jMWE - Multiword Expressions Detector Library
User's Guide

Version 1.0.*

Mark A. Finlayson & Nidhi Kulkarni

September 26, 2011

1 Purpose of the Software

jMWE is a Java library for constructing and testing Multi-Word Expression detectors. A Multi-Word Expression (MWE) is a group of words that (1) occurs together more often than would be expected by pure chance and, (2) is arbitrarily restricted with regard to their syntactic or semantic flexibility. Examples of common MWEs are compound nouns such as *world record* or verb-particle constructions such as *look up*, as, for example in the sentence:

(1) She looked up the world record.

The library has three main facilities: (1) a detector API, (2) a MWE index facility, and (3) a test harness. The detector API defines a detector interface which provides a single method for detecting MWE tokens in a list of individual tokens; anyone interested in taking advantage of jMWEs testing infrastructure or writing their own MWE token detection algorithm need only implement this interface. jMWE provides several baseline MWE token detection strategies. Also provided are detector filters, which apply a specific constraint to or resolve conflicts in the output another detector. The MWE index provides classes for constructing, storing, and accessing indices of valid MWE types. An MWE index allows an algorithm to retrieve a list of MWE types given a single word token and part of speech. The index also lists how frequently, in a particular concordance, a set of tokens appears as a particular MWE type rather than as independent words. The test harness allows one to run an MWE detector over a given corpus and measure its precision and recall. The library has no GUI elements.

The freely available version of jMWE is licensed for use for **non-commercial purposes only**, as long as proper acknowledgement is made. Details can be found in the license, which is included at the end of this document. The copyright on the software is owned by MIT; if you wish to use the software for commercial purposes, please contact the MIT Technology Licensing Office for more information on how to obtain a commercial license.

1.1 Citing jMWE

If you have used jMWE in your work and wish to cite a published paper, please cite either of the following two papers. Links to these papers may be found on the jMWE website.

- Finlayson, Mark A. and Kulkarni, Nidhi (2011) "Detecting Multi-Word Expressions improves Word Sense Disambiguation", in Proceedings of the 2011 Workshop on Multiword Expressions, pp. 20-24, held at ACL'2011 in Portland, OR.
- Kulkarni, Nidhi and Finlayson, Mark A. (2011) "jMWE: A Java Toolkit for Detecting Multi-Word Expressions" in Proceedings of the 2011 Workshop on Multiword Expressions, pp 122-124, held at ACL'2011 in Portland, OR.

1.2 Software Dependencies

For some functionality jMWE depends on external software packages that are not included in the standard Java Runtime Environment JRE. In particular, some classes in the library depend on one or more of the following libraries. Individual classes that depend on these external libraries will say so in their documentation. Listed here are the versions of the external libraries with which jMWE 1.0.* was compiled - earlier or later versions may also work, depending on API changes in those libraries.

1. MIT Java Wordnet Interface (JWI) v2.1.5
   `http://projects.csail.mit.edu/jwi`

2. JSemcor v1.0.0
   `http://projects.csail.mit.edu/jSemcor`

3. Stanford POS Tagger v2010-05-26
   `http://nlp.stanford.edu/software/tagger.shtml`

2 Getting Started

The main interface for detecting MWEs is `edu.mit.jmwe.detect.IMWEDetector`. There are numerous implementations of this interface included in the library, which fall into three types: (1) Basic Detectors, (2) Filters, and (3) Resolvers.

Basic Detectors: These types of detectors use an MWE index, or other source of information, to detect MWE tokens in a stream of tokens. jMWE includes several implementations of basic detectors, including the following:

1. Exhaustive - Given a MWE type index, finds all possible MWE tokens regardless of inflection, order,or continuity.
2. Continuous - Given a MWE type index, finds all MWE tokens whose constituent tokens occur without other tokens interspersed.
3. Simple Proper Noun - Finds all continuous sequences of proper noun tokens, and marks them as proper noun MWE tokens.

Filters: These MWE detectors filter to the output of another, wrapped, detector. Detectors which function as filters will implement the `edu.mit.jmwe.detect.IMWEDetectorFilter` interface. Examples of implemented filters are:

1. In Order - Only returns MWE tokens whose constituent tokens are in the same order as the constituents listed in the MWE types definition.
2. No Inflection - Removes inflected MWE tokens.
3. Observed Inflection - Returns base form MWEs, as well as those whose inflection has been observed in a specified concordance. For the standard MWE index data file distributed with jMWE, this corpus is Semcor 1.6
4. Pattern Inflection - Only return MWE tokens whose inflection matches a pre-defined set of part of speech patterns.

Resolvers: These MWE detectors resolve conflicts in the output of another, wrapped, detector. Detectors which function as filters will implement the `edu.mit.jmwe.detect.IMWEDetectorResolver` interface. Examples of implemented resolvers are:

1. Longest-Match-Left-to-Right - For a set of conflicting MWE tokens, picks the one that starts earliest. If all of the conflicting MWE tokens start at the same point, picks the longest.
2. Observed Probability - For a set of conflicting MWE tokens, picks the one whose constituents have most often been observed occurring as an MWE token rather than as isolated words. For the standard MWE index data file distributed with jMWE, this corpus is Semcor 1.6

3. Variance Minimizing: For a set of conflicting MWE tokens, picks the MWE token with the fewest interstitial spaces.

An example of constructing an `IMWEIndex` object and running a simple detector with it can be found in Listing 1 in the `simpleDetectorExample()` method. In that method, the first line of code (5) retrieves a file handle pointing to the location of the standard MWE index data file, which is distributed with jMWE. Wherever this file is located on your system, have this method return that file. The second block of code, two lines long (8-9), constructs an instance of the default MWEIndex object, and opens it by calling the `open()` method. On line 12, we construct a simple detector that uses the index. In the next block of eight lines (16-23) we construct a simple sentence on which to run the detector. In the final block (26-28) we run the detector on the sentence and print out the results. Listing 2 shows the console output of the method.

```
1  public void simpleDetectorExample() throws IOException {
2
3      // get handle to file containing MWE index data,
4      // e.g., mweindex_wordnet3.0_Semcor1.6.data
5      File idxData = getMWEIndexDataFile();
6
7      // construct an MWE index and open it
8      IMWEIndex index = new MWEIndex(idxData);
9      index.open();
10
11     // make a basic detector
12     IMWEDetector detector = new Consecutive(index);
13
14     // construct a test sentence:
15     // "She looked up the world record."
16     List<IToken> sentence = new ArrayList<IToken>();
17     sentence.add(new Token("She",    "DT"));
18     sentence.add(new Token("looked", "VBD", "look"));
19     sentence.add(new Token("up",     "RP"));
20     sentence.add(new Token("the",    "DT"));
21     sentence.add(new Token("world",  "NN"));
22     sentence.add(new Token("record", "NN"));
23     sentence.add(new Token(".",      "."));
24
25     // run detector and print out results
26     List<IMWE<IToken>> mwes = detector.detect(sentence);
27     for(IMWE<IToken> mwe : mwes)
28         System.out.println(mwe);
29 }
```

Listing 1: Basic use of jMWE

```
look_up_V={looked_VBD,up_RP}
world_record_N={world_NN,record_NN}
```

Listing 2: Output of the `simpleDetectorExample()` method in Listing 1 for the standard index included in the jMWE distribution

A more complex example is shown in Listing 3 in which we instantiate two compound detectors and compare their results over a longer sentence. Listing 4 shows the console output of the method.

```java
public void complexDetectorExample(IMWEIndex alreadyOpenIndex){

    // make some detectors
    IMWEDetector pnDetector = ProperNouns.getInstance();
    IMWEDetector exhaustive = new TrulyExhaustive(alreadyOpenIndex);
    IMWEDetector goodDetector = new MoreFrequentAsMWE(new
       InflectionPattern(new Consecutive(alreadyOpenIndex)));

    // construct two detectors to test
    IMWEDetector d1 = new CompositeDetector(pnDetector, exhaustive);
    IMWEDetector d2 = new CompositeDetector(pnDetector, goodDetector);

    // construct test sentence:
    // "Ben Bitdiddle watched as the ship passed out of sight."
    List<IToken> sentence = new ArrayList<IToken>();
    sentence.add(new Token("Ben",       "NNP"));
    sentence.add(new Token("Bitdiddle", "NNP"));
    sentence.add(new Token("watched",   "VBD", "watch"));
    sentence.add(new Token("as",        "IN"));
    sentence.add(new Token("the",       "DT"));
    sentence.add(new Token("ship",      "NN"));
    sentence.add(new Token("passed",    "DT", "pass"));
    sentence.add(new Token("out",       "IN"));
    sentence.add(new Token("of",        "IN"));
    sentence.add(new Token("sight",     "NN"));
    sentence.add(new Token(".",         "."));

    // run bad detector and print out results
    System.out.println("\nBad Detector:");
    for(IMWE<IToken> mwe : d1.detect(sentence))
        System.out.println(mwe);

    // run good detector and print out results
    System.out.println("\nGood Detector:");
    for(IMWE<IToken> mwe : d2.detect(sentence))
        System.out.println(mwe);
}
```

Listing 3: A more complex example

```
Bad Detector:
ben_bitdiddle_P={Ben_NNP,Bitdiddle_NNP}
watch_out_V={watched_VBD,out_IN}
pass_out_V={passed_DT,out_IN}
out_of_sight_J={out_IN,of_IN,sight_NN}
out_of_sight_R={out_IN,of_IN,sight_NN}
out_of_O={out_IN,of_IN}
out_of_R={out_IN,of_IN}

Good Detector:
ben_bitdiddle_P={Ben_NNP,Bitdiddle_NNP}
out_of_sight_J={out_IN,of_IN,sight_NN}
```

Listing 4: Output of the `complexDetectorExample()` method in Listing 3 for the standard index data.

3 Generating an MWE Index file

For basic use of jMWE, there is no need to generate your own index data file – that is, the file that contains the data which backs the default implementations of the `IMWEIndex` interface – you can download the standard data file (called *mweindex_wordnet3.0_Semcor1.6.data*) from the jMWE website. This file contains all the multi-word expressions, plus inflections, extracted from Wordnet 3.0 and Semcor 1.6, as well as various statistics regarding the number of time each form is observed in the corpus.

3.1 Creating an Index data file programmatically

If you have a small number of MWEs, the easiest way to create a index data file for later use is to manually create one. This is illustrated in Listing 5. If you wish to have full control over the contents of the index, including occurrence statistics and inflected forms, use the `MWEIndex(Map)` constructor instead.

```
public void createSimpleIndex(OutputStream out) throws IOException {

    // create a list of MWE types
    List<String> mweForms = new LinkedList<String>();
    mweForms.add("cruise_ship_N");
    mweForms.add("look_up_V");
    mweForms.add("pass_out_V");
    mweForms.add("out_of_sight_R");

    // create an index and open it
    IMWEIndex index = new MWEIndex(mweForms);
    index.open();

    // this will go at the top of the index data file
    List<String> headerLines = new LinkedList<String>();
    headerLines.add("My first index data file");
    headerLines.add("Generated on " + new Date());

    // write the data file
    IndexBuilder.writeDataFile(index, out, headerLines);
}
```

Listing 5: Manual creation of a MWE index and index data file

```
// My first index data file
// Generated on Sun May 08 14:48:44 EDT 2011
cruise_ship_N 0,0,0,0,0
look_up_V 0,0,0,0,0
out_of_sight_R 0,0,0,0,0
pass_out_V 0,0,0,0,0
```

Listing 6: Output written to the stream during the `createSimpleIndex` method in Listing 5.

3.2 Creating an Index data file from Wordnet and Semcor

If you have a specific version of Wordnet and Semcor, you can use the `edu.mit.jmwe.index.IndexBuilder` class directly. This class requires both JWI and JSemcor to be on the classpath. Running the main method of that class will present you with a number of file selection prompts, during which you locate the versions of Wordnet (required) and Semcor (optional) from which you want to generate the index. If you want all the various statistics, as well as inflected forms, before running `IndexBuilder` you must generated a tagged concordance file, which can be done using the `edu.mit.jmwe.data.concordance.ConcordanceTagger`. See the javadoc of that class for more information.

4 Using the Test Harness

If you want to test the performance of your detectors, measuring their precision, recall, and f-measure over some corpus, jMWE provides facilities that assist you in doing this. The central class for this is the `edu.mit.jmwe.harness.TestHarness` class. To use this class, you need at least one detector, a set of sentences over which to run, and an `IAnswerKey` object that can tell the harness what the correct answers are for each sentence. If you want to run your detectors over Semcor (or a corpus that conforms to the Semcor format), all the necessary classes are in place. You can generate a list of POS-tagged sentences by using the `ConcordanceTagger` class in conjunction with the Stanford POS Tagger, and interface with Semcor via the JSemcor library.

```
public void runTestHarness(IMWEIndex alreadyOpenIndex, Iterator<IConcordanceSentence> ss, int numSentences, IConcordanceSet cs){

    // make detector
    IMWEDetector pnDetector = ProperNouns.getInstance();
    IMWEDetector goodDetector = new MoreFrequentAsMWE(new InflectionPattern(new Consecutive(alreadyOpenIndex)));
    IMWEDetector dt = new CompositeDetector(pnDetector, goodDetector);

    // make answer key
    IAnswerKey key = new ConcordanceAnswerKey(cs);

    // results will be reported to this object
    IResultBuilder<IConcordanceToken, IConcordanceSentence> results = new MWEResultBuilder<IConcordanceToken, IConcordanceSentence>();

    // construct and run harness
    IProgressBar pb = new ProgressBar(numSentences);
    TestHarness.getInstance().run(dt, results, ss, key, pb);

    // print out results
    System.out.println(results);
}
```

Listing 7: Using the `TestHarness` class

```
Started Sun May 08 16:03:23 EDT 2011
100% bar: .............................. (total=20138)
Progress: .............................. (count=20138, time=307.287s)
Finished Sun May 08 16:08:30 EDT 2011
POS          Found   Actual  Correct Pr         Re         F1
-------------------------------------------------------------------
ADJECTIVE    1119    1218    976     0.872      0.801      0.835
NOUN         4918    5031    4625    0.940      0.919      0.930
OTHER        302     391     225     0.745      0.575      0.649
PROPER_NOUN  4225    3755    3043    0.720      0.810      0.763
ADVERB       2575    2782    2174    0.844      0.781      0.812
VERB         3484    3418    2764    0.793      0.809      0.801
-------------------------------------------------------------------
Totals       16623   16595   13807   0.831      0.832      0.831
```

Listing 8: Results of running the `runTestHarness` method in Listing 7, using the `brown1` and `brown2` concordances of Semcor 1.6, for the standard index data distributed with jMWE.

5 Frequently Asked Questions

5.1 How do I report a bug / request a feature / heap praise upon you for making such cool software?

If you find a bug, have a feature request, or wish to send your praises, please contact me via my permanent email forwarding address markaf@alum.mit.edu.

6 License

The following is the full text of the jMWE license.

This program and the accompanying materials are made available by MIT under the terms of this license. Contact markaf@alum.mit.edu for further details, or to inquire about a commercial license.

The following constitutes the jMWE License. It is identical to the Creative Commons Attribution Version 3.0 Unported License, which can be found on the web at:

http://creativecommons.org/licenses/by/3.0/legalcode

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

1. Definitions

a. "Adaptation" means a work based upon the Work, or upon the Work and other pre-existing works, such as a translation, adaptation, derivative work, arrangement of music or other alterations of a literary or artistic work, or phonogram or performance and includes cinematographic adaptations or any other form in which the Work may be recast, transformed, or adapted including in any form recognizably derived from the original, except that a work that constitutes a Collection will not be considered an Adaptation for the purpose of this License. For the avoidance of doubt, where the Work is a musical work, performance or phonogram, the synchronization of the Work in timed-relation with a moving image ("synching") will be considered an Adaptation for the purpose of this License.

b. "Collection" means a collection of literary or artistic works, such as encyclopedias and anthologies, or performances, phonograms or broadcasts, or other works or subject matter other than works listed in Section 1(f) below, which, by reason of the selection and arrangement of their contents, constitute intellectual creations, in which the Work is included in its entirety in unmodified form along with one or more other contributions, each constituting separate and independent works in themselves, which together are assembled into a collective whole. A work that constitutes a Collection will not be considered an Adaptation (as defined above) for the purposes of this License.

c. "Distribute" means to make available to the public the original and copies of the Work or Adaptation, as appropriate, through sale or other transfer of ownership.

d. "Licensor" means the individual, individuals, entity or entities that offer(s) the Work under the terms of this License.

e. "Original Author" means, in the case of a literary or artistic work, the individual, individuals, entity or entities who created the Work or if no individual or entity can be identified, the publisher; and in addition (i) in the case of a performance the actors, singers, musicians, dancers, and other persons who act, sing, deliver, declaim, play in, interpret or otherwise perform literary or artistic works or expressions of folklore; (ii) in the case of a phonogram the producer being the person or legal entity who first fixes the sounds of a performance or other sounds; and, (iii) in the case of broadcasts, the organization that transmits the broadcast.

f. "Work" means the literary and/or artistic work offered under the terms of this License including without limitation any production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression including digital form, such as a book, pamphlet and other writing; a lecture, address, sermon or other work of the same nature; a dramatic or dramatico-musical work; a choreographic work or entertainment in dumb show; a musical composition with or without words; a cinematographic work to which are assimilated works expressed by a process analogous to cinematography; a work of drawing, painting, architecture, sculpture, engraving or lithography; a photographic work to which are assimilated works expressed by a process analogous to photography; a work of applied art; an illustration, map, plan, sketch or three-dimensional work relative to geography, topography, architecture or science; a performance; a broadcast; a phonogram; a compilation of data to the extent it is protected as a copyrightable work; or a work performed by a variety or circus performer to the extent it is not otherwise considered a literary or artistic work.

g. "You" means an individual or entity exercising rights under this License who has not previously violated the terms of this License with respect to the Work, or who has received express permission from the Licensor to exercise rights under this License despite a previous violation.

h. "Publicly Perform" means to perform public recitations of the Work and to communicate to the public those public recitations, by any means or process, including by wire or wireless means or public digital performances; to make available to the public Works in such a way that members of the public may access these Works from a place and at a place individually chosen by them; to perform the Work to the public by any means or process and the communication to the public of the performances of the Work, including by public digital performance; to broadcast and rebroadcast the Work by any means including signs, sounds or images.

i. "Reproduce" means to make copies of the Work by any means including without limitation by sound or visual recordings and the right of fixation and reproducing fixations of the Work, including storage of a protected performance or phonogram in digital form or other electronic medium.

2. Fair Dealing Rights.

Nothing in this License is intended to reduce, limit, or restrict any uses free from copyright or rights arising from limitations or exceptions that are provided for in connection with the copyright protection under copyright law or other applicable laws.

3. License Grant

Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

a. to Reproduce the Work, to incorporate the Work into one or more Collections, and to Reproduce the Work as incorporated in the Collections;

b. to create and Reproduce Adaptations provided that any such Adaptation, including any translation in any medium, takes reasonable steps to clearly label, demarcate or otherwise identify that changes were made to the original Work. For example, a translation could be marked "The original work was translated from English to Spanish," or a modification could indicate "The original work has been modified."

c. to Distribute and Publicly Perform the Work including as incorporated in Collections;

d. and, to Distribute and Publicly Perform Adaptations.

e. For the avoidance of doubt:

i. Non-waivable Compulsory License Schemes. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme cannot be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License;

ii. Waivable Compulsory License Schemes. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme can be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License if Your exercise of such rights is for a purpose or use which is otherwise than noncommercial as permitted under Section 4(b) and otherwise waives the right to collect royalties through any statutory or compulsory licensing scheme; and,

iii. Voluntary License Schemes. The Licensor reserves the right to collect royalties, whether individually or, in the event that the Licensor is a member of a collecting society that administers voluntary licensing schemes, via that society, from any exercise by You of the rights granted under this License that is for a purpose or use which is otherwise than noncommercial as permitted under Section 4(c).

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such