D3.1 Machine Learning and Artificial Intelligence for Digital Interaction Intelligence; State-of-the-art and Guidance Report

WP3 Digital Interaction Intelligence Techniques – T3.1 DII software SoTA and guidance

Delivery Date:	*Project Number*:	*Responsible partner*:
M12 - 30/11/2017	ITEA3 Call2 15011	SIVECO ROMANIA SA

DOCUMENT CONTRIBUTORS

Name	Company	Email
Mirela Ardelean	SIVECO Romania	mirela.ardelean@siveco.ro
Cristina Ivan	SIVECO Romania	cristina.ivan@siveco.ro
Raúl Santos de la Cámara	HI Iberia Ingeniería y Proyectos	rsantos@hi-iberia.es
Arun Kumar	Taiger Spain	Arun.kumar@taiger.com

DOCUMENT HISTORY

Version	Date	Author	Description
0.1	03.08.2017	SIVECO Romania	Initial ToC
0.2	09.11.2017	HIB	Add section Natural Language Processing and Opinion Mining
0.3	22.11.2017	Taiger Spain	Add section Machine learning
0.4	10.12.2017	SIVECO Romania	Add contribution in Pattern Recognition section First integrated version
0.5	20.12.2017	Taiger Spain	Added elaborated Machine learning section
0.6	03.01.2018	Taiger Spain	Reviewed
1.0	05.01.2018	SIVECO Romania	Release final version

TABLE OF CONTENTS

ABBREVIATIONS

BNP	Bayesian non parametric
BFGS	Broyden–Fletcher–Goldfarb–Shanno
DID	Digital Interaction Data
DII	Digital Interaction Intelligence
NER	Named Entity Recognition
NLP	Natural Language Processing
POS	Part of Speech
SoTA	State of the Art
WP	Work package

REFERENCES

[1]. *Jignian Chen*. Feature selection for text classification with Naïve Bayes
[2]. http://sentiment.christopherpotts.net/classifiers.html#nb
[3]. Introduction to pattern recognition system. *http://shodhganga.inflibnet.ac.in/bitstream/10603/25143/7/07_chapter%201.pdf (3)*
[4]. http://www.oracle.com/technetwork/database/options/advanced-analytics/odm/odm-techniques-algorithms-097163.html
[5]. https://www.redhat.com/en/about/press-releases/red-hat-unveils-big-data-and-open-hybrid-cloud-direction
[6]. https://www.cs.waikato.ac.nz/~ihw/papers/99IHW-EF-LT-MH-GH-SJC-WEKA.pdf
[7]. https://www.rtinsights.com/using-apache-spark-machine-learning-to-predict-consumer-behavior/
[8]. http://www.ijste.org/articles/IJSTEV3I10225.pdf
[9]. http://www.corejavaguru.com/bigdata/storm/introduction
[10]. http://sci-hub.cc/10.1145/2723372.2742788
[11]. https://dzone.com/articles/6-sparkling-features-apache
[12]. http://www.metistream.com/comparing-hadoop-mapreduce-spark-flink-storm/
[13]. http://www.developintelligence.com/blog/2017/02/comparing-contrasting-apache-flink-vs-spark/
[14]. https://www.imaginea.com/apache-spark/
[15]. http://www.computerweekly.com/feature/Apache-Spark-speeds-up-big-data-decision-making
[16]. https://spark-summit.org/east-2016/events/insights-into-customer-behavior-from-clickstream-data/
[17]. *Bishop, C. M.* (n.d.). Pattern recognition and machine learning
[18]. *Go, A.* (n.d.). Twitter Sentiment Classification using Distant Supervision.
[19]. *Križan, V.* (n.d.). Social media analysis using pattern recognition.
[20]. *Nicolov, N.* (n.d.). Spring Symposium on Computational Approaches to Analysing Weblogs.
[21]. *Y. Liu, X. H.* (n.d.). A sentiment-aware model for predicting.
[22]. *Pineda, F. J.* (1987). Generalization of back-propagation to recurrent neural networks. Physical review letters, 59(19), 2229.
[23]. *Sepp Hochreiter and Jürgen Schmidhuber. 1997*. Long Short-Term Memory. Neural Comput. 9, 8 (November 1997), 1735-1780.
[24]. *Sutton, C., & McCallum, A.* (2012). An introduction to conditional random fields. Foundations and Trends® in Machine Learning, 4(4), 267-373.
[25]. *Forney, G. D.* (1973). The viterbi algorithm. Proceedings of the IEEE, 61(3), 268-278.
[26]. *Choi, Y., Cardie, C., Riloff, E., & Patwardhan, S.* (2005, October). Identifying sources of opinions with conditional random fields and extraction patterns. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (pp. 355-362). Association for Computational Linguistics.
[27]. *Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B.* (1998). Support vector machines. IEEE Intelligent Systems and their applications, 13(4), 18-28.
[28]. *Tony Mullen, Nigel Collier.* 2004. Sentiment Analysis using Support Vector Machines with Diverse Information Sources. Proceedings of EMNLP 2004, pages 412–418, Barcelona, Spain. Association for Computational Linguistics.
[29]. *Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J. ... & Kudlur, M.* (2016, November). TensorFlow: A System for Large-Scale Machine Learning. In OSDI (Vol. 16, pp. 265-283).
[30]. *Kingma, D., & Ba, J.* (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[31]. *Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., ... & Bengio, Y.* (2016). Theano: A Python framework for fast computation of mathematical expressions. arXiv preprint.
[32]. *Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C.* (2011, June). Learning word vectors for sentiment analysis. In Proceedings of the 49th Annual Meeting of the

Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 142-150). Association for Computational Linguistics.

[33]. *McCallum, Andrew Kachites.* MALLET: A Machine Learning for Language Toolkit. http://mallet.cs.umass.edu. 2002.

[34]. *Chih-Chung Chang and Chih-Jen Lin*, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.

[35]. bnpy: Reliable and scalable variational inference for Bayesian nonparametric models." *Michael C. Hughes and Erik B. Sudderth*. Probabilistic Programming Workshop at NIPS 2014.

1. INTRODUCTION

The overall objective of Task 3.1 is to provide a living or dynamic document that describes the state of the art in the domain of Digital Interaction intelligence and a guidance report captured from SoMeDi experience. The focus will be on techniques and approaches in machine learning and artificial intelligence. The output of Task 3.1 consists of this current deliverable D3.1.

Deliverable D3.1 will have two iterations with the objective to have an ongoing view of state-of-the-art in DII domain. The first iteration compiles the current trendy algorithms and open source tools from machine learning, pattern recognition, natural language processing and opinion mining while the second iteration – that will be delivered one year later - will update the state-of-the-art section and will provide a guidance report, construct based on our experience gathered during the first two years of SoMeDi project.

The current document represents the first iteration of D3.1. The material presented here is the result of the work performed by the responsible partners: they conducted a literature review and a field research, based on their knowledge and experience, to identify the most suitable algorithms and the available tools that can be used to develop the SoMeDi's DID toolkit. The field research was conducted in the above mentioned domains: machine learning, pattern recognition, natural language processing and opinion mining.

This first iteration of D3.1 document is organised as follows:

- Section 2 describes algorithms and open tools available for Machine Learning
- Section 3 describes algorithms and open tools available for Pattern Recognition
- Section 4 describes algorithms and open tools available for Natural Language Processing and Opinion mining
- Section 5 concludes the document

2. DATA MINING AND MACHINE LEARNING

Digital data analysis depends heavily on machine learning and data mining techniques.

2.1. DATA MINING

Data mining is the computing process of discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems. It is an essential process where intelligent methods are applied to extract data patterns. It is an interdisciplinary subfield of computer science. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Aside from the raw analysis step, it involves database and data management aspects, data pre-processing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and online updating. Data mining is the analysis step of the "knowledge discovery in databases" process, KDD.

The process of data mining has the following steps:

(1) Pre-processing

(2) Data mining

(3) Interpretation/evaluation.

(1) Pre-processing

Before data mining algorithms can be used, a target data set must be assembled. As data mining can only uncover patterns actually present in the data, the target data set must be large enough to contain these patterns while remaining concise enough to be mined within an acceptable time limit. A common source for data is a data mart or data warehouse. Pre-processing is essential to analyse the multivariate data sets before data mining. The target set is then cleaned. Data cleaning removes the observations containing noise and those with missing data.

(2) Data Mining

Data mining involves six common classes of tasks:

- Anomaly detection (outlier/change/deviation detection) – The identification of unusual data records, that might be interesting or data errors that require further investigation. Anomaly detection (also outlier detection) is the identification of items, events or observations which do not conform to an expected pattern or other items in a dataset. Typically, the anomalous items will translate to some kind of problem such as bank fraud, a structural defect, medical problems or errors in a text. Anomalies are also referred to as outliers, novelties, noise, deviations and exceptions. In particular, in the context of abuse and network intrusion detection, the interesting objects are often not rare objects, but unexpected bursts in activity. This pattern does not adhere to the common statistical definition of an outlier as a rare object, and many outlier detection methods (in particular unsupervised methods) will fail on such data, unless it has been aggregated appropriately. Instead, a cluster analysis algorithm may be able to detect the micro clusters formed by these patterns
- Association rule learning (dependency modelling) – Searches for relationships between variables. For example, a supermarket might gather data on customer purchasing habits. Using association rule learning, the supermarket can determine which products are frequently bought together and use this information for marketing purposes. This is sometimes referred to as market basket analysis. Moreover, association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using some measures of interestingness.

- Clustering – is the task of discovering groups and structures in the data that are in some way or another "similar", without using known structures in the data. Moreover, Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters). Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem. The appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) depend on the individual data set and intended use of the results. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It is often necessary to modify data pre-processing and model parameters until the result achieves the desired properties.
- Classification – is the task of generalizing known structure to apply to new data. Classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. An example would be assigning a given email into "spam" or "non-spam" classes or assigning a diagnosis to a given patient as described by observed characteristics of the patient (gender, blood pressure, presence or absence of certain symptoms)
- Regression – attempts to find a function which models the data with the least error that is, for estimating the relationships among data or datasets. Moreover, Regression analysis is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed. Most commonly, regression analysis estimates the conditional expectation of the dependent variable given the independent variables – that is, the average value of the dependent variable when the independent variables are fixed. Less commonly, the focus is on a quantile, or other location parameter of the conditional distribution of the dependent variable given the independent variables. In all cases, a function of the independent variables called the regression function is to be estimated. In regression analysis, it is also of interest to characterize the variation of the dependent variable around the prediction of the regression function using a probability distribution.
- Summarization – providing a more compact representation of the data set, including visualization and report generation.

These tasks are typically done using machine learning.

2.2. MACHINE LEARNING

Machine Learning is a branch of computer science that enables computers to act without explicitly programmed. Machine learning uses statistical and pattern matching techniques to accomplish prediction and data analysis task. Moreover, machine learning is a subfield of artificial intelligence. Its goal is to enable computers to learn on their own. A machine's learning algorithm enables it to identify patterns in observed data, build models that explain the world, and predict things without having explicit pre-programmed rules and models.

In machine learning, the major objective is to find a model or function that maps input variables to output variables. Consider we have set of input variables (X) and set of output variables (Y), the objective of machine learning is to find a function *f* that maps the input variable *(X)* to *(Y)*.

$$Y = f(X)$$

This function *f* can be learned from the data. Based on how we learn the function *f,* machine learning techniques are classified into three major categories:

1. Supervised Learning
2. Unsupervised Learning
3. Deep learning

2.2.1. Supervised Learning

In supervised learning, target function *f* is learned from set of labelled examples. As a reason the training data consist of both input and desired result. In the case of sentiment analysis, supervised machine learning requires training samples that are classified to both positive and negative. The requirement of labelled data for training leads other problems; if the training dataset size is small the models are highly probable to over-fitting. Typically, supervised learning is known as classification.

The major algorithms in this category for sequence labelling. Sequence labelling is a type of pattern recognition task that involves the algorithmic assignment of a categorical label to each member of a sequence of observed values. A common example of a sequence labelling task is part of speech tagging, which seeks to assign a part of speech to each word in an input sentence or document. Sequence labelling can be treated as a set of independent classification tasks, one per member of the sequence. However, accuracy is generally improved by making the optimal label for a given element dependent on the choices of nearby elements, using special algorithms to choose the globally best set of labels for the entire sequence at once. The major algorithms for sequence labelling are:

A) Conditional Random Fields (CRF)

Conditional Random Fields [24] are class of probabilistic models that are typically used for structured predictions. They are discriminative undirected probabilistic graphical models that capture dependency between features and target predictions. The model as follows:

$$P(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{Z(\mathbf{x})} \exp \sum_{i=1}^{n} \sum_{j} \lambda_j f_j(y_{i-1,} y_i, \mathbf{x}, i)$$

where *y* is the output variable, ***x*** is the input variable and $f_j\ (y_{i-1,}\ y_{i,}\ x\ i)$ is the feature function that explains the connection between input and output variables.

Z(x) can be represented as follows

$$Z(\mathbf{x}) = \sum_{y \in Y} \sum_{i=1}^{n} \sum_{j} \lambda_j f_j(y_{i-1,} y_i, \mathbf{x}, i)$$

The inference on these models are done using Viterbi algorithm [25] as exact inference is not tractable. Sentiment analysis problem is modelled using CRF model due to the ability of structured prediction [26].

B) Support Vector Machines (SVM)

Support Vector Machines [27] are machine learning algorithms that can be used for both classification and regression. Moreover, they are discriminative classifiers work by defining various hyperplanes for classification and regression. Given a set of training data, SVM outputs an optimized hyperplane that categorizes the training data; due to this property SVMs are successfully used in sentiment analysis and opinion mining [28].

More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outliers detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

Open Source Tools of Supervised Learning

- Mallet

MALLET (MAchine Learning for LanguagE Toolkit) [33] is a Java-based package for statistical natural language processing, document classification, clustering, topic modeling, information extraction, and other machine learning applications to text. It supports various machine learning algorithms such as Naïve Bayes and Conditional Random Fields. It also supports sequence labeling.

- LIBSVM

LIBSVM [34] is an integrated software for support vector classification, (C-SVC, nu-SVC), regression (epsilon-SVR, nu-SVR) and distribution estimation (one-class SVM). It supports multi-class classification. It is open source and written in C++. It is supported by National University of Taiwan.

- CRF ++

CRF++ is a simple, customizable, and open source implementation of Conditional Random Fields (CRFs) for segmenting/labeling sequential data. CRF++ is designed for generic purpose and will be applied to a variety of NLP tasks, such as Named Entity Recognition, Information Extraction and Text Chunking. It supports fast training based on L-BFGS (Limited-memory BFGS), a quasi-newton algorithm for large scale numerical optimization problems. It is licensed under GNU Lesser General Public License.

2.2.2. Unsupervised Learning

In unsupervised learning, the task is to infer a function that describes unlabelled data. In unsupervised learning setup the machine learning algorithms are provided with un labelled data set for training. As the machine learning function is learned from unlabelled data, these methods are less susceptible to overfitting problem. Unsupervised machine learning algorithms typically known as clustering algorithms. One of the most important clustering algorithms is K-mean clustering.

A) K-means clustering

The major steps of K-means as follows:

Given a set of data points in set X with set of labels K clusters:

1. Define the *k* centroids. Initialize these at random

2. Find the closest centroid and update cluster assignments. Assign each data point in X to one of the *k* clusters. Each data point is assigned to the nearest centroid's cluster. Here, the measure of "nearness" is a hyper parameter — often Euclidean distance.

3. Move the centroids to the center of their clusters. The new position of each centroid is calculated as the average position of all the points in its cluster.

Keep repeating steps 2 and 3 until the centroid stop moving a lot at each iteration (i.e., until the algorithm converges).

B) Bayesian Non Parametric Models

Bayesian non parametric (BNP) models are statistical models that have infinite number of parameters. They are typically used for density estimation and clustering. As the density of the probability models are estimated from the data set, these models not require any labelled data sets. The major drawback of BNP models is large training time as BNP models do not have exact equations to get the prediction from the models. The inference of the models is typically done using sampling techniques, which is time consuming. For data analysis tasks Bayesian models are rarely applied.

Open Source Tools of Unsupervised Learning

- Bnpy

The goal of bnpy is to make it easy for Python programmers to train state-of-the-art clustering models on large datasets. It focuses on nonparametric models based on the Dirichlet process, especially extensions that handle hierarchical and sequential datasets. Traditional parametric counterparts (like finite mixture models) are also supported. Training a model with bnpy requires the user to specify the dataset, the model, and the algorithm to use. Flexible keyword options allow advanced users lots of control, but smart defaults make it simple for beginners. bnpy's modular implementation makes it possible to try many variants of models and algorithms, to find the best fit for the data at hand. bnpy trains a model in two ways: (1) from a command line/terminal, or (2) from within a Python script (of course). Both options require specifying a dataset, an allocation model, an observation model (likelihood), and an algorithm. Optional keyword arguments with reasonable defaults allow control of specific model hyper parameters, and algorithm parameters.

2.2.3. Deep Learning

Deep learning is a new branch of machine learning, where layers of neural networks are trained to perform an analysis task. Neural network is a beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data. In all these models the training of the neural network is done using backpropagation. Backpropagation, short for "backward propagation of errors" is an algorithm for supervised learning of artificial neural networks using gradient descent. Given an artificial neural network and an error function, the method calculates the gradient of the error function with respect to the neural network's weights. It is a generalization of the delta rule for perceptron to multilayer feedforward neural networks. There are various kind of neural networks that could be used for various data analysis applications. However, the data analysis tasks such as sentiment analysis are typically carried out by modelling it using Long Term Short Term Memory (LSTMS) and Recurrent Neural Networks.

A) Recurrent Neural Networks (RNN)

Recurrent Neural Networks [22] are a special category of neural networks that designed to capture sequential information. A traditional neural network fails to assume dependency between inputs and outputs as a reason it fails to account sequential information. Sequential information is important for modelling language data hence sentiment analysis. A recurrent neural network has the following structure:

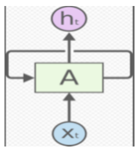


FIGURE 1. RECURRENT NEURAL NETWORK

The above diagram represents a recurrent neural network, where A takes an input x_t and outputs h_t and the loops allows which information to be passed to next step of the neural networks.

If unroll the network A, it will look like this:

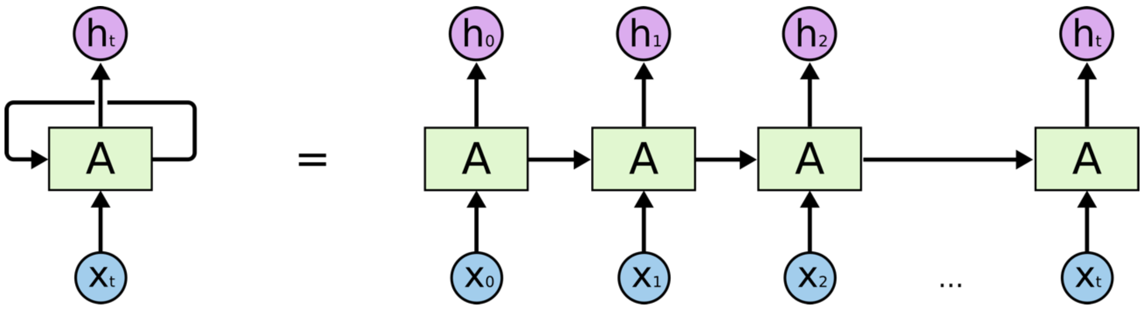


FIGURE 2. RECURRENT NEURAL NETWORKS ELABORATED

The above diagram can be represented in mathematical notation in following way:

$$h_t = \tanh(w_{ih} * x_t + b_{ih} + w_{hh} * h_{(t-1)} + b_{hh})$$

where h_t hidden state at time *t*, *tanh* is hyperbolic tangent function or a squeezing function, W_{ih} is the weights for each neurons in a particular layer, x_t is state of the previous layer or input layer and b_{hh} is the bias for neuron a particular layer.

B) Long Term Short Term Memory (LSTMS)

LSTMS [23] are a special kind of Recurrent Neural Networks that can capture long term dependencies between inputs, which is important for sentiment analysis and other natural language processing tasks.

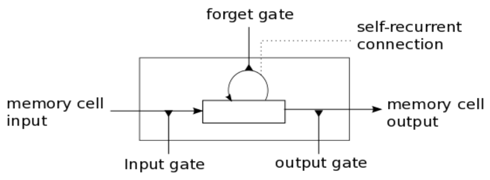


FIGURE 3. LONG TERM / SHORT TERM MEMORY

Compared to normal neuron, a LSTM has got a forget gate that enables the neuron to remember its states. An LSTM layer consists of a set of recurrently connected blocks, known as memory blocks. These blocks can be thought of as a differentiable version of the memory chips in a digital computer. Each one contains one or more recurrently connected memory cells and three multiplicative units – the input, output and forget gates – that provide continuous analogues of write, read and reset operations for the cells. The net can only interact with the cells via the gates.

The Long Short Term Memory architecture was motivated by an analysis of error flow in existing RNNs which found that long time lags were inaccessible to existing architectures, because back propagated error either blows up or decays exponentially. LSTMs are successfully used in sentiment analysis, where the sentiment is modelled as long range dependencies between neurons [32]

The equations for LSTMs as follows:

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}
$$

Variables in LSTM:

- $x_t \in R^d$: input vector to the LSTM block
- $f_t \in R^h$: forget gate's activation vector
- $i_t \in R^h$: input gate's activation vector
- $o_t \in R^h$: output gate's activation vector
- $h_t \in R^h$: output vector of the LSTM block
- $c_t \in R^h$: cell state vector
- $W \in R^{h\times d}, U \in R^{h\times h}$ and $b \in R^h$: weight matrices and bias vector parameters which need to be learned during training

The where the activation function can be Sigmoid or hyperbolic tangent function.

Open Source Tools for Deep Learning

- TensorFlow

Tensorflow [29] is a large scale heterogeneous machine learning library for deep learning. It supports various neural network architectures such as RNNs and LSTMs. The major advantage of this framework is the computation is done in graph structures. As a reason machine learning using this framework can be distributed to many systems in a cluster or inside Graphical Processing Units (GPU). Moreover, it also supports fast optimization algorithms and training algorithms. For instance, it supports ADAM [30] which is a new fast algorithm for optimization. One another major advantage of Tensorflow library is the large online community of supporters.

- Theano

Theano [31]is a python based library for machine learning. It can be used to design, develop and optimize mathematical expressions that involve multi-dimensional arrays. Furthermore, it supports symbolic differentiation, which is important for fast computation in a distributed setup. This library supports most of the neural network architectures including RNN and LSTMs.

- DeepLearning4J

DeepLearning4J is a java based machine learning library for deep learning. It supports various neural network architectures including RNN and LSTMS. The major advantage of this library is it is distributed in Java Virtual Machine. Furthermore, it also supports Graphical Processing Units (GPUs) which makes the computation faster.

- Apache Maxnet

Apache MAXNET is a deep learning library that supports various programming languages including C++, Julia, and Python. It is an open source library licensed under apache license.

The major components of Apache Maxnet are:

1. Runtime Dependency Engine: Schedules and executes the operations according to their read/write dependency.

2. Storage Allocator: Efficiently allocates and recycles memory blocks on host (CPU) and devices (GPUs).
3. Resource Manager: Manages global resources, such as the random number generator and temporal space.
4. NDArray: Dynamic, asynchronous n-dimensional arrays, which provide flexible imperative programs for MXNet.
5. Symbolic Execution: Static symbolic graph executor, which provides efficient symbolic graph execution and optimization.
6. Operator: Operators that define static forward and gradient calculation (backprop).
7. SimpleOp: Operators that extend NDArray operators and symbolic operators in a unified fashion.
8. Symbol Construction: Symbolic construction, which provides a way to construct a computation graph (net configuration).
9. KVStore: Key-value store interface for efficient parameter synchronization.
10. Data Loading(IO): Efficient distributed data loading and augmentation.

- Keras: The Python Deep Learning library

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. It can seamlessly on CPU and GPU.

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.

Keras is:

- **User friendly**: Keras is an API designed for human beings, not machines. It puts user experience front and center. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear and actionable feedback upon user error.
- **Modular:** A model is understood as a sequence or a graph of standalone, fully-configurable modules that can be plugged together with as little restrictions as possible. In particular, neural layers, cost functions, optimizers, initialization schemes, activation functions, regularization schemes are all standalone modules that you can combine to create new models.
- **Extensible:** New modules are simple to add (as new classes and functions), and existing modules provide ample examples. To be able to easily create new modules allows for total expressiveness, making Keras suitable for advanced research.

2.2.4. Summary of Machine Learning Tools

The table below illustrates a comparison between the presented open tools.

Name of the tool	Development	Latest release	Operating System	Licence	Language
Mallet	McCallum Andrew and team	MALLET 2.0	Cross Platform	CLP 1.0	Java
LIBSVM	National University of Taiwan	2.88	Cross Platform	BSD	Java/C/ Python
CRF++	Nara Institute of Science and Technology	CRF++ 0.58	Cross Platform	BSD	C++
Bnpy	Brown University	0.1	Cross Platform	BSD	Python
TensorFlow	Google Brain Team	1.3	Cross Platform	Apache 2.0	Python
DeepLearning4J	Skymind.	0.9.1	Cross Platform	Apache 2.0	Java
Theano	Université de Montréal	1.0.0	Cross Platform	BSD	Python
Apache MXNet	The Apache Software Foundation	1.0	Cross Platform	Apache 2.0	Python/ C++
Keras	François Chollet and team	2.0.8	Cross Platform	MIT	Python

TABLE 1 SUMMARY OF MACHINE LEARNING TOOLS

Machine Learning for Sentiment Analysis

With the rapid development of information, the era of information networks is affecting people's lives at unprecedented rates. At the same time, social media also presents a variety of forms, BBS, blogs and others network media have developed rapidly, and the network user's participation is continuing to raise, for the use of network also has a huge change. The users are no longer a passive knowledge of access network, but to become a maker of network information. The change has resulted in a present a large number of network media used to express the user emotions, emotions and ideas of all kinds of forms of subjective information, and the text is one of the most main forms. According to these subjective information, how to more effective use of these vast amounts of data, extract, carrying people interested in view of the text, and its accurate analysis is so far in the field of natural language processing is one of the important research topic.

Text Sentiment Analysis is a very important research topic in the field of natural language processing, and it is focus on people's ideas, feelings, attitudes, and for such products, services, organizations, individuals, events, themes, such as entity emotions tend to make the effective mining and Analysis, and then further to dig out the category of technology information induction and reasoning.

Various machine learning techniques have been applied to sentiment analysis task. The best performing method for this task is Recurrent Neural Networks (RNN) and LSTMs, where a layer of LSTMs are trained for the task.

3. PATTERN RECOGNITION

Pattern recognition is concerned with the investigation of adaptive and analytic techniques for processing large amounts of data, the extraction of useful information to reduce the data, and the classification the data as required. It is closely akin to machine learning, and also finds applications in fast emerging areas such as biometrics, bioinformatics, multimedia data analysis and most recently data science. The great variety of pattern-recognition problems makes it difficult to say precisely what pattern recognition is. However, a good idea of the scope of the field can be gained by considering some typical pattern-recognition tasks. Pattern recognition has its origins in engineering, and the term is popular in the context of computer vision. In pattern recognition, there is a higher interest to formalize, explain and visualize the pattern, while machine learning focuses on maximizing the recognition rates.

Previous studies have examined many different factors that play a role in the recognition of opportunities for new business ventures. Among these, however, three have been identified as especially important and received most attention: engaging in an active search for opportunities: alertness to opportunities (the capacity to recognize them when they emerge); and prior knowledge of a market, industry, or customers as a basis for recognizing new opportunities in these areas.

The field of pattern recognition is concerned mainly with the description and analysis of measurements taken form physical or mental process. It consists of acquiring raw data and taking actions based on the class of the patterns recognized in the data. Earlier it was studied as a specialized subject due to higher cost of the hardware for acquiring the data and to compute the answers. The fast developments in computer technology and resources enhanced possible various practical applications of pattern recognition, which in turn contributed to the demands for further theoretical developments.

3.1. PATTERN RECOGNITION APPROACHES

Patterns generated from the raw data depend on the nature of the data. Patterns may be generated based on the statistical feature of the data. There are different mathematical techniques in order to obtain the attributes for pattern recognition system. The four best-known approaches for the pattern recognition are:

1. Template matching
2. Statistical classification
3. Syntactic matching
4. Neural networks

1. In **template matching**, the prototype of the pattern to be recognized is compared against the pattern to be recognized. In the statistical approach, the patterns are described as random variables, from which class densities can be inferred. Classification is done based on the statistical modelling of data. In the syntactic approach, a pattern is seen as being composed of simple sub-patterns which are themselves built from yet simpler sub-patterns, the simplest being the primitives. Inter relationships between these primitive patterns are used to represent a more complex pattern. The neural network approach to pattern recognition is strongly related to the statistical methods, since they can be regarded as parametric models with their own learning scheme. The comparison of different approaches is summarized in TABLE 2 [3]

Approach	Representation	Recognition Function	Typical Criterion
Template matching	Samples, pixels, curves	Correlation, distance measure	Classification error
Statistical classification	Features	Discriminant function	Classification error
Syntactic or structural	Primitives	Rules, grammar	Acceptance error
Neural network	Samples, pixels, features	Network function	Mean square error

TABLE 2 PATTERN RECOGNITION MODELS

3.2. PATTERN RECOGNITION SYSTEM

A pattern recognition system, in general, consists of two parts, namely, feature extractor and classifier. The function of feature extractor is to extract or to measure the important characteristics from the input patterns. The extracted characteristics are called features, and they are supposed to best characterize all the possible input patterns. The function performed by a classifier is the one of making a proper decision to assign each input pattern to one of the possible pattern classes. The decision is made on the basis of the feature measurements supplied by the feature extractor in a recognition system. The classification criterion is usually [17].

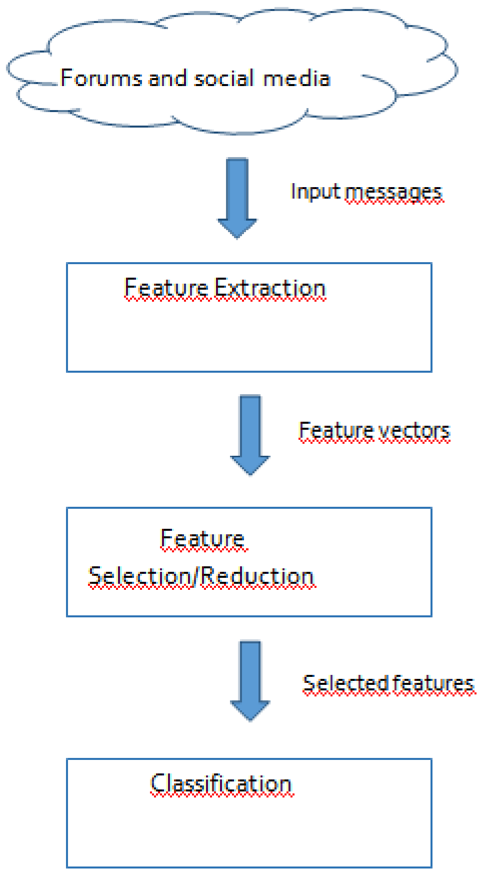


FIGURE 4. PATTERN RECOGNITION SYSTEM

3.2.1. FEATURE EXTRACTION

Text contains word, letters, punctuation or other symbols, and may be arrange into patterns. These characteristics are known features (or attributes). Features are a part of the learning and evaluation process, and they are used by classifier for finding the differences between classes. Frequently used features by researches are n-grams, patterns, punctuation characters, emoticons (especially in blogs and chats) uppercases and count features (counts of a specific feature) [19].

N-GRAMS

N-grams are sequences of n items, such as words, letters, syllables, etc. The purpose of n-grams is to compute the joint of sequence probabilities or probability of an upcoming item in the sequence. A simple n-gram model is the unigram model, where n is 1. The next most used n-grams are bigrams, which are suitable for two-word phrases, including negative phrases, like "not good". The disadvantage of n-gram is that most languages have long-distant dependencies. This drawback makes the n-gram model unsuitable for text data information extraction when using it as the only model.

The n-grams might not be efficient for sentiment analysis. Unigrams can be supported by part-of-speech (POS) tags as meta features. Besides of the POS tags, they mapped the words to their prior subjectivity and polarity. The prior polarity is switched from positive to negative, or vice-versa, by the occurrence of negative word (not) or negative expression. They derived the subjectivity and polarity of the tag from the subjectivity lexicon.

PUNCTUATION AND EMOTION

Punctuations are a part of sentences, and can be used as generic features. On the one hand the exclamation marks and question marks can emphasize the textual emotional content. For example, the repetition of exclamation marks significantly indicates anger.

On the other hand, emoticons are textual representations of emotions. They are usually written with ASCII characters. Nowadays, ideograms, represented by pictograph, are getting more popular in electronic messages and social networks. These pictographs are called emojis, and are represented by a UTF-8 encoded character. As an analogy to the ASCII emoticons, the emojis represent their emotions. Emoticons and emojis are considered as the most significant carries of information in the textual content.

PATTERN-BASED FEATURES

One of the pattern based feature is based on the arrangement of texts into high frequency words (HFW) and content words (CW). HFW is defined as words appearing more than T_H times, per million words, and CW as words appearing more than T_C times, per million words. T_H and T_C are thresholds, which were set to 1000 words per million for T_C, and 100 words per million for T_H.
Class sequential rules (CSR) are the next approach of pattern-based features. CSR is a sentence-level pattern for the sentiment/emotion classification. Two emotion labels are obtained for each sentence, by lexicon-based and SVM-based methods. The derived CSR feature is subsequently used for SVM- based emotion classification. Sentences with conjunction words, such as “but” or “where”, are separated before the feature extraction. These conjunction words are added into the sequence, since they reflect relationship between the sub-sentences.

3.2.2. FEATURE SELECTION/REDUCTION

Extraction of all possible features from the training set is not an effective approach, since it can cause over fitting or misclassification due to the inclusion of redundant or irrelevant features. To solve these issues, the number of input features should be reduced by either selecting a certain number of them or by transforming the whole feature space to a new space with smaller number of dimensions.

In the feature selection, the issue is to acquire a considerable reduction of the features, without affecting the performance of the classifier.

For example, according to this study [18] , Twitter language model has many unique properties, thus it is possible to use these properties in order to reduce the feature space.

Usernames: Users often include Twitter usernames in their tweets in order to direct their messages, where they have to include the @ symbol before the username. For example, an equivalence class token (USERNAME) replaces all words that start with this symbol.

Usage of links: Users very often include links in their tweets. An equivalence class is used for all URLs. One option is converting a URL like “http://” to the token “URL”

Repeated letters: Tweets contains very casual language. Many users write the same word with a different number of letters. For instance, the search of hungry can be like “huungry” or “huuungry “. In this case can be use a pre-processing, in which any letter can occurring more than two times in a row is replaced with two occurrences. In the samples above, these words would be converted into the token huungry.

These three reductions shrink the feature set down to 45.85% of its original size.

In the feature selection, the issue is to acquire a considerable reduction of the features, without affecting the performance of the classifier.

3.2.3. TEXT CLASSIFICATION

Classification is the last step of the pattern recognition process, assigns the input feature vector to a certain class. There are several algorithms for the classification, the most known among them for emotion recognition are: decision trees, rules, Bayes methods, nearest neighbour classifiers, SVM classifiers, and neural networks.

Sentiment classification techniques can be differentiated into machine learning approach and lexicon based approach. The machine learning approach (ML) applies the well-known ML algorithms and uses semantic features. The Lexicon-based Approach depends with the respect to sentiment lexicon. It is partitioned into dictionary-based approach and corpus-based approach which utilize statistical or semantic methods to find sentiment polarity. The hybrid Approach combines both approaches/methodologies and is exceptionally basic with sentiment lexicons playing a key part in the majority of methods.

Machine Learning Approach

The text classification methods using Machine Learning Approach can be differentiated into supervised and unsupervised learning methods. In general, the machine learning approach relevant to sentiment analysis mostly belongs to supervised classification and text classification techniques in particular.

Machine learning techniques like Naive Bayes (NB), maximum entropy (ME), and support vector machines (SVM) have made extraordinary success in text categorization.

A) Naive Bayes.

The Naive Bayes classification method is based on Bayes rule, and relies on a simple representation of document. The aim of this method is to choose the class with a highest probability. The accuracy of Naive Bayes using unigrams and bigrams in Twitter is the 82.7%. Sentiment was represented as positive, negative and neutral.

There are two different models of Naïve Bayes: The Multi-Variate Bernoulli Event Model and the Multinomial Event Model. In the multi-variate Bernoulli event model, a vocabulary V is given. A document is represented with a vector of |V| dimensions. The dimension of the vector corresponds to word wk from V and is either 1 or 0, indicating whether word wk occurs in the document. To simplify the calculation of probability, the Naïve Bayes assumption is made in this model: that in a document the probability of the occurrence of each word is independent of the occurrence of other words.

In the multinomial event model, a document is regarded as a “bag of words”. No order of the words is considered, but the frequency of each word in the document is captured. In this model, a similar Naïve Bayes assumption is made: that the probability of the occurrence of each word in a document is independent of the word’s position and the occurrence of other words in the document [1].

The Naïve Bayes model is tremendously appealing because of its simplicity, elegance, and robustness. It is one of the oldest formal classification algorithms. It is widely used in areas such as text classification and spam filtering.

B) MaxEnt

Maximum Entropy (MaxEnt) is a feature-based model that uses search-based optimization to find weights for the features that maximize the likelihood of the training data. The MaxEnt model does not make an independence assumption of features, like in the Naïve Bayes, and therefore overlapping features have no effect on the classification.

The MaxEnt classifier is closely related to a Naïve Bayes classifier, except that, rather than allowing each feature to have its say independently, the model uses search-based optimization to find weights for the features that maximize the likelihood of the training data.

The MaxEnt model can also handle mixtures of Boolean, integer and real-valued features. The MaxEnt classifier can be built in the following steps, assuming that the only features are word-level features.

1. For each word w and class c Є C, define a joint feature f(w,c) = N where N is the number of times that w occurs in a document in class c.
2. Via iterative optimization, assign a weight to each joint feature so as to maximize the log-likelihood of the training data.
3. The probability of class c given a document d and weights λ is

$$P(c|d,\lambda) \overset{def}{=} \frac{\exp \sum_i \lambda_i f_i(c,d)}{\sum_{c' \in C} \exp \sum_i \lambda_i f_i(c',d)}$$

MaxEnt models are more difficult to implement than Naïve Bayes model, but MaxEnt has an excellent software packages available [2].

C) Support Vector Machines

Support Vector Machines (SVMs) is another popular classification technique. It pre-processes data in high dimensions, much higher than the original feature space. The main idea behind the SVMs, is to find a separating hyperplane, with the largest margin in the learning process. SVMs are defined by vectors of points closet to the separating hyperplane. The best hyperplane is defined by the largest margin between two classes.

Lexicon-based Approach

The lexicon-based approach based on finding the opinion lexicon which is used to analyse the text. Opinion words are utilized in many sentiment classification tasks. Positive opinion words are utilized to express some desired sates, while negative opinion words are utilized to express some undesired states. There are also opinion phrases and expression which together are called opinion lexicon. The lexicon-based approach is partitioned into two methods. The dictionary-based approach based on finding opinion seed words, and then searches the dictionary of their equivalent synonyms and antonyms. The corpus-based methodology starts with a seed list of opinion words, and then finds other opinion words in a large corpus to get opinion words with context specific orientations. This should be possible by utilizing statistical or semantic methods.

Usage statics

Readers of the content, who may or may not also be contributors, provide valuable information about the items they find interesting. In particular, usage statistics such as the number of clicks on the item and the time have been shown useful in the context of identifying high quality web search results, and are complementary to link-analysis based methods. Intuitively, usage statistics measures are useful for social media content, but require different interpretation from the previously studied settings.

3.3. TOOLS FOR PATTERN RECOGNITION

In order to enable the processing of large quantities of data SOMEDI will propose different strategies such as Apache Spark. This section will depict the different tools and the reason why Apache has been chosen.

3.3.1. APACHE SPARK

Apache Spark is a powerful open source processing engine for Hadoop data built around speed, easy to use, and sophisticated analytics. It was originally developed in UC Berkeley's AMPLab and later-on it moved to Apache. Apache Spark is basically a parallel data processing framework that can work with Apache Hadoop to make it extremely easy to develop fast, Big Data applications combining batch, streaming, and interactive analytics on all the data. In fact, a recent Sync sort report shows that 70 percent of IT managers and BI analysts are interesting in Spark over MapReduce

Spark is also well-suited for running machine-learning queries in parallel. That is useful in almost context where pattern detection is key.

This section will show some features which are really highlighting it in the Big data world:

<u>Lighting Fast Processing</u>

In Big Data processing is essential the speed. Spark enables applications in Hadoop clusters to run up to 100x faster in memory, and 10x faster even when running on disk. Spark makes it possible by reducing number of read/write to disc. It stores this intermediate processing data in-memory. It uses the concept of a Resilient Distributed Dataset (RDD), which allows it to transparently store data on memory and persist it to disc only it's needed. This helps to reduce most of the disc read and write – the main time-consuming factors – of data processing

<u>Ease of use as It supports multiple languages</u>

Spark lets you quickly write applications in Java, Scala or Python. This helps developers to create and run their applications on their familiar programming languages.

<u>Support for sophisticated analytics</u>

In addition to simple map and reduce operations, Spark support SQL queries, streaming data, and complex analytics such as machine learning and graph algorithms out-of-the-box. Furthermore, users can combine all these capabilities seamlessly in a single workflow

<u>Real time Stream Processing</u>

Spark can handle real time streaming. Map-reduce mainly handles and process the data stored already. However, Spark can also manipulate data in real time using Spark Streaming. Not ignoring that there are other frameworks with their integration we can handle streaming in Hadoop.
Here there is some abilities about Sparks:

Easy: Built on Spark's lightweight yet powerful APIs, Spark Streaming lets you rapidly develop streaming applications
Fault tolerant: Unlike other streaming solutions (e.g. Storm), Spark Streaming recovers lost work and delivers exactly-once semantics out of the box with no extra code or configuration
Integrated: Reuse the same code for batch and stream processing, even joining streaming data to historical data

Ability to integrate with Hadoop and Existing Hadoop Data.

Spark can run independently. Apart from that it can run on Hadoop 2's cluster manager, and can read any existing Hadoop data, such as HBase and HDFS. This feature of Spark makes it suitable for migration of existing pure Hadoop applications

Active and Expanding community.

Apache Spark is built by a wide set of developers from over 50 companies and is expanding continuously.

APACHE SPARK IN ORDER TO EVALUATE CUSTOMER BEHAVIOUR

In the case of consumer behaviour, because consumers are shopping for different products and exhibit varying behaviours, multiple algorithms are run to find the one best-suited to tackle a predictive task, and that is where Apache Spark excels [7].

Many different classification algorithms could be used in order to analyse, classify or predict data. These algorithms differ in their performance and results. With Apache, it is possible to use different techniques, such as Naïve Bayes and Support Vector Machines in order to select the best choice.

Spark's combination of high-performance distributed processing and extensive libraries make it an attractive platform for tackling customer segmentation. Spark's speed and flexibility make it ideal for rapid, iterative processes such as machine learning, with Postcodeanywhere. Postcodeanywhere is a provider of address data to popular e-commerce and retail websites, has been using Spark internally for more than a year to help understand and predict customer behaviour on its platform, enabling the company to improve service [15].

On the other hand, through Apache Spark is possible to use different algorithms in order to predict the CTR [14]. Click Trough Rates (CTR) is essential for measure the success of an online advertising campaign for a particular website as well as the effectiveness of email campaigns. For that reason, this parameter would be very useful for SOMEDI.

3.4. ALTERNATIVES TO APACHE SPARK

Apache Hadoop is an open source software framework. Hadoop consists of two main components: a distributed processing framework named MapReduce and a distributed file system known as the Hadoop distributed file system, or HDFS.

HDFS is used for storage. In short, HDFS provides a distributed architecture for extremely large-scale storage. On the other hand, MapReduce is a programming model for processing and generating large data sets

Through MapReduce is possible to study the consumer behaviour. For example, in this study [8] have been analysed customer behaviour using big data analytics.

Hadoop is an open-source, distributed computing technology. This configuration enables users to distribute processing of large data sets across clusters of computers to run simple programming models.

Users can scale Hadoop from a single server to thousands of servers. And, each connected server can provide its own local computation and storage [12].

For persistence this project will use HDFS in order to store the data due to its simplicity, efficiency, distribution architecture and ease of integration with Apache Spark.

3.4.1. SPARK VS HADOOP

Between Hadoop/HDFS and Spark are two fundamentally different systems and used for different purposes. Essentially, HDFS is a storage solution and Spark is fast and general engine for large-scale data processing.

Companies use Hadoop/HDFS to store data in a reliable and secure manner. Companies use Spark to make sense of that data.

Spark offers a fast technology for large-scale data processing. This framework provides high-level APIs in Java, Scala and Python. And, it provides a very rich set of data stores including streaming processing and machine learning [12].

Google introduced MapReduce as a programming model to facilitate its search processes. This framework serves two basic functions: It parcels out work to various nodes within a map (or cluster), and it organizes the workload to reduce the results into a relevant answer to a query.

One of the limitations of Hadoop MapReduce is that the framework can only batch process one job at a time [12].

3.4.2. SPARK VS MAPREDUCE

One of the significant challenges for systems developed with this framework is its high-latency, batch-mode response. Since MapReduce has evolved over the last 10 years some developers would complain that it’s difficult to maintain because of inherent inefficiencies in its design and code.

Spark, on the other hand, is faster and better suited for big data analytics. It can run on a variety of file systems and databases. Some users report that processes work on Spark 100 times faster than the same process on MapReduce.

Spark is also more developer-friendly with an API that is easier for most developers to use when compared to MapReduce.

While MapReduce is limited in “map” and “reduce” operations, Spark supports SQL queries, streaming data and complex analytics such as machine learning and graph algorithms. Furthermore, users can combine all these capabilities seamlessly in a single workflow [11].

Apache Storm is an open source, scalable, fault-tolerant, distributed real-time computation system. It is popular because of it real-time features and many organizations have implemented it a part of their system for this very reason.

Storm provides Twitter’s publisher analytics product, processing every tweet and click that happens on Twitter to provide analytics for Twitter’s publisher partners. Storm integrates with the rest of Twitter’s infrastructure, including Cassandra, the Kestrel infrastructure, and Mesos. Apache Storm is able to process customer behaviour on web pages and mobile apps [9].

Storm has long served as the main platform for real-time analytics at Twitter. However, as the scale of data being processed in real-time at Twitter has increased, along with an increase in the diversity and the number of use cases, many limitations of Storm have become apparent [10].

3.4.3. SPARK VS STORM

One key difference between these is that Spark performs Data-Parallel computations while Storm performs Task- Parallel computations. While Spark and Storm provide fault tolerance and scalability, each framework uses a different processing model. Spark uses micro-batches to process events while Storm processes events one by one. This difference in process handling means that Spark has a latency of seconds while Storm provides a millisecond of latency. Spark Streaming provides a high-level abstraction called a Discretized Stream or DStream, which represents a continuous sequence of RDDs [12].

Heron is built with a wide array of architectural improvements that contribute to high efficiency gains. It has powered all real-time analytics with varied use cases at Twitter since 2014. In addition is an API compatible with Apache Storm and hence no code change is required for migration.

On the other hand, Heron architecture is very complex. Multiple tasks are written into a single file, for that reason it is hard to identify any errors or exceptions that are associated with a particular task. Furthermore, an unhandled exception in a single task takes down the entire worker process [10].

3.4.4. SPARK VS HERON

The main disadvantage in Heron is that is not an open source, while Spark is it. Heron has already replaced the internal usage of Storm at Twitter, running hundreds of development and production topologies. However, most of the IoT platforms are using a combination of Apache Kafka, Apache Storm and Apache Spark for data ingestion, data aggregation and data analytics.

Apache Flink is an open-source stream processing framework for distributed, high-performing, always-available, and accurate data streaming applications. It is fast, reliable, scalable and easy to use. Flink can be used in different environments, like in the optimization of e-commerce search results in real-time.

Flink applications are fault-tolerant in the event of machine failure and support exactly-once semantics. Programs can be written in Java, Scala, Python and SQL and are automatically compiled and optimized into dataflow programs that are executed in a cluster or cloud environment.

3.4.5. SPARK VS FLINK

Apache Spark is considered a replacement for the batch-oriented Hadoop system. But it includes a component called Apache Spark Streaming as well. Contrast this to Apache Flink, which is specifically built for streaming [13].

The main difference is Flink was built from the ground up as a streaming product. Spark added Streaming onto their product later. Spark Streaming uses a fast-batch operation. It works on a portion of incoming data during a set period of time. While this approach to processing data works well for many situations, it may not be the best in all use cases. On the one hand, Apache Flink can provide a purer stream-processing capability with lower latency. Flink can lead Spark in the future, but actually the community of the users is lower.

Oracle Advanced Analytics provides a broad range of in-database, parallelized implementations of machine learning algorithms to solve many types of business problems.

Oracle has different techniques, such as classification, regression, attribute importance, anomaly detection, clustering, association and feature selection and extraction. In relation to the feature selection and extraction produces new attributes as linear combination of existing attributes, applicable for pattern recognition [4].

Oracle offers advertisers something akin to a real-time target marketing list, where machine learning already has determined which customers are most likely to buy the advertiser’s products.

Predictive consumer behaviour is considered the most important goal of marketing, but a classic problem is filtering out the noise from customers who are ready to buy. Web activity such as search and browsing may generate petabytes of data. This problem was solved moving Oracle Data Cloud from an on premise single machine for data processing to cloud-based Apache Hive and ultimately Apache Spark [7].

4. NATURAL LANGUAGE PROCESSING AND OPINION MINING

In SoMeDi, in order to automate the process of understanding the social media data that is usually done by human experts, we will use tools and algorithms on the Natural Language Processing realm (henceforth NLP) that try to understand the language generated by users of the social networks and then represents it in data formats that are useful for analysis by computers.

We will first explain a series of general aspects in the NLP theory. After this, in subsections 5.2 onwards we will explain in detail tools and approaches used for this and how they can be leveraged in SoMeDi to achieve the global understanding task.

4.1. General Aspects of Natural Language Processing

The following is a list of some of the most commonly researched task in NLP that are discussed in this deliverable. For each one there is typically a well-defined problem setting, a standard metric for evaluating the task, standard corpora on which the task can be trained or evaluated and competitions devoted to the specific task.

- **Sentence breaking:** given a chunk of text, find the sentence boundaries. Sentence boundaries are often marked by periods, colons or other punctuation marks, but these same characters can serve other purposes (e.g. marking abbreviations). Usually, this task is carried out by the *sentence splitter* module.

- **Word segmentation** carries out a lexical analysis, splitting texts into very simple tokens such as words, numbers, punctuation, space tokens and symbols. Usually, this task is carried out by the *tokenizer* module.

- **Part-of-speech (POS) tagging:** given a sentence, determine the part of speech for each word. Many words can serve as multiple parts of speech. For example, “set” can be a noun, verb or adjective. Usually, this task is carried out by the part-of-speech tagger module.

- **Named entity recognition (NER):** given a stream of text, determine which items in the text map to proper names, such as people or places, and what type of each such name is (e.g. person, location, organization). This identification is based on gazetteer lists. These are plain text files, with one entry per line. Each list represents a set of names, such as names of cities, organizations, days of the week… Usually, this task is carried out by the gazetteer module.

- **Parsing** makes the grammatical analysis determine a parse tree for each given sentence. The grammar for natural languages is ambiguous and typical sentences have multiple possible analyses. Usually, this task is carried out by the *parser* module.

- **Co-reference resolution:** given a sentence or larger chunk of text, determine which words (“mentions”) refer to the same objects (“entities”). The more general task of co-reference resolution also includes identifying so-called “bridging relationships” involving referring expressions. For example, in a sentence such as “He entered John´s house through the front door”, “the front door” is a referring expression and the bridging relationship to be identified is the fact that the door being referred to is the front door of John´s house (rather than of some other structure that might also be referred to). Usually, this task is carried out through a *co-reference tagger* based on co-reference chains.

- **Relationship extraction:** given a chunk of text, identify the relationships among named entities (e.g. who is the wife of whom).

- **Word sense disambiguation:** many words have more one meaning; this task is in charge of selecting which makes the most sense in context. For this problem, it is usually given a list of words and associated word senses.

In some cases, sets of related task are grouped into complete subfields of NLP that are often considered separately from NLP as a whole. Examples include:

- **Information retrieval:** this is concerned with storing, searching and retrieving shallower elements information from corpora of text. This includes search engines, indexers and similar software modules.

- **Information extraction:** this is concerned in general with the extraction of deeper semantic information from text. This covers tasks such as named entity recognition, co-reference resolution, ...

As well as traditional natural language processing, statistical language processing will be also considered. Some of the most commonly used methods are:

- **TF-IDF:** mathematically, TF-IDF is expressed as the product of the term frequency (TF, it represents the importance of a term in a specific document and IDF, it represents the importance of a term relative to the entire corpus). Toolkits such as NLTK provide lists of stop words that can be used to filter out terms such as “the”, “a”, “and”, etc. The TF-IDF means that the order of terms in both the document and the query itself does not matter.

- **Bigram Analysis:** using NLTK to compute bigrams and collocations for a sentence, i.e., if someone were to tell you that a few of the most common term in a document are “open”, “source” and “government”, could you necessarily say that the text is probably about “open source”, “open government”, both or neither? Bigram offers the option of find out how often a pair of words occurs in a sentence.

- **Vector Space Model** is a large multidimensional space that contains one vector for each document, and the distance between any vectors indicates similarity of the corresponding documents. One of the most powerful things about vector space models is that they can also represent a query as a vector and find the most relevant documents for the query by finding the document vectors with the shortest distance to the query vector.

- **Cosine similarity** can be used to cluster documents. For example, suppose *document 1* contained the terms (A,B,C) and had the corresponding vector of TF-IDF weights (0.1, 0.15, 0.12), while *document2* contained the terms (C,D,E) with the corresponding vector of TF-IDF weights (0.05, 0.1, 0.09). The derived vector for *document1* would be (0.1, 0.15, 0.12, 0, 0) and the derived vector for *document2* would be (0, 0, 0.05, 0.1, 0.09). Each of these vectors could be passed into the NLTK´s cosine distance function, which yields the cosine similarity. Although we use TF-IDF calculations, the exact scoring function could be any useful metric.

After this initial definition of terms we will begin to discuss concrete implementations of algorithms that might prove useful for SoMeDi analyses.

4.2. Text processing

Before the in-depth analysis of the text by the Language Processing Software, the text gathered by the crawler has to be treated:

- If the language data stored is already available as text, it is most likely to be stored in files. Apache parser libraries like **POI**[1] and **PDFBox**[2] extract text from the most common formats (Microsoft documents and PDF, respectively). **Apache Tika**[3] is a toolkit that uses such libraries to extract text and other types of metadata from most types of documents. Tika unifies these parsers under a single interface to allow easily parsing over a thousand different file types. It is useful for search engine indexing, content analysis, translation and identifying the document language if needed. A great way of organizing and accessing text data is by using a search engine [13][14].

- If the language data is on the web, most likely it is part of some other larger website. **BoilerPipe**[4] provides algorithms to detect and remove the surplus “clutter” around the main textual content of a web page, like navigational menu or ads. We can use the results of complete solutions for web scraping such as **Scrapy**[5] or similar.

- If the language data is stored in images, you need to use OCR to extract the text. **Tesseract**[6] can read a wide variety of image formats and convert them to text in over 60 languages (including English, Italian and Spanish). It can be tuned to get commercial grade quality

- Finally, if the language data is stored as audio, **CMU Sphinx**[7] converts speech to text. Depending on how diverse the speakers are and what they say, the results may be quite usable. It is important to highlight that the support that CMU provides via email, forums and IRC is actually better than the support most commercial vendors offer. In SoMeDi we can target content in social media that has elements of speech and hence could benefit from speech recognition.

Independently of the type of language data, the software used will depend on what are the needs of the user: text classification, topic modeling, key word extraction, entity linking, summary, parsing or co-reference resolution. We will analyse these in the following sections.

4.3. Text Classification

Methods that fall under text classification can be quite versatile and powerful. They can be used for guessing genre, estimating overall attitude or sentiment or even predict characteristic of the text author, like their age or gender. But these methods also rely on two assumptions: a) classes are known in advance and b) training data available for each of these classes. The quality of a text classification approach always depends on the number of examples you are training on and how distinct are the classes. Some of the toolkits which offer text classification as one of their features are:

[1] Apache POI – the Java API for Microsoft Documents: http://poi.apache.org/
[2] Apache PDFBox – a Java PDF Library: https://pdfbox.apache.org/
[3] Apache Tika – a content analysis toolkit:http://tika.apache.org/
[4]BoilePipe: http://tika.apache.org/
[5] Scrapy: https://scrapy.org/
[6]Tesseract-ocr: https://code.google.com/p/tesseract-ocr/
[7]CMU Sphinx- Speech Recognition Toolkit: http://cmusphinx.sourceforge.net/

4.3.1. Natural Language Toolkit (NLTK) 8

The NLTK is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for the Python programming language. NLTK includes graphical demonstrations and sample data.
It is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, artificial intelligence, information retrieval and machine learning. NLTK has been used successfully as a platform for prototyping and building research systems.
NLTK comes with a large collection of corpora, extensive documentation, and hundreds of exercises, making NLTK unique in providing comprehensive framework to develop a computational understanding of languages. NLTK´s code base of 100,000 lines of Python code includes support for corpus access, tokenizing, stemming, tagging, chunking, parsing, clustering, classification, language modelling, semantic interpretation, unification and much else besides.

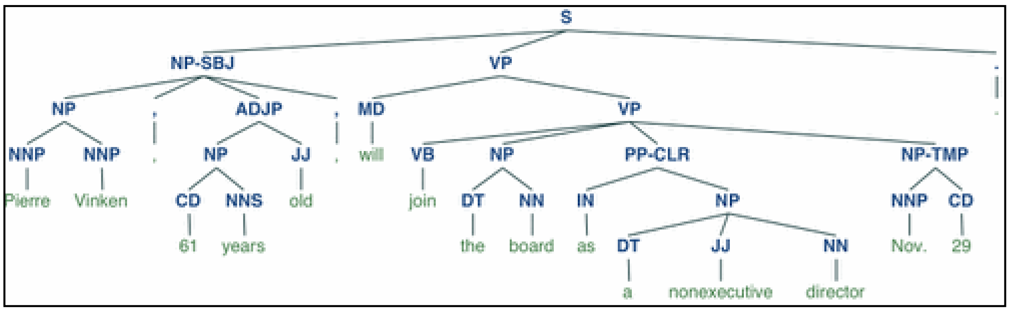


FIGURE 5. PARSE TREE IN NLTK

Although NTLK includes a Spanish corpus, CESS Treebanks (it is derived from Freeling project), in Spanish and Catalan, these resources do not allow to develop a tagging process of new documents. Some reasons are:

- CESS corpus is designed to show results regarding the queries that we make inside them, but do not allow implementing some learning process for a Spanish tagger.
- It hasn't developed any method to train a Spanish tagger yet.
- There isn't any tagged corpus of reference in Spanish, for free use and which belongs to the libraries of NLTK.
- There aren't any common criterions about the tagging process in Spanish.

For Romanian the situation is even less encouraging, with limited availability of corpora or adaptations. However, NTLK it is a powerful tool in Statistical Language Processing where the language isn't so important (i.e. a term frequency).

4.3.2. LingPipe[9]

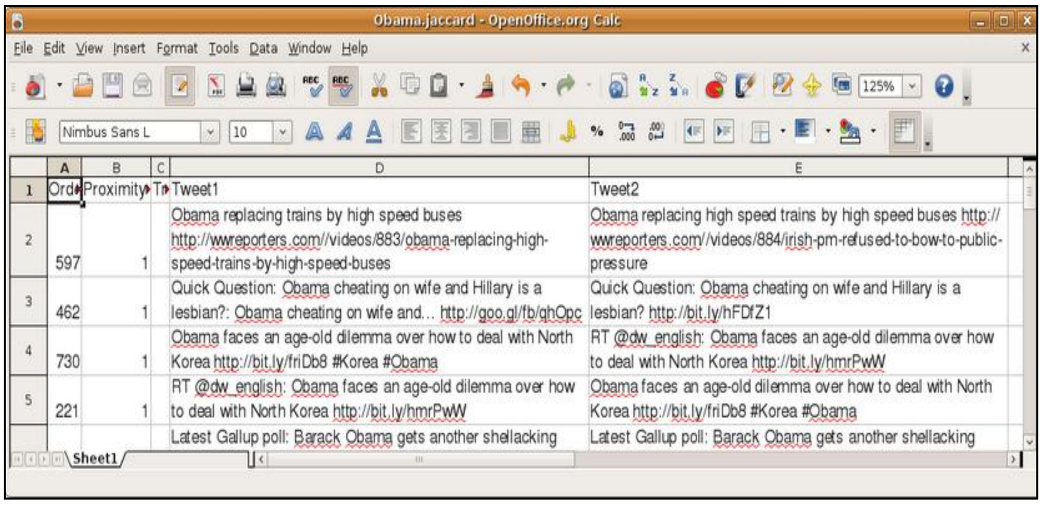
LingPipe is tool kit for processing text using computational linguistics. LingPipe is used to do tasks like:

- Find the names of people, organizations or locations in news. Also, the "standard" model supplies GSP (GeoSpatial Political Entity). This can be changed if the named entity detector is retrained.
- Automatically classify Twitter search results into categories
- Suggest correct spelling of queries

[8] http://cesaraguilar.weebly.com/uploads/2/7/7/5/2775690/met_class12.pdf
[9] http://alias-i.com/lingpipe/

LingPipe´s architecture is designed to be efficient, scalable, reusable and robust. Highlights include:

- Java API with source code and unit tests
- Multi-lingual, multi-domain, multi-genre models
- Training with new data for new tasks
- N-best output with statistical confidence estimates
- Online training (learn-a-little, tag-a-little)
- Thread-safe models and decoders for concurrent-read exclusive-write (CREW) synchronization
- Character encoding-sensitive I/O


Obama.jaccard - OpenOffice.org Calc

	A	B	C	D	E
1	Ord	Proximity	Tr	Tweet1	Tweet2
2	597	1		Obama replacing trains by high speed buses http://wwreporters.com//videos/883/obama-replacing-high-speed-trains-by-high-speed-buses	Obama replacing high speed trains by high speed buses http://wwreporters.com//videos/884/irish-pm-refused-to-bow-to-public-pressure
3	462	1		Quick Question: Obama cheating on wife and Hillary is a lesbian?: Obama cheating on wife and... http://goo.gl/fb/qhOpc	Quick Question: Obama cheating on wife and Hillary is a lesbian? http://bit.ly/hFDfZ1
4	730	1		Obama faces an age-old dilemma over how to deal with North Korea http://bit.ly/friDb8 #Korea #Obama	RT @dw_english: Obama faces an age-old dilemma over how to deal with North Korea http://bit.ly/hmrPwW
5	221	1		RT @dw_english: Obama faces an age-old dilemma over how to deal with North Korea http://bit.ly/hmrPwW	Obama faces an age-old dilemma over how to deal with North Korea http://bit.ly/friDb8 #Korea #Obama
				Latest Gallup poll: Barack Obama gets another shellacking	Latest Gallup poll: Barack Obama gets another shellacking

FIGURE 6. PROCESSING TWEETS WITH LINGPIPE

As it was mentioned above, named entity recognition finds mentions of things in text. The interface in LingPipe provides character offset representations as chunkings.

Name entity recognizers in LingPipe are trained from a corpus of data. Although they are only providing English data, there is training data available (usually for research purposes only) in a number of languages, including Spanish. Many of these training sets may be purchased for commercial applications. LingPipe also supports Unicode and can process documents in any language with Unicode support.

4.3.3. WEKA[10]

Weka is a library containing a large collection of machine learning algorithms, implemented in Java. It supports several standard data mining task. Some of the most important are listed below:

- **Classification:** given a labelled set of observations, learn to predict labels for new observations
- **Regression:** numeric value instead of value
- **Attribute selection:** find attributes of observations that are important for prediction

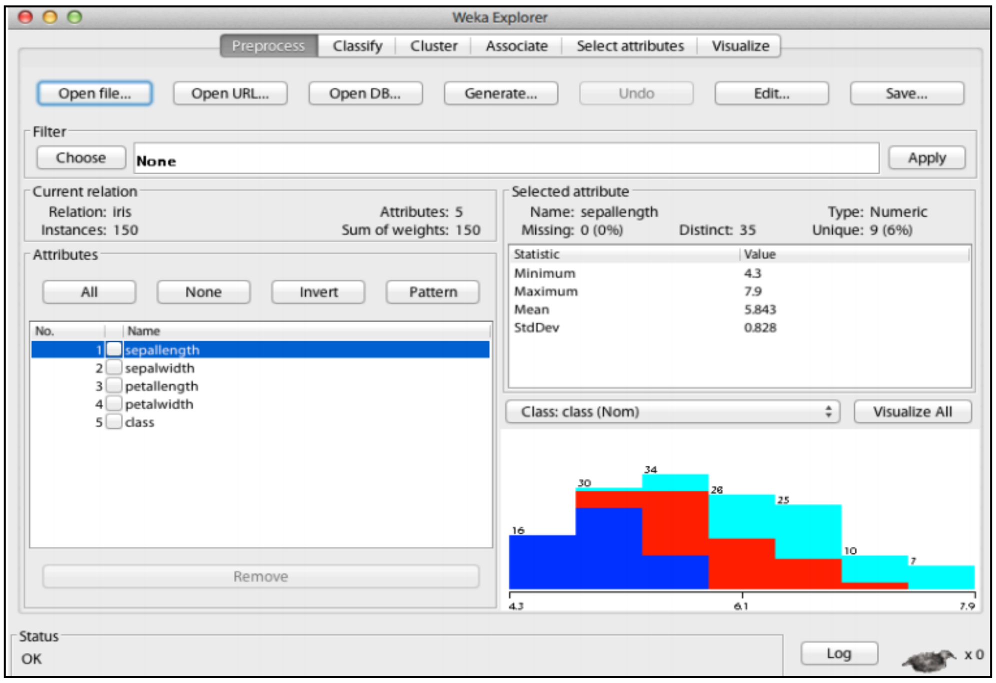
[10] http://www.cs.waikato.ac.nz/~eibe/WEKA_Ecosystem.pdf

- **Clustering:** no labels, just identify groups of similar observations (clusters)
- There is also some support for association rule mining and (conditional) density estimation

All of Weka's techniques are predicted on the assumption that the data is available as a single flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric or nominal attributes, but some other attribute types are also supported). Weka provides access to SQL databases using Java Database Connectivity and can process the result returned by a database query. It is not capable of multi-relational data mining, but there is separate software for converting a collection of linked database tables into a single table that is suitable for processing using Weka. Another important area that is currently not covered by the algorithms included in the Weka distribution is sequence modeling.

Weka's main user interface is the Explorer, but essentially the functionality can be accessed through the component-based Knowledge Flow interface and from the command line. There is also the Experimenter, which allows the systematic comparison of the predictive performance of Weka's machine learning algorithms on a collection of datasets.

- **Explorer:** still the most popular interface for batch data processing; tab-based interface to algorithms. It features several panels providing access to the main components of the workbench:
  - The *Preprocess* panel has facilities for importing data from a database, a CSV file, etc. and for preprocessing this data using a so-called *filtering* algorithm. These filters can be used to transform the data and make it possible to delete instances and attributes according to specific criteria.
  - The *Classify* panel enables the user to apply classification and regression algorithms (indiscriminately called *classifiers* in Weka) to the resulting dataset, to estimate the accuracy of the resulting predictive model, and to visualize erroneous predictions, ROC curves, etc. or the model itself (if the model is amenable to visualization like, e.g., a decision tree).
  - The *Associative* provides access to association rule learners that attempt to identify all important interrelationships between attributes in the data.
  - The *Cluster* panel gives access to the clustering techniques in Weka, e.g., the simple k-means algorithms. There is also an implementation of the expectation maximization algorithm for learning a mixture of normal distributions.
  - The *Select attributes* panel provides algorithms for identifying the most predictive attributes in a dataset
  - The *Visualize* panel shows a scatter plot matrix, where individual scatter plots can be selected and enlarged, and analysed further using various selection operators.



Filter: None

Current relation
Relation: iris　Attributes: 5
Instances: 150　Sum of weights: 150

Selected attribute
Name: sepallength　Type: Numeric
Missing: 0 (0%)　Distinct: 35　Unique: 9 (6%)

Statistic	Value
Minimum	4.3
Maximum	7.9
Mean	5.843
StdDev	0.828

Attributes

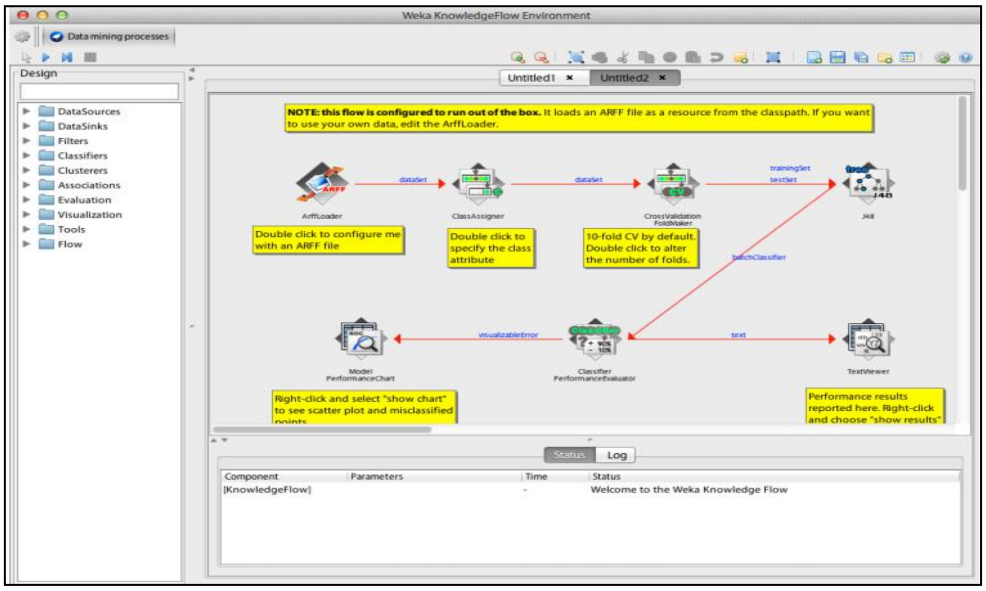
No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class

Class: class (Nom)

Status
OK

FIGURE 7. WEKA EXPLORER

- **Knowledge Flow:** users lay out and connect widgets representing WEKA components



NOTE: this flow is configured to run out of the box. It loads an ARFF file as a resource from the classpath. If you want to use your own data, edit the ArffLoader.

Double click to configure me with an ARFF file

Double click to specify the class attribute

10-fold CV by default. Double click to alter the number of folds.

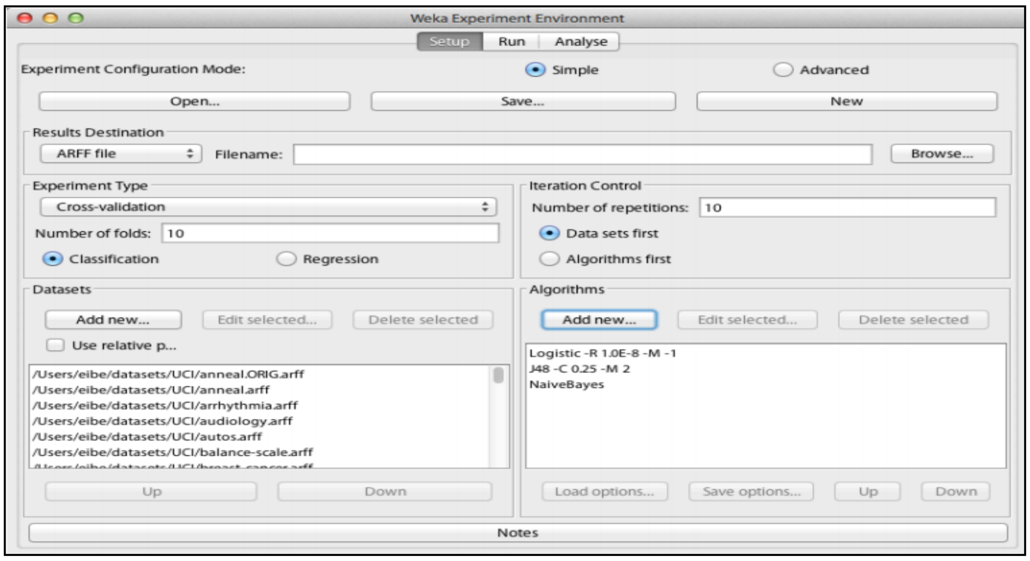
Right-click and select "show chart" to see scatter plot and misclassified points

Performance results reported here. Right-click and choose "show results"

Component	Parameters	Time	Status
[KnowledgeFlow]		-	Welcome to the Weka Knowledge Flow

FIGURE 8. WEKA KNOWLEDGE FLOW

- **Experimenter:** enables large scale comparison of predictive performance of learning algorithms.



Experiment Configuration Mode: [✓ Simple] Advanced

Results Destination: ARFF file — Filename:

Experiment Type: Cross-validation
Number of folds: 10
[✓ Classification] Regression

Iteration Control
Number of repetitions: 10
[✓ Data sets first] Algorithms first

Datasets
Use relative p...

```
/Users/eibe/datasets/UCI/anneal.ORIG.arff
/Users/eibe/datasets/UCI/anneal.arff
/Users/eibe/datasets/UCI/arrhythmia.arff
/Users/eibe/datasets/UCI/audiology.arff
/Users/eibe/datasets/UCI/autos.arff
/Users/eibe/datasets/UCI/balance-scale.arff
```

Algorithms

```
Logistic -R 1.0E-8 -M -1
J48 -C 0.25 -M 2
NaiveBayes
```

FIGURE 9. WEKA EXPERIMENTER

Weka also has a command-line interface and its functionality can be accessed through the OS shell.

Only Knowledge Flow and command-line interface enable incremental processing of data.

Related to the use of external interfaces, Weka provides a unified interface to a large collection of learning algorithms and is implemented in Java.

There is a variety of software through which one can make use of this interface:

- Octave/ Matlab
- R statistical computing environment: RWeka
- Python: python-weka-wrapper

Other software through which one can access WEKA are Mathematica, SAS, KNIME and RapidMiner.

4.3.4. LIBSHORTTEXT[11]

LibShort Text is an open source library for short-text classification and analysis. Properties of short text are considered in its design and implementation. The package supports effective text pre-processing and fast training/ prediction procedures. Because of the short length, details of each short text can be investigated easily. It has the following features:

1. For large-scale short-text classification, it is more efficient than general text-mining tools.
2. Users not try many options to get the best performance for their applications
3. An interactive tool to perform error analysis. In particular, because of short lengths, details of each text can be investigated.
4. The package is mainly written in Python for simplicity and extensibility, while time-consuming operations are implemented in C/C++ for efficiency.
5. Full documentation including class references and examples

[11] http://www.csie.ntu.edu.tw/~cjlin/libshorttext/

The workflow of LibShort Text is shown in the following figure. Each step corresponds to a library in the core Python module libshorttext.

Workflow	Libraries	Command-line tools
Short texts ↓		
Pre-processing feature generation ↓	libshorttext.converter	text-train.py text2svm.py
Training Testing ↓	libshorttext.classifier	text-train.py text-predict.py
Analysis	libshorttext.analyzer	

TABLE 3. LIBSHORT TEXT WORKFLOW

The ease of use, efficiency and extensibility of LibShort Text make it very useful for practitioners working on short-text classification and analysis, like Facebook comments and tweets.

4.4. Topic Modelling

In situations in which there is an abundance of data and no idea about the discussed concepts or topics, topic modeling allows to cluster documents while providing a series of possible labels (topics) for each cluster. Some of the toolkits which offer topic modeling are

4.4.1. Mallet

Mallet[12] is an integrated collection of Java code useful for statistical natural language processing, document classification, cluster analysis, information extraction, topic modeling and other machine learning applications to text.

Mallet includes sophisticated tools for **document classification:** efficient routines for converting text to “features”, a wide variety of algorithms (including Naïve Bayes, Maximum Entropy and Decision Trees) and code for evaluating classifier performance using several commonly used metrics.

In addition to classification, Mallet includes tools for sequence tagging for applications such as named-entity extraction from text. Algorithms include Hidden Markov Models, Maximum Entropy Markov Models and Conditional Random Fields. These methods are implemented in an extensible system for finite state transducers.

Topic models are useful for analyzing large collections of unlabeled text. The Mallet **topic modeling** toolkit contains efficient, sampling-based implementations of Latent Dirichlet Allocation, Pachinko Allocation and Hierarchical LDA.

In addition to sophisticated Machine Learning applications, Mallet includes routines for transforming text documents into numerical representations that can then be processed efficiently. This process is implemented through a flexible system of “pipes”, which handle distinct tasks such as tokenizing strings, removing stop words and converting sequences into count vectors.

An add-on package to Mallet, called GRMM, contains support for interference in general graphical models and training of CRFs with arbitrary graphical structure.

4.4.2. Gensim

[12]http://mallet.cs.umass.edu/

Gensim[13] is a free Python library designed to automatically extract semantic topics from documents, as efficiently (computer-wise) and painlessly (human-wise) as possible.

The algorithms in Gensim, such as Latent Semantic Analysis, Latent Dirichlet Allocation or Random Projections, discover semantic structure documents, by examining word statistical co-occurrence patterns within a corpus of training documents. These algorithms are unsupervised, which means no human input is necessary – only a corpus of plain text documents is needed.

Once these statistical patterns are found, any plain text documents can be succinctly expressed in the new, semantic representation, a queried for topic similarity against other documents.

The main features of Gensim are:

- Memory independence – there is no need for the whole training corpus to reside fully in RAM at any one time (can process large, web-scale corpora)
- Efficient implementation for several popular vector space algorithms, including TF-IDF, distributed incremental Latent Semantic Analysis, distributed incremental Latent Dirichlet Allocation (LDA) or Random Projection; adding new ones is easy
- I/O wrappers and converters around several popular data formats
- Similarity queries for documents in their semantic representation

4.5. Key Word Extraction

Key Word Extraction consists of extracting relevant terms from a given corpus automatically. Compared to text classification and topic modeling, key word extraction assumes that the number of possible topics (or categories, or tags) can be large (thousands or more), that each document may have different number of matching topics and that these topics are well-formed phrases. Key words can be chosen from document text or from a predefined vocabulary, which ensures their consistency.

4.5.1. KEA

KEA[14] is an algorithm for extracting key phrases from text documents. It can be either used for free indexing or for indexing with a controlled vocabulary.

The functioning of Kea is the following:

1. **Documents.** Kea gets a directory name and processes all documents in this directory that have the extension ".txt". The default language and the encoding is set to English, but this can be changed as long as a corresponding stop word file and a stemmer is provided.
2. **Thesaurus.** If a vocabulary is provided, Kea matches the documents' phrases against this file. For processing SKOS[15] files stored as RDF files, Kea uses the Jena API.
3. **Extracting candidates.** Kea extracts n-grams for a predefined length that do not start or end with a stop word. In controlled indexing, it only collects those n-grams that match thesaurus terms. If the thesaurus defines relations between non-allowed terms (non-descriptors) and allowed terms (descriptors), it replaces each descriptor by an

[13]http://radimrehurek.com/gensim/intro.html
[14]http://www.nzdl.org/Kea/index.html
[15] SKOS is a W3C recommendation for representation of thesauri, classification schemes, taxonomies, subject-heading systems or any other type of structured controlled vocabulary. SKOS is part of the Semantic Web family of standards built upon RDF and RDFS.

equivalent non-descriptor. In the bellow diagram, **pseudo-phrase matching** means removing stop words from the phrase, and then stemming and ordering the remaining words.

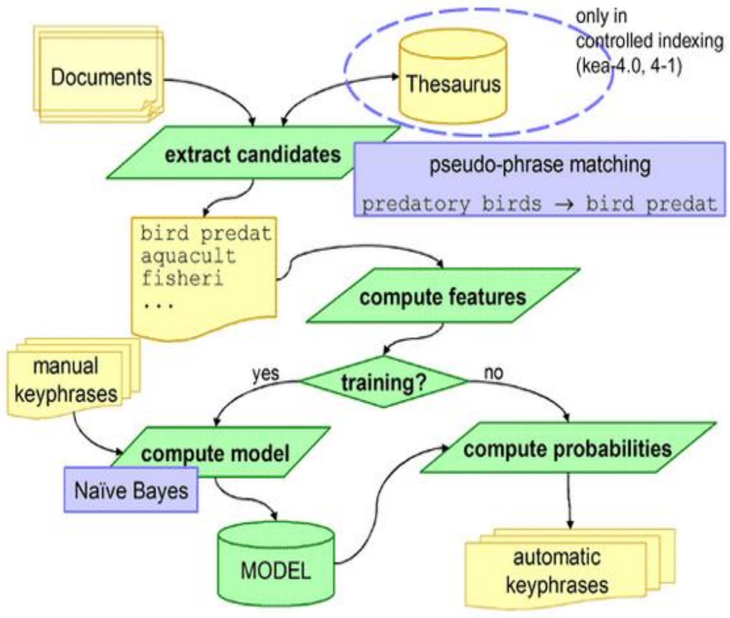


FIGURE 10. KEA FUNCTIONING

4. **Features.** For each candidate phrase Kea compute 4 feature values:
   - **TF/IDF**
   - **First occurrence.** It is the percentage of the document preceding the first occurrence of the term in the document. Terms that tend to appear at the start or at the end of a document are more likely to be key phrases
   - **Length** of a phrase is the number of its component words. Two-word phrases are usually preferred by human indexers
   - **Node degree** of a candidate phrase is the number of phrases in the candidate set that is semantically related to this phrase. This is computed with the help of thesaurus. Phrases with high degree are more likely to be key phrases.

5. **Building the model.** Before being able to extract key phrases from new documents, Kea first needs to create a model that learns the extraction strategy from manually indexed documents. This means, for each document in the input directory they must be a file with the extension ".key" and the same name as the corresponding document. This file should contain manually assigned key phrases, one per line.
   Given the list of the candidate phrases (.3), Kea marks those that were manually assigned as positive example and all the rest as negative examples. By analysing the feature values (.4) for positive and negative candidate phrases, a model is computed, which reflects the distribution of feature values for each phrase.

6. **Extracting key phrases.** When extracting key phrases from new documents, Kea takes the model (.5) and feature values for each candidate phrase and computes its probability of being a key phrase. Phrases with the highest probabilities are selected

into the final set of key phrases. The user can specify the number of key phrases that need to be selected.

4.5.2. Maui

Maui[16],[17] automatically identifies main topic in text documents. Depending on the task, topics are tags, keywords, keyphrases, vocabulary terms, descriptors, index terms or titles of Wikipedia articles.

Maui performs the following tasks:

- Term assignment with a controlled vocabulary (or thesaurus)
- Subject indexing
- Topic indexing with terms from Wikipedia
- Key phrase extraction
- Terminology extraction
- Automatic tagging
- Semi-automatic topic indexing

Maui builds on the key phrase extraction algorithm Kea by adopting its two-stage indexing process and inheriting some of its components. The Maui algorithm consists in four main steps:

1. Generating candidate topics (*training and extraction*)
2. Computing features for the candidates (*training and extraction*)
3. Building the topic indexing model (*training only*)
4. Applying the topic indexing model (*extraction only*)

After the candidate generation Maui computes features for each of candidates. The following features are computed during the candidate generation step:

- *Term frequency*
- *First occurrence* – the position of the first occurrence for each candidate relative to the number of words in the document
- *Last occurrence* – the position of the last occurrence for each candidate relative to the number of words in the document

In the training stage Maui also creates tables with the following values:

- n_t is how many documents contain each candidate topic
- N is the total number of documents
- m_t is the frequency of a topic that appears in the manually assigned topic sets

These features are then used in the computation of the following features:

- *Inverse document frequency* (IDF)
- TF-IDF
- *Spread = last occurrence – first occurrence*
- *Domain keyphraseness* is 0 if the candidate topic never appears in a manually assigned topic set and m_t otherwise

[16] https://code.google.com/p/maui-indexer/
[17] http://www.ke.tu-darmstadt.de/lehre/arbeiten/bachelor/2011/Seifert_Viktor.pdf

- *Wikipedia keyphraseness* involves matching candidate title against anchors appearing in the Wikipedia corpus. Values are pre-computed during candidate generation
- *Inverse Wikipedia frequency* is computed by retrieving the most likely Wikipedia article for the current candidate (unless the candidate is a Wikipedia article itself) and counting the number of its incoming links
- *Total Wikipedia keyphraseness* is the sum of *Wikipedia keyphraseness* values over all *n-grams* that were mapped to the Wikipedia article corresponding to the given candidate
- *Semantic relatedness* is the total relatedness of the Wikipedia article representing the candidate to Wikipedia articles identified for all other candidates computed using Wikipedia Miner
- *Term length* is the number of words in the candidate topic's name
- *Generality* is composed for candidates that were mapped to Wikipedia articles, and is the distance between the category corresponding to the article and the root of the category tree, normalized by the tree depth
- *Class value,* which is only known for training documents, is 1 if the candidate has been assigned manually, and 0 otherwise

In the third step *Building the topic indexing model*, Maui uses Machine Learning to build an indexing model.

In the fourth step Maui determines for each candidate a probability of it being a candidate according to the previously built indexing model. The *k* candidates with highest probabilities are chosen as topics.

Maui supports stemmers and stop words for English, French and Spanish, but can be extended to work in many other languages, including languages that require special encoding.

After Maui is installed, there are two ways of using it: from the command line and from the Java code. Either way, the input data is required first.

4.6. Text summarization

Whereas text classification, topic modeling and key word extraction all summarize the content of the document categorically, summarizing the very text by extracting the most relevant sentences is also possible.

4.6.1. MEAD

MEAD[18] is a publicly available toolkit for multi-lingual summarization and evaluation using standard metrics. The toolkit implements multiple summarization algorithms such as position-based, Centroid, TF*IDF and query-based methods. Methods for evaluating the quality of the summaries include co-selection and content-based measures.

MEAD can perform many different summarization tasks. It can summarize individual documents or clusters of related documents (multi-document summarization).

MEAD includes two baseline summarizers: lead-base and random. Lead-based summaries are produced by selecting the first sentence of each document, then the second sentence of each, etc.

[18] http://www.summarization.com/mead/documentation/meaddoc.pdf

until the desired summary size is met. A random summary consists of enough randomly selected sentences (from the cluster) to produce a summary of the desired size.

MEAD has been primarily used for summarizing documents in English, but recently, Chinese capabilities have been added to the publicly available version of MEAD. It is relatively easy to coerce MEAD to produce summaries of other languages.

Query-based summarization is often used in natural language circles, and is (not coincidentally) included in MEAD as well.

The MEAD evaluation toolkit (MEAD Eval), allows evaluation of human-human, human-computer and computer-computer agreement. It currently supports two general classes of evaluation metrics: co-selection and content-based metrics. Co-selection metrics include precision, recall, Kappa and Relative Utility, a more flexible cousin of Kappa. MEAD's content-based metrics are cosine (which uses TF*IDF), simple cosine (which doesn't) and unigram-and bigram-overlap. Relevance correlation has previously been used in conjunction with MEAD but is not included with the current distribution.

4.7. Entity Linking

Going beyond key categories and phrases, concepts and entities mentioned in text could be extracted and identified. If the focus is in names of people, organizations and locations, Name Entity Recognition (NER) is the technique to use.

Entity linking performs this by disambiguating entities to their unique ids in a knowledge base.

4.7.1. Wikipedia Miner

Wikipedia Miner[19],[20] is a toolkit for tapping the rich semantics encoded within Wikipedia. It makes easy to integrate Wikipedia's knowledge into the applications, by:

- Providing simplified, object-oriented access to Wikipedia's structure and content
- Measuring how terms and concepts in Wikipedia are connected to each other
- Detecting and disambiguating Wikipedia topics when they are mentioned in documents

This toolkit uses Wikipedia to train a program to automatically recognize topics when they are mentioned. This learns from the categories and internal links that pepper Wikipedia's articles.

The same program can be run on any webpage or document, yielding a number of links to Wikipedia topics. A search engine can use this to see people, places, events and ideas rather than just letters.

4.7.2. Illinois Wikifier

Illinois Wikifier[21],[22] identifies important entities and concepts in text, disambiguates them and links them to Wikipedia. Wikification is an important step in helping to facilitate Information Access, in knowledge acquisition from text and in helping to inject background knowledge into NLP applications. The main decisions the Wikifier must take are:

- What expression to link to Wikipedia
- Disambiguating the ambiguous expressions and entities. For example, four possible types of features could be:

[19] http://wikipedia-miner.cms.waikato.ac.nz/
[20] http://www.nzdl.org/wikification/about.html
[21] http://cogcomp.cs.illinois.edu/page/demo_view/Wikifier
[22] https://sites.google.com/site/vaprdemos/projects/wikifier

- String matching and prevalence of entities in Wikipedia
- Lexical similarity between the input document and the Wikipedia pages
- "Semantic similarity" between the ESA summary of the input document and the Wikipedia pages
- How likely is a set of Wikipedia pages to be linked from a single document (this is gotten by looking at the linkage patterns in Wikipedia)

The Wikifier provides a quick way to learn the meaning of important concepts; it links mentions of people, organizations and other entities across documents and is an easy way to learn more about key concepts and entities that appear in free form text. For example, if a mention of "Armstrong" in text is seen, knowing who is mentioned could be needed – is it Lance Armstrong, Louis Armstrong, Neil Armstrong or another prominent Armstrong? Also, knowing something about them beyond what is mentioned in the text, which was probably written under the assumption that the reader knows what is meant, could be interesting. The Wikifier will disambiguate the correct Armstrong based on the context, will provide some information about the entity (e.g., he is an astronaut) and a link to Wikipedia about the entity.

4.7.3. Illinois Named Entity Tagger

Illinois Named Entity Tagger[23] is a state of the art Name Entity Recognition tagger that tags plain text with named entities. The newest version tags entities with either the "classic" 4-label type set (people/ organizations/ locations/ miscellaneous), while the most recent can also tag entities with a larger 18-label type set (based on the OntoNotes corpus[24]). It uses gazetteers extracted from Wikipedia, word class models derived from unlabeled text and expressive non-local features. The tagger is robust and has been evaluated on a variety of datasets.

4.7.4. OpenNLP[25],[26]

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text. It supports the most common NLP tasks via components such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing and co-reference resolution. Components contain parts which enable one to execute the respective natural language processing task, to train a model and often also to evaluate a model. Each of these facilities is accessible via its application program interface (API). In addition, a command line interface (CLI) is provided for convenience of experiments and training. These tasks are usually required to build more advanced text processing services.

There is independent named entity recognition for Spanish and Dutch and currently there are data files available for Spanish.

4.7.5. Stanford CoreNLP

Stanford CoreNLP[27,28] provides a set of natural language analysis tools which can take raw text input and give the base forms of words, their parts of speech, whether they are name of companies, people, etc. (Named Entity Recognition), normalized dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases and work dependencies, and indicate which noun phrases refer to the same entities (co-reference resolution). Stanford CoreNLP is an integrated framework, which makes very easy to apply a bunch of language analysis tools to a piece of text.

[23] http://cogcomp.cs.illinois.edu/page/software_view/NETagger
[24] OntoNotesRelease 5.0: https://catalog.ldc.upenn.edu/LDC2013T19
[25] http://en.wikipedia.org/wiki/OpenNLP
[26] http://opennlp.apache.org/documentation/1.5.3/manual/opennlp.html#tools.corpora.conll
[27]https://www.npmjs.org/package/stanford-simple-nlp
[28]http://www-nlp.stanford.edu/

Starting from plain text, running all the tools on it with just two lines is possible. Its analysis provides the foundational building blocks for higher-level and domain-specific text understanding applications.

Stanford CoreNLP integrates all NLP tools, including the part-of-speech (POS) tagger, the named entity recognizer (NER), the parser and the co-reference resolution system and provides model files for analysis of English.

Another distinguishing feature of Stanford NLP is the combination of deep linguistic modeling and data analysis with probabilistic and machine learning approaches to NLP.

4.8. Co-reference Resolution

When people speak or write, they use pronouns and other ways to refer to the same entities in order to avoid repetition and sound nice. The computer first needs to understand which pronoun and expression refers to which entity. The technique to do this is called co-reference resolution. The previous libraries mentioned in the above section also offer these capabilities (e.g., Stanford NLP, OpenNLP, IllinoisNLP). Now we will detail one of the dedicated tools for resoluting coreferences.

4.8.1. MaltParser

MaltParser[29,30] is a system for data-driven dependency parsing, which can be used to induce a parsing model from treebank data and to parse new data using an induced model. Parsers developed using MaltParser have achieved state-of-the-art accuracy for a number of languages. However, MaltParser is a fairly complex system with many parameters that need to be optimized. Simply using the system "out of the box" with default settings is therefore likely to result in suboptimal performance.

MaltParser can be characterized as a data-driven parser-generator which constructs a parser given a treebank. MaltParser is an implementation of inductive dependency parsing, where the syntactic analysis of a sentence amounts to the derivation of a dependency structure, and where inductive machine learning is used to guide the parser at nondeterministic choice points. The parsing methodology is based on three essential components:

1. Deterministic parsing algorithms for building labeled dependency graphs
2. History-based models for predicting the next parser action at nondeterministic choice points
3. Discriminative learning to map histories to parser actions

In addition, is important to mention that the training data for MaltParser consists of sentences annotated with dependency trees. If the source treebank does not contain dependency trees, there are usually standard procedures for conversion that can be used (with varying degrees of precision depending on the nature of the original annotation). MaltParser supports several input formats, but by default it is assumed that the data is in the CoNLL representation format (Buchholz and Marsi, 2006) for data-driven dependency parsing.

MaltParser has a module for Spanish, espmalt-1.0. mco and is also able to analyze an Italian treebank.

[29] http://www.maltparser.org/optiondesc.html
[30] http://www.maltparser.org/guides/opt/quick-opt.pdf

4.9. Opinion Mining

Opinion mining or sentiment analysis refers to the use of automated language analysis tools to evaluate personal attitudes and subjective information about digital or physical assets. This is used frequently in fields such as social media analytics, dialogue management and e-commerce monitoring.

Opinion mining is based upon psychological theories on emotions and their representation: models to categorize emotional feedback from a person so it can be stored, processed, compared or shared with others. Among these models we can cite the fundamental emotional work of Paul Ekman[31] that informs many of these models. The main models used in opinion mining are dimensional analysis frameworks that attempt to systematize the emotional representation around two or three dimensions. Most dimensional models incorporate valence/polarity (quality factor between *positive* and *negative*) and arousal (quantity or intensity factor such as *relaxed* or *excited*) dimensions. Many times in the sentiment analytics this is simplified to the valence/polarity axis to simplify the analytics process. Common models to represent emotion in the opinion mining are the classic PAD model (Pleasure, Arousal and Dominance)[32], the also three-dimensional Lövheim cube of emotion[33] and the more modern and much more featured model proposed by Cowen and Keltner[34] that, based on massive machine learning analytics, identifies 27 distinct vectors of emotion.

Sentiment analysis is often very subjective and based on very deep layers of language understanding. A sentence such as “The movie was surprising with plenty of unsettling plot twists” could be quickly dismissed as pertaining to negative emotions because of used adjectives that are often used to express negative emotions (‘surprising’, ‘unsettling’) yet here are rotundly expressing a positive factor. Hence, sentiment analytics many times hinges on many of the NLP tools that we have mentioned in this chapter.

Many of the tools mentioned in past sections (such as OpenNLP or Standford CoreNLP) can be used to construct sentiment analysis pipelines. However, dedicated tools also exist. We will now examine tools commonly used to do sentiment analysis on texts:

4.9.1. TextBlob

TextBlob[35] is a Python (2 and 3) library for processing textual data. It provides a consistent API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, and more. It effectively wraps NTLK libraries with a high level API to generate some extra results.

For sentiment analysis, a primitive is available for every sample text that is provided. The sentiment property returns a value including a score for ‘polarity’ and other for ‘subjectivity’. The polarity score is a float within the range [-1.0, 1.0]. The subjectivity is a float within the range [0.0, 1.0] where 0.0 is very objective and 1.0 is very subjective.

4.9.2. Pattern

[31] Ekman, Paul (1992). "An Argument for Basic Emotions". Cognition and Emotion. 6 (3/4): 169–200. doi:10.1080/02699939208411068.
[32] Mehrabian, Albert (1980). Basic dimensions for a general psychological theory. pp. 39–53. ISBN 0-89946-004-6.
[33] Lövheim H. A new three-dimensional model for emotions and monoamine neurotransmitters. Med Hypotheses (2011), Epub ahead of print. doi:10.1016/j.mehy.2011.11.016 PMID 22153577
[34] Alan S. Cowen and Dacher Keltner (2017-09-05). "Self-report captures 27 distinct categories of emotion bridged by continuous gradients". Pnas.org.
[35] TextBlob website: https://textblob.readthedocs.io/en/dev/

Pattern[36] is a geneal workbench of NLP tools provided open source by the University of Antwerpen. It is divided into different sub-modules for uses or languages. For example, the pattern.en module contains a fast part-of-speech tagger for English (identifies nouns, adjectives, verbs, etc. in a sentence), sentiment analysis, tools for English verb conjugation and noun singularization & pluralization, and a WordNet interface.

The sentiment analytics are available out of the box for English. The pattern.en module bundles a lexicon of adjectives (e.g., good, bad, amazing, irritating, ...) that occur frequently in product reviews, annotated with scores for sentiment polarity (positive ↔ negative) and subjectivity (objective ↔ subjective).

- The sentiment() function returns a (polarity, subjectivity)-tuple for the given sentence, based on the adjectives it contains, where polarity is a value between -1.0 and +1.0 and subjectivity between 0.0 and 1.0. The sentence can be a string, Text, Sentence, Chunk, Word or a Synset (see below).
- The positive() function returns True if the given sentence's polarity is above the threshold. The threshold can be lowered or raised, but overall +0.1 gives the best results for product reviews. Accuracy is about 75% for movie reviews.

Even with the limitation of English-only out of the box, it will be explored to see whether it can be extended to other languages of interest such as Spanish or Romanian for use in SoMeDi.

4.10. Frameworks

The majority of NLP software is available as a set of Java or Phyton toolkits. The advantage of using toolkits is that they make easy to pass the output from one NLP component to another. However, sometimes combining components from different libraries is needed. The below mentioned tools allows to mitigate this problem by offering frameworks, which can combine components from different authors into another systems.

4.10.1. UIMA[37] [38]

UIMA is software architecture for the development, discovery, composition and deployment of multi-modal analytics for the analysis of unstructured information and its integration with search technologies developed by IBM.

One potential use of UIMA is a logistics analysis software system that could convert unstructured data such as repair logs and service notes into relational tables. These tables can then be used by automated tools to detect maintenance or manufacturing problems. UIMA analyzes large volumes of unstructured information in order to discover knowledge that is relevant to an end-user. An example UIMA application might ingest plain text and identify entities, such as persons, places, organizations; or relations, such as works-for or located-at.

[36] De Smedt, T. & Daelemans, W. (2012). *Pattern for Python*. Journal of Machine Learning Research, 13: 2031–2035.
[37] http://en.wikipedia.org/wiki/UIMA
[38] http://uima.apache.org/

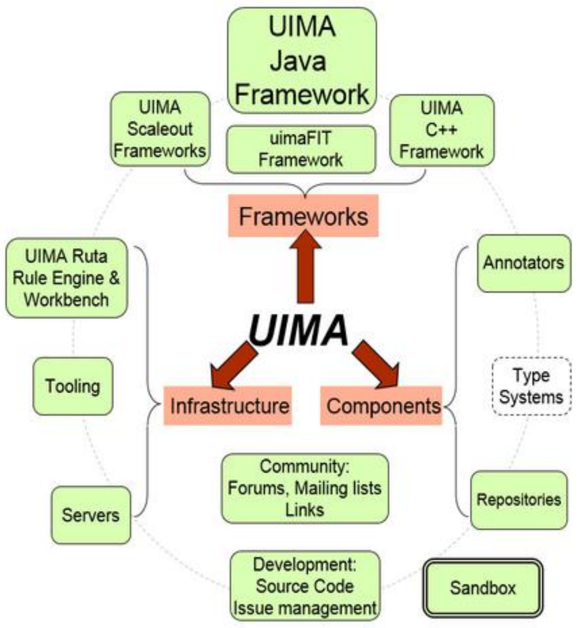


FIGURE 11. UIMA FRAMEWORK

The UIMA architecture can be thought of in four dimensions:

1. It specifies component interfaces in an analytics pipeline
2. It describes a set of Design patterns.
3. It suggests two data representations: an in-memory representation of annotations for high-performance analytics and an XML representation of annotations for integration with remote web services.
4. It suggests development roles allowing tools to be used by users with diverse skills.

UIMA enables applications to be decomposed into components, for example “language identification” => “language specific segmentation” => “sentence boundary detection” => “entity detection (person/place names etc.)”. Each component implements interfaces defined by the framework and provides self-describing metadata via XML descriptor files. The framework manages these components and the data flow between them. Components are written in Java or C++; the data that flows between components is designed for efficient mapping between these languages.

UIMA can scale to very large volumes by replicating processing pipelines over a cluster of networked nodes.

Apache UIMA is an Apache-licensed open source implementation of the UIMA specification.

The frameworks run the components, and are available for both Java and C++. The Java Framework supports running both Java and non-Java components (using C++ framework) The C++ framework, besides supporting annotators written in C/C++, also supports Perl, Python, and TCL annotators.

The frameworks support configuring and running pipelines of Annotator components. These components do the actual work of analyzing the unstructured information. Users can write their own annotators, or configure and use pre-existing annotators. Some annotators are available as part of this project; others are contained in various repositories on the internet.

Additional infrastructure support components include a simple server that can receive REST requests and return annotation results, for use by other web services.

4.10.2. GATE [39],[40]

GATE is a Java suite of tools used worldwide for all sorts of natural language processing task, including information extraction in many languages. GATE includes an information extraction system called ANNIE (A Nearly-New Information Extraction System) which is a set of modules comprising: tokenizer, part-of-speech tagger, sentence splitter, gazetteer, parser and co-reference tagger.

ANNIE can be used as-is to provide basic information functionality, or provide a starting point for more specific task.

Languages currently handled in GATE include Spanish.

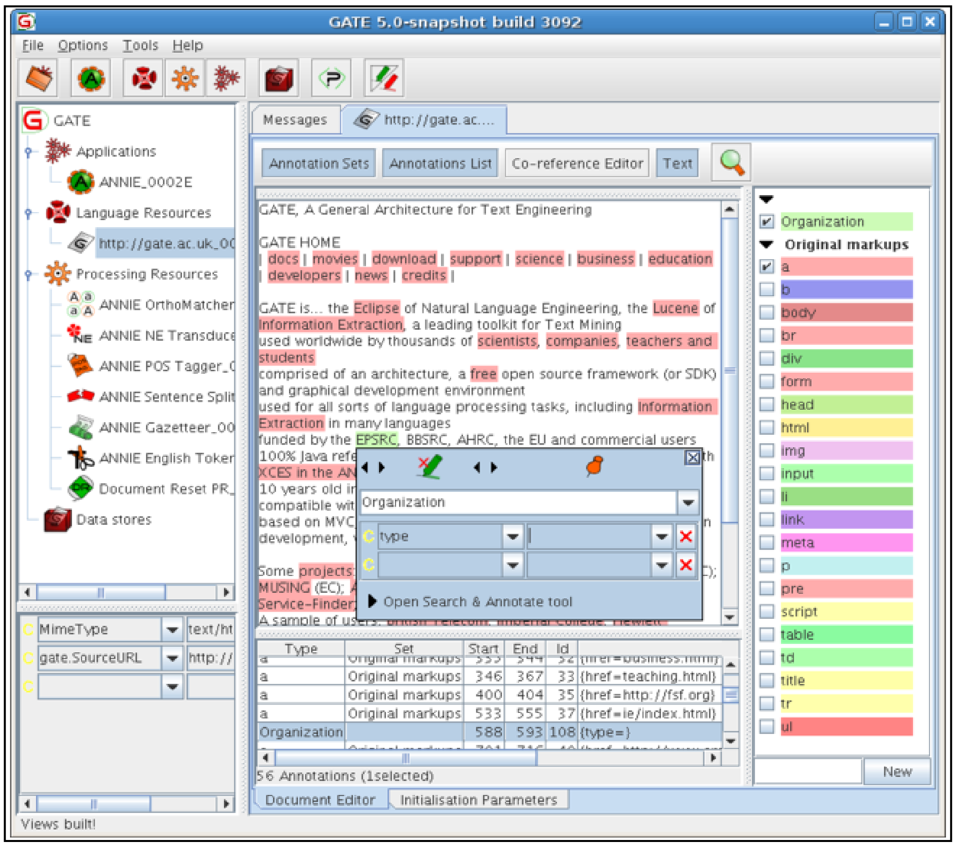
Plug-ins are included for machine learning with Weka, as well as a LIBSVM integration and an in-house perceptron implementation, for managing ontologies like WordNet, for querying search engines like Google or Yahoo.

GATE accepts input in various formats, such as TXT, HTML, Doc, PDF documents, and Java Serial, PostgreSQL, Lucene, Oracle Databases with help of RDBMS storage over JDBC.

JAPE transducers are used within GATE to manipulate annotations on text.

GATE Developer and GATE Embedded are licensed under the GNU Lesser General Public License.

[39] http://en.wikipedia.org/wiki/General_Architecture_for_Text_Engineering
[40] http://gate.ac.uk/gate/doc/index.html



GATE, A General Architecture for Text Engineering

GATE HOME
| docs | movies | download | support | science | business | education
| developers | news | credits |

GATE is... the Eclipse of Natural Language Engineering, the Lucene of
Information Extraction, a leading toolkit for Text Mining
used worldwide by thousands of scientists, companies, teachers and
students
comprised of an architecture, a free open source framework (or SDK)
and graphical development environment
used for all sorts of language processing tasks, including Information
Extraction in many languages
funded by the EPSRC, BBSRC, AHRC, the EU and commercial users
100% Java refe
XCES in the AN
10 years old in
compatible wit
based on MVC
development,

Some projects:
MUSING (EC);
Service-Finder;
A sample of users:

Type	Set	Start	End	Id	
a	Original markups	346	367	33	{href=teaching.html}
a	Original markups	400	404	35	{href=http://fsf.org}
a	Original markups	533	555	37	{href=ie/index.html}
Organization		588	593	108	{type=}

56 Annotations (1selected)

FIGURE 12. GATE DEVELOPER MAIN WINDOW

The license of GATE allows embedding the software as a library in other tools. The only restriction is that distributed binaries that incorporate GATE must also include copies of GATE, and acknowledge the use of their system. Modifications made to GATE itself must be clearly marked and also distributed.

4.11. Text Mining and cluster analysis

4.11.1. CARROT

Carrot2[41] is an Open Source Search Results Clustering Engine. It can automatically organize small collections of documents, e.g. search results, into thematic categories. Carrot2 integrates very well with both Open Source and proprietary search engines. It should successfully deal with up to a few thousands of documents, a few paragraphs each. For algorithms designed to process millions of documents, the **Mahout** project provides better performance. It can cluster content in other languages than English, like Spanish. Carrot2 is not a search engine on its own, there is no common query syntax in Carrot2. The syntax depends on the underlying search engine you set it to use, e.g. Bing, Solr, Lucene or any other.

Features:

- Two high-quality document clustering algorithms

[41]http://project.carrot2.org/

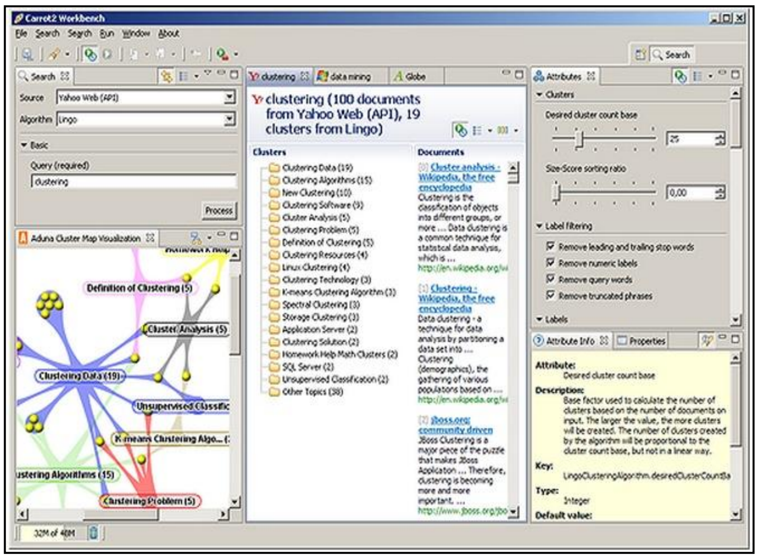
- Integrates with public and open source text search engines (Solr, Lucene)
- Easy to integrate with Java and non-Java software
- Ships a GUI application for tuning clustering for specific collections
- Ships with a simple web application
- Native C#/.NET API

Carrot2 comes with a suite of tools and APIs that can be used to quickly set up clustering on datasets, tune clustering results and call Carrot2 clustering from Java or C# code or access Carrot2 as a remote service.

Carrot2 distribution contains the following elements:

- Carrot2 Document Clustering Workbench: which is a standalone GUI application you can use to experiment with Carrot2 clustering on data from common search engines or your own data. It can fetch and cluster documents from a number of sources, including major search engines, indexing engines (Lucene, Solr) as well as generic XML feeds and files, live tuning of clustering algorithm attributes, performance benchmarking, attractive visualizations, modular architecture and extendibility (it is based on Eclipse Rich Client Platform, which makes it easily extendable).

  It is distributed for Windows, Linux 32-bit and 64-bit versions and OSX.



Source: Yahoo Web (API)
Algorithm: Lingo
Query (required): clustering

clustering (100 documents from Yahoo Web (API), 19 clusters from Lingo)

Clusters
- Clustering Data (19)
- Clustering Algorithms (15)
- New Clustering (10)
- Clustering Software (9)
- Cluster Analysis (5)
- Clustering Problem (5)
- Definition of Clustering (5)
- Clustering Resources (4)
- Linux Clustering (4)
- Clustering Technology (3)
- K-means Clustering Algorithm (3)
- Spectral Clustering (3)
- Storage Clustering (3)
- Application Server (2)
- Clustering Solution (2)
- Homework Help Math Clusters (2)
- SQL Server (2)
- Unsupervised Classification (2)
- Other Topics (38)

Documents

[0] Cluster analysis - Wikipedia, the free encyclopedia
Clustering is the classification of objects into different groups, or more ... Data clustering is a common technique for statistical data analysis, which is ...
http://en.wikipedia.org/wi

[1] Clustering - Wikipedia, the free encyclopedia
Data clustering - a technique for data analysis by partitioning a data set into ... Clustering (demographics), the gathering of various populations based on ...
http://en.wikipedia.org/wi

[2] JBoss.org: community driven
JBoss Clustering is a major piece of the puzzle that makes JBoss Application ... Therefore, clustering is becoming more and more important. ...
http://www.jboss.org/jbo

Clusters
Desired cluster count base: 25
Size-Score sorting ratio: 0,00

Label filtering
- [✓] Remove leading and trailing stop words
- [✓] Remove numeric labels
- [✓] Remove query words
- [✓] Remove truncated phrases

Labels

Attribute: Desired cluster count base
Description: Base factor used to calculate the number of clusters based on the number of documents on input. The larger the value, the more clusters will be created. The number of clusters created by the algorithm will be proportional to the cluster count base, but not in a linear way.
Key: LingoClusteringAlgorithm.desiredClusterCountBa
Type: Integer
Default value:

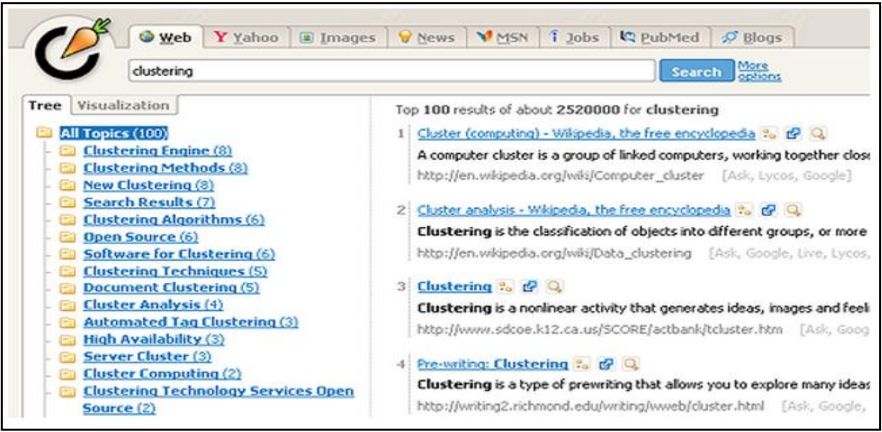
FIGURE 13. CARROT2 DOCUMENT CLUSTERING WORKBENCH SCREENSHOT

- Carrot2 Java API: for calling Carrot2 document clustering from your Java code. This package contains Carrot2 JAR files along with all dependencies, JavaDoc API reference and Java code examples. You can use this package to integrate Carrot2 clustering into your Java software.
- Carrot2 C# API: for calling Carrot2 document clustering from your C# or .NET code. This package contains all DLL libraries required to run Carrot2, C# API reference and code examples.

- Carrot2 Document Clustering Server which exposes Carrot2 clustering as a REST service. You can use Carrot2 Document Clustering Server to: integrate Carrot2 with your non-Java software, build a high-throughput document clustering system by setting up a number of load-balanced instances of the DCS. It features include: XML and JSON response formats, various document source included (including major search engines and indexing engines), direct XML feed, quick start screen that will let you make your first DCS request straight from your browser.

- Carrot2 Command Line Interface: applications which allow invoking Carrot2 clustering from command line. Currently, the only available CLI application is Carrot2 Batch Processor, which performs Carrot2 XML format and saves the results as XML or JSON. Apart from clustering large number of documents sets at one time, you can use the Carrot2 Batch Processor to integrate Carrot2 with your non-Java applications.

- Carrot2 Web Application[42]which exposes Carrot2 clustering as a web application for end users. It features include: two views of the clusters generated by Carrot2, contains a large number of document sources, including major search engines, XSLT and JavaScript-based presentation layer, high-performance front-end being optimized by using technique as JavaScript and CSS merging and minification.



All Topics (100)
- Clustering Engine (8)
- Clustering Methods (8)
- New Clustering (8)
- Search Results (7)
- Clustering Algorithms (6)
- Open Source (6)
- Software for Clustering (6)
- Clustering Techniques (5)
- Document Clustering (5)
- Cluster Analysis (4)
- Automated Tag Clustering (3)
- High Availability (3)
- Server Cluster (3)
- Cluster Computing (2)
- Clustering Technology Services Open Source (2)

Top **100** results of about **2520000** for **clustering**

1 Cluster (computing) - Wikipedia, the free encyclopedia
A computer cluster is a group of linked computers, working together clos
http://en.wikipedia.org/wiki/Computer_cluster [Ask, Lycos, Google]

2 Cluster analysis - Wikipedia, the free encyclopedia
Clustering is the classification of objects into different groups, or more
http://en.wikipedia.org/wiki/Data_clustering [Ask, Google, Live, Lycos,

3 Clustering
Clustering is a nonlinear activity that generates ideas, images and feel
http://www.sdcoe.k12.ca.us/SCORE/actbank/tcluster.htm [Ask, Goog

4 Pre-writing: Clustering
Clustering is a type of prewriting that allows you to explore many ideas
http://writing2.richmond.edu/writing/wweb/cluster.html [Ask, Google,

FIGURE 14. CARROT2 WEB APPLICATION RESULTS SCREEN

[42] http://search.carrot2.org/stable/search

4.12. Natural Language Processing and Opinion Mining Summary

The table below presents a summary of NLP and opinion mining tools.

Name of the tool	Developers	Latest release	Written in	Operating System	Type	Licence
NLTK	NLTK-Developers and contributors	3.2.5	Python	Cross-platform	Text classification/NLP framework	Apache 2.0
LingPipe	Alias-i	4.1.0	Java	Cross-platform	Entity Recognition/ Text classification	Alias-I royalty free license
Weka	University of Waikato	3.8.1	Java	Cross-platform	Text classification/ Data mining	GPL
LibShort Text	LibShortText project	1.1	Python and C/C++	Cross-platform	Text classification	BSD
Mallet	Andrew McCallum, graduate students and staff	2.0.8	Java	Cross-platform	Topic modeling/ Text classification	Common Public License 1.0
Gensim	RadimRehurek	3.1.0	Pyton	Cross-platform	Topic modeling	LGPL
KEA	Gordon Paynter	5.0	Java	Cross-platform	Key word extraction	GPL
Maui	AlyonaMedelyan	2.0	Google code	Cross-platform	Key word extraction	GPL v3
MEAD	University of Michigan	2.2.0	Perl: requires XML-related modules and external software package	Cross-platform	Text summarization	GPL
Wikipedia Miner	University of Waikato	1.2.0.	Java	Cross-platform	Entity Linking	GPL v2
Illinois Wikifier	University of Illinois	2.0	N/A	Cross-platform	Entity Linking	GPL
Illinois Name Entity Tagger	University of Illinois	2.7	Java	Cross-platform	Name Entity Recognition	GPL

OpenNLP	Apache Software Foundation	1.8.3 April 2013	Java	Cross-platform	Entity Recognition/ Coreference resolution/ Parsing	LGPL v2
Stanford NLP	Stanford university	3.8.0	Java 1.6 or later for the last version	Cross-platform	Entity Linking/ Co-reference resolution	GPL v2
GATE	GATE research team, Dept. Computer Science, University of Sheffield	8.4.1	Java	Cross-platform	Entity Recognition	LGPL
UIMA	IBM, Apache Software Foundation	2.10.2	Java with C++ Enablement	Cross-platform	NLP Framework	Apache License 2.0
Carrot2	Carrot Search	3.15.1	Java	Cross-platform	Text mining and cluster analysis	BSD

TABLE 4. SUMMARY OF NLP TOOLS

5. CONCLUSIONS

This document summarized the algorithm and tools that can be used in three areas of Artificial Intelligence and Data mining: Machine Learning, Pattern Recognition and Natural Language Processing and Opinion Mining. In each section we had also presented a short evaluation of each tool, mentioning what type of problem can be solved by using that specific tool.

As is already mentioned, this document represents only the first version of a living document. The involve partners will continue the research in the fields presented in this document with the goal to update D3.1 with the new tools that will be available in the next months and that will be suitable, form the technical point of view, with the DID Toolkit.

During the next phase of the project all involved partners will continue to evaluate the tools and will choose the ones that will be used for development of DID toolkit. In this way, we will be able to construct a guiding report, based on the experience gather during development of the tool, report that will be deliver as the second version of D3.1.