Bayesian Deep Convolutional Belief Networks for Subjectivity Detection

Iti Chaturvedi, Erik Cambria
School of Computer Science and Engineering
Nanyang Technological University
Singapore
{iti,cambria}@ntu.edu.sg

Soujanya Poria, Rajiv Bajpai
Temasek Laboratories
Nanyang Technological University
Singapore
{sporia,rbajpai}@ntu.edu.sg

***Abstract*—Subjectivity detection aims to distinguish natural language as either opinionated (positive or negative) or neutral. In word vector based convolutional neural network models, a word meaning is simply a signal that helps to classify larger entities such as a document. Previous works do not usually consider prior distribution when using sliding windows to learn word embeddings and, hence, they are unable to capture higher-order and long-range features in text. In this paper, we employ dynamic Gaussian Bayesian networks to learn significant network motifs of words and concepts. These motifs are used to pre-train the convolutional neural network and capture the dynamics of discourse across several sentences.**

***Index Terms*—Multiple Kernel Learning, Deep Convolutional Neural Networks, Sentiment Analysis.**

I. Introduction

Subjectivity detection is an important subtask of sentiment analysis [1] that can prevent a sentiment classifier from considering irrelevant or potentially misleading text in online social platforms such as Twitter and Facebook. Subjective extraction can reduce the amount of review data to only 60% and still produce the same polarity results as full text classification [2]. This allows analysts in government, commercial and political domains who need to determine the response of people to different crisis events [2], [3], [4].

Similarly, online reviews need to be summarized in a manner that allows comparison of opinions, so that a user can clearly see the advantages and weaknesses of each product merely with a single glance, both in unimodal [5], [6] and multimodal [7], [8] contexts. Further, we can do in-depth opinion assessment, such as finding reasons or aspects [9] in opinion-bearing texts. For example, 'Poor acting', which makes the film 'awful'. Several works have explored sentiment composition through careful engineering of features or polarity shifting rules on syntactic structures. However, sentiment accuracies for classifying a sentence as positive/negative/neutral has not exceeded 60%.

Early attempts used general subjectivity clues to generate training data from un-annotated text [10]. Next, bag-of-words (BOW) classifiers were introduced that represent a document as a multi set of its words disregarding grammar and word order. These methods did not work well on short tweets. Co-occurrence matrices also were unable to capture difference in antonyms such as 'good/bad' that have similar distributions.

Subjectivity detection hence progressed from syntactic to semantic methods in [10], where the authors used extraction pattern to represent subjective expressions. For example, the pattern 'hijacking' of $< x >$, looks for the noun 'hijacking' and the object of the preposition $< x >$. Extracted features are used to train machine-learning classifiers such as SVM [11] and ELM [12]. Subjectivity detection is also useful for constructing and maintaining sentiment lexicons, as objective words or concepts need to be omitted from them [12].

Since, subjective sentences tend to be longer than neutral sentences, recursive neural networks were proposed where the sentiment class at each node in the parse tree was captured using matrix multiplication of parent nodes [13], [14]. However, the number of possible parent composition functions is exponential, hence in [15] recursive neural tensor network was introduced that use a single tensor composition function to define multiple bilinear dependencies between words.

In [16], the authors used logistic regression predictor that defines a hyperplane in the word vector space where a word vectors positive sentiment probability depends on where it lies with respect to this hyperplane. However, it was found that while incorporating words that are more subjective can generally yield better results, the performance gain by employing extra neutral words is less significant [17]. Another class of probabilistic models called Latent Dirichlet Allocation assumes each document is a mixture of latent topics.

Lastly, sentence-level subjectivity detection was integrated into document-level sentiment detection using graphs where each node is a sentence. The contextual constraints between sentences in a graph led to significant improvement in polarity classification [18]. Similarly, in [19] the authors take advantage of the sequence encoding method for trees and treat them as sequence kernels for sentences.

Templates are not suitable for semantic role labeling, because relevant context might be very far away. Hence, deep neural networks have become popular to process text. In word2vec, for example, a word's meaning is simply a signal that helps to classify larger entities such as documents. Every word is mapped to a unique vector, represented by a column in a weight matrix. The concatenation or sum of the vectors is then used as features for prediction of the next word in a sentence [20].

Related words appear next to each other in a d dimensional vector space. Vectorizing them allows us to measure their similarities and cluster them. For semantic role labeling, we need to know the relative position of verbs, hence the features can include prefix, suffix, distance from verbs in the sentence etc. However, each feature has a corresponding vector representation in d dimensional space learned from the training data.

Recently, recurrent convolutional neural networks are being used for subjectivity detection [21]. These show high accuracy on certain datasets such as Twitter we are also concerned with a specific sentence within the context of the previous discussion, the order of the sentences preceding the one at hand results in a sequence of sentences also known as a time series of sentences [21]. However, their model suffers from over-fitting, hence in this paper we consider deep convolutional neural networks, where temporal information is modeled via dynamic Gaussian Bayesian networks.

II. Related Work and Contributions

Convolutional neural networks (CNN) are sensitive to the order of words in a sentence and do not depend on external language specific features such as dependency or constituency parse trees [13]. Here narrow or wide convolution is achieved by applying filters such as pattern templates across the input sequence of words. A convolution layer in the network is obtained by convolving a matrix of weights with the matrix of activations at the layer below and the weights are trained using back propagation [22]. Each convolution layer is interleaved with a max-pool layer that eliminates redundant values. Simple pooling will forget the order in which the features occur, hence k-max pooling is used where the order of k highest values corresponds to the original order in the sentence. However, this method does not work well at the margins that are considered fewer times in the computation of the convolution. In addition, higher-order and long range features cannot be easily incorporated into a CNN model.

In order to model the sentiment in text in addition to the syntactic context, in [4], the authors propose sentiment specific word embedding that predicts the sentiment distribution of input text based on n-gram however it leaves out the context of words. The output neuron is a two dimensional vector for the syntactic and semantic scores computed using hinge loss. However, it is convenient to engineer features for Twitter sentiment classification instead of learning the word vectors from scratch. Simple additive or multiplicative models that do not take into account the word order or structure are found to outperform structured models at certain phrase similarity.

[21] used recurrent CNN to classify a sequence of sentences as a whole for use in dialogue tracking and question answering systems. They propose a hierarchical CNN with kernels of increasing sizes to preserve the order of words in a sentence. However, again their discourse model does not use any prior features and has limited memory, as the input is the current sentence and the label of the previous sentence.

To locate sensible responses for a tweet a heterogeneous model that leverages on the intrinsic hierarchy in the language was proposed in [23]. Here, a bilinear kernel metric for matching two tweets was determined using an interaction matrix between the words. The same kernel was extended to a deep model, where instead of patches of images, the patches capture the text segments of rich inherent structure. To determine the word vector representation in terms of a tuple of previous few words the Log-Bilinear model of context matrices was used in [24]. Finally, to model the flow of sentiment in a document [25] consider a sequence model that predicts the sequence of sentiments based on a sequence of sentences. The sequence of sentiments is used to design new author dependent features in the document.

In this paper, we propose use of Gaussian Bayesian networks to extract higher-order features from time series of sentences in a document. These features in the form of network motifs are used to screen sentences in the training data and pre-train the CNN. The significance and contributions of the research work presented in this paper can be summarized as follows:

- We introduce a new Bayesian Deep Convolutional Neural Network (BCDBN) capable of model higher-order features across consecutive sentences in a document. From our knowledge, no previous work has considered Gaussian networks to learn features in a CNN.
- Network motifs made of top subjectivity clue words in the training dataset are extracted using BOW model. These motifs are then used to pre-train the weights of CNN for sentence classification.
- Layers of kernels of increasing sizes are used to preserve the order of words in a sentence and combine small delays to model long delays with few training samples. In this way, filters in higher layers can capture syntactic relations between phrases far apart in the input sentence.

To verify the effectiveness of BCDBN in capturing dependencies in high-dimensional data, we consider the MPQA corpus [11], which is a collection of 535 English-language news articles from a variety of news sources manually annotated for subjectivity. From the total of 9,700 sentences in this corpus, 55% of the sentences are labeled as subjective while the rest are objective. We also consider the movie review benchmark dataset [18], that contains 5000 subjective movie review snippets and another 5000 objective sentences from plot summaries available from the Internet Movies Database. The classification accuracy obtained using the proposed BCDBN is shown to outperform the baseline by over $5-10\%$ on both real datasets.

The rest of the paper is organized as follows: Section 3 provides the preliminary concepts necessary to comprehend the proposed BCDBN algorithm of the present work. In section 4, we introduce the proposed BCDBN for sentences and describe the algorithm for learning the weights of the framework. Lastly, in section IV, we validate our method on real world benchmark dataset on subjectivity detection.

III. PRELIMINARIES

In this section, we briefly review the theoretical concepts necessary to comprehend the present work. We begin with a description of maximum likelihood estimation of edges in dynamic Gaussian Bayesian networks where each node is a word in a sentence. Next, we show that weights in the CNN can be learned by minimizing a global error function that corresponds to an exponential distribution over a linear combination of input sequence of word features.

Notations : Consider a Gaussian network (GN) with time delays which comprises a set of N nodes and observations gathered over T instances for all the nodes. Nodes can take real values from a multivariate distribution determined by the parent set. Let the dataset of samples be $X = \{x_i(t)\}_{\mathrm{N}\times\mathrm{T}}$, where $x_i(t)$ represents the sample value of the i^{th} random variable in instance t. Lastly, let $\boldsymbol{a}_i$ be the set of parent variables regulating variable i.

A. Gaussian Bayesian Networks

In tasks where one is concerned with a specific sentence within the context of the previous discourse, capturing the order of the sequences preceding the one at hand may be particularly crucial. We take as given a sequence of sentences $s(1), s(2), \ldots, s(T)$, each in turn being a sequence of words so that $s(t) = (x_1(t), x_2(t), \ldots, x_{\mathrm{L}}(t))$, where L is the length of sentence $s(t)$. Thus, the probability of a word $p(x_i(t))$ follows the distribution :

$$p(x_i(t)) = P(x_i(t)|(x_1(t), x_2(t), \ldots, x_{i-1}(t)), (s(1), s(2), \ldots, s(t-1)) \quad (1)$$

A Bayesian network is a graphical model that represents a joint multivariate probability distribution for a set of random variables [26]. It is a directed acyclic graph S with a set of parameters $\boldsymbol{\theta}$ that represents the strengths of connections by conditional probabilities.

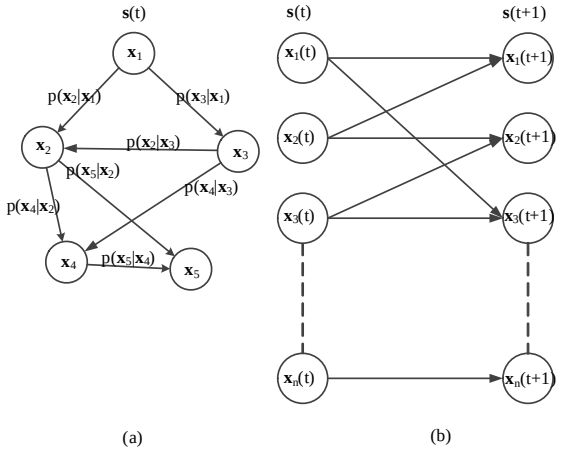

Fig. 1. State space of different Bayesian models

The BN decomposes the likelihood of node expressions into a product of conditional probabilities by assuming independence of non-descendant nodes, given their parents.

$$p(X|S, \boldsymbol{\theta}) = \prod_{i=1}^{\mathrm{N}} p(\mathbf{x}_i|\boldsymbol{a}_i, \theta_{i,\boldsymbol{a}_i}), \quad (2)$$

where $p(\mathbf{x}_i|\boldsymbol{a}_i, \theta_{i,\boldsymbol{a}_i})$ denotes the conditional probability of node expression $\mathbf{x}_i$ given its parent node expressions $\boldsymbol{a}_i$, and $\theta_{i,\boldsymbol{a}_i}$ denotes the maximum likelihood(ML) estimate of the conditional probabilities.

Figure 1 (a) illustrates the state space of a Gaussian Bayesian network (GBN) at time instant t where each node $x_i(t)$ is a word in the sentence $s(t)$. The connections represent causal dependencies over one or more time instants. The observed state vector of variable i is denoted as $\mathbf{x}_i$ and the conditional probability of variable i given variable j is $p(\mathbf{x}_i|\mathbf{x}_j)$. The optimal Gaussian network S^* is obtained by maximizing the posterior probability of S given the data X. From Bayes theorem, the optimal Gaussian network S^* is given by:

$$S^* = \arg\max_S p(S|X) = \arg\max_S p(S)p(X|S), \quad (3)$$

where $p(S)$ is the probability of the Gaussian network and $p(X|S)$ is the likelihood of the expression data given the Gaussian network.

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

$$p(X|S) = \int p(X|S, \boldsymbol{\theta})p(\boldsymbol{\theta}|S)d\theta, \quad (4)$$

To find the likelihood in (4), and to obtain the optimal Gaussian network as in (3), Gaussian BN assumes that the nodes are multivariate Gaussian. That is, expression of node i can be described with mean μ_i and covariance matrix Σ_i of size $\mathrm{N} \times \mathrm{N}$. The joint probability of the network can be the product of a set of conditional probability distributions given by:

$$p(\mathbf{x}_i|\boldsymbol{a}_i) = \theta_{i,\boldsymbol{a}_i} \sim \mathcal{N}\left(\mu_i + \sum_{j\in\boldsymbol{a}_i} (\mathbf{x}_j - \mu_j)\beta,\ \Sigma_i^{'}\right), \quad (5)$$

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 $\mathbf{x}_i$ given its parent set $\boldsymbol{a}_i$, $\Sigma_{i,\boldsymbol{a}_i}$ is the covariance between observations of $\mathbf{x}_i$ and the variables in $\boldsymbol{a}_i$, and $\Sigma_{\boldsymbol{a}_i}$ is the covariance matrix of $\boldsymbol{a}_i$. The acyclic condition of BN does not allow feedback among nodes, and feedback is an essential characteristic of real world GN.

Therefore, dynamic Bayesian networks have recently become popular in building GN with time delays mainly due to their ability to model causal interactions as well as feedback regulations [27]. A first-order dynamic BN is defined by a transition network of interactions between a pair of Gaussian networks connecting nodes at time instants τ and $\tau + 1$. In time instant $\tau + 1$, the parents of nodes are those specified in the time instant τ.

Similarly, the Gaussian network of a R-order dynamic system is represented by a Gaussian network comprising $(\mathrm{R}+1)$ consecutive time points and N nodes, or a graph of $(\mathrm{R}+1)\times\mathrm{N}$ nodes. In practice, the sentence data is transformed to a BOW model where each sentence is a vector of frequencies for each word in the vocabulary. Figure 1 (b) illustrates the state space of a first-order Dynamic GBN models transition networks among words in sentences $s(t)$ and $s(t+1)$ in consecutive time points, the lines correspond to first-order edges among the words learned using BOW. Hence, a sequence of sentences results in a time series of word frequencies. It can be seen that such a discourse model produces compelling discourse vector representations that are sensitive to the structure of the discourse and promise to capture subtle aspects of discourse comprehension, especially when coupled to further semantic data and unsupervised pre-training.

B. Convolutional Neural Networks

The idea behind convolution is to take the dot product of a vector of k weights w_k also known as kernel vector with each k-gram in the sentence $s(t)$ to obtain another sequence of features $c(t) = (c_1(t), c_2(t), \ldots, c_{\mathrm{L}}(t))$.

$$c_j = {w_k}^T . \mathbf{x}_{i:i+k-1} \tag{6}$$

We then apply a max pooling operation over the feature map and take the maximum value $\hat{c}(t) = \max\{\mathbf{c}(t)\}$ as the feature corresponding to this particular kernel vector. Similarly, varying kernel vectors and window sizes are used to obtain multiple features [13].

For each word $x_i(t)$ in the vocabulary, an d dimensional vector representation is given in a look up table that is learned from the data [20]. The vector representation of a sentence is hence a concatenation of vectors for individual words. Similarly, we can have look up tables for other features. One might want to provide features other than words if these features are suspected to be helpful. Now, the convolution kernels are applied to word vectors instead of individual words.

We use these features to train higher layers of the CNN that can represent bigger groups of words in sentences. We denote the feature learned at hidden neuron h in layer l as F_h^l. Multiple features may be learned in parallel in the same CNN layer. The features learned in each layer are used to train the next layer

$$F^l = \sum_{h=1}^{n_h} w_k^h * F^{l-1} \tag{7}$$

where * indicates convolution and w_k is a weight kernel for hidden neuron h and n_h is the total number of hidden neurons. Training a CNN becomes difficult as the number of layers increases, as the Hessian matrix of second-order derivatives often does not exist. Recently, deep learning has been used to improve the scalability of a model that has inherent parallel computation. This is because hierarchies of modules can provide a compact representation in the form of input-output pairs. Each layer tries to minimize the error between the original state of the input nodes and the state of the input nodes predicted by the hidden neurons.

This results in a downward coupling between modules. The more abstract representation at the output of a higher layer module is combined with the less abstract representation at the internal nodes from the module in the layer below. In the next section, we describe deep CNN that can have arbitrary number of layers.

C. Convolution Deep Belief Network

A deep belief network (DBN) is a type of deep neural network that can be viewed as a composite of simple, unsupervised models such as restricted Boltzmann machines (RBMs) where each RBMs hidden layer serves as the visible layer for the next RBM. RBM is a bipartite graph comprising two layers of neurons: a visible and a hidden layer; it is restricted such that the connections among neurons in the same layer are not allowed.

To compute the weights W of an RBM, we assume that the probability distribution over the input vector $\mathbf{x}$ is given as:

$$p(\mathbf{x}|W) = \frac{1}{Z(W)} \exp^{-\mathrm{E}(\mathbf{x};W)} \tag{8}$$

where $Z(W) = \sum_{\mathbf{x}} \exp^{-\mathrm{E}}(\mathbf{x};W)$ is a normalisation constant. Computing the maximum likelihood is difficult as it involves solving the normalisation constant, which is a sum of an exponential number of terms. The standard approach is to approximate the average over the distribution with an average over a sample from $p(\mathbf{x}|W)$, obtained by Markov chain Monte Carlo until convergence.

To train such a multi-layer system, we must compute the gradient of the total energy function E with respect to weights in all the layers. To learn these weights and maximize the global energy function, the approximate maximum likelihood contrastive divergence (CD) approach can be used. This method employs each training sample to initialize the visible layer. Next, it uses the Gibbs sampling algorithm to update the hidden layer and then reconstruct the visible layer consecutively, until convergence [28]. As an example, here we use a logistic regression model to learn the binary hidden neurons and each visible unit is assumed to be a sample from a normal distribution [29]. The continuous state $\hat{h}_j$ of the hidden neuron j, with bias b_j, is a weighted sum over all continuous visible nodes $\boldsymbol{v}$ and is given by:

$$\hat{h}_j = b_j + \sum_i v_i w_{ij}, \tag{9}$$

where w_{ij} is the connection weight to hidden neuron j from visible node v_i. The binary state h_j of the hidden neuron can be defined by a sigmoid activation function:

$$h_j = \frac{1}{1 + e^{-\hat{h}_j}}. \tag{10}$$

Similarly, in the next iteration, the binary state of each visible node is reconstructed and labeled as $\boldsymbol{v}_{recon}$. Here, we determine the value to the visible node i, with bias c_i, as a random sample from the normal distribution where the mean

is a weighted sum over all binary hidden neurons and is given by:

$$\hat{v}_i = c_i + \sum_j h_i w_{ij}, \tag{11}$$

where w_{ij} is the connection weight to hidden neuron j from visible node v_i. The continuous state v_i is a random sample from $\mathcal{N}(\hat{v}_i, \sigma)$, where σ is the variance of all visible nodes. Lastly, the weights are updated as the difference between the original and reconstructed visible layer using:

$$\triangle w_{ij} = \alpha(< v_i h_j >_{data} - < v_i h_j >_{recon}), \tag{12}$$

where α is the learning rate and $< v_i h_j >$ is the expected frequency with which visible unit i and hidden unit j are active together when the visible vectors are sampled from the training set and the hidden units are determined by (9). Finally, the energy of a DNN can be determined in the final layer using $\mathrm{E} = -\sum_{i,j} v_i h_j w_{ij}$.

To extend the deep belief networks to convolution deep belief network (CDBN) we simply partition the hidden layer into Z groups. Each of the Z groups is associated with a $k \times d$ filter where k is the width of the kernel and d is the number of dimensions in the word vector. Let us assume that the input layer has dimension $\mathrm{L} \times d$ where L is the length of the sentence. Then the convolution operation given by (6) will result in a hidden layer of Z groups each of dimension $(\mathrm{L} - k + 1) \times (d - d + 1)$. These learned kernel weights are shared among all hidden units in a particular group. The energy function is now a sum over the energy of individual blocks given by:

$$\mathrm{E} = -\sum_{z=1}^{\mathrm{Z}} \sum_{i,j}^{\mathrm{L}-k+1,1} \sum_{r,s}^{k,d} v_{i+r-1,j+s-1} h_{ij}^z w_{rs}^k \tag{13}$$

The CNN sentence model preserve the order of words by adopting convolution kernels of gradually increasing sizes that span an increasing number of words and ultimately the entire sentence [21]. However, several word dependencies may occur across sentences hence, in this paper we propose a Bayesian CNN model that uses dynamic Bayesian networks to model a sequence of sentences.

IV. Deep Bayesian CNN for Sentences

In this paper, we propose to integrate a higher-order GBN for sentences into the first layer of the CNN. The GBN layer of connections β is learned using maximum likelihood approach on the BOW model of the training data. The input sequence of sentences $s(t : t-2)$ are parsed through this layer prior to training the CNN. Only sentences or groups of sentences containing high ML motifs are then used to train the CNN. Hence, motifs are convolved with the input sentences to generate a new set of sentences for pre-training.

$$F^0 = \sum_{h=1}^{M} \beta^h * \boldsymbol{s} \tag{14}$$

where M is the number of high ML motifs and $\boldsymbol{s}$ is the training set of sentences in a particular class.

Fig. 2 illustrates the state space of Bayesian CNN where the input layer is pre-trained using a dynamic GBN with up-to two time point delays shown for three sentences in a review on iPhone. The dashed lines correspond to second-order edges among the words learned using BOW. Each hidden layer does convolution followed by pooling across the length of the sentence. To preserve the order of words we adopt kernels of increasing sizes.

Since, the number of possible words in the vocabulary is very large, we consider only the top subjectivity clue words to learn the GBN layer. Lastly, In-order to preserve the context of words in conceptual phrases such as 'touchscreen'; we consider additional nodes in the Bayesian network for phrases with subjectivity clues. Further, the word embeddings in the CNN are initialized using the log-bilinear language model (LBL) where the d dimensional vector representation of each word $x_i(t)$ in (2) is given by :

$$x_i(t) = \sum_{k=1}^{i-1} C_k x_k(t) \tag{15}$$

where C_k are the $d \times d$ co-occurrence or context matrices computed from the data.

A. Pre-training using Gaussian features

The time series of sentences is used to generate a subset of sentences containing high ML motifs using (14). The frequency of a sentence in the new dataset will also correspond to the corresponding number of high ML motifs in the sentence. In this way, we are able to increase the weights of the corresponding causal features among words and concepts extracted using Gaussian Bayesian networks. The new set of sentences is used to pre-train the deep neural network prior to training with the complete dataset. Each sentence can be divided into chunks or phrases using POS taggers. The phrases have hierarchical structures and combine in distinct ways to form sentences. The k-gram kernels learned in the first layer hence correspond to a chunk in the sentence.

V. Experiments

We use the MPQA corpus [11], a collection of 535 English news articles from a variety of sources manually annotated with subjectivity flag. From the total of 9,700 sentences in this corpus, 55% of the sentences are labeled as subjective while the rest are objective. We also compare with the Movie Review (MR) benchmark dataset [18], that contains 5000 subjective movie review snippets from Rotten Tomatoes website and another 5000 objective sentences from plot summaries available from the Internet Movies Database. All sentences are at least ten words long and drawn from reviews or plot summaries of movies released post 2001.

The data pre-processing included removing top 50 stop words and punctuation marks from the sentences. Next, we used a POS tagger to determine the part-of-speech for each word in a sentence. Subjectivity clues dataset [10] contains a list of over 8,000 clues identified manually as well as automatically using both annotated and unannotated data. Each clue is a word and the corresponding part of speech.

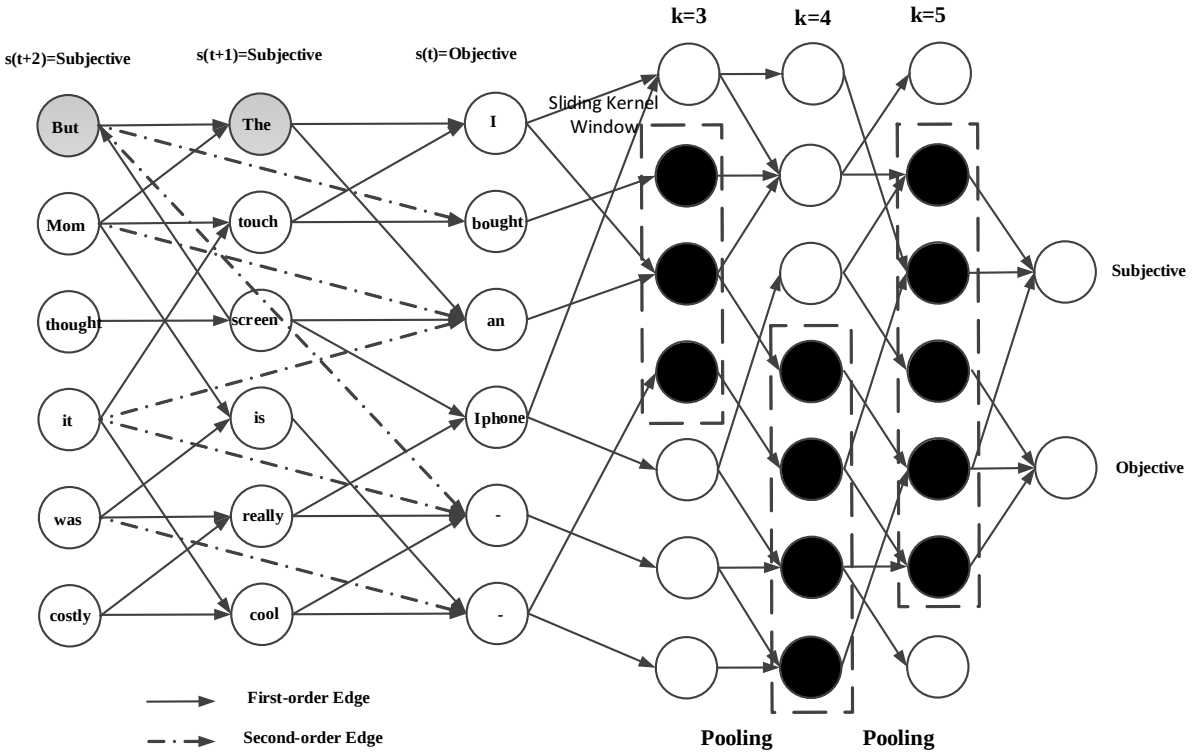

Fig. 2. State space of Bayesian CNN where the input layer is pre-trained using a dynamic GBN

The frequency of each clue was computed in both subjective and objective sentences of the MPQA corpus. Here we consider the top 50 clue words with highest frequency of occurrence in the subjective sentences. We also extracted 25 top concepts containing the top clue words using the method described in [30].

A. Time-delayed Gaussian features

In order to determine the optimal structure among the top words and concepts in subjective and objective sentences, each of the 9,700 sentences was transformed to a binary feature vector where presence of a top word is denoted as '1' and absence is denoted as '0'.

Since each sentence is dependent on the previous sentence in the article, the resulting matrix of words versus frequency is a time series. It must be noted that each word in a sentence is also dependent on the preceding words. Subsequently, we divide the matrix into subjective and objective datasets. We use multivariate Gaussian Bayesian fitness function to extract the maximum likelihood probabilities of each word given up-to three parent words and up-to two time points delay. Such sub-structures are referred to as network motifs. Top 20% of Motifs with high ML are used to select the training sentences for the CNN.

B. Comparison with Baselines

The CNN is collectively pre-trained with both subjective and objective sentences that contain high ML word and concept motifs. The word vectors are initialized using the LBL model and a context window of size 5 and 30 features. Each sentence is wrapped to a window of 50 words to reduce the number of parameters and hence the over-fitting of the model. A CNN with three hidden layers of 100 neurons and kernels of size $\{3, 4, 5\}$ is used. The output layer corresponds to two neurons for each class of sentiments.

We used 10 fold cross validation to determine the accuracy of classifying new sentences using the trained CNN classifier. A comparison is done with classifying the time series data using baseline classifiers such as Naïve Bayes SVM (NBSVM) [31], Multichannel CNN (CNN-MC) [32], Subjectivity Word Sense Disambiguation (SWSD) [33] and Unsupervised-WSD (UWSD) [34]. Table 2 shows that BCDBN outperforms previous methods by $5-10\%$ in accuracy on both datasets. Almost 10% improvement is observed over NBSVM on the movie review dataset. In addition, we only consider word vectors of 30 features instead of the 300 features used by CNN-MC and hence are 10 times faster.

C. Visualizing learned features

To visualize the learned features we consider the 5-grams in the test set that show highest activation when convolved with the learned kernels. Here, we simply consider the root mean square error between predicted 5-gram kernel vectors and the prior word-vectors for each 5-gram learned using co-occurrence data.

Table 1 shows Top 5-grams correlated to features learned at the hidden neurons in proposed BCDBN for 'Subjective' and 'Objective' sentences in the Movie Review dataset.

TABLE I
TOP 5-GRAMS CORRELATED TO FEATURES LEARNED AT THE HIDDEN NEURONS IN PROPOSED BCDBN FOR 'SUBJECTIVE' AND 'OBJECTIVE' SENTENCES IN THE MOVIE REVIEW DATASET.

Model	1	2	3	4	5
BCDBN(S)	stiff good help better	ponderous actors us fiction	charmless good clearly concocted	mechanical poetry see there	apparatus good world still
BCDBN(O)	american squirrel scandalous southern	adventurers must laurel gothic	follow go canyon tale	mysterious deep murders true	clues undercover which love

TABLE II
F-MEASURE BY DIFFERENT MODELS FOR CLASSIFYING SENTENCES IN A DOCUMENT AS SUBJECTIVE AND OBJECTIVE IN MPQA AND MR DATASET.

Dataset	NBSVM	CNN-MC	SWSD	UWSD	BCDBN
MPQA	86.3	89.4	80.35	60	**93.2**
MR	93.2	93.6	-	55	**96.4**

It can be seen that BCDBN captures subjective and objective sentiments in 5-grams very accurately, the objective 5-grams are factual describing plots of the movies while the objective 5-grams are strongly positive or negative reviews.

VI. CONCLUSION

In this paper, we have proposed a Bayesian deep convolutional belief network to classify a sequence of sentences as either subjective or objective. Our simulation and experimental study show that BCDBN outperforms several baseline approaches in terms of prediction accuracy. On the real benchmark dataset, it could achieve almost $5-10\%$ improvement in prediction accuracy to previous approaches and it was 10 times faster.

Previous methods for subjectivity detection learn word embedding from the data and do not have suitable prior features to initialize the model. Hence, here we propose the use of Gaussian Bayesian networks to extract high ML concepts and word motifs from the data and use them to pre-train the model.

REFERENCES

[1] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 102–107, 2016.

[2] M. Bonzanini, M. Martinez-Alvarez, and 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*, ser. SIGIR '12, 2012, pp. 1121–1122.

[3] G. Murray and G. Carenini, "Subjectivity detection in spoken and written conversations," *Natural Language Engineering*, vol. 17, pp. 397–418, 7 2011.

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

[5] H. Tang, S. Tan, and X. Cheng, "A survey on sentiment detection of reviews," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10 760 – 10 773, 2009.

[6] E. Cambria and A. Hussain, *Sentic Computing: A Common-Sense-Based Framework for Concept-Level Sentiment Analysis*. Cham, Switzerland: Springer, 2015.

[7] S. Poria, E. Cambria, A. Hussain, and G.-B. Huang, "Towards an intelligent framework for multimodal affective data analysis," *Neural Networks*, vol. 63, pp. 104–116, 2015.

[8] S. Poria, E. Cambria, N. Howard, G.-B. Huang, and A. Hussain, "Fusing audio, visual and textual clues for sentiment analysis from multimodal content," *Neurocomputing*, vol. 174, pp. 50–59, 2016.

[9] S. Poria, E. Cambria, and A. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network," *Knowledge-Based Systems*, vol. 108, 2016.

[10] E. Riloff and 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.

[11] J. Wiebe and 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.

[12] E. Cambria, S. Poria, R. Bajpai, and B. Schuller, "SenticNet 4: A semantic resource for sentiment analysis based on conceptual primitives," in *COLING*, 2016.

[13] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, 2014, pp. 655–665.

[14] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *In Proceedings of the Twenty-eight International Conference on Machine Learning, ICML*, 2011.

[15] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," 2013.

[16] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, 2011, pp. 142–150.

[17] C. Lin, Y. He, and R. Everson, "Sentence subjectivity detection with weakly-supervised learning," in *The 5th International Joint Conference on Natural Language Processing (IJCNLP)*, 2011.

[18] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 2004.

[19] J. Suzuki and H. Isozaki, "Sequence and tree kernels with statistical feature mining," in *Advances in Neural Information Processing Systems 18*, 2006, pp. 1321–1328.

[20] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.

[21] N. Kalchbrenner and P. Blunsom, "Recurrent convolutional neural networks for discourse compositionality," *CoRR*, vol. abs/1306.3584, 2013.

[22] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08, 2008, pp. 160–167.

[23] Z. Lu and H. Li, "A deep architecture for matching short texts." in *NIPS*, 2013, pp. 1367–1375.

[24] R. Kiros, R. Salakhutdinov, and R. Zemel, "Multimodal neural language models," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 595–603.

[25] Y. Mao and G. Lebanon, "Isotonic conditional random fields and local

sentiment flow," in *Advances in Neural Information Processing Systems*, 2007, pp. 961–968.

[26] A. Prinzie and D. Van den Poel, *Dynamic Bayesian Networks for Acquisition Pattern Analysis: A Financial-Services Cross-Sell Application New Frontiers in Applied Data Mining*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2009, vol. 5433, pp. 123–133.

[27] N. Friedman, K. Murphy, and S. Russell, "Learning the structure of dynamic probabilistic networks," *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, pp. 139–14, 1998.

[28] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771 – 1800, 2002.

[29] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007, pp. 1345–1352.

[30] S. Poria, E. Cambria, A. Gelbukh, F. Bisio, and A. Hussain, "Sentiment data flow analysis by means of dynamic linguistic patterns," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 26–36, 2015.

[31] S. Wang and C. Manning, "Fast dropout training," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 118–126.

[32] Y. Kim, "Convolutional neural networks for sentence classification," *CoRR*, vol. abs/1408.5882, 2014.

[33] R. Ortega, A. Fonseca, Y. Gutiérrez, and A. Montoyo, "Improving subjectivity detection using unsupervised subjectivity word sense disambiguation." *Procesamiento del Lenguaje Natural*, vol. 51, pp. 179–186, 2013.

[34] C. Akkaya, J. Wiebe, and R. Mihalcea, "Subjectivity word sense disambiguation," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 2009, pp. 190–199.