SENSE4US

Models and Tools to Analyse and Predict Discussion Dynamics and Sentiment Towards Policy

Project acronym:	**SENSE4US**
Project full title:	**Data Insights for Policy Makers and Citizens**
Grant agreement no.:	**611242**
Responsible:	**Miriam Fernandez, Open University**
Contributors:	**Hassan Saif, Open University**
Document Reference:	**D5.2**
Dissemination Level:	**PU**
Version:	**1.0**
Date:	**30/03/2015**

Project co-funded by the European Union under the Seventh Framework Programme

History

Version	*Date*	*Modification reason*	*Modified by*
0.1	30/03/2015	Initial draft	Miriam Fernandez
0.2	15/03/2015	Code Released	Hassan Saif
0.3	15/03/2015	Functionality Description	Hassan Saif
0.4	30/03/2015	Quality check	Steve Taylor
1.0	4/01/2014	Final reviewed deliverable	Miriam Fernandez

Table of contents

Executive summary

In this document we describe the software produced as part of D5.2 to represent the social media citizen's discussions and to extract their sentiment by using our proposed SentiCircle representation [6][7][5]. This document aims to act as a manual for other developers and researchers who want to use the technology developed within Sense4us and more particularly, within WP5.

We also report in this document the multiple optimisations and extensions built since our previous software released as part of D5.1. In particular, this new version of the software provides a functionality to find the relation between two terms based on their co-occurrence in the social media conversations and on the sentiment of these conversations. This functionality aims to help the policy modelling process developed by WP6 by supporting the Policy Maker on setting up influence scores between concepts based on evidence extracted from social media conversations.

In addition to the developed functionalities we have also provided command line execution capabilities for the released sentiment analysis library. This will allow researchers and practitioners not familiar with software development to make use of the Sense4us technology.

To conclude, this document presents specific details about the sentiment prediction models developed as part of WP5. The results of our evaluation show that our models can predict sentiment with up to 0.664 F-measure by making use of content as well as social features.

List of figures

List of tables

List of abbreviations

<Abbreviation>	<Explanation>
HCR	Health Care Reform Dataset
IDF	Inverse Document Frequency
OMD	Obama-MaCain Debate Dataset
PM	Policy Maker
TF	Term Frequency

1 SentiCircle

In this section we present a brief overview of SentiCircle, our developed lexicon-based approach for sentiment analysis on Microblogs. Different from typical lexicon-based approaches, which offer fixed and static prior sentiment polarities of words regardless of their context, SentiCircle takes into account the co-occurrence patterns of words in different contexts in social media posts to capture their contextual semantics and update their pre-assigned strength and polarity accordingly.

Contextual semantics (aka statistical semantics) have been traditionally used in diverse areas of computer science, including Natural Language Processing (NLP) and Information Retrieval (IR). The main principle behind the notion of contextual semantics comes from the dictum- "You shall know a word by the company it keeps!" This suggests that words that co-occur in a given context tend to have certain relation or semantic influence, which we try to capture with our SentiCircle approach.

SentiCircle allows for (i) *entity-level sentiment* detection, which detects sentiment towards a particular entity (e.g., `Obama`, `Microsoft`, `iPad`), and (ii) *tweet-level sentiment* detection, which identifies the overall sentiment of *individual* tweets.

In the subsequent section we summarise the SentiCircle approach and explain how it captures the contextual semantic and sentiment of words and how this representation is used to compute sentiment at tweet and entity levels. For more details the reader is referred to the following publications. [5][6][7][8]

1.1 The SentiCircle Model

SentiCircle aims to represent the sentiment orientation of words with respect to their contextual semantics. The main notion behind this is that the sentiment of a term is not static, as in traditional lexicon-based approaches, but rather depends on the context in which the term is used, i.e., it depends on its contextual semantics. For example, most existing sentiment analysis methods fail to detect the sentiment of the tweet "`#Syria. Execution continues with smile! :( #ISIS`", since they consider the existence of the word "`smile`" positive, even though it is used within the context of the negative word "`Execution`".

To capture the contextual semantics of a term, we follow the distributional semantic hypothesis that words that occur in similar contexts tend to have similar meaning. Therefore, the contextual semantics of a term in our approach is computed from its co-occurrence patterns with other terms. Note that we define context as a textual corpus or a set of tweets.

Thus, in order to understand the semantics and the sentiment of a target word like "ISIS"(Islamic State in Iraq and Syria), our method relies on the words that co-occur with the target word in a given collection of tweets. These co-occurrences are mathematically represented as a 2D geometric circle; where the target word ("ISIS") is at the centre of the circle and each point in the circle represents a context word that co-occurs with "ISIS" in the tweet collection (see Figure 1).

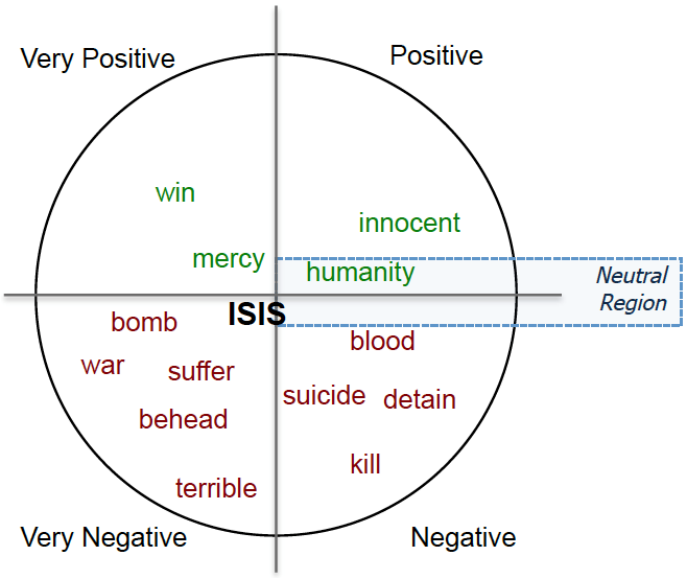

Figure 1: Example SentiCircle for the word "ISIS". Terms positioned in the upper half of the circle have positive sentiment while terms in lower half have negative sentiment

The position of each context term (e.g., "`kill`", "`mercy`", etc.) in the circle determines its importance and sentiment influence toward the target term. Such position can be defined by an angle (representing the prior sentiment of the context term extracted from a given sentiment lexicon) and a radius (representing the correlation between the context term and the target term). For example the angle of the context word "`kill`" is calculated as:

$$\theta_{kill} = Prior_{Sentiment(kill)} * \pi$$

where the prior sentiment (kill) is extracted from a given sentiment lexicon, and has a value between -1 and 1. Note that, by calculating the angle in this way, we are considering two independent semicircles. The upper semicircle, where the angle varies from 0 to π (0 being neutral and π most positive), capturing contextual words with positive sentiment, and the lower semicircle, where the angle varies from 0 to – π (0 being neutral and -π most negative); capturing contextual words with negative sentiment. If the context term, in this case "`kill`", appears in the vicinity of a negation (negation words like "`no`", "`can't`", etc.), its prior sentiment score is negated. The negation words are collected from the General Inquirer under the NOTLW category.[1]

The degree of correlation between a context word, e.g., "`kill`", and a target word, in this case "`ISIS`", is computed based on the co-occurrence of these words within a given context, or set of posts (e.g., tweets).

Terms in the two upper quadrants of the SentiCircle have a positive sentiment (θ between 0 and π), with upper left quadrant representing stronger positive sentiment since it has larger angle values than those in the top right quadrant. Similarly, terms in the two lower quadrants have negative sentiment values (θ between 0 and - π). Moreover, a small region called the "*Neutral Region*" can be defined. This region, as shown in Figure 1, is located very close to X-axis in the "*Positive*" and the "*Negative*" quadrants only, where terms lie in this region have very weak sentiment (i.e., $|\theta| \approx 0$)

[1] http://www.wjh.harvard.edu/~inquirer/NotLw.html

Points representing context terms the circle have different radii ($0 \leq ri \leq 1$), which reflect how important a context term is to the target term. The larger the radius, i.e., the distance to the origin, the more important the context term is to the target term.

The rationale behind using this circular representation shape is to benefit from the trigonometric properties it offers for estimating the sentiment orientation, and strength, of terms. It also enables us to calculate the impact of context words on the sentiment orientation (i.e., positive, negative, neutral) and on the sentiment strength (e.g., weak positive, strong negative, etc.) of a target-word separately, which is difficult to do with traditional vector representations.

For specific mathematical details about the SentiCircle model the reader is referred to the following publication. [6]

1.2 SentiCircle for Sentiment Analysis

The SentiCircles representation is used for detecting the positive and negative sentiment expressed on social media. SentiCircles can be used in the following two sentiment analysis tasks:

- **Entity-level Sentiment Analysis**: which aim to detect the sentiment of a given named entity (e.g., "Obama", "David Cameron", "ISIS")
- **Post-level Sentiment Analysis**: which aim to detect the overall sentiment of a given post (e.g., "*#Syria #ISIS. Execution continues with smile! :(*")

To compute sentiment at entity-level we use the geometric median of the SentiCircle (or **Senti-Median**). The Senti-Median is a point within the circle, capturing the overall sentiment and the sentiment strength of the target entity or term (which sits at the centre of the SentiCircle) by using all the previously positioned contextual terms. The geometric median of the SentiCircle (or Senti-Median) is calculated as the point where the Euclidean distances[2] to all the points within the circle is minimum.

To compute sentiment and tweet level we propose three different methods:

- **The Median Method**: The method calculates the sentiment of a post by capturing the SentiCircle of each term within the post and computing the Senti-Median of each of these SentiCircles. The final sentiment score of the tweet is computed as the median of all the previously computed Senti-Medians.
- **The Pivot Method:** This method favours some terms in a post over others, based on the assumption that sentiment is often expressed towards one or more specific targets, which we refer to as "Pivot" terms. For simplicity, we assume that the pivot terms are those having the Part of Speech (POS) tags: {Common Noun, Proper Noun, Pronoun} in a post.
- **The Pivot-Hybrid Method**: This last method is a combination of two previous ones. In the cases where the Pivot-Method fails to find a pivot term (because the post is too short, or it contains many ill-formed words) the Median Method is applied to compute the sentiment of the post.

[2] http://en.wikipedia.org/wiki/Euclidean_distance

2 SentiCircle Implementation

In this section we provide specific details of the implemented code as well as relevant details of how to execute it. Additional functionalities have been added, and multiple software improvements have been performed with respect to the previous release (D5.1). In particular we have conducted the following optimisations:

- ✓ **Optimisation at design level**: Once we experimented in different settings [Saif et al., 2014b; Saif et al., 2014c], including multiple Twitter datasets, different sentiment lexicons and different baselines for comparison, we have taken the lessons learned from these experiments to optimise the architectural design of the SentiCircle representation. In particular, two optimisation setups have been mainly made. First, we chose to use both, Thelwall-Lexicon [Thelwall et al., 2012] and the MPQA [Wilson et al., 2005] lexicon together to derive the prior sentiment of words. This is because our experiments showed that using both lexicons in our model allows for higher coverage of opinionated words and better sentiment extraction than using each lexicon solely. Secondly, our evaluation on several datasets allowed us to set the thresholds of the neutral region's boundaries in the SentiCircle model, which consequently allows for optimising the analysis of neutral sentiment.
- ✓ **Optimisation at data transfer level**: JSON format has been selected as the standard input/output format for our methods. Note that this format is highly used by the developers' community and will therefore facilitate the manipulation of this software by other developers.
- ✓ **Optimisation at source code level**: Beyond the algorithms and their implementations concrete source code level choices have been taken to provide a readable and easy to understand code for other developers who may want to extend or reuse our model.
- ✓ **Optimisation at run time:** optimisations have been carefully considered to cope with large vocabularies of terms. Note that the larger the vocabulary size associated to the corpus, the more expensive will be the initial computation of our method. Specific details of our research behind this optimisation can be found in the following publication. [Saif et al., 2014a]
- ✓ **Microblogging platform independence:** although SentiCircles have been tested with Twitter data, our code is generic and can be applied to compute sentiment from other micro blogging services.
- ✓ **Platform independence:** our code has been packed as a java library, and it is therefore executable in different Operating Systems.
- ✓ **Command Line execution:** the released library is executable via command line. This will allow researchers not familiar with software development to make use of the provided functionalities for sentiment analysis.

Extensions: this new software release also provides a novel functionality to find the relation between two terms based on their co-occurrence in the social media conversations and on the sentiment of these conversations. This functionality aims to help the policy modelling process developed by WP6 by supporting the Policy Maker on setting up influence scores between concepts based on evidence extracted from social media conversations.

2.1 Overview

The SentiCircle model is implemented using Java 6, providing several functionalities for semantic representation and sentiment extraction of words.

2.1.1 Binaries and Distribution

This release of SentiCircle is packaged in a "zip" archive, containing the following files and directories:

- **senticircle.jar** Java binaries of the SentiCircle implementation.
- **data/model** folder where the generated SentiCircle models will be saved.
- **resources** folder that contains configuration files, sentiment lexicons and lexical resources required for the functioning of SentiCircle.

2.1.2 Comments and Bug Reports

SentiCircle is a standalone model that is developed from scratch with no reliance on external sentiment analysis models or tools. However, SentiCircle uses several third-party libraries for text-processing, syntactic parsing, and object serialization such as OpenCSV,2 TweetNLP3, and Google-Json.4

2.1.3 License

SentiCircle is licensed under the Apache License, Version 2.0 (the "License"). A copy of the License can be obtained at http://www.apache.org/licenses/LICENSE-2.0.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

2.2 Main Functions and Usage

The main class that provides all the functionalities of SentiCircle is: "`MainController`" which is found under Package `org.kmi.controller`. This class provides the following main functions:

2.2.1 Generate the SentiCircle Model

```
public void generateModel(java.lang.String dataFile) throws
java.io.IOException
```

Generates the SentiCircle model for a given tweet collection. The generated model will be stored under the "data/model" directory. Note that, for every term in the tweet collection, a sentiCircle will be generated.

Parameters:

• `dataFile` – the file path of the tweet collection (see Input Format)

Returns:

- name and directory of the generated model in JSON format (see Output Format)

Throws:

- `java.io.IOException` - input file not found

Input Format

- `dataFile:` a semicolon-separated CSV file. Each line in the file represents a tweet message, as follows:

```
"tweet_id";"tweet_message"
```

where *tweet_id* is a number representing the id of a tweet in the file, and *tweet_message* is the text of the tweet.

Note that: no header is required for the CSV file. *Example an input data file for the above function:*

```
"2000270661";"1 vs 100 on Xbox Live was fun"
"2195918085";"I want some fries from McDonalds. Im so hungry"
"2229956563";"Jogging, isnt REALLY that cool if u've got fever"
```

Output Format

JSON format as follows:

```
{
    "modelName": "the name of the generated SentiCircle Model",
    "modelDir": "the relative path of the model's directory"
}
```

2.2.2 Sentiment Detection of a Tweet

```
public java.lang.String inferTweet(java.lang.String tweet,
java.lang.String modelName) throws java.io.IOException,
org.kmi.utils.IFaultException
```

Detects the sentiment (positive, negative) of a given tweet message

Parameters:

- `tweet` - the tweet message
- `modelName` - the name of the SentiCircle model to be used for sentiment detection

Returns:

- The sentiment of the given tweet in JSON format (see Output Format)

Throws:

- `java.io.IOException`
- `com.google.gson.JsonSyntaxException`
- `org.kmi.utils.IFaultException`

Output Format

JSON format as follows:

```
{
    "id": 1,  // The id the tweet message
    "text": "I like iPhone", // the body of the tweet message
    "inferredPolarity": 4   // The inferred sentiment
}
```

Note that: the sentiment returned by this function is one of the following values:

$$inferredPolarity = \begin{Bmatrix} 0\ Negative \\ 4\ Positive \\ -1\ Sentiment\ can\ not\ be\ determined \end{Bmatrix}$$

2.2.3 Sentiment Detection of a Collection of Tweets

```
public java.lang.String inferTweets(java.lang.String tweetFile,
java.lang.String modelName) throws java.io.IOException,
com.google.gson.JsonSyntaxException, org.kmi.utils.IFaultException
```

Detects the sentiment (positive, negative) of tweets in a given data file

Parameters:

- `tweetFile` - the file path of the tweet collection
- `modelName` - the name of the SentiCircle model to be used for sentiment detection

Returns:

- the sentiment of each tweet in the data file (see Output Format)

Throws:

- `java.io.IOException`
- `com.google.gson.JsonSyntaxException`
- `org.kmi.utils.IFaultException`

Input Format

`dataFile`: a semicolon-separated CSV file. Each line in the file represents a tweet message, as follows:

```
"tweet_id";"tweet_message"
```

where *tweet_id* is a number representing the the id of a tweet in the file, and *tweet_message* is the text of the tweet.

Output Format

A list of tweet-sentiment JSON object, as follows:

```
[
    {
        "id": 1,  // The id the tweet message
```

```
        "text": "I like iPhone", // the body of the tweet message
        "inferredPolarity": 4   // The inferred sentiment
    },
    {
        "id": 2,  // The id the tweet message
        "text": "I do not like Samsung", // the body of the tweet
message
        "inferredPolarity": 0   // The inferred sentiment
    }
]
```

Note that: the sentiment returned by this function is one of the following values:

$$inferredPolarity = \begin{Bmatrix} 0\ Negative \\ 4\ Positive \\ -1\ Sentiment\ can\ not\ be\ determined \end{Bmatrix}$$

2.2.3.1 Sentiment Detection of an Entity/Term

```
public java.lang.String inferEntity(java.lang.String entity,
java.lang.String modelName) throws java.io.IOException,
com.google.gson.JsonSyntaxException, org.kmi.utils.IFaultException
```

Detects the sentiment (positive, negative, neutral) of a given entity or term

Parameters:

- `entity` - a given term or entity
- `modelName` - the name of the SentiCircle model to be used for sentiment detection

Returns:

- the sentiment, SentiCircle, and SentiMedian of the given entity/term in JSON format (see Output Format)

Throws:

- `java.io.IOException`
- `com.google.gson.JsonSyntaxException`
- `org.kmi.utils.IFaultException`

Output Format

JSON format as follows:

```
{
    "id": 1,    // id of the entity
    "label": "obama",   // entity
    "inferredSentiment": 0, // the inferred sentiment
    "sentiCircle": {// the entity's SentiCircle
        "label": "obama",
```

```
        "median":
        {   // The position of the median point (SentiMedian) of the
entity's SentiCircle
            "x": -0.022478999321236006, // X-position
            "y": -0.018401050804494827, // Y-position
            "region": 0 // The sentiment region of the point
        },
        "points": [ // List of all the points (context-terms) within
the SentiCircle
            {
                "label": "fail", // context-term
                "x": -0.01965002715587616,  // X-position
                "y": -0.06047656759619713,  // Y-position
                "region": 0 // The sentiment region of the point
            },
            {
                "label": "cheney",  // context-term
                 "x": 0.0943480134010315,   // X-position
                 "y": 0.0,  // Y-position
                 "region": 2    // The sentiment region of the point
            }
        ]
    }
}
```

Note1: the sentiment returned by this function is one of the following values:

$$inferredPolarity = \begin{Bmatrix} 0\ Negative \\ 2\ Neutral \\ 4\ Positive \\ -1\ Sentiment\ can\ not\ be\ determined \end{Bmatrix}$$

Note2: the **sentiment region** of the a point within the SentiCircle returned by this function is one of the following values:

$$region = \begin{Bmatrix} 0\ Negative \\ 2\ Neutral \\ 4\ Positive \end{Bmatrix}$$

Note3: If the provided entity does not exist in the model's vocabulary, the returned output will be:

```
{
    "id": 1,
    "label": "ipad",
    "inferredSentiment": -1 // the entity "ipad" can not be found in
the vocabulary
}
```

2.2.4 Sentiment Detection of a Collection of Entities

```
public java.lang.String infereEntities(java.lang.String entityFile,
java.lang.String modelName) throws java.io.IOException,
com.google.gson.JsonSyntaxException, org.kmi.utils.IFaultException
```

Detects the sentiment (positive, negative, neutral) of entities in a given data file

Parameters:

- `entityFile` - the file path of the entity collection
- `modelName` - the name of the SentiCircle model to be used for sentiment detection

Returns:

- the sentiment of each entity in the data file (see Output Format)

Throws:

- `java.io.IOException`
- `com.google.gson.JsonSyntaxException`
- `org.kmi.utils.IFaultException`

Input Format

`dataFile`: a semicolon-separated CSV file. Each line in the file represents an entity, as follows:

```
"entity_id";"entity"
```

where *tweet_id* is a number representing the the id of a tweet in the file

Note that: no header is required for the CSV file. *Example an entity data file:*

```
"1";"obama"
"2";"iPod"
"3";"fever"
```

Output Format

A list of entity-sentiment JSON object. The output for each entity in the list is similar to the one for Function `inferEntity` in the previous section.

```
[
    {
        // Entity.1 sentiment output - See the output format of
Function inferEntity()
    },
    {
        // Entity.2 sentiment output
    }
]
```

Note that: the sentiment returned by this function for each entity in the list is one of the following values:

$$inferredPolarity = \left\{ \begin{array}{c} 0\ Negative \\ 2\ Neutral \\ 4\ Positive \\ -1\ Sentiment\ can\ not\ be\ determined \end{array} \right\}$$

2.2.4.1 Sentiment-based Correlation between Terms

```
public java.lang.String calcTermCorrelation(java.lang.String
firstTerm, java.lang.String secondTerm, java.lang.String modelName)
throws com.google.gson.JsonSyntaxException,
org.kmi.utils.IFaultException
```

Compute the correlation between two terms with respect to their sentiment orientations. This functionality has been developed to provide input to the modelling work in WP6. It will help the user to better understand whether two concepts (terms) are related within the social media conversations and with which sentiment

Parameters:

- `firstTerm` - the first term
- `secondTerm` - the second term
- `modelName` - the name of the SentiCircle model to be used for correlation extraction

Returns:

- The sentiment-based correlation (percentage) between the two terms

Throws:

- `com.google.gson.JsonSyntaxException`
- `org.kmi.utils.IFaultException`

Output Format

JSON format as follows:

```
{
    "firstTerm": "obama",    // the first term
    "secondTerm": "iran",    // the second term
    "sentiment": 2, // the sentiment where the correlation between
both terms occurs
    "correlation": 23.170949302396654   // the correlation
(percentage) with respect the sentiment label
}
```

Note1: the sentiment under which the correlation is computed is one of the following values:

$$sentiment = \begin{Bmatrix} 0\ Negative \\ 4\ Positive \\ -1\ No\ correlation\ is\ detected \end{Bmatrix}$$

Note3: if there is no correlation between the first and second terms (i.e, the two terms do not occur together in the given data collection), the output of the function would be:

```
{
    "firstTerm": "obama",
    "secondTerm": "ipad",
```

```
    "sentiment": -1, // there is no sentiment under which the
correlation between both terms happens
    "correlation": 0.0
}
```

2.3 Using SentiCircle from the command line

We have packed our code so that the library can be directly executed from the command line without the need to explore the code. In the following section we explain how SentiCircle can be used from the command line.

Note that the input and output formats here are similar to those explained in the previous Section.

Hint: for lowering the processing time of the SentiCirlce's functions, we recommend increasing the heap size of JVM by, for example, using the command `-Xmx8000M`.

2.3.1 Generate the SentiCircle Model

The command line for generating the SentiCircle model for a given collection of tweets is:

```
java -jar senticircle.jar -gen -i <string>
```

where:

- -gen: generate the SentiCircle model from scratch
- -i: the input data file (collection of tweets)

Example:

```
java -jar senticircle.jar -gen -i data/datasets/sts_gold_binary.csv
```

2.3.2 Tweet-level Sentiment Detection

```
java -jar senticircle.jar -inf -model <string> -tweet [-t|-i] <string>
[-o <string>]
```

where:

- -inf: generate the SentiCircle model from scratch
- -model: the name of the SentiCircle model to be used for sentiment detection
- -tweet: perform tweet-level sentiment detection
- -t: a single tweet message
- -i: the input data file containing a collection of tweet messages
- -o: the output file. If not provided, the output will be directly printed on the screen.
-

Example1: detect the sentiment of a single tweet

```
java -jar senticircle.jar -inf -model sts_gold_binary.csv -tweet -t "I
like my iPad"
```

Example2: detect the sentiment of a tweet collection

```
java -jar senticircle.jar -inf -model sts_gold_binary.csv -tweet -i
data/datasets/sts_test_binary.csv -o results.json
```

2.3.3 Entity-level Sentiment Detection

```
java -jar senticircle.jar -inf -model <string> -entity [-e|-i]
<string> [-o <string>]
```

where:

- -inf: generate the SentiCircle model from scratch
- -model: the name of the SentiCircle model to be used for sentiment detection
- -entity: perform entity-level sentiment detection
- -e: a single entity
- -i: the input data file containing the entity collection
- -o: the output file. If not provided, the output will be directly printed on the screen.

Example1: detect the sentiment of a single entity

```
java -jar senticircle.jar -inf -model sts_gold_binary.csv -entity -e
Obama
```

Example2: detect the sentiment of entities in a given file

```
java -jar senticircle.jar -inf -model sts_gold_binary.csv -entity -i
data/datasets/sts_gold_entity.csv -o entity_results.json
```

2.3.4 Sentiment-based Correlation between Terms

```
java -jar senticircle.jar -corr -model <string> -f <string> -s
<string> [-o <string>]
```

where:

- -corr: find the correlation between two terms with respect to their sentiment
- -model: the name of the SentiCircle model to be used for sentiment detection
- -first: the first term
- -s: the second term
- -o: the output file. if not provided, results will be directly printed on the screen.

Example1: detect the sentiment of a single entity

```
java -jar senticircle.jar -corr -model sts_gold_binary.csv -f obama -s
iran
```

As explained before, this functionality has been developed to help shaping the predictive models / simulations of policy impacts. In WP6 we need to identify related concepts of the policy issue, to introduce them to the user who will employ them in structuring the problem situation for simulation. The relations extracted from the social media conversations do not aim to reflect casual relations, but relations between two variables (indicators) based on the sentiment of the social media conversations around the two indicators.

3 Sentiment Prediction

In D5.1 we presented our developed functionalities for predicting sentiment. In this deliverable we explain further details of our approach to developed sentiment prediction models and their evaluation.

3.1 Goal

The goal of the sentiment prediction task is to predict the sentiment of a particular theme or topic. In this sense the task is different than classifying the sentiment for a particular entity or tweet. We do not focus on learning the characteristics of individual tweets in order to determine the language and context that characterise positive vs. negative tweets, but we focus on learning the characteristics of tweet sets at time t in order to predict the global sentiment of topic at time t+1.

3.2 Approach

To conduct this task we first need to define how a topic or theme is identified. In the particular case of Twitter, topics are generally identified by the use of hashtags. Hashtags are manually generated terms, preceded by the symbol #, agreed by users to refer to a common topic. For example the hashtag #earthhour15 refers to the campaign launched by the World Wildlife Fund (WWF) to raise awareness about climate change and sustainability issues. In the case of microblogging platforms where the use of hashtags is not available, a topic can be identified by a subset of relevant keywords.

To predict sentiment for a particular topic we follow a similar approach to research works focused on predict engagement dynamics.[18] In these works, researchers aim to predict how many retweets/replies/favourites (i.e., how much engagement) a tweet will obtain by generating regression models[3] based on different features. These models do not only allow for engagement prediction, but also allow studying which features are aligned or characterise those tweets that are followed by a high number of engagement actions.

In our particular case the aim is not to predict the number of engagement actions that a tweet will generate, but the sentiment that a particular topic will have in the future considering the tweets that were posted about that topic in the past. We capture the sentiment at time *t* as the ratio between the number of positive vs. the number of negative tweets. If this ratio is greater than 1 the sentiment is positive, it this ratio is smaller than 1 the sentiment is negative. If the ratio is equal to 1 the sentiment is considered neutral and/or controversial.

$$sentiment\_score_{t=} \frac{|positive\ tweets|}{|negative\ tweets|}$$

To predict the sentiment_score at time t+1 we consider different features to characterise the set of tweets at time t as well as the sentiment evolution until time t. These features include *content features* as well as *social features*. While content features reflect the characteristics of the posts, social features aim to characterise the social standing of authors of those posts. In particular, the set of features considered for this work include:

Content features:

- Number of posts: number of unique posts produced between the times t-1 and t. The rationale to choose this feature is that discussions that generate a high number of

[3] http://en.wikipedia.org/wiki/Regression_analysis

posts may be more polar (positive/negative) than discussions generating a lower number of posts.

- Average number of mentions: average number of mentions to other users within the set of posts produced between the times t-1 and t. The rationale to choose this feature is that, when mentioning users in their messages, people are explicitly spreading their arguments (messages) and their polarity to other users. Therefore, mentions may provide an indication of sentiment spread.
- Average number of hyperlinks: average number of hyperlinks within the set of posts produced between the times t-1 and t. The rationale to choose this feature is that posts containing links to external information may be more likely to be retweeted and therefore to propagate their sentiment.
- Sentiment Evolution: increase/decrease of the sentiment_score value between the times t-1 and t. This feature indicates the evolution of sentiment in the previous step. If sentiment (positive or negative) have increased or decreased in the previous step by a large margin this may be an indication that the tendency will continue in future time steps.

Social features

- Number of unique users: number of unique users who have posted between the times t-1 and t. The rationale to choose this feature is that, a topic involving a high number of users discussing about it may be more polar (positive/negative) than a topic that only few users are discussing.
- Average in-degree: average number of followers for the users who posted between the times t-1 and t. The rationale to choose this feature is that the higher the in-degree, the more popular users are. Therefore, messages and their associated sentiment are more likely to be spread around the network.
- Average out-degree: average number of users followed by the users who posted between the times t-1 and t. The rationale to choose this feature is that users who follow lots of users may be more prone to observe different opinions and therefore be less polar.
- Average user age: average time that the users who posted between the times t-1 and t have been members of the microblogging platform. The rationale to choose this feature is that users participating in the social network during longer periods of time may have higher reputation and their messages and sentiment may have higher chances to be propagated.

Once the topic is identified, and the different features that describe the content at different time periods are established, a key element is to determine the *time granularity* needed to develop the prediction models. In the case of our work we consider different time slots, from 10 minutes to 24h depending on the evaluation dataset.

3.3 Evaluation

To evaluate our approach we need evaluation datasets that: (i) are based on a particular topic, preferably related to policy, (ii) contain time information and, (iii) are manually annotated with sentiment labels. For the purpose of this evaluation we have considered nine different Twitter datasets commonly used in the literature of sentiment analysis. The complete description of these datasets can be found in [4].

These datasets generally include the text of the tweet as well as its associated sentiment label. However, none of these datasets includes time information.

936734619	0	#debate08 Lehrer pounding on ' What can't we do , due to cost of bailout ? ' = job interview "" What would you do if bear broke into yr house ? ""
936735826	0	Both candidates are unable to tell the difference between the Republican Guard and the Revolutionary Guard . #debate08
936468736	4	@current I've got the live stream going #current .... awesome !
936469906	4	Current TV is pretty cool . Streaming Twitter feedback #tweetdebate
936469907	4	#tweetdebate the first debate is about to start

Figure 2: Example of Tweets for the OMD dataset

To recover the creation date and time of the tweets we needed the dataset to contain the original tweet IDs. For those datasets containing the original tweet IDs we have collected information about the tweets by using the Twitter Search API. After discarding those datasets for which timestamp information could not be gathered, as well as those datasets that are not focused on a particular topic (i.e., have not been collected by considering few hashtags related to the same topic), we remained with two standard evaluation datasets to perform our experiments, HRC and OMD.

- *Health Care Reform (HCR)*: The Health Care Reform (HCR) dataset was built by crawling tweets containing the hashtag "#hcr" (health care reform) in March 2010 [15]. A subset of this corpus was manually annotated with 5 labels (positive, negative, neutral, irrelevant, unsure (other)) and split into training (839 tweets), development (838 tweets) and test (839 tweets) sets. For this experiment, we consider all the three subsets (training, development and test) as one unique dataset for the analysis. The final dataset, consists of 2,516 tweets including 1,381 negative, 470 neutral and 541 positive tweets.

- *Obama-McCain Debate*: The Obama-McCain Debate (OMD) dataset was constructed from 3,238 tweets crawled during the first U.S. presidential TV debate in September 2008. [16] Sentiment labels were acquired for these tweets using Amazon Mechanical Turk, where each tweet was rated by at least three annotators as positive, negative, mixed, or other. The dataset is provided at https://bitbucket.org/speriosu/updown along with the annotators' votes on each tweet. We considered those sentiment labels, which two-thirds of the voters agreed on, as final labels of the tweets. This resulted in a set of 1,196 negative, 710 positive and 245 mixed tweets. The OMD dataset is a popular dataset, which has been used to evaluate various supervised learning methods [15], as well as unsupervised methods [9] for polarity classification of tweets. Tweets' sentiments in this dataset were also used to characterize the Obama-McCain debate event in 2008. [17]

For our experiments we induced a regression model. We trained the model using different feature sets (i.e., social, content, social+content) to see which feature set performs best and how this differs between the different evaluation datasets. Results are reported in the following tables in terms of Precision, Recall and F1 measure (see the following reference for further explanations about these evaluation metrics http://en.wikipedia.org/wiki/Precision_and_recall).

In the case of HCR the selected time period to compute the features was 24 hours, while, in the case of OMD, the selected time period was 10 minutes. Note that the OMD dataset includes tweets for the 97 minute debate + the 53 minutes following the event, i.e., 150 minutes, while HCR contains information for several days during March 2010. The different use case scenarios therefore determined the granularity of the selected time period. In the case of the Obama-McCain Debate it is relevant to predict rapidly the evolution of sentiment since the debate only spans an hour. In the case of the Health Care Reform the discussion spanned several weeks.

Feature sets	*Precision*	*Recall*	*F1*
Social	0.562	0.561	0.561
Content	0.662	0.664	0.659
Social + Content	0.661	0.670	**0.664**

Table 1: Results for the HCR dataset.

Feature sets	*Precision*	*Recall*	*F1*
Social	0.528	0.540	0.537
Content	0.612	0.612	0.611
Social + Content	0.651	0.642	**0.645**

Table 2: Results for the OMD dataset

3.4 Discussion

As we can see from the results obtained, in both cases the use of content features is more useful to predict sentiment than the use of social features, i.e., the content of the posts is more discriminative than the standing of the user in the social network when predicting sentiment. However, the combination of user and content features was more successful in both cases. The models reached 0.645 F1-measure in the case of OMD and 0.664 F1-measure in the case of HCR when using both content and social features to predict sentiment.

While these results are encouraging, multiple different aspects can still be explored in order to reach a greater understanding on sentiment prediction. Note that different factors may influence the prediction of sentiment including: the selected model (regression, signal estimation or other types of time series analysis, etc.), the selected features and the length of the time period selected to build the prediction models. In our next experiments we plan to incorporate features derived from SentiCircles in order to improve the accuracy of the developed models.

Assessing the prediction of sentiment is also a difficult task because of the lack of evaluation datasets prepared for this matter. Note that existing evaluation datasets are mainly focused on assessing polarity (positive vs. negative) and subjectivity (polar vs. neutral) detection models at tweet level. Evaluation datasets need to be developed to assess sentiment prediction over longer time periods and ideally, capturing particular topics and/or events. It is therefore part of our future work to develop evaluation datasets based on automatic sentiment computation rather than on manual sentiment labels. While the evaluation datasets may not be as “clean”, they will allow us to evaluate our sentiment predictors over longer time periods and over selected events.

Additional research in terms of sentiment prediction is planned for the next months of the project and will be provided in D5.3 which is due M24.

Conclusions

This deliverable summarises the sentiment analysis software released by WP5. The goal of this WP has been to develop tools to monitor and analyse policy discussions in social media and the citizens' sentiment towards policies based on these discussions.

The design and development of these tools have been driven by the conducted research. During the first year of the project, WP5 investigated the use of contextual and conceptual semantics for calculating sentiment as well as the use of user and content features to track discussion dynamics around a variety of topics. This research was translated into a series of publications [5][6][7][8]. During this period we have focused on encapsulating all this research into a sentiment analysis library to encourage other researchers and practitioners to make use of the technology developed as part of the Sense4us project. The current deliverable aims therefore to be a manual for other researchers and developers who may want to make use of or extend our research.

We also include in this deliverable further details of our research about sentiment prediction. We present our approach to develop sentiment prediction models as well as an assessment of our approach by making use of two standard evaluation datasets. Our results show that our development models can predict sentiment with up to 0.664 F-measure. Further research is needed to incorporate additional features, which may help improving upon the obtained results, and to automatically generate evaluation datasets that can help us to assess sentiment prediction models over long periods of time.

4 References

[1] Kouloumpis, E., Wilson, T., Moore, J.: Twitter sentiment analysis: The good the bad and the omg! In: Proceedings of the ICWSM. Barcelona, Spain (2011)

[2] Saif, H., He, Y., Alani, H.: Alleviating data sparsity for twitter sentiment analysis. In:Proc. Workshop on Making Sense of Microposts (#MSM2012) in WWW 2012. Lyon, France (2012)

[3] Saif, H., He, Y., Alani, H.: Semantic sentiment analysis of twitter. In: Proc. 11th Int. Semantic Web Conf. (ISWC). Boston, MA (2012)

[4] H. Saif, M. Fernandez, Y. He, and H. Alani. Evaluation datasets for twitter sentiment analysis a survey and a new dataset, the sts-gold. In Proceedings, 1st Workshop on Emotion and Sentiment in Social and Expressive Media (ESSEM) in conjunction with AI*IA Conference, Turin, Italy, 2013.

[5] H. Saif, M. Fernandez, Y. He, and H. Alani. 2014a. On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter. In Proc. 9th Language Resources and Evaluation Conference (LREC), Reykjavik, Iceland, 2014.

[6] H. Saif, M. Fernandez, Y. He, and H. Alani. 2014b. SentiCircles for Contextual and Conceptual Semantic Sentiment analysis of Twitter. Extended Semantic Web Conference (ESWC), Crete, 2014.

[7] H. Saif, M. Fernandez, Y. He, and H. Alani. 2014c. Adapting Sentiment Lexicons using Contextual Semantics for Twitter Sentiment Analysis. In Proceeding of the first semantic sentiment analysis workshop: conjunction with the eleventh Extended Semantic Web conference (ESWC). Crete, Greece.

[8] H. Saif, Y. He, M. Fernandez and H. Alani. 2014d Semantic Patterns for Sentiment Analysis of Twitter, The 13th International Semantic Web Conference (ISWC), Riva del Garda - Trentino Italy

[9] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Computational linguistics 37(2), 267–307 (2011)

[10] Thelwall, M., Buckley, K., Paltoglou, G.: Sentiment strength detection for the social web. J. American Society for Information Science and Technology 63(1), 163–173 (2012)

[11] Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., Kappas, A.: Sentiment strength detection in short informal text. J. American Society for Info. Science and Technology 61(12) (2010)

[12] Cambria, E.: An introduction to concept-level sentiment analysis. In: Advances in Soft Computing and Its Applications, pp. 478–483. Springer (2013)

[13] Turney, P.D., Pantel, P.,et al.: From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research 37(1), 141–188 (2010)

[14] Wittgenstein, L.: Philosophical Investigations. Blackwell, London, UK (1953, 2001)

[15] Speriosu, M., Sudan, N., Upadhyay, S. and Baldridge, J. Twitter polarity classification with label propagation over lexical links and the follower graph. In Proceedings of the EMNLP First workshop on Unsupervised Learning in NLP, Edinburgh, Scotland, 2011.

[16] Shamma, D., Kennedy, L., Churchill, E.: Tweet the debates: understanding community annotation of uncollected sources. In: Proceedings of the first SIGMM workshop on Social media. pp. 3–10. ACM (2009)

[17] Hu, X., Tang, J., Gao, H., Liu, H.: Unsupervised sentiment analysis with emotional signals. In: Proceedings of the 22nd international conference on World Wide Web. pp. 607–618. International World Wide Web Conferences Steering Committee (2013)

[18] Rowe, M., & Alani, H. (2014, June). Mining and comparing engagement dynamics across multiple social media platforms. In Proceedings of the 2014 ACM conference on Web science (pp. 229-238). ACM.