
PositiveSentiment**_Keyword_Frequs.csv**	
PositiveSentiment**_NoMatchDocsWithKeywords.csv**	
Sentences_AllWord_Frequs.csv	Contains frequencies of all single words in the source text collection (Sentences.csv).
Sentences_AllWord_WordPair_Frequs.csv	Contains frequencies of all word pairs in the source text collection (Sentences.csv).