Multilingual Subjectivity Detection Using Deep Multiple Kernel Learning

Iti Chaturvedi
School of Computer Eng.
Nanyang Technological Univ.
iti@ntu.edu.sg

Erik Cambria
School of Computer Eng.
Nanyang Technological Univ.
cambria@ntu.edu.sg

Feida Zhu
School of Information Systems
Singapore Management Univ.
fdzhu@smu.edu.sg

Lin Qiu
School of Social Sciences
Nanyang Technological Univ.
linqiu@ntu.edu.sg

Wee Keong Ng
School of Computer Eng.
Nanyang Technological Univ.
wkn@ntu.edu.sg

ABSTRACT

Subjectivity detection can prevent a sentiment classifier from considering irrelevant or potentially misleading text. Since, different attributes may correspond to different opinions in the lexicon of different languages, we resort to multiple kernel learning (MKL) to simultaneously optimize the different modalities. Previous approaches to MKL for sentence classifiers are computationally slow and lack any hierarchy when grouping features into different kernels. In this paper, we consider deep recurrent convolution neural networks to reduce the dimensionality of the problem. Further, the lower layers in a deep model are abstract and the higher layers become more detailed connecting attributes to opinions. Hence, the features learned automatically in the multiple intermediate layers can be used to train MKL classifiers depending on the application. The proposed deep recurrent MKL outperforms the accuracy of baselines by over 5-30% and is several times faster on two benchmark datasets for subjectivity detection. It can also be used to develop subjectivity lexicons in other languages using English.

Keywords

Subjectivity Detection, Deep Convolution Neural Network, Multiple Kernel Learning, Recurrent Neural Networks, Gaussian Networks

1. INTRODUCTION

Four problems dominate sentiment classification, namely: subjectivity detection, word sentiment classification, document sentiment classification, and opinion extraction. Subjectivity detection is the task of labeling a document as either neutral or opinionated (i.e., positive or negative) and can prevent sentiment classifiers from considering irrelevant or potentially misleading text [24].

This paper was presented at the 4th International Workshop on Issues of Sentiment Discovery and Opinion Mining (WISDOM'15), held in conjunction with the 21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'15) in Sydney on 10 August 2015.

Sentiment classifiers aim to detect the sentiment information contained in both text [31] and videos [29]. The presence of neutral reviews, however, results in a low accuracy when classifying negative and positive examples. Document sentiment classification is used for comparison of consumer opinions of different products. Here again, with rapidly growing reviews on a product it becomes it becomes difficult for a potential consumer to make an informed decision on whether to purchase the product. Opinion extraction is used to summarize opinions in articles by presenting the sentiment polarities of correlated events. It is now convenient for consumers to clearly see the advantages and weaknesses of each product by merely a single glance [37].

For example in [14], the authors used domain adaptation on the Amazon dataset containing 340,000 reviews regarding 22 different product types and for which reviews are labelled as either positive, negative or neutral. There was a vast disparity between domains in the total number of instances and in the proportion of negative examples. For example, the word *conspiracy* is negative in many domains, but in the mystery novel domain, it is a favourable factor indicating positive sentiment.

Previous methods use well established general subjectivity clues to generate training data from un-annotated text [34, 24, 36, 3]. Bag of words (BOW) classifiers represent a document as a multi set of its words disregarding grammar and word order. The use of limited number of domain dependent words is however not enough, when dealing with social blogs like Twitter, as the content is often diverse and noisy. Hence, in [34], the authors used extraction pattern learning to automatically generate patterns that 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 state-of-the-art classifiers such as support vector machine (SVM) and Naive Bayes (NB) that assume that the class of a particular feature is independent of the class of other features given the training data [44].

Alternatively, knowledge-based approaches [6] can be applied, together with either linear [7] or non-linear [5] clustering techniques, to infer the subjectivity of words or multiword expressions. In general, however, templates are not suitable for semantic role labeling, because relevant context might be very far away.

For semantic role labeling, we need to know the relative position of verb, hence the features can include prefix, suffix, distance from verbs in the sentence, etc. As a result, it was found that manual lexicons focused on emotional words, while the lexicon learned by automatic methods tend to include many neutral words, introducing noise in the detection of subjectivity. Further, it can be seen that neutral n-grams are short while longer phrases tend to be subjective. Therefore, matrix representations for long phrases and matrix multiplication to model composition are being used to evaluate sentiment. For example, recursive neural networks predict the sentiment class at each node in the parse tree and try to capture the negation and its scope in the entire sentence [19, 14].

Lastly, news spreads very quickly via Twitter, hence to model the temporal nature of sentiment flow in [22], the authors use a sequence of sentences to design new author dependent features. Recently, Nate Silver has made forecasting elections a close to real-time experience. They proposed a diffusion model that predicts how a phenomenon spreads through a network like a disease [17].

The methods described focus on English language, hence to allow for portability to foreign languages such as Spanish or Arabic, dictionary based deep convolution neural networks are commonly used. For instance, we can assume that synonyms convey the same orientation and antonym relations convey an inverse sentiment in the foreign language after translation. Next, feature relatedness graphs are built for the foreign language using mappings from foreign senses to the English senses available in WordNet.

Convolution 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 [19]. 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 [12]. Next to model sentiment flow, in [18], the authors used recurrent CNN to model the dynamics in dialogue tracking and question answering systems. However, they assume that the data is uni-modal.

Since, different attributes such as *quality* or *cost* may correspond to different opinions such as *good* or *bad*, we resort to a deep multiple kernel learning framework to simultaneously optimize the different modalities. The proposed meta-level feature representation does not depend on the vocabulary size of the collection and hence provides considerable dimensionality reduction in comparison to unigram or n-gram models. In the next section, we review some recent work on the application of Multiple Kernel Learning (MKL) to natural language processing.

2. RELATED WORK AND OUTLINE

Subjectivity detection is a key problem in both knowledge-based [8] and statistical sentiment analysis [9], as sentiment classifiers are usually optimized for the detection of either positive or negative polarity. Different linear classification models have been proposed in the past but recently kernel methods have become increasingly popular, as non-linear kernels such as radial basis functions (RBF) show a considerably higher accuracy as compared to linear models.

It is often desirable to use multiple kernels simultaneously as multiple feature representations are derived from the sentences or because different kernels such as RBF or polynomial are used to measure the similarity between two sentences for the same feature representation. MKL is a feature selection method where features are organized into groups and each group has its own kernel function [35, 48]. However, the choice of kernel coefficients can have significant impact on the classification accuracy and efficiency of MKL [4].

Most previous applications of MKL have been in image and video classification and object recognition. For example in [15], multiple kernel learning (MKL) was used simultaneously optimize different modalities in Alzheimer disease images since different types of tests may reveal different aspects of the diagnosis. Recently, MKL with Fourier transform on the Gaussian kernels have been applied to Alzheimer disease classification using both sMRI and fMRI images [21]. MKL was also used to detect presence of large lump in images using a convolution kernel on Gaussian features [27].

In [40], higher order kernels are used to enhance the learning of MKL. Here, block co-ordinate Gradient optimization is used that approximates the Hessian matrix of derivatives, as a diagonal resulting is loss of information. Group-sensitive MKL for object recognition in images integrates a global kernel clustering method with MKL for sharing of group-sensitive information [47]. They showed that their method outperformed baseline-grouping strategies on the WikipediaMM data of real-world web images. The drawback of this method is that a looping strategy is used to relabel the groups and may not reach the global optimum solution. In [39],

MKL was also used to combine and re-weight multiple features by using structured latent variables during video event detection [39]. Here, two different types of kernels are used to group global features and segments in the test video that are similar to the training videos. The concept of kernel slack variables for each of the base kernels was used to classify YouTube videos in [46]. In order to select good features and discard bad features that may not be useful to the kernel, [25] used a beta prior distribution. Lastly, Online MKL shows good accuracy on object recognition tasks by extending online kernel learning to online MKL, however the time complexity of the methods is dependent on the dataset [45].

In the case of sentiment analysis, MKL was applied to Polish opinion aggregator service contains textual opinions of different products in [43], however they did not consider the hierarchical relation of different attributes of products. Video and text multi-modal features were also fused at different levels of fusion for indexing of web data in [26], however they are computationally very slow. It can be seen that the main challenges in using MKL is the computational time and the choice of suitable grouping strategy.

In this paper, we propose use of deep recurrent convolution neural networks to extract significant time dependent phrases and features from time series of sentences in a document or blog. These different feature representations at intermediate levels are optimized simultaneously in a multiple kernel learning classifier for subjectivity detection. The significance and contributions of the research work presented in this paper can be summarized as follows:

- In this paper, we consider deep recurrent multiple ker-

nel learning to learn subjectivity features from paragraphs. From our knowledge, no previous work has considered MKL to simultaneously optimize the features learned in the different layers of deep recurrent convolution neural networks.

- In a deep model, the lower layers are abstract and the higher layers become more detailed connecting attributes to opinions. The subsets of features in the intermediate layers are hence portable to other languages such as Spanish after training in English. Further, pre-training of the deep neural network is done using Gaussian Bayesian networks over Subjectivity clues in English.

- The k-gram sliding window features learned by deep convolution neural network are used to train a recurrent neural network that can determine the temporal dependence among learned features. These features are then used to learn the MKL classifier using with Gaussian kernels.

Figure 1 illustrates the state space of the proposed Deep Recurrent Multiple Kernel Learning with time delays for three sentences in a review on iPhone. The CNN extracts significant k-gram features and reduce the dimensionality of the data. Next, recurrent neural network (RNN) is used to learn time-delayed features and reduce the dimensionality further. The hidden neurons are interconnected and the dashed lines correspond to time delay edge for output of hidden neurons that becomes a part of the input at next time point. Lastly, MKL is used to classify the sentences.

To verify the effectiveness of deep recurrent MKL in capturing dependencies in high-dimensional data, we consider the MPQA corpus [44], which is a collection of 535 English-language news articles from a variety of news sources manually annotated for subjectivity. From the 9,700 sentences in this corpus, 55% of the sentences are labelled as subjective while the rest are objective. Further, to measure the portability of the proposed method on language translation task we consider a corpus of 504 sentences manually annotated for subjectivity in Spanish [23]. Here, we try to develop a Subjectivity lexicon for Spanish language using the available resources in English. The classification accuracy obtained using the proposed deep recurrent MKL is shown to outperform the baseline by over 5-30% on the real datasets.

The rest of the paper is organized as follows: Section 3 provides the preliminary concepts necessary to comprehend the proposed deep recurrent MKL algorithm of the present work. In section 4, we introduce the proposed deep recurrent MKL 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.

3. PRELIMINARIES

In this section, we briefly review the theoretical concepts necessary to comprehend the present work. We begin with a description of maximum likelihood (ML) estimation of edges in dynamic Gaussian Bayesian networks where each node is a word in a sentence. Next, we show how high ML word network motifs predicted by GBN can be used to pre-train the weights of a deep convolution neural network that classifies sentences by minimizing a global error function over a linear combination of words in a sentence.

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 $i^{\text{ th}}$ random variable in instance t. Lastly, let $\boldsymbol{a}_i$ be the set of parent variables regulating variable i.

3.1 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 $s(1), s(2), \ldots, s(T)$ and the corresponding target labels $y(t) \in \{Subj, Obj\}$. Each sentence in turn is 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 [33]. It is a directed acyclic graph S with a set of parameters $\boldsymbol{\theta}$ that represents the strengths of connections by conditional probabilities. 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 $\boldsymbol{a}_i$ in the current or previous time points, and $\theta_{i,\boldsymbol{a}_i}$ denotes the ML estimate of the conditional probabilities.

To find the likelihood in (2), and to obtain the optimal Gaussian network, 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 N × N. The joint probability of the network can be the product of a set of conditional probability distributions given by:

$$p(\mathbf{x}_i|\boldsymbol{a}_i) = \theta_{i,\boldsymbol{a}_i} \sim \mathcal{N}\left(\mu_i + \sum_{j\in\boldsymbol{a}_i}(\mathbf{x}_j - \mu_j)\beta,\ \Sigma_i^{'}\right), \quad (3)$$

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 $\boldsymbol{a}_i$.

To compute the likelihood of $\Sigma_i^{'}$ efficiently we can use Cholesky decomposition :

$$\Sigma_i^{'} = R^T R \quad (4)$$

where R is an upper triangular matrix, and the likelihood of $\Sigma_i^{'}$ is simply the sum of the log of the diagonal elements of R.

To extract dynamic network motifs, we simply compute conditional probabilities using parent word expressions in the previous r time points. Next, high ML word network motifs compute using (3) can be used as prior features for training convolution neural network models described in the next section.

3.2 Deep Convolution 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 $c(t) = (c_1(t), c_2(t), \ldots, c_{\mathrm{L}}(t))$.

$$c_j = {w_k}^T.\mathbf{x}_{i:i+k-1} \tag{5}$$

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 [19]. 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 [13]. 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.

Since, the computation of gradient becomes difficult with increasing number of layers, we consider a deep belief network for learning the subjectivity features. 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:

$$p(\mathbf{x}|W) = \frac{1}{Z(W)}\exp^{-\mathrm{E}(\mathbf{x};W)} \tag{6}$$

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 normalization 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 [16]. As an example, here we use a logistic regression model to learn the binary hidden neurons and each visible unit is assumed a sample from a normal distribution [38].

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:

$$\hat{h}_j = b_j + \sum_i v_i w_{ij}, \tag{7}$$

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:

$$h_j = \frac{1}{1 + e^{-\hat{h}_j}}. \tag{8}$$

Similarly, in the next iteration, the binary state of each visible node is reconstructed and labelled 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:

$$\hat{v}_i = c_i + \sum_j h_i w_{ij}, \tag{9}$$

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:

$$\triangle w_{ij} = \alpha(< v_i h_j >_{data} - < v_i h_j >_{recon}), \tag{10}$$

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 (7). 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 (5) 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:

$$\mathrm{E} = -\sum_{z=1}^{\mathrm{Z}} \sum_{i,j}^{\mathrm{L}-k+1,1} \sum_{r,s}^{k,d} v_{i+r-1,j+s-1} h_{ij}^z w_{rs}^k \tag{11}$$

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 [18]. Since, different attributes may correspond to different opinions in the next section we resort to a deep multiple kernel learning framework to simultaneously optimize the different modalities in text.

4. DEEP RECURRENT MKL

In this section, we describe the deep recurrent MKL framework. Here, the low dimensional features learned in the intermediate layer of a deep recurrent CNN are used to train the MKL classifier.

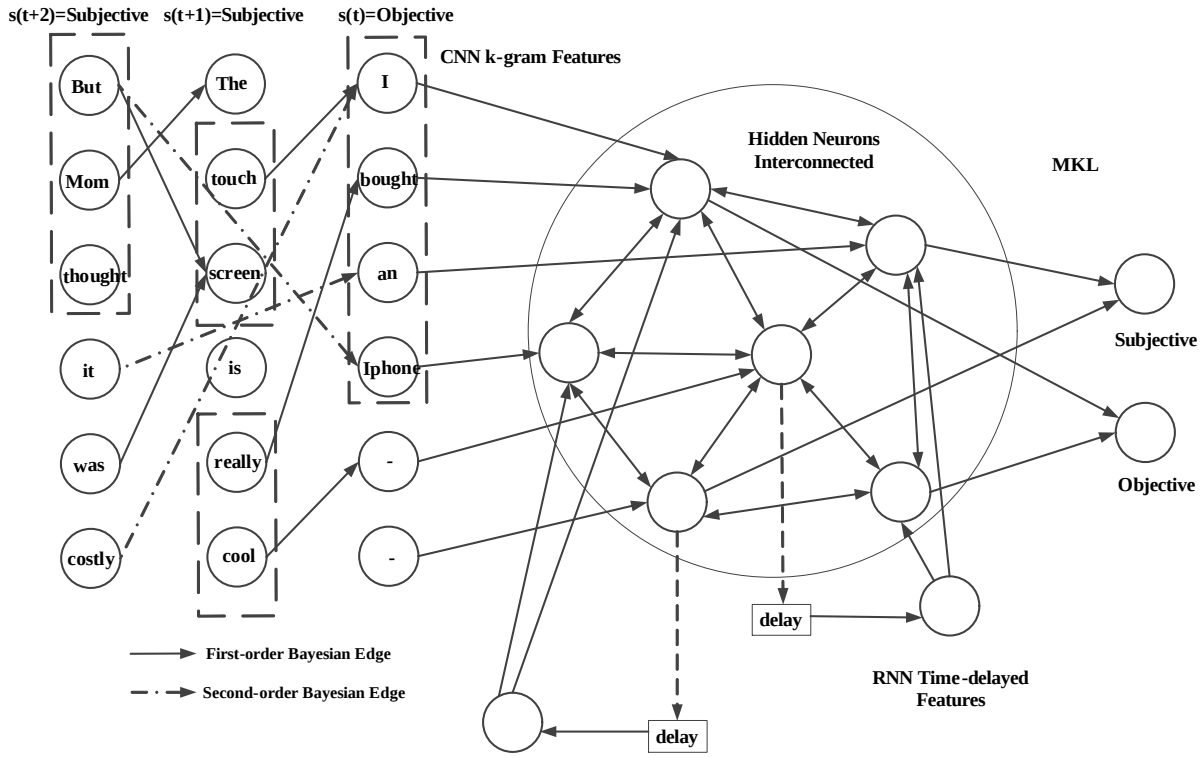


Figure 1: Illustrates the state space of a Deep Recurrent Multiple Kernel Learning with time delays for three sentences in a review on iPhone. The CNN extracts significant k-gram features and reduce the dimensionality of the data. Next, RNN is used to learn time-delayed features and reduce the dimensionality further. The hidden neurons are interconnected and the dashed lines correspond to time delay edge for output of hidden neurons that becomes a part of the input at next time point. Lastly, MKL is used to classify the sentences.

To pre-train the deep recurrent CNN, we use the training data to extract high ML network motifs on top subjectivity clue words similar to the approach described in [11]. To create the modified training data the time series of sentences is used to generate a sub-set of sentences containing high ML motifs using (3). 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.

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. In-order to preserve the context of words in conceptual phrases such 'touch-screen'; we consider additional nodes in the Bayesian network for phrases with subjectivity clues. The new set of sentences is used to learn the word vectors and pre-train the deep neural network prior to training with the complete dataset.

Algorithm 1 shows the pseudo code for creating a modified training data using Gaussian Bayesian networks of subjectivity clues. Here, we consider up-to r previous sequences when computing the likelihood of network motifs and the maximum number of possible parent words is denoted as g. Lastly, the threshold γ is used to select high ML network motifs and create the new training data $\hat{4}$ as described in line no's 3:11.

However, several word dependencies may occur across sentences hence, in this paper we further consider a recurrent neural network layer to extract significant time-delayed features in the sentences.

4.1 Recurrent Neural Networks

The standard RNN output, $\mathbf{x}_l(t)$, at time step t for each layer l is calculated using the following equations :

$$\mathbf{x}_l(t) = f(W^l_{\mathrm{R}}.\mathbf{x}_l(t-1) + W_l.\mathbf{x}_{l-1}(t)) \tag{12}$$

where W_{R} is the interconnection matrix among hidden neurons and W_l is the weight matrix of connections between hidden neurons and the input nodes, $\mathbf{x}_{l-1}(t-1)$ is the input vector at time step t from layer $l-1$, vectors $\mathbf{x}_l(t)$ and $\mathbf{x}_l(t-1)$ represent hidden neuron activations at time steps t and $t-1$ and f is the non-linear activation function. To learn the weights of the RNN, back propagation through time is used where the hidden layer is unfolded in time using duplicate hidden neurons. Figure 1 illustrates the state space of a recurrent neural network with inter-connected hidden neurons and feedback delays.

Each new layer of hidden neurons models a single time delay and is trained using time shifted input data. The features learned at the hidden neurons can then be used to train standard classifiers such as MKL. **Algorithm 1** shows the pseudo code for learning temporal features using recurrent neural networks in line no 28:40.

Algorithm 1 Deep Recurrent Multiple Kernel Learning

Input : Training sequence of sentences $s = \{s(1), s(2), \ldots, s(\mathrm{T})\}$ with max length L and corresponding class labels $y(t) \in \{Subj, Obj\}$.
Output : Class labels of Test Sentences
% *Extract modified training data $\hat{s}$ using Gaussian Bayesian Networks of subjectivity clues*
$\boldsymbol{\theta} = \{\theta_{i,\boldsymbol{a}_i} = p(\mathbf{x}_i|\boldsymbol{a}_i), |\boldsymbol{a}_i| \leq g\} \forall i, \forall \boldsymbol{a}_i$ using (3) and time series s and parent expression from up-to r previous time points.
$\boldsymbol{M} = \{\theta_{i,\boldsymbol{a}_i} \geq \gamma\}, \forall \theta_{i,\boldsymbol{a}_i} \in \boldsymbol{\theta}$
$\hat{s} =$
for $t = 1$ to T **do**
  for $m = 1$ to $|\boldsymbol{M}|$ **do**
    if $\{i, a_i\}^m \in \{s(t) : s(t-r)\}$ **then**
      $\hat{s} = \hat{s} \bigcup \{s(t) : s(t-r)\}$
    end if
  end for
end for
$\hat{s} = \hat{s} \bigcup s$
% *Extract k-gram features using deep convolution neural network*
Construct a minimal deep CNN with visible layer of $\mathrm{L} \times d$ nodes and first hidden convolution layer l of n_l k-gram neurons
repeat
  for $t = 1$ to $|\hat{s}|$ **do**
    Initialize the visible layer with t^{th} training sample in $\hat{s}(t)$
    Use (11) to do convolution
    Update W_l using CD given by (10) $\forall l$
  end for
  Compute change in reconstruction error $\triangle\epsilon$ on training data $\hat{s}$
  if $\triangle\epsilon$ is significant **then**
    Construct another convolution hidden layer of neurons
  end if
until Adding a layer does not change visible layer reconstruction error
Construct a penultimate hidden logistic layer of n_h neurons and n_d output neurons
Fine-tune weights using known class label of training samples
The expression of $\hat{s}$ samples at n_h learned features of the logistic layer form the new dataset $\hat{s}_2$ of dimension $n_h \times \mathrm{T}$.
% *Extract temporal features using recurrent neural network*
Construct a minimal RNN with visible layer of with n_h nodes and first layer of n_r interconnected hidden neurons with time-delays
repeat
  for $t = 1$ to $|\hat{s}_2|$ **do**
    Initialize the visible layer with t^{th} training sample in $\hat{s}_2(t)$
    Update W_l layer weights and W_{R}^l neuron interconnection weights using CD given by (10)
  end for
  Compute change in reconstruction error $\triangle\epsilon$ on training data $\hat{s}_2$
  if $\triangle\epsilon$ is significant **then**
    Construct another hidden layer of interconnected neurons
  end if
until Adding a layer does not change visible layer reconstruction error
The expression of $\hat{s}_2$ samples at n_r learned features of the hidden neurons form the new dataset $\hat{s}_3$ of dimension $n_r \times \mathrm{T}$.
% *Classification using Multiple Kernel Learning*
Train an MKL classifier with n_r features and T samples using (13)
Each test sample is used to generate n_h outputs from deep CNN and n_r outputs from RNN and finally classified using MKL

The weight matrix W_{R}^l computes the interconnection matrix of hidden neurons using outputs at previous R time steps. To determine the number of hidden neurons in each new layer it is convenient to use the number of significant principle components in the input data.

4.2 Multiple Kernel Learning

Multiple kernel learning uses the sequence of sentences $s(1), s(2), \ldots, s(T)$ and the corresponding target labels $y(t) \in \{Subj, Obj\}$ to train a classifier of the dual form :

$$\max_{\beta} \min_{\alpha} \frac{1}{2} \sum_{i=1}^{T} \sum_{j=1}^{T} \alpha_i \alpha_j y(i) y(j) \left(\sum_{m=1}^{M} \beta_m K_m(s(i), s(j)) \right) - \sum_{i=1}^{T} \alpha_i,$$

$$\text{s.t} \sum_{i=1}^{T} \alpha_i y(i) = 0, \ \sum_{m=1}^{\mathrm{M}} \beta_m = 1, \ 0 \leq \alpha_i \leq C \forall i. \qquad (13)$$

where M is the total number of positive definite Gaussian kernels $K_m(s(i), s(j))$ each with a set of different parameters and α_i, b and $\beta_m \geq 0$ are co-efficients to be learned simultaneously from the training data using quadratic programming.

4.3 Deep MKL Framework

Algorithm 1 describes the complete framework for predicting time-delayed networks using the proposed deep recurrent MKL. We first construct a minimal deep CNN with visible layer of $\mathrm{L} \times d$ nodes, where L is length of the sentence and d is the number of features for each word; first hidden convolution layer of k-gram neurons, second hidden logistic layer of n_h neurons and n_d output neurons. The n_h features expressed at logistic layer after training form the new input data of T samples. Next, we construct a RNN with n_h input nodes and n_r hidden neurons with time-delays. The n_r features expressed at the hidden neurons after training form the new input data of T samples. Lastly, we train an MKL classifier with n_r features and T samples. Each test sample is used to generate n_h outputs from deep CNN and n_r outputs from RNN and finally classified using MKL.

To determine the number of hidden layers in the deep CNN and the RNN, we compute the change in visible layer reconstruction error $\triangle\epsilon$ on the training samples. This is the root means square error between input training sample and reconstructed sample at each visible node. If there is a significant change in the error $\triangle\epsilon$, a new hidden layer is added. The layer weights are then learned and the reconstruction error is recomputed. The above progresses iteratively until further addition of hidden layers does not change the classification precision error significantly, and the optimal configuration is achieved. To determine the optimal number of hidden neurons in a single layer, we consider the number of significant principal components in the training data for that layer.

Each hidden neuron in the final output layer will correspond to a particular class. The contrastive divergence approach will sample features with high frequency into the upper layers, resulting in the formation of phrases at hidden neurons in the first layer, bigger sentences at hidden neurons in second hidden layer and so on. We iterate through the algorithm until there is no significant change in the weights at the l^{th} layer.

5. EXPERIMENTS

In order to evaluate the performance of our method on a large dataset, we use the MPQA corpus [44], which is a collection of 535 English-language news articles from a variety of news sources manually annotated for subjectivity after machine translation from Spanish. There are 9,700 sentences in this corpus, 55% of the sentences are labelled as subjective while the rest are objective.

Next, to measure the portability of the proposed method on language translation task we consider another MPQA Gold corpus of 504 sentences manually annotated for subjectivity in Spanish. The annotation resulted in 273 subjective and 231 objective sentences as described in [23]. Lastly, the sentences are machine translated into English to obtain the training dataset.

The second corpus is small, as the annotators need to be trained with annotation guidelines in Spanish. Some sentences are difficult to annotate as Objective or Subjective and hence are annotated by several different annotators. However, it is a popular benchmark used by previous authors, and can evaluate the robustness of Deep MKL when few training samples are present. Hence, we aim to provide a comparison with baselines on different training data sizes.

5.1 Preprocessing

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 [34] contains a list of over 8,000 clues identified manually as well as automatically using both annotated and un-annotated data. Each clue is a word and the corresponding part of speech. 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, 32].

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 (ML) 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 convolution neural network.

Table 1: F-measure by different models for classifying sentences in a document as Subjective and Objective in MPQA dataset.

Dataset	NBSVM	CNN-MC	SWSD	UWSD	Deep MKL
MPQA	86.3	89.4	80.35	60	**97.2**

Table 2: F-measure by different models for classifying Spanish sentences in a document as Subjective and Objective in MPQA Gold dataset. Precision, Recall and F-measure of correctly classifying test data from both classes is reported.

Model	Type	Precision	Recall	F-measure
Rule Based [23]	Obj	0.56	0.48	0.52
	Subj	0.8	0.2	0.32
	Total	0.62	0.33	0.44
Bootstrapping [10]	Obj	0.56	0.48	0.52
	Subj	0.8	0.21	0.32
	Total	0.62	0.33	0.43
Deep MKL	Obj	0.7	0.7	0.69
	Subj	0.81	0.8	0.8
	Total	**0.75**	**0.75**	**0.75**
SVM [2]	NB	0.62	0.62	0.62
	SVM	0.62	0.62	0.62

5.2 Comparison with Baselines

Lastly, 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 deep CNN with three hidden layers of 100 neurons and kernels of size $\{3, 4, 5\}$ and one logistic layer of 300 neurons is used. The output layer corresponds to two neurons for each class of sentiments. The 300 feature outputs of deep CNN are used to train a recurrent NN with 10 hidden neurons and up-to 2 time point delays. These 10 features are then used to train the simpleMKL classifier.

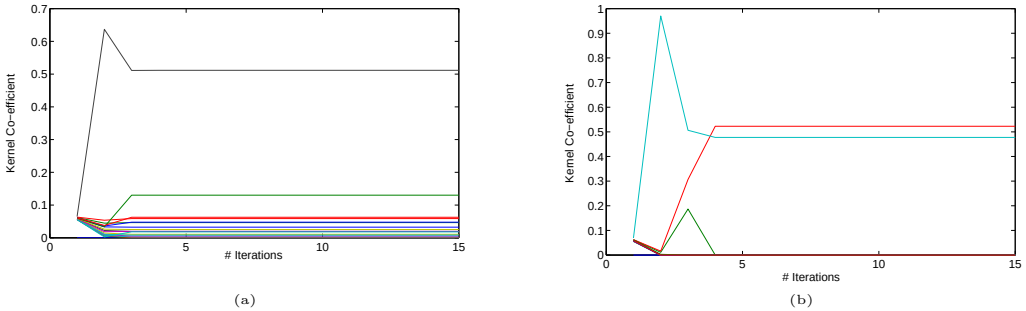
We used 10 fold cross validation to determine the accuracy of classifying new sentences using the trained convolution neural network classifier. A comparison is done with classifying the time series data using baseline classifiers such as Naive Bayes Support Vector Machine (NBSVM) [41, 42], Multichannel Convolution Neural Network (CNN-MC) [20], Subjectivity Word Sense Disambiguation (SWSD) [28] and Unsupervised-WSD (UWSD) [1]. Table 5.1 shows that deep recurrent MKL outperforms previous methods by 5-12% in accuracy. Almost 12% improvement is observed over NBSVM. 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.

We can also theoretically justify the higher accuracy of the proposed Deep MKL compared to baselines. NBSVM is a variant of SVM that uses NB log-odds ratios as input features. However, it assumes a single kernel function for the entire dataset. Since, different attributes may correspond to different opinions in the lexicon of different languages. Hence, in this paper, Deep MKL divides the data into a hierarchy of groups and a different kernel is used for each group.

Similarly, CNN-MC used a single hidden layer of neurons that lacks any hierarchy when learning features. However, in a deep sentence model, the lower layers are abstract concepts and the higher layers become more detailed connecting attributes to opinions. Hence, we propose to use several hidden layers with kernels of increasing width [14]. It can be justified that sentences of different lengths will be optimally classified at different depths.

Table 3: Top 3-grams correlated to features in English and Spanish learned at the hidden neurons in proposed deep recurrent MKL for 'Subjective' and 'Objective' sentences in the Gold MPQA corpus. The Subjectivity clues are shown in bold

	English			Spanish		
Model	1	2	3	1	2	3
Subjective	Communist group baseball traitors	**victory** commended men **assured**	dealt **ability** **agree** our	**Victoria** grupo los hombres de tratados	comunista elogio la beisbol **aseguraron**	trato **capacidad** **coinciden** nuestra
Objective	1954 throats spectators grandmother	when shantung white watching	hit like crush event	1954 gargantas Espectadores abuela	cuando shantung aplastamiento evento	es golpeado como blanco viendo



(a) (b)

Figure 2: Kernel Co-efficients evolving with iterations in MKL. (a) Kernel Co-efficients for input features learned by Deep CNN (b) Kernel Co-efficients evolving for input features learned by Deep CNN followed by RNN.

The Bootstrapping method starts with a set of seed words in Spanish and iteratively includes new words into the lexicon with maximum similarity in each Bootstrap or iteration. Such a method is unable to capture the temporal dependence between sentences. By using a layer of recurrent neurons, we are able to learn time-delayed features for polarity changes within a single review. Lastly, WSD and rule based classifiers are heavily dependent on templates and do not consider the relative positions between nouns and verbs.

Similarly, for the MPQA gold corpus a comparison was done with baseline classifiers such as rule-based classifier [23], bootstrapping based classifier [10], SVM and Naive Bayes [2]. Table 2 shows that the F-measure of deep recurrent MKL outperforms previous methods by almost 5-30% . Almost 30% improvement was observed over rule based classifiers. In addition, it is much faster than baseline classifiers such SVM.

5.3 Visualizing learned Text features

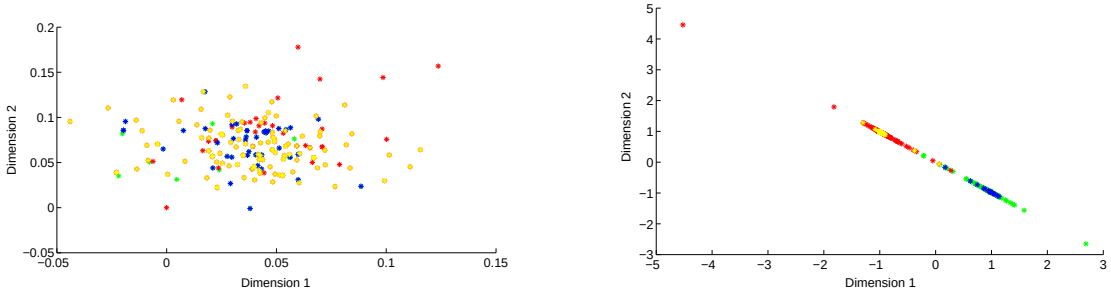
To visualize the learned features we consider the 3-grams in the test set that show highest activation when convolved with the learned kernels. Here, we simply consider the root mean square error between predicted 3-gram kernel vectors and the prior word-vectors for each 3-gram learned using co-occurrence data.

Table 3 shows Top 3-grams correlated to features learned at the hidden neurons in proposed deep recurrent MKL for 'Subjective' and 'Objective' sentences in the Gold MPQA dataset. It can be seen that our method captures subjective and objective sentiments in 3-grams very accurately, the objective 3-grams are factual while the objective 3-grams are strongly positive or negative comments. Further, by translating them into Spanish we can determine new Subjective clues and their context using the clues in English language.

5.4 Visualizing learned Support Vectors

Figure 3 shows the support Vectors predicted by MKL. Subjective data is in Green and Objective data is in Red. Subjective Support Vectors are Blue and Objective Support Vectors are Yellow. Figure 2(a) shows the support Vectors for two features learned by Deep CNN and Figure 2 (b) shows the support Vectors for two features learned by Deep CNN followed by RNN. It can be seen that the features learned by Deep CNN are overlapping and MKL is not able to classify them easily. However, by using a recurrent neural network we learn new features that are linearly separated with high accuracy.

Figure 2 shows the number of kernels used by both datasets and the evolution of kernel co-efficients with iterations in MKL. Figure 2(a) shows the kernel co-efficients for input features learned by Deep CNN and Figure 2 (b) shows the kernel co-efficients evolving for input features learned by Deep CNN followed by RNN. The features learned by deep CNN only require 3 distinct kernels to capture the multi-modal effect however, the features learned by RNN are multi-modal and require 18 distinct kernels that evolve to classify the data with high accuracy. Hence, we can conclude that by using RNN prior to MKL we can extract significant multi-modal features in the data.



(a) (b)

Figure 3: Support Vectors predicted by MKL. Subjective data is in Green and Objective data is in Red. Subjective Support Vectors are Blue and Objective Support Vectors are Yellow (a) Support Vectors for two Features learned by Deep CNN (b) Support Vectors for two features learned by Deep CNN followed by RNN.

6. CONCLUSION

In this paper, we have proposed deep recurrent multiple kernel learning to classify a sequence of sentences as Subjective or Objective. Our simulation and experimental study show that the method outperforms several baseline approaches in terms of prediction accuracy. On the real benchmark dataset, it could achieve almost 5-30% improvement in prediction accuracy to previous approaches and was much faster.

Multiple kernel learning is extremely slow for natural language tasks. Hence, we consider deep recurrent convolution neural networks to reduce the dimensionality of the problem and learn a hierarchy of phrases such that the lower layers are more abstract and the higher layers combine phrases to connect attributes and opinions. The different phrase representations learned in the intermediate layers are simultaneously optimized using the MKL classifier.

Deep CNN is able to extract significant k-gram phrases and then recurrent neural network further reduces the dimensionality by selecting significant time-delayed features to train the MKL classifier. Since, different features of the same sentence may have different effects on the sentiment of sentence, the MKL classifier is able to better capture the multi-modal nature of the dataset.

7. REFERENCES

[1] C. Akkaya, J. Wiebe, and R. Mihalcea. Subjectivity word sense disambiguation. In *EMNLP*, pages 190–199, 2009.

[2] C. Banea, R. Mihalcea, J. Wiebe, and S. Hassan. Multilingual subjectivity analysis using machine translation. In *EMNLP*, pages 127–135, 2008.

[3] M. Bonzanini, M. Martinez-Alvarez, and T. Roelleke. Opinion summarisation through sentence extraction: An investigation with movie reviews. In *SIGIR*, SIGIR '12, pages 1121–1122, 2012.

[4] S. Bucak, R. Jin, and A. Jain. Multiple kernel learning for visual object recognition: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(7):1354–1369, 2014.

[5] E. Cambria, P. Gastaldo, F. Bisio, and R. Zunino. An ELM-based model for affective analogical reasoning. *Neurocomputing*, 149:443–455, 2015.

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

[7] E. Cambria, T. Mazzocco, A. Hussain, and C. Eckl. Sentic medoids: Organizing affective common sense knowledge in a multi-dimensional vector space. In D. Liu, H. Zhang, M. Polycarpou, C. Alippi, and H. He, editors, *Advances in Neural Networks*, volume 6677 of *Lecture Notes in Computer Science*, pages 601–610, Berlin, 2011. Springer-Verlag.

[8] E. Cambria, B. Schuller, B. Liu, H. Wang, and C. Havasi. Knowledge-based approaches to concept-level sentiment analysis. *IEEE Intelligent Systems*, 28(2):12–14, 2013.

[9] E. Cambria, B. Schuller, B. Liu, H. Wang, and C. Havasi. Statistical approaches to concept-level sentiment analysis. *IEEE Intelligent Systems*, 28(3):6–9, 2013.

[10] R. M. Carmen Banea and J. Wiebe. A bootstrapping method for building subjectivity lexicons for languages with scarce resources. In *LREC*, 2008.

[11] I. Chaturvedi, Y.-S. Ong, and R. V. Arumugam. Deep transfer learning for classification of time-delayed gaussian networks. *Signal Processing*, 110(0):250 – 262, 2015.

[12] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, ICML '08, pages 160–167, 2008.

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

[14] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, 2011.

[15] C. Hinrichs, V. Singh, G. Xu, and S. Johnson. Mkl for robust multi-modality ad classification. *Med Image Comput Comput Assist Interv*, 5762:786–94, 2009.

[16] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771 – 1800, 2002.

[17] V. Kagan, A. Stevens, and V. Subrahmanian. Using twitter sentiment to forecast the 2013 pakistani

election and the 2014 indian election. *Intelligent Systems, IEEE*, 30(1):2–5, 2015.

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

[19] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. In *ACL*, pages 655–665, 2014.

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

[21] F. Liu, L. Zhou, C. Shen, and J. Yin. Multiple kernel learning in the primal for multimodal alzheimers disease classification. *Biomedical and Health Informatics, IEEE Journal of*, 18(3):984–990, 2014.

[22] Y. Mao and G. Lebanon. Isotonic conditional random fields and local sentiment flow. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 961–968. MIT Press, 2007.

[23] R. Mihalcea, C. Banea, and J. Wiebe. Learning multilingual subjective language via cross-lingual projections. In *ACL*, 2007.

[24] G. MURRAY and G. CARENINI. Subjectivity detection in spoken and written conversations. *Natural Language Engineering*, 17:397–418, 7 2011.

[25] B. Ni, T. Li, and P. Moulin. Beta process multiple kernel learning. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 963–970, 2014.

[26] U. Niaz and B. Merialdo. Fusion methods for multi-modal indexing of web data. In *Image Analysis for Multimedia Interactive Services (WIAMIS), 2013 14th International Workshop on*, pages 1–4, 2013.

[27] S. Nilufar, N. Ray, and H. Zhang. Object detection with dog scale-space: A multiple kernel learning approach. *Image Processing, IEEE Transactions on*, 21(8):3744–3756, 2012.

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

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

[30] S. Poria, E. Cambria, G. Winterstein, and G.-B. Huang. Sentic patterns: Dependency-based rules for concept-level sentiment analysis. *Knowl.-Based Syst.*, 69:45–63, 2014.

[31] S. Poria, A. Gelbukh, E. Cambria, A. Hussain, and G.-B. Huang. EmoSenticSpace: A novel framework for affective common-sense reasoning. *Knowledge-Based Systems*, 69:108–123, 2014.

[32] S. Poria, A. Gelbukh, D. Das, and S. Bandyopadhyay. Fuzzy clustering for semi-supervised learning – case study: Construction of an emotion lexicon. In *Advances in Artificial Intelligence*, volume 7629 of *Lecture Notes in Computer Science*, pages 73–86. Springer Berlin Heidelberg, 2013.

[33] 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*, volume 5433 of *Lecture Notes in Computer Science*, pages 123–133. Springer Berlin / Heidelberg, 2009.

[34] E. Riloff and J. Wiebe. Learning extraction patterns for subjective expressions. In *EMNLP*, pages 105–112, 2003.

[35] N. Subrahmanya and Y. Shin. Sparse multiple kernel learning for signal processing applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(5):788–798, 2010.

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

[37] H. Tang, S. Tan, and X. Cheng. A survey on sentiment detection of reviews. *Expert Systems with Applications*, 36(7):10760 – 10773, 2009.

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

[39] A. Vahdat, K. Cannons, G. Mori, S. Oh, and I. Kim. Compositional models for video event detection: A multiple kernel learning latent variable approach. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1185–1192, 2013.

[40] S. Wang, S. Jiang, Q. Huang, and Q. Tian. Multiple kernel learning with high order kernels. *Pattern Recognition, International Conference on*, 0:2138–2141, 2010.

[41] S. Wang and C. Manning. Fast dropout training. In *ICML*, pages 118–126, 2013.

[42] S. Wang and C. D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *ACL*, pages 90–94, 2012.

[43] A. Wawer. Mining opinion attributes from texts using multiple kernel learning. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 123–128, 2011.

[44] J. Wiebe and E. Riloff. Creating subjective and objective sentence classifiers from unannotated texts. In *COLING*, pages 486–497, 2005.

[45] H. Xia, S. Hoi, R. Jin, and P. Zhao. Online multiple kernel similarity learning for visual search. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(3):536–549, 2014.

[46] X. Xu, I. Tsang, and D. Xu. Soft margin multiple kernel learning. *Neural Networks and Learning Systems, IEEE Transactions on*, 24(5):749–761, 2013.

[47] J. Yang, Y. Tian, L.-Y. Duan, T. Huang, and W. Gao. Group-sensitive multiple kernel learning for object recognition. *Image Processing, IEEE Transactions on*, 21(5):2838–2852, 2012.

[48] Z. Zhang, Z.-N. Li, and M. Drew. Adamkl: A novel biconvex multiple kernel learning approach. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 2126–2129, 2010.