Bayesian Network based Extreme Learning Machine for Subjectivity Detection

Iti Chaturvedi[†], Edoardo Ragusa[‡],
Paolo Gastaldo[‡], Rodolfo Zunino[‡], Erik Cambria[†⋆]

[†]*School of Computer Science and Engineering, Nanyang Technological University, Singapore*

[‡]*Dept. of Electrical, Electronic, Telecommunications Engineering and Naval Architecture, DITEN, University of Genoa, Genova, Italy*

[⋆]*Corresponding author (cambria@ntu.edu.sg)*

Abstract

Subjectivity detection is a task of natural language processing that aims to remove 'factual' or 'neutral' content, i.e., objective text that does not contain any opinion, from online product reviews. Such a pre-processing step is crucial to increase the accuracy of sentiment analysis systems, as these are usually optimized for the binary classification task of distinguishing between positive and negative content. In this paper, we extend the extreme learning machine (ELM) paradigm to a novel framework that exploits the features of both Bayesian networks and fuzzy recurrent neural networks to perform subjectivity detection. In particular, Bayesian networks are used to build a network of connections among the hidden neurons of the conventional ELM configuration in order to capture dependencies in high-dimensional data. Next, a fuzzy recurrent neural network inherits the overall structure generated by the Bayesian networks to model temporal features in the predictor. Experimental results confirmed the ability of the proposed framework to deal with standard subjectivity detection problems and also proved its capacity to address portability across languages in translation tasks.

Preprint submitted to Journal of The Franklin Institute *January 2, 2018*

Keywords: Sentiment Analysis, Subjectivity Detection, Extreme Learning Machine, Bayesian Networks

1. Introduction

In recent years, sentiment analysis of social media data on blogs, online communities, Wikis, microblogging platforms, and other online collaborative media has become increasingly popular [1]. The distillation of knowledge from such a big amount of unstructured information, however, is an extremely difficult task, as the contents of today's Web are perfectly suitable for human consumption, but remain hardly accessible to machines. Sentiment analysis is a branch of affective computing research [2] that aims to classify text (but sometimes also audio and video [3]) into either positive or negative. Sentiment analysis systems can be broadly categorized into knowledge-based and statistics-based. While most works approach it as a simple categorization problem, sentiment analysis is actually a 'suitcase' research problem that requires tackling many natural language processing (NLP) sub-tasks, including aspect extraction [4], named entity recognition [5], word polarity disambiguation [6], personality recognition [7], sarcasm detection [8], and subjectivity detection. In the Big Data landscape, in particular, subjectivity detection is of utmost importance, because it allows for filtering out the huge amount of objective text, e.g., neutral sentences that are not useful for polarity detection. For example, objective sentences are often spam or off-topic questions such as 'What is the resolution of the Camera?'. Subjective sentences, instead, are of wishful nature, which indicates purchasing interest [9]. In this paper, we consider two key applications of subjectivity detection namely: summarizing product reviews on dedicated sites like 'Rotten-tomatoes' or 'Amazon' [10]

and summarizing multi-perspective news articles [11, 12].

In the above applications, the sentence model has to be very sensitive as users have different levels of expertise on the topic of discussion and may be from diverse economic and educational backgrounds, as well as being often separated by large geographical distances. Furthermore, subjectivity detection in product reviews targets the psychology of an investor by breaking down factual information, which may imply positive or negative sentiment that is otherwise undetected by coarse-grained methods (as these merely focus on detecting explicit sentiments [13]). Another application is monitoring response of people to different crisis situations. This is done by processing microblogging platforms such as Twitter and Facebook. Here, some of the main challenges are the use of abbreviations and hashtags. Tweets may also possess dual meanings due to the potential contexts of discussion. For example, in political tweets the word 'grun' - 'green' is used for the political party 'Die Grunen' - 'The Greens', but it is also used in reference to the color green [14].

In [15], the authors show that subjective sentences in online forums can be identified by 'Dialog Acts' such as 'Question', 'Repeated Question', 'Clarification', etc. They also show that subjective sentences are longer than objective sentences and often contain inappropriate content such as abusive language. Traditional sentence models extract significant k-gram features and classify them using a Naïve Bayes model. For example, $cloud\_computing$ is a bi-gram of two words frequently used together. Since the number of such features is exponential, a convolutional neural networks (CNN) is usually adopted to automatically learn these from large datasets. We have previously proposed the use of a deep CNN to extract subjectivity features in Spanish and English and combined features from

different languages multiple kernel learning [16]. Although very promising for long texts, this model was unable to compute parameters for short tweets and was dependent on human annotation, which is often inaccurate for long sentences.

For example, in the sentence "Those digging graves for others, get engraved themselves', he [Abdullah] said while citing the example of Afghanistan" there is clearly an objective frame for the writer and a direct subjective frame for Abdullah with the text anchor "said". However, it is ambiguous whether the texts anchor "citing" is objective or subjective in nature [17]. Lastly, subjectivity in tweets is a temporal phenomenon, e.g., the support for a candidate during elections will diffuse through a social network from nearby tweets [18]. To overcame these challenges, in this paper we propose a new version of extreme learning machine (ELM), termed Bayesian network based ELM (BNELM), which learns non-linear relationships between the hidden-layer neurons. For the purpose of subjectivity detection, we first extract k-gram features using a deep CNN and, hence, train BNELM with the low-dimensional features learnt by the CNN.

The trained BNELMs are efficient compared to traditional ELMs, as they are able to prune redundant hidden neurons by learning a prior for weights. BNELMs also outperform sparse Bayesian ELM (SBELM) [19] because they do not need to compute the Hessian matrix of second-order derivatives that often does not exist for noisy datasets. Instead, BNELM employs heuristic Markov chain Monte Carlo (MCMC) sampling with a Gaussian Bayesian network fitness function to determine the weights between hidden neurons. Another shortcoming of the traditional ELM is that it does not generalize to non-linear datasets such as a sequence of sentences. Hence, we introduce a recurrent layer of hidden neurons to model temporal features in long tweets. Lastly, since recurrent neurons may

become unstable on noisy data, a fuzzy classifier is used to stabilize the model and predict the output labels. We evaluate our model on three different datasets: the first is a question-answering corpus from news articles, the second is about product reviews, and the third is a multi-class Twitter corpus.

2. Contributions

ELMs are single-hidden-layer, feed-forward neural networks. In general, feed-forward neural networks provide a powerful paradigm for inductive learning; however, the learning procedure may require one to tackle a few major issues, such as convergence speed, setting of free parameters, and overfitting. Several authors have shown that it is possible to exploit randomization in feed-forward neural network configuration resulting in a simplified training procedure, while maintaining a notable generalization performance [20, 21, 22, 23, 24]. This is the peculiar aspect of ELM: the hidden weights are set randomly and can be tuned using a regularized least squares (RLS) problem in a linear space [25]. However, in most applications ELM requires a very large number of hidden neurons, resulting in a huge amount of used memory and slow computational performance. Furthermore, the trained classifier, in general, is not sparse.

To achieve sparsity, SBELM [19, 26] were proposed. In SBELM, one imposes a hierarchical independent prior on each weight, also known as automatic relevance determination (ARD) prior. In this way, several weights are pruned and the trained predictor only uses a small number of hidden neurons. The common approach to learning the prior parameters is to maximize the log likelihood of the dataset after marginalization of the weights. To tackle the intractable integral required for marginalization, Laplace approximation is used [26]. This involves

the computation of Hessian matrix, which requires computation of second order derivatives. The whole process is sensitive to noisy data and could be affected by numerical instability. Moreover, the optimization process involved in the maximization of the likelihood relies on a non-convex function.

In contrast, we propose to augment the conventional SBELM model with Bayesian networks, which employ heuristics to determine the prior parameters of weights in the output layer [27, 28]. In practice, the resulting BNELM allows one to replace Laplace approximation with a heuristic process. The rationale behind such a solution is twofold. First, Laplace approximation of SBELM is often difficult to compute, as the gradient may not exist on several noisy datasets. Second, by removing Laplace approximation one discards the corresponding non-convex optimization problem. The heuristic process in Bayesian networks samples a Markov chain of weights that always converges to the global maximum at equilibrium, resulting in a higher accuracy of the predictor.

The proposed framework for subjectivity detection integrates CNNs, ELMs, and Bayesian networks. Each element in the model has a specific role in the process of classification. First, a deep CNN automatically learns significant k-gram features from the training sentences. Then, the BNELM model receives these features as input. The recurrent layer supports BNELM in embedding temporal dynamics. Since the traditional recurrent layer is often unstable, BNELM employs a layer of fuzzy recurrent neurons to the specific purpose of achieving stability. A crucial property of this framework is the ability to learn a dictionary of features that are portable to new languages and domains. Such a characteristic becomes important when it is difficult and expensive to generate training data in new languages.

The experiments are designed to both verify the effectiveness of BNELM in capturing dependencies in high-dimensional data and to assess the ability of the framework to address portability in language translation tasks. Hence, first we considered the multi-perspective question answering (MPQA) corpus of 504 sentences manually annotated for subjectivity in Spanish [29, 30]. Here, we tried to develop a subjectivity lexicon for the Spanish language using available resources in English. Next, in order to evaluate the method on a multi-class problem, we considered the multimodal opinion utterances dataset (MOUD) [31]. This dataset contains videos of product reviews from YouTube. On average, each video has 6 utterances and each utterance is 5-second long. Each utterance in a video is annotated separately. Hence, sentiment can change during the course of a product review. The dataset contains 498 utterances labeled as 'positive', 'negative', or 'neutral'. In our experiment, only the text transcript of each spoken utterance was considered. Lastly, to evaluate the method on a very large noisy dataset, we consider the four-class TASS corpus, which is a collection of Spanish tweets commonly used for the evaluation of social media analysis tasks [32]. The classification accuracy obtained using the proposed BNELM was shown to outperform the baseline by over 20% on all three real datasets. The rest of the paper is organized as follows: Section 3 provides the preliminary concepts necessary to comprehend the proposed approach; Section 4 introduces the BNELM model for subjectivity detection; Section 5 validates the model on real-world benchmark datasets; finally, Section 6 concludes the paper.

3. Background

This section briefly reviews the theoretical models that support the proposed framework. First, a description of Bayesian networks for sentences is provided. Next, we detail the heuristic approach to find the optimal structure of a Bayesian network, termed MCMC. Finally, the SBELM is described.

3.1. Gaussian Bayesian Networks

A Bayesian network is a graphical model that represents a joint multivariate probability distribution for a set of random variables [33]. It is a directed acyclic graph that has a structure s with N nodes and a set of parameters $\boldsymbol{\theta}$, which represent the strengths of connections by conditional probabilities. Given a set $\mathbf{X}$ of Z samples $\{\mathrm{x}_i; \mathbf{x} \in \mathbb{R}^{\mathrm{N}}; i = 1, .., \mathrm{Z}\}$, the Bayesian network decomposes the likelihood of node expressions into a product of conditional probabilities by assuming the independence of non-descendant nodes, given their parents.

$$p(\mathbf{x}|s, \boldsymbol{\theta}) = \prod_{i=1}^{\mathrm{N}} p(\mathrm{x}_i|\boldsymbol{a}_i, \theta_{i,\boldsymbol{a}_i}), \tag{1}$$

where $p(\mathrm{x}_i|\boldsymbol{a}_i, \theta_{i,\boldsymbol{a}_i})$ denotes the conditional probability of node expression x_i given its parent node expressions $\boldsymbol{a}_i$, and $\theta_{i,\boldsymbol{a}_i}$ denotes the maximum likelihood (ML) estimate of the conditional probabilities. Fig. 1(a) Bayesian network for a multivariate system with five nodes. Each node is a variable in the state-space of the system that can be observed or measured. The connections represent causal dependencies within a single time instance. The observed state of variable i is denoted as x_i and the regulation or conditional probability of variable i given variable j is $p(\mathrm{x}_i|\mathrm{x}_j)$.

An example of a Bayesian network representing inter-dependencies between the words of the sentence “The escalation must end any time soon” is illustrated

in Fig. 1(b). Once the structure of the Bayesian network is determine heuristically using the training data, the context (that is, the parents for each word) can be established. For example, the context of the word 'soon' is 'must' and 'end'. Hence, our hypothesis is that structurally related words, among all the words within the sentence, provide the best contextual information for polarity detection.

The vocabulary in most languages is very large, hence, instead of individual words as shown in Fig. 1(b), we can consider a Bayesian network over N patterns of maximum length k words, and the corresponding frequency in each training sentence. Subjectivity patterns can be handcrafted or learnt automatically using a software like AutoSlog [34]. For example, the pattern '$< x >$ was asked' would extract 'he was asked to leave the premises' and is strongly subjective. On the other hand, the pattern '$< x >$ was expected' would 'he was expected to retire' and is objective as it is a mere fact.

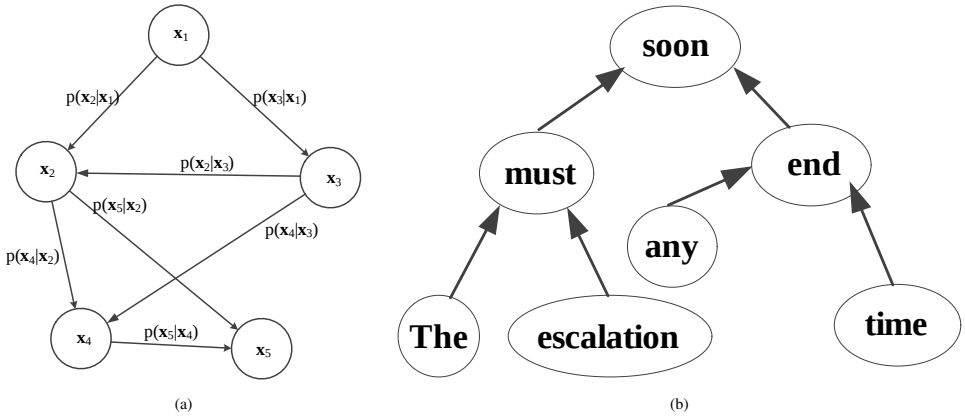

Figure 1: (a) Bayesian network for a multivariate system with five nodes. Each node is a variable in the state-space of the system that can be observed or measured. The connections represent causal dependencies within a single time instant. (b) Bayesian network representing inter-dependencies between the words of the sentence 'The escalation must end any time soon'.

The optimal structure s^* is obtained by maximizing the posterior probability of s given the data $\mathbf{X}$. From Bayes theorem, the optimal structure s^* is given by

$$s^* = \arg\max_s p(s|\mathbf{X}) = \arg\max_s p(s) \cdot p(\mathbf{X}|s), \tag{2}$$

where $p(s)$ is the probability of the network structure and $p(\mathbf{X}|s)$ is the likelihood of the expression data given the network structure.

Given the set of conditional distributions with parameters $\boldsymbol{\theta} = \{\theta_{i,\boldsymbol{a}_i}\}_{i=1}^{\mathrm{N}}$, the likelihood of the data is given by

$$p(\mathbf{X}|s) = \int p(\mathbf{X}|s, \boldsymbol{\theta}) \cdot p(\boldsymbol{\theta}|s) d\boldsymbol{\theta}, \tag{3}$$

To find the likelihood in (3), and to obtain the optimal structure as in (2), we can use the Laplace approximation of integrals [35]. A Gaussian Bayesian network assumes that the nodes are multivariate Gaussian. The parameters $\theta_{i,\boldsymbol{a}_i}$ are then defined by the mean μ and the covariance matrix Σ of size $\mathrm{N} \times \mathrm{N}$ [36]. The joint probability of the network can be the product of a set of conditional probability distributions is then given by:

$$p(\mathrm{x}_i|\boldsymbol{a}_i, \theta_{i,\boldsymbol{a}_i}) = \mathcal{N}\left(\mu_i + \sum_{j \in \boldsymbol{a}_i} (\mathrm{x}_j - \mu_j)\beta, \Sigma_i^{'}\right), \tag{4}$$

where $\Sigma_i^{'} = \Sigma_i - \Sigma_{i,\boldsymbol{a}_i} \Sigma_{\boldsymbol{a}_i}^{-1} \Sigma_{i,\boldsymbol{a}_i}^T$ and β denotes the regression coefficient matrix, $\Sigma_i^{'}$ is the conditional variance of x_i given its parent set $\boldsymbol{a}_i$, $\Sigma_{i,\boldsymbol{a}_i}$ is the covariance between observations of x_i and the variables in $\boldsymbol{a}_i$, and $\Sigma_{\boldsymbol{a}_i}$ is the covariance matrix of $\boldsymbol{a}_i$.

3.2. Markov Chain Monte Carlo

In order to find the optimal structure, a Markov chain of structures is formed using a MCMC simulation, which converges to the optimal structure at the equilibrium. The Metropolis-Hastings method of MCMC is adopted, which associates

an acceptance mechanism with newly drawn sample structures. The acceptance of a new structure s^{new} is given by the following equation:

$$\min\left\{1, \frac{p(s^{\mathrm{new}})}{p(s)} \cdot \frac{p(s^{\mathrm{new}}|\mathbf{X})}{p(s|\mathbf{X})} \cdot \frac{p(s^{\mathrm{new}}|s)}{p(s|s^{\mathrm{new}})}\right\} \tag{5}$$

where the Metropolis-Hastings acceptance ratio:

$$\alpha \;=\; \frac{p(s^{\mathrm{new}}|\mathbf{X})}{p(s|\mathbf{X})} \cdot \frac{p(s^{\mathrm{new}}|s)}{p(s|s^{\mathrm{new}})}. \tag{6}$$

when adding an edge and the prior ratio is inverted when deleting an edge.

Sampling new structures with the use of the above-listed procedure generates a Markov chain, which converges in distribution to the approximate posterior distribution. Taking the average over sampled structures after a burn-in period, we can compute the integral over parameters in (3). In practice, a new network structure is proposed by applying one of the elementary operations such as deleting, reversing, or adding an edge, and then discarding structures that violate the acyclic condition. The first and second term of the acceptance ratio, the ratio of likelihoods, is computed using (4). The third term, is obtained by

$$\frac{p(s^{\mathrm{new}}|s)}{p(s|s^{\mathrm{new}})} = \frac{\mathrm{N_n}^{\mathrm{new}}}{\mathrm{N_n}} \tag{7}$$

where $\mathrm{N_n}$ denotes the size of the neighborhood obtained by elementary operations on structure s as well as counting the valid structures.

3.3. Sparse Bayesian ELM

ELMs are single layer feed-forward neural networks that are trained several times faster than traditional neural networks. The main idea behind ELM lies in

the random initialization of hidden-layer input weights and biases. The hidden-layer output weights are then computed using a Moore-Penrose generalized inversion of an input layer weights and the known target values of training data. In this section, we provide the details of ELM algorithm.

Let D be a labeled training set $\{(\mathbf{x}, \mathrm{y})_i; \mathbf{x} \in \mathbb{R}^{\mathrm{N}}; y \in \{Pos, Neu, Neg\}; i = 1, .., \mathrm{Z}\}$. The hypothesis space of ELM can be formalized as follows

$$\mathbf{y} = \sum\nolimits_{i=1}^{\mathrm{N}} \mathrm{w}_i h_i(\mathbf{x}, \boldsymbol{\beta}_i, \mathrm{b}_i) \tag{8}$$

where N is the number of hidden neurons, $\boldsymbol{\beta_i} \in \mathbb{R}^N$ and $b_i \in \mathbb{R}$ are the parameters of the i^{th} hidden neuron, $\mathrm{w}_i \in \mathbb{R}$ and h_i is a non-linear piecewise function satisfying ELM universal approximation capability theorems [37]. The ELM randomly assigns input weights $\boldsymbol{\beta}_i$ and biases b_i. Let $\mathbf{H}$ denote a $\mathrm{Z} \times \mathrm{N}$ matrix, where $h_{i,j} = h_j(\mathbf{x}_i)$ and Z is the number of samples in the training set; then, the minimization problem can be expressed as:

$$\min_{\mathbf{w}} ||\mathbf{y} - \mathbf{H}\mathbf{w}||^2 + \lambda ||\mathbf{w}||^2 \tag{9}$$

The vector of weights $\mathbf{w}$ is then obtained as follows:

$$\mathbf{w} = (\mathbf{H^T H} + \lambda \mathbf{I})^{-1} \mathbf{H^T y} \tag{10}$$

Bayesian ELM (BELM) is based on the use of Bayesian linear regression to optimize the weights of the output layer. The Bayes law states that the posterior distribution of model parameters is proportional to the product of the prior distribution and the likelihood:

$$p(\mathbf{w}|D) \propto p(\mathbf{w}) \cdot (D|\mathbf{w}) \tag{11}$$

Next, the output distribution of the model y_{new} for new input $\mathbf{x}_{new}$ is given by the integral of the posterior distribution of the parameters $\mathbf{w}$. Therefore, the predictive distribution for a new input is given by:

$$p(\mathrm{y}_{new}|\mathbf{x}_{new}, D) = \int p(\mathrm{y}_{new}|\mathbf{x}_{new}, \mathbf{w}) \cdot p(\mathbf{w}|D) d\mathbf{w} \tag{12}$$

where the data is assumed Gaussian, and the maximum likelihood estimate of the weights maximizes the posterior probability.

The core of SBELM resides in the application of an ARD prior to the linear weight, formally

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \mathcal{N}(\mathbf{w}, \boldsymbol{\alpha}) \tag{13}$$

The training consists of learning the parameters $\boldsymbol{\alpha}$ maximizing:

$$p(\mathbf{y}|\boldsymbol{\alpha}, \mathbf{H}) = \int p(\mathbf{y}|\mathbf{w}, \mathbf{H}) \cdot p(\mathbf{w}|\boldsymbol{\alpha}) d\mathbf{w}. \tag{14}$$

The integral (14) is intractable. However, one can address (14) by using Laplace approximation, which involves a quadratic Taylor expansion of the log form of posterior probability. This in turn requires the computation of the Hessian matrix, which is a matrix of second-order derivatives. Here, we note that presence of noise in the dataset can heavily affect this approximation.

Finally, the predictive distribution is obtained by

$$p(\mathrm{y}_{new}|\mathbf{x}_{\boldsymbol{new}}, \hat{\mathbf{w}}) = \frac{1}{1 + e^{-\mathbf{h}_{\mathbf{new}}\hat{\mathbf{w}}}} \tag{15}$$

where $\hat{\mathbf{w}}$ is the Laplace's mean and $\mathbf{h}_{\boldsymbol{new}}$ correspond to the activation of the ELM random layer with input $\mathbf{x}_{new}$.

4. BNELM for Subjectivity Detection

In this section, we introduce the novel BNELM, which extends the traditional ELM to non-linear datasets (e.g., sequence of sentences) by integrating the features of Bayesian networks and fuzzy recurrent neural networks (RNNs). In the proposed model, Bayesian networks provide an effective tool to build a network of connections among the hidden neurons of the conventional ELM configuration. Such a step relies on unsupervised learning, as labels are not involved in the process. A fuzzy RNN inherits the overall structure generated by the Bayesian networks to the purpose of supporting a predictor that can model also temporal features. In the following sections, we will formalize BNELM and will clarify its advantages over SBELM (Sec. 4.1) and we will present the framework that uses BNELM to perform subjectivities detection (Sec. 4.2).

4.1. Bayesian Network Based Extreme Learning Machine

The proposed BNELM augments the standard structure of a RNN to generate a predictor that can take advantage of two main features. First, the weight matrix of connections between input nodes and hidden neurons can be learnt by combining the abilities of ELMs and Bayesian networks. Second, temporal features can be suitably modeled. In a standard RNN, the output $\mathrm{y}(t)$ at time step t is calculated using the following equation:

$$\mathrm{y}(t) = f(W_{\mathrm{R}} \cdot \mathrm{y}(t-1) + W \cdot \mathbf{x}(t)) \tag{16}$$

where W_{R} is the interconnection matrix among hidden neurons, W is the weight matrix of connections between hidden neurons and the input nodes, and f is a non-linear activation function. In BNELM, matrix W is learnt by using the ELM's hidden neurons outputs $\boldsymbol{h}$ to train a Bayesian network.

Hence, the number of nodes N of the Bayesian network is equal to the number of hidden neurons. The computation of the connection matrix is achieved simply by finding the optimal structure s^* of dimension $\mathrm{N} \times \mathrm{N}$ defined in (2); accordingly, the hidden neurons outputs of the ELM become the inputs for a Gaussian Bayesian network. Next, the learnt structure s^* replaces the W in the recurrent layer. Algorithm 1 illustrates the complete training procedure for W with a Gaussian Bayesian network fitness function and MCMC simulation.

The new structure for the recurrent layer becomes:

$$\mathrm{y}(t) = f(W_{\mathrm{R}} \cdot \mathrm{y}(t-1) + s^* \cdot \mathbf{h}(t)) \tag{17}$$

Back propagation through time is utilized to learn W_{R}. As recurrent neurons can prove to be unstable on noisy data, a fuzzy classifier is actually exploited in this work to stabilize the model. Therefore, $\mathrm{y}(t)$ eventually becomes $\mathbf{y}(t)$ by introducing fuzzy membership functions.

The BNELM model has two main advantages with respect to SBELM. Firstly, training a SBELM involves the computation of second-order derivatives. The performance of this procedure is highly sensitive to noisy data. In BNELM training, this problem is overcome by using a heuristic MCMC as described in Section 3.1 and Section 3.2. Secondly, SBELM involves a non-convex optimization problem. On the other hand, the heuristic process in Bayesian networks samples a Markov chain of weights that always converges to the global maximum at equilibrium, resulting in a higher accuracy of the predictor.

Moreover, one should consider that determining the number of optimal hidden neurons is one of the biggest challenges in ELM. The MCMC algorithm will only consider edges with frequency above a threshold in all samples when determining the optimal structure s^*. Hence, neurons with no edges are removed from the

structure. In this way, the Bayesian network is able to determine the optimal number of hidden neurons, thus pruning redundant neurons.

4.2. A Framework for Subjectivity Detection

In this paper, BNELM provides a suitable tool to support the development of a framework for subjectivity detection. Figure 2 schematizes the framework by proposing the whole flowchart.

The BNELM model receives as input a vector $\tilde{\boldsymbol{x}}$, which is the outcome of a pre-processing step involving a deep CNN model. First, a sentence is transformed into word vector representation $\mathbf{X}$ of dimensions $L \times d$, where L is the maximum number of words in a sentence, and d is the number of Google word vectors used. The transformed data is then fed to the deep CNN model, which automatically performs dimensionality reduction. Deep CNNs are recently being used extensively to extract patterns automatically from large datasets such as Twitter. Such a model looks for highly activated k-grams in a CNN as our pattern set. For example, $cloud\_computing$ is a bi-gram composed by the words $cloud$ and $computing$. Details of such an implementation can be found in [16]. However, the training of deep models can be very time consuming. To this end, here we use the activations at the penultimate layer of the deep CNN as a new training data for a fast ELM model. The first CNN hidden layer contains kernels of size $k \times d$ to learn k-gram patterns. There are several layers of kernels and an output layer of n_d sentiment labels namely 'positive', 'negative' and 'neutral'. The features learnt by the deep CNN are expressed in the penultimate layer and can be used as input $\tilde{\boldsymbol{x}}$ to the BNELM model.

BNELM in practice tries to learn the context of the k-gram features learnt by the CNN. Since BNELM embeds a recurrent layer of hidden neurons, it is also

Algorithm 1 Bayesian Network based Extreme Learning Machine training

Given a labeled training set $\{(\mathbf{x}, \boldsymbol{y})_i; \mathbf{x} \in \mathbb{R}^{\mathrm{N}}; \boldsymbol{y} \in \mathbb{R}^{\mathrm{P}}; i = 1, .., \mathrm{Z}\}$;

Procedure:

Randomly initialize weights of the input hidden layer, with N neurons
Build a Bayesian network with N nodes, one node for each neuron in hidden layer activation $\mathbf{h}$
Initialize topology of $s = s_0$
repeat
    Generate s^{new} by elementary operations
    Find N^{new} and N^{old} corresponding to s^{new} and s
    Find acceptance ratio α given by (6)
    if $\alpha \geq 1$ **then**
        $s = s^{new}$
    else
        if $rand[0, 1] \geq \alpha$ **then**
            $s = s^{new}$
        end if
    end if
until convergence
$s^* \cdot \mathbf{h}$ is used to train the final fuzzy recurrent classifier

able to model temporal features in long product reviews. In Figure 2, the fuzzy membership function is designed to tackle a three-class problem. The complete training procedure is proposed in Algorithm 2. The corresponding state diagram for BNELM is illustrated in Figure 3. Starting from the bottom, the training of the model is organized as follows:

- The first two layers starting from the bottom of Figure 3 correspond to the Bayesian network used in the pre-processing step.
- The significant k-gram features are extracted from sentences using a deep CNN. In the figure, the CNN corresponds to the third layer. The bold lines identify the size of the kernel window, which has been set to 4.
- The extracted features are then fed to the random layer of BNELM; in this figure, the links are represented with dashed lines. The output weights of the random layer have been learnt using a Bayesian network.
- The ELM output is used to train a layer of recurrent neurons with feedback connections, marked as "Inter-connected recurrent neurons" in the graph.
- The two fuzzy membership functions are used to facilitate stable convergence of the model.
- The top layer represents the output layer of the network. It has three neurons; one per class.

4.3. Computational Complexity

The computational complexity of a single training epoch for the l^{th} convolutional layer of a CNN is given by $O(n_{l-1}.s_l^2.n_l.m_l^2)$, where n_{l-1} and n_l are the

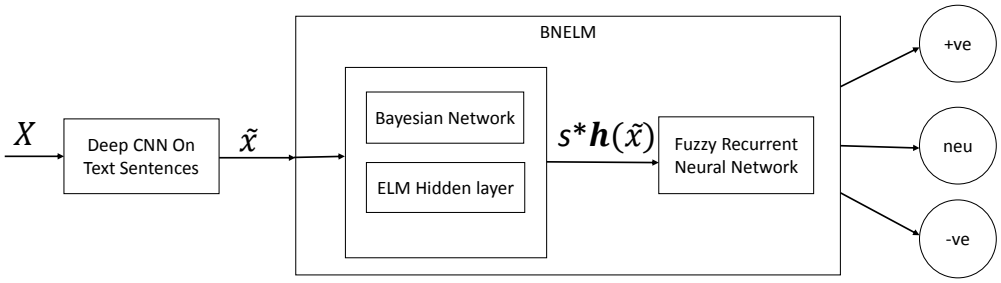

Figure 2: Flowchart of the BNELM framework. First, we run a conventional deep CNN on sentences. The extracted features are then fed to an ELM classifier, where the output layer weights are determined heuristically using Bayesian networks. Features learnt are further evolved by BNELM using a fuzzy RNN. The output layer has three nodes for classifying sentences as positive, negative, or neutral.

number of input and output feature maps, respectively; $s_l = n_x^{l-1} \times n_y^{l-1}$ and $m_l = n_x^l \times n_y^l$ are the dimensions of the input and output feature maps, respectively. This computational cost clearly characterizes the computational complexity of the overall framework. In contrast, the computational complexity of a layer of recurrent hidden neurons is much lower being $O(2 \times N^2)$, where N is the number of neurons and a single time delay is considered. Similarly, the complexity of the neuro-fuzzy classifier with two membership functions is also very small being $O(\sum_{i=1}^{n_{l-1}} 2n_l + \sum_{i=1}^{n_{l-1}} 2)$ [38].

In general, training a CNN as a classifier is indeed time consuming due to the

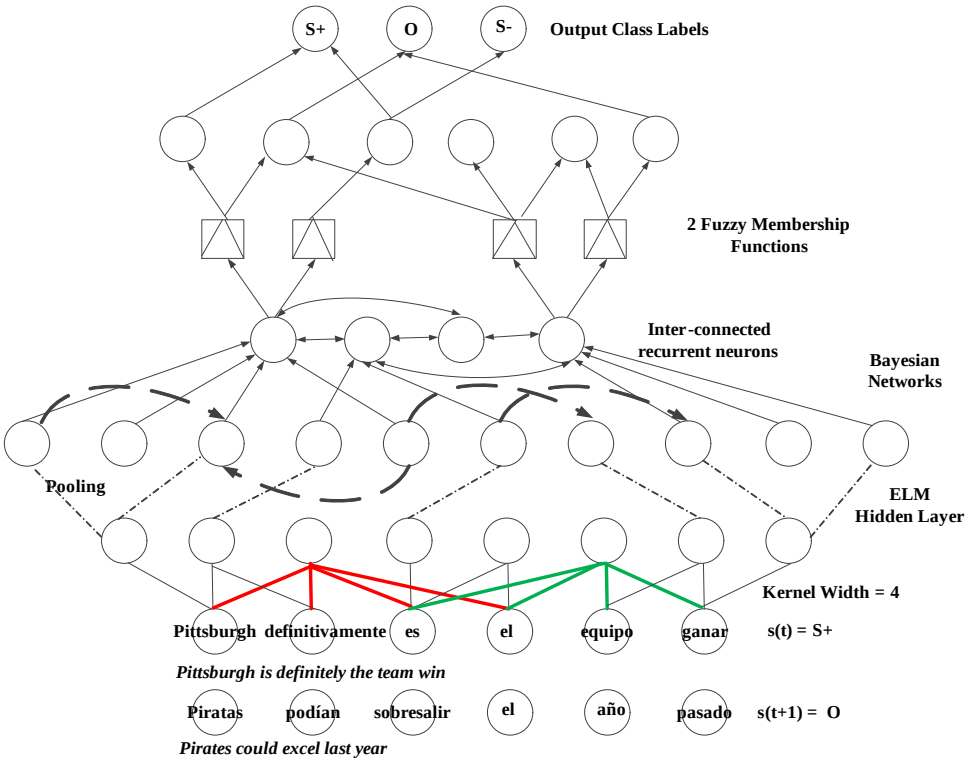

Figure 3: State space of a BNELM for a subjective sentence in online forums. Features are extracted from Spanish sentences using a deep CNN. The bold lines correspond to kernels. The extracted features are then used to train a Bayesian ELM, where output layer weights are determined using Bayesian networks. The bold dashed arcs correspond to causal edges predicted by a Bayesian network. The ELM output is subsequently used to train a layer of recurrent neurons with feedback connections. Lastly, we introduce a layer of fuzzy neurons with two membership functions in order to achieve stable convergence of the model.

huge number of training iterations required. In the proposed method, however, the CNN is utilized as a feature extractor; eventually, such features are fed to a low-dimensional Bayesian ELM model. As a result, in this case, the CNN only involves a reduced number of epochs; this in turn means a significant reduction

Algorithm 2 Subjectivity detection framework training

Given a labeled training set $\{(\mathbf{x}, \mathrm{y})_i; i = 1, .., \mathrm{Z}\}$ where each $\mathbf{x}$ is a sentence of length L and labels $\mathrm{y} \in \{Pos, Neu, Neg\}$

1: Transform each sentences in vector representation with word vector dimension d
2: Construct a deep CNN with visible layer as a 2-d vector of $L \times d$ input features
3: Construct hidden layer with kernels of size $k \times d$ to learn k-gram patterns
4: Construct several hidden layers with kernels and output layer with n_d neurons
5: The features learnt by the deep CNN are expressed in penultimate layer and can be used as input layer to the BNELM model

of the computational cost. Lastly, the present framework is designed to exploit a small number of features learnt by deep learning. Thus, the computational cost of the MCMC with Bayesian network fitness function also decreases. It is worth noting that model obtained after the training process is sparse; this is a major difference with the model one would obtain by exploiting a standard ELM model. As a consequence, we can improve the computational performance of the predictor.

5. Experiments and Results

The proposed experiments aim at (1) evaluating the generalization performance of BNELM and (2) providing a comparison between the proposed method and two alternative models (namely, a classifier based on the standard regularized ELM [37] and a classifier based on a SBELM [26]). Three different benchmarks have been used in the experimental evaluation: MPQA [29, 30], MOUD [39] and TASS 2015 Corpus. All datasets involve sentences expressed in Spanish. The

rationale behind this setup is that present research is geared towards assessing the proposed methods ability to deal with non-English language documents. In all the experiments, standard model-selection procedures support the setup of the ELM regularization parameter λ. The following settings have been used:

$\lambda \in \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}, 10^{4}, 10^{5}, 10^{6}\}$

5.1. Pre-processing

A pre-processing stage has been applied to all the datasets in the experiments. The first step consisted in removing the top 50 stopwords and punctuation marks from the sentences[1]. Next, a part of speech (POS) tagger was used to determine the POS tag for each word in a sentence. Words may have different subjectivity levels when used in different forms such as 'noun' or 'verb', hence POS tagging was applied to all the Spanish training sentences.

After POS tagging, subjectivity clue words were identified. The subjectivity clues dataset [40] contains a list of more than 8,000 clues identified both manually as well as automatically, using both labeled and unlabeled data. As this dataset includes only English words, the corresponding Spanish list was created using the Bing translator API. For each clue word, the number of occurrences in the dataset was computed. Eventually, the top 50 clue words with highest occurrences in the subjective sentences were considered [41]. Each sentence was then transformed to a binary feature vector of length 50, where the presence of a clue word is denoted as '1' and an absence is denoted as '0'.

The resulting binary matrix 'clue words versus sentences' has been processed as a time series. Thus, the sentences have been used as input for a Gaussian

[1] http://www.ranks.nl/stopwords/

Bayesian network. The ML probabilities of each word, given up-to three parent words and up-to two time points delay, was also computed. Such sub-structures are referred to as network motifs. The top 20% of motifs with the highest ML were exploited to select the pre-training sentences for the deep CNN, done by simply selecting the sentences containing these motifs.

The deep CNN was employed to extract features in the form of 3-grams and 4-grams in each language separately. It was configured as follows: three hidden layers with 100 neurons each, kernels of size 3, 4 and 5, respectively, and one logistic layer with 300 neurons. The output layer included two neurons for each class of sentiments. The 300 feature outputs of the deep CNN (from both languages) were used to train BNELM with an additional fuzzy recurrent layer of 10 hidden neurons and up to 2 time point delays.

5.2. Multi-Perspective Question Answering Dataset

The MPQA corpus is a collection of 504 sentences manually annotated for subjectivity in Spanish. The annotation resulted in 273 subjective and 231 objective sentences [42]; the corpus includes sentences of an uncertain nature that were assigned to a definite class after assessment by multiple annotators. MPQA is a popular benchmark, which can be used to evaluate the robustness of the proposed framework when a small training set is involved. The sentences were eventually machine translated into English to obtain the final dataset, which after preprocessing lay in a 20-dimensional space. A 5-fold cross validation has been used to estimate the accuracy of the trained classifier when applied to new sentences, i.e., sentences not included in the training set. The performance of the three predictors (BNELM, ELM, and SBELM) was assessed by using the average value of the accuracy computed over 10 runs, i.e., 10 different randomizations of the

mapping layer.

Figures 4 and 5 provide the outcomes of the experiments. Figure 4 assesses the performances of ELM, SBELM, and BNELM for four different sizes of the mapping layer: $L = \{25; 50; 100; 600\}$, note that SBELM has not been tested for a random hidden layer of 600 neurons, due to the expensive training phase. All the experiments were run using a fixed value for the number of iterations, $nItr$, in the MCMC procedure. In Figure 4, the x axis gives L, while the y axis gives the classification accuracy (expressed as the percentage over the size of the test set). The bar graph compares the performance of the standard ELM (white bar) with the performance of the SBELM (grey bar) and the performance of BNELM (black bar). On an overall basis, the graphs clearly show that BNELM can improve over standard ELM and SBELM in terms of classification performance. Figure 5 analyses the performance of the BNELM for different values of the parameter $nItr$. Here, the x axis gives number of iterations $nItr$, while the y axis gives the classification accuracy. The experiments refer to a configuration with $L = 50$; the number of iterations were allowed to take the following values: $nItr = \{25; 50; 100\}$. The graph shows that the accuracy of the classifier reaches a maximum when the number of iterations is 25. Nonetheless, it is interesting to note that a good accuracy can be obtained with all the proposed values of $nItr$.

5.3. Multimodal Opinion Utterances Dataset

MOUD consists of 498 short video fragments containing one sentence each. The items are manually tagged for sentiment polarity, which can be positive, negative, or neutral. The videos are in MP4 format with a resolution of 360 $\times$ 480 pixels; the duration of the clips is about 5 seconds on average. About 80$\%$ of the clips involve female speakers. The transcripts of the videos were used as a

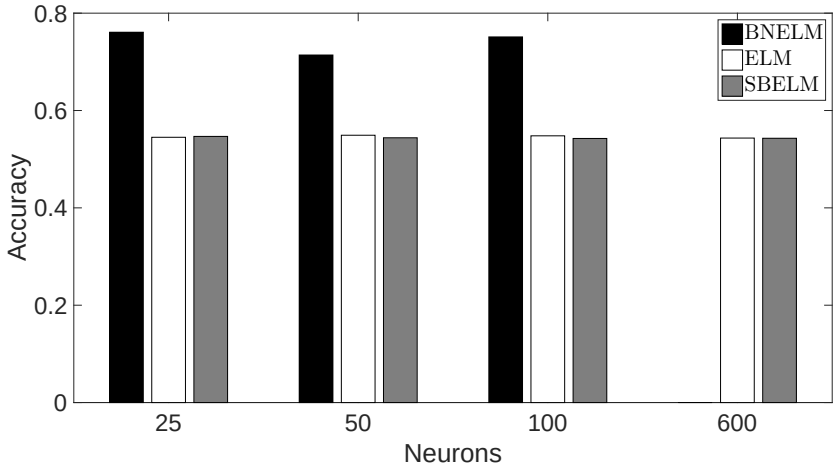

Figure 4: Accuracy of experiments involving MPQA dataset; Average accuracy of the three different classifiers;

dataset. After pre-processing, the patterns lay in a 20-dimensional space. Similar to the MPQA experiment, a 5-fold cross validation has been used to estimate the performance of the trained classifier. Figures 6 and 7 provide the outcomes of the experiments by adopting the format of Fig. 4 and Fig. 5, respectively. The experiments showed in Fig. 6 involved the following settings: $L = \{25; 50; 100; 600\}$; $nItr = 50$. The settings of the experiments showed in Fig. 7 are: $L = 50$; $nItr = \{25; 50; 100\}$. In general, these results confirm the tendency identified with the MPQA dataset. This in turn proves that BNELM can indeed serve as an effective tool to deal with sentiment analysis.

5.4. Sentiment Classification on the TASS 2015 Corpus

In order to evaluate the model on a noisy dataset, we consider the four-class TASS corpora of Spanish tweets [32]. Each tweet belongs to one of the four

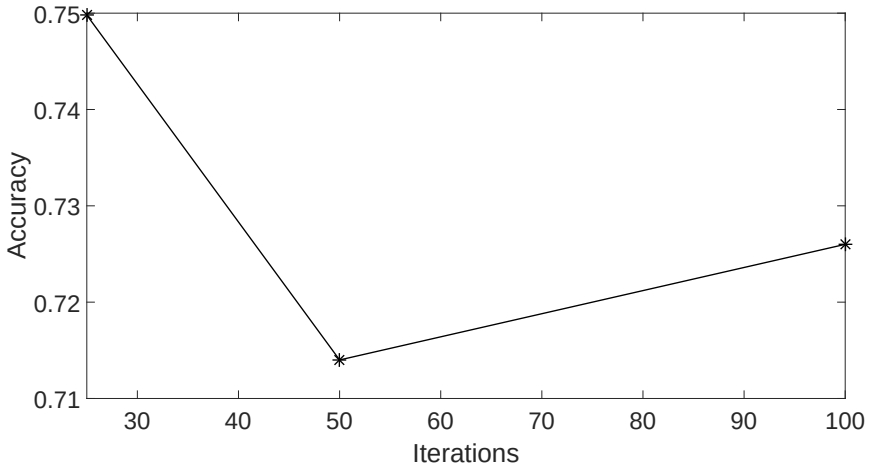

Figure 5: Accuracy of experiments involving MPQA dataset; Average accuracy of the gmm-ELM for different values of the parameters $nItr$;

categories: *positive*, *neutral*, *negative*, or *without opinion*. In this paper, we are using the training set of 7219 tweets and the test set of 1000 tweets.

Table 1 shows the accuracy of different models in classifying sentences in a document as Positive (subjective), Negative (subjective), Neutral (objective) or None in TASS test dataset. A simple CNN model for sentences learns features of two or three words using sliding window kernels. We also compare our approach with different models evaluated at the TASS workshop (see the overview paper [32] for a detailed description of all approaches).

In LYS [43], the authors used classical logistic regression with linguistic features. Their approach was limited as they heavily relied on polarity lexicons that are not available in Spanish; in this paper, instead, we use ELMs to automatically learn features from both English and Spanish. Furthermore, we use Bayesian heuristics to determine parameters for a large set of noisy tweets and, hence, we

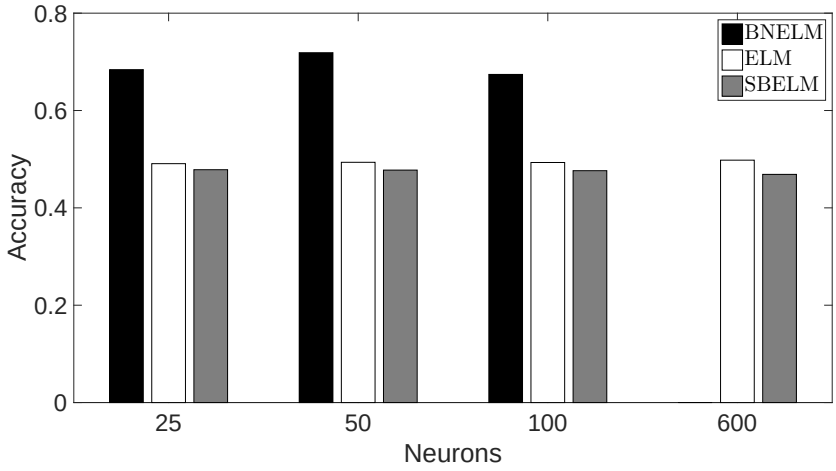

Figure 6: Accuracy of experiments involving MOUD; average accuracy of the three different classifiers;

are able to outperform the baselines by over 15% in accuracy.

Table 1: Accuracy of different models in classifying sentences in a document as Positive (subjective), Negative (subjective), Neutral (objective) or None in TASS dataset.

	CNN [44]	LYS [43]	LIF [32]	BNELM
TASS 2015	0.66	0.637	0.692	**0.89**

6. Conclusion

Subjectivity detection represents a challenging task for sentiment analysis tools as these are usually optimized for the binary classification task of polarity detection (positive versus negative). In this paper, we introduce a novel architecture for filtering out neutral content in a time- and resource-effective manner. The proposed BNELM model augments the standard RNN structure with the purpose

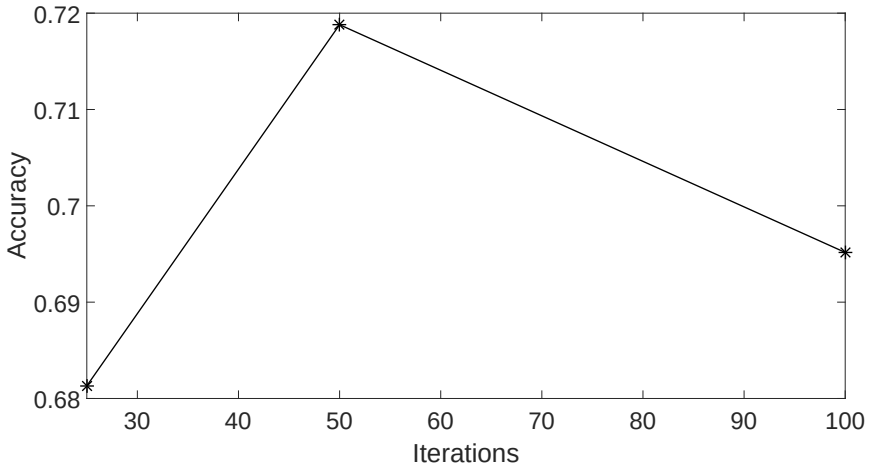

Figure 7: Accuracy of experiments involving MOUD; Average accuracy of the gmm-ELM for different values of the parameters $nItr$;

of generating a predictor that can take advantage of the fruitful properties of ELM and Bayesian networks.

The BNELM architecture has two main advantages with respect to SBELM: firstly, training BNELM does not involve the computation of second-order derivatives; secondly, BNELM tackles the optimization problem by exploiting an heuristic procedure that relies on MCMC. As a result, the optimization process converges to the global maximum at the equilibrium, resulting in a higher accuracy of the predictor. Moreover, the use of Bayesian networks inherently leads to a model that is able to determine the optimal number of hidden neurons, thus pruning redundant neurons.

The final framework for subjectivity detection relies on the effective integration of CNNs and BNELM, where the former learns significant features from the training set and, hence, feeds them to the latter. Experimental results proved that

this framework has the ability to learn a dictionary of features that are portable to new languages and domains. Such a characteristic becomes particularly important when it is difficult and expensive to generate training data in resource-scarce languages.

References

[1] E. Cambria, D. Das, S. Bandyopadhyay, A. Feraco, A Practical Guide to Sentiment Analysis, Springer, Cham, Switzerland, 2017.

[2] S. Poria, E. Cambria, R. Bajpai, A. Hussain, A review of affective computing: From unimodal analysis to multimodal fusion, Information Fusion 37 (2017) 98–125.

[3] S. Poria, I. Chaturvedi, E. Cambria, A. Hussain, Convolutional MKL based multimodal emotion recognition and sentiment analysis, in: ICDM, Barcelona, 2016, pp. 439–448.

[4] S. Poria, E. Cambria, A. Gelbukh, Aspect extraction for opinion mining with a deep convolutional neural network, Knowledge-Based Systems 108 (2016) 42–49.

[5] Y. Ma, E. Cambria, S. Gao, Label embedding for zero-shot fine-grained named entity typing, in: COLING, Osaka, 2016, pp. 171–180.

[6] Y. Xia, E. Cambria, A. Hussain, H. Zhao, Word polarity disambiguation using bayesian model and opinion-level features, Cognitive Computation 7 (3) (2015) 369–380.

[7] N. Majumder, S. Poria, A. Gelbukh, E. Cambria, Deep learning based document modeling for personality detection from text, IEEE Intelligent Systems 32 (2) (2017) 74–79.

[8] S. Poria, E. Cambria, D. Hazarika, P. Vij, A deeper look into sarcastic tweets using deep convolutional neural networks, in: COLING, 2016, pp. 1601–1612.

[9] S. Vázquez, Ó. Muñoz-García, I. Campanella, M. Poch, B. Fisas, N. Bel, G. Andreu, A classification of user-generated content into consumer decision journey stages, Neural Networks 58 (0) (2014) 68 – 81.

[10] G. Murray, G. Carenini, Subjectivity detection in spoken and written conversations, Natural Language Engineering 17 (2011) 397–418.

[11] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, B. Qin, Learning sentiment-specificword embedding for twitter sentiment classification (2014).

[12] M. Bonzanini, M. Martinez-Alvarez, T. Roelleke, Opinion summarisation through sentence extraction: An investigation with movie reviews, in: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12, 2012, pp. 1121–1122.

[13] M. V. de Kauter, D. Breesch, V. Hoste, Fine-grained analysis of explicit and implicit sentiment in financial news articles, Expert Systems with Applications 42 (11) (2015) 4999 – 5010.

[14] S. Rill, D. Reinel, J. Scheidt, R. V. Zicari, Politwi: Early detection of emerging political topics on twitter and the impact on concept-level sentiment analysis, Knowledge-Based Systems 69 (0) (2014) 24 – 33.

[15] P. Biyani, S. Bhatia, C. Caragea, P. Mitra, Using non-lexical features for identifying factual and opinionative threads in online forums, Knowledge-Based Systems 69 (0) (2014) 170 – 178.

[16] I. Chaturvedi, E. Cambria, D. Vilares, Lyapunov filtering of objectivity for Spanish sentiment model, in: IJCNN, Vancouver, 2016, pp. 4474–4481.

[17] T. Wilson, Fine-grained subjectivity and sentiment analysis: Recognizing the intensity, polarity, and attitudes of private states, Ph.D. thesis, Intelligent Systems Program, University of Pittsburgh (2007).

[18] S. Krishnamoorthy, Linguistic features for review helpfulness prediction, Expert Systems with Applications 42 (7) (2015) 3751 – 3759.

[19] E. Soria-Olivas, J. Gomez-Sanchis, J. D. Martin, J. Vila-Frances, M. Martinez, J. R. Magdalena, A. J. Serrano, Belm: Bayesian extreme learning machine, IEEE Transactions on Neural Networks 22 (3) (2011) 505–509. doi:10.1109/TNN.2010.2103956.

[20] G.-B. Huang, E. Cambria, K.-A. Toh, B. Widrow, Z. Xu, New trends of learning in computational intelligence, IEEE Computational Intelligence Magazine 10 (2) (2015) 16–17.

[21] P. Gastaldo, R. Zunino, E. Cambria, S. Decherchi, Combining elm with random projections, IEEE intelligent systems 28 (6) (2013) 46–48.

[22] F. Bisio, S. Decherchi, P. Gastaldo, R. Zunino, Inductive bias for semi-supervised extreme learning machine, Neurocomputing 174 (2016) 154–167.

[23] P. Gastaldo, F. Bisio, S. Decherchi, R. Zunino, Sim-elm: Connecting the elm model with similarity-function learning, Neural Networks 74 (2016) 22–34.

[24] P. Gastaldo, F. Bisio, C. Gianoglio, E. Ragusa, R. Zunino, Learning with similarity functions: A novel design for the extreme learning machine, Neurocomputing.

[25] W. Deng, Q. Zheng, L. Chen, Regularized extreme learning machine, in: Computational Intelligence and Data Mining, 2009. CIDM '09. IEEE Symposium on, 2009, pp. 389–395.

[26] J. Luo, C. M. Vong, P. K. Wong, Sparse bayesian extreme learning machine for multi-classification, IEEE Transactions on Neural Networks and Learning Systems 25 (4) (2014) 836–843.

[27] I. Chaturvedi, E. Cambria, F. Zhu, L. Qiua, W. K. Ng, Multilingual subjectivity detection using deep multiple kernel learning, Proceedings of KDD, Sydney.

[28] S. Konishi, T. Ando, S. Imoto, Bayesian information criteria and smoothing parameter selection in radial basis function networks, Biometrika 91 (1) (2004) 27–43. doi:10.1093/biomet/91.1.27.

[29] R. Mihalcea, C. Banea, J. Wiebe, Learning multilingual subjective language via cross-lingual projections, in: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Association for Computational Linguistics, 2007, pp. 976–983.

[30] J. Wiebe, E. Riloff, Creating subjective and objective sentence classifiers

from unannotated texts, in: Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing, 2005, pp. 486–497.

[31] L.-P. Morency, R. Mihalcea, P. Doshi, Towards multimodal sentiment analysis: Harvesting opinions from the web, in: Proceedings of the 13th international conference on multimodal interfaces, ACM, 2011, pp. 169–176.

[32] J. Villena-Román, J. García-Morera, M. García-Cumbreras, E. Martínez-Cámara, M. Martín-Valdivia, L. Ureña-López, Overview of tass 2015, in: Proceedings of TASS 2015: Workshop on Sentiment Analysis at SEPLN, 2015.

[33] A. Prinzie, D. V. den Poel, Dynamic bayesian networks for acquisition pattern analysis: A financial-services cross-sell application, in: New Frontiers in Applied Data Mining, PAKDD 2008 International Workshops, Osaka, Japan, May 20-23, 2008. Revised Selected Papers, 2008, pp. 123–133.

[34] E. Riloff, J. Wiebe, Learning extraction patterns for subjective expressions, in: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, 2003, pp. 105–112.

[35] E. Castillo, J. M. Gutierrez, A. S. Hadi, Expert Systems and Probabiistic Network Models, 1st Edition, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.

[36] E. Castillo, U. Kjærulff, Sensitivity analysis in gaussian bayesian networks using a symbolic-numerical technique, Reliability Engineering and System Safety 79 (2) (2003) 139 – 148.

[37] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1) (2006) 489–501.

[38] P. Baranyi, K.-F. Lei, Y. Yam, Complexity reduction of singleton based neuro-fuzzy algorithm, in: Systems, Man, and Cybernetics, 2000 IEEE International Conference on, Vol. 4, 2000, pp. 2503–2508 vol.4.

[39] V. Pérez-Rosas, R. Mihalcea, L.-P. Morency, Utterance-level multimodal sentiment analysis., in: ACL (1), 2013, pp. 973–982.

[40] E. Riloff, J. Wiebe, Learning extraction patterns for subjective expressions, in: Proceedings of the 2003 conference on Empirical methods in natural language processing, Association for Computational Linguistics, 2003, pp. 105–112.

[41] E. Cambria, S. Poria, R. Bajpai, B. Schuller, SenticNet 4: A semantic resource for sentiment analysis based on conceptual primitives, in: COLING, 2016, pp. 2666–2677.

[42] R. Mihalcea, C. Banea, J. Wiebe, Learning multilingual subjective language via cross-lingual projections, in: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Association for Computational Linguistics, 2007, pp. 976–983.

[43] D. Vilares, Y. Doval, M. A. Alonso, C. Gómez-Rodríguez, Lys at tass 2014: A prototype for extracting and analysing aspects from spanish tweets, Proceedings of the TASS workshop at SEPLN.

[44] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences, in: Proceedings of the 52nd Annual Meeting

of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, 2014, pp. 655–665.