COAL: Convolutional Online Adaptation Learning for Opinion Mining

Iti Chaturvedi
Information Technology,
James Cook University, Australia
iti.chaturvedi@jcu.edu.au

Edoardo Ragusa, Paolo Gastaldo
DITEN,
University of Genoa, Italy
{edoardo.ragusa,paolo.gastaldo}@unige.it

Erik Cambria
School of Computer Science and Engineering,
Nanyang Technological University, Singapore
cambria@ntu.edu.sg

2020 International Conference on Data Mining Workshops (ICDMW) | 978-1-7281-9012-9/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICDMW51313.2020.00012

***Abstract*—Thanks to recent advances in machine learning, some say AI is the new engine and data is the new coal. Mining this 'coal' from the ever-growing Social Web, however, can be a formidable task. In this work, we address this problem in the context of sentiment analysis using convolutional online adaptation learning (COAL). In particular, we consider semi-supervised learning of convolutional features, which we use to train an online model. Such a model, which can be trained in one domain but also used to predict sentiment in other domains, outperforms the baseline in the range of 5-20%.**

***Index Terms*—Online SVM, Sentiment Prediction, Domain Adaptation**

I. Introduction

In recent years, sentiment analysis has become increasingly popular for processing social media data on online communities, blogs, wikis, microblogging platforms, and other online collaborative media. Sentiment analysis is a branch of affective computing research that aims to mine opinions from text (but sometimes also videos) based on review helpfulness and user intent. Sentiment analysis has raised growing interest in the business world, due to the remarkable benefits to be had from financial forecasting [1], marketing [2], healthcare [3], dialogue systems [4], and many more. The aim of the paper is to predict the sentiments of text posted on the social media. However, such content is continuously posted. Hence, we need a model that can adapt to new sentences without the need for retraining. Table I illustrates two product reviews with positive and negative star ratings. We can see that there are 'neutral' or 'negative' comments in the positive review and vice-versa.

Sentiment analysis techniques can be broadly categorized into symbolic and sub-symbolic approaches: the former include the use of lexicons, ontologies, and semantic networks [5] to encode the polarity associated with words and multiword expressions; the latter consist of supervised [6], semi-supervised [7] and unsupervised [8] machine learning techniques that perform sentiment classification based on word co-occurrence frequencies. Among these, the most popular recently are algorithms based on deep neural networks, generative adversarial networks [9] and capsule networks [10]. There are also some hybrid frameworks that leverage both symbolic and sub-symbolic approaches, e.g., sentic computing [11]. Most machine learning based models use pre-trained word vectors are good at capturing analogy questions [12].

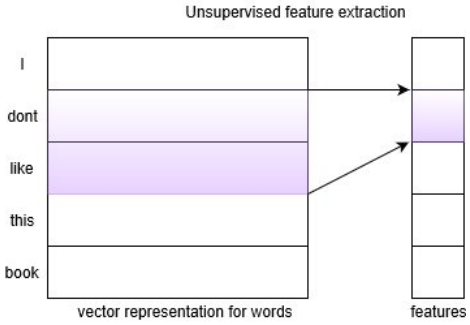


Fig. 1. Learning phrases in a sentence using convolution. Each word is represented by a pre-trained vector. A bi-gram feature 'dont like' is learned to classify negative reviews.

For example, a gender relation from man to women is same as one from king to queen. Another example is plural relation from king to kings is same as one from queen to queens in word vectors. For effective business decision-making, it is critical to keep track of all product and service reviews in real time as these get continuously posted online [13]. It would be computational slow to keep retraining the classifier using the entire dataset. To this end, we use online learning that can update the model with a single sample and then discard it from the memory. This is achieved by constructing a Gaussian kernel classifier where the mean and variance are updated using each new sample. Since it has no prior knowledge as to how many training observations will be presented, it can generalize better to sequential data [14].

While online learning has extremely low complexity, however the convergence to global optima is slow and the accuracy of the model is low. To overcome this problem, we use convolutional neural networks to extract significant phrases from the sentences [15]. In such a model, we are able to learn a dictionary of features that is portable across products and tasks [16]. In order to determine the context of words in different domains we use known word vectors of high-frequency bi-grams as a prior to the online update. For each domain we identify the most common bi-grams with positive and negative polarity. Word vectors are currently available only for a few bi-grams, hence we construct a prior based on these bi-grams.

2375-9259/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDMW51313.2020.00012

TABLE I
CHANGE IN POLARITY OF SENTENCES IN A SINGLE REVIEW OVER TIME. WEAK REVIEWS HAVE MIXED SENTIMENTS.

Review Polarity	Sentence Polarity	Sentence
Negative	Neutral	I have bought and returned three of these units now
	Negative	Each one has been defective, and finally I just gave up on returning the system
	Negative	The DVD player constantly gives "Bad Disc" errors and skips if there is even the slightest smudge on a disc
Positive	Positive	It's speedy and space saving and inexpensive
	Neutral	I bought this to replace my Belkin because the Belkin needed to be plugged in
	Positive	This one is powered by your computer so there's no extra power cords, which is a big plus to me

The organization of the paper is as follows: Section III describes the CDBN model and how a RNN is used to model a sequence of sentences; next, Section IV describes online learning of non-stationary sentences; Section V validates the proposed method on two benchmark datasets; finally, Section VI proposes concluding remarks.

II. RELATED WORK AND CONTRIBUTIONS

Traditional machine learning models need to use all the available training data to create a classifier. Hence, if new training data is provided the model has to be re-trained again with the update dataset. This is extremely slow for microblogs such as Twitter where people are constantly posting new reviews every second. In contrast, online learning models update the model using only a single new samples and discard it from memory. Lifelong learning is another batch learning algorithm for multi-task sentiment problems [17]. Here, the model learned on the first task is used as a prior for the second. The second is used as a prior for the third and so on. Hence lifelong learning still requires the entire data for a domain to be available during training.

To improve the accuracy of online learning, in [18], the authors introduced a new type of regularization that considers the average weight learned in all iterations. Their method required a lot of memory and was computationally slow. We address this problem by reducing the dimensionality of the sentences using convolutional kernels. Another challenge while training such as model is that reviews from different products may have opposite polarity for the same words. In order to create a generalized classifier, a semi-supervised classifier was proposed in [19]. Since their model used offline learning it is not suitable for live Twitter streams.

Another author in [20], applied online learning to word vector model of sentences. Since it is difficult to understand the difference in word vectors of words such as 'actor' and 'actress', they consider a continuous bag-of-words model where context is represented by multiple words for a given target word. Such a model is not portable to new domains where the context of words may change. In contrast, in this paper we consider a convolutional deep belief network (CDBN) to learn word vectors of n-grams from the input sentences. For example, in Fig. 1 we can learn the feature 'dont like' using a kernel of width two.

The content posted Online is often sequential. For example, a long product review is made of many individual sentences. We use a recurrent neural network (RNN) to remember the polarity of the previous sentence in a sequence. For an Online model it may not be suitable to use LSTM that captures long term dependencies. This is because the model parameters are changing rapidly. Hence, we feel a vanilla RNN is suitable.

Lastly, in unsupervised deep learning, each layer is trained independently of the layer below, thus over-fitting is avoided. For example, in [21] the authors showed that speech recognition accuracy improves and then saturates at eight hidden layers. Furthermore, online learning has much lower complexity as only a single data sample is used to update the model. The following is a summary of the significance and contributions of the research work presented in this paper:

- To improve the accuracy of online learning for sentiment prediction, we extract n-gram features using a CDBN for training the online classifier.
- To enable learning from unlabeled data we update weights in the deep model in a semi-supervised manner.
- We propose a prior based on word vectors of bi-grams with high frequency when updating the online model

Validation of the proposed method is performed on two real-world benchmarks taken from Twitter and Amazon. The Twitter dataset contains short tweets on Movie reviews classified as positive and negative [22]. The Amazon dataset consists of product reviews with multi-class labels such as 'very positive', 'neutral' and 'very negative'. The products are from four categories such as 'Books' and 'DVD' [23].

The proposed model outperforms the baselines in terms of accuracy by $5-20\%$ on both benchmark dataset. The CDBN has two stages of training. The first stage called pre-training is completely unsupervised. The unlabelled data samples can be used in this stage. The second stage called fine-tuning requires labelled samples. For the purpose of our experiments, we used the same dataset for both stages. This way, we could compare with previous authors. However, in practice the pre-training may be done with any unlabelled dataset.

III. CONVOLUTIONAL DEEP LEARNING

In this section, we begin with a description of the probabilistic sentence sequence model. Next, we describe restricted Boltzmann machines (RBM). Lastly, we describe how to stack RBMs in a hierarchical way where the hidden layer of one RBM serves as the visible layer of the next RBM, resulting in a deep belief network.

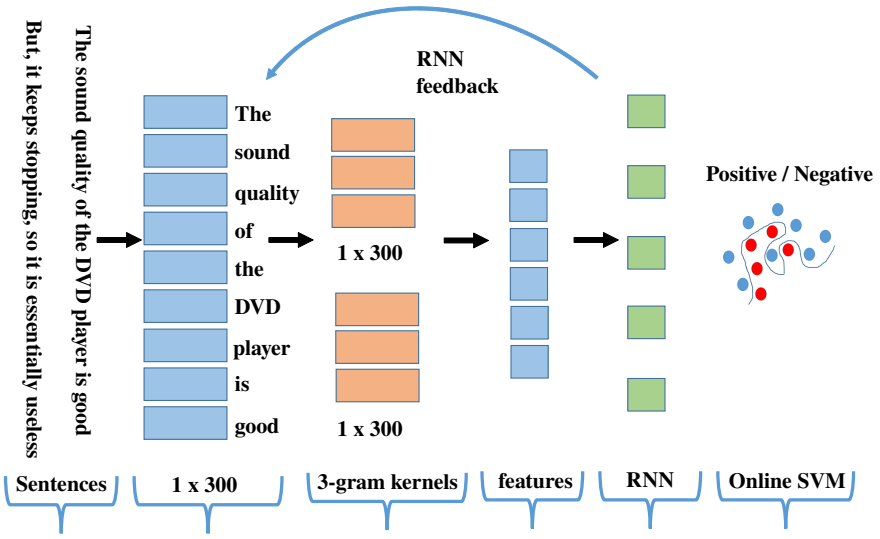


Fig. 2. State diagram of COAL learning. A sequences of sentences are input to the model. Each word is replaced by a pre-trained vector of length 300. Each kernel takes the shape of three words with length 300. Convolution results in extracting a feature vector where each neuron represents a significant tri-gram. These features are used to train an RNN classifier where the label is used to predict the next sentence using a feedback loop. Lastly, the RNN features are classified using an online SVM.

A. Sentence Model

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 $y(t) \in \{P^+, P, 0, N, N^-\}$ where 'P^+' is strongly positive, 'O' is neutral and 'N^-' is strongly negative review. 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:

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

The word vector of each sentence is randomly initialized to a vector of d dimension. Next, the word-vector model represents each word as a d dimensional vector that is computed from co-occurrence data using Eq(1). A larger value of d will be more accurate and computationally slow. When we concatenate the word vectors of all words in a sentence of length L, it results in a 2D input vector of dimension $\mathrm{L} \times d$. In the next section, we describe convolutional learning to learn n-gram features from this transformed 2D input.

B. Convolutional Deep Belief Network

A RBM [24] is a shallow neural model consisting of two layers (known as visible and hidden layers) organized in a bipartite graph. To compute the weights W of an RBM, we assume that the probability distribution over the input vector $\mathbf{x}$ is given as :

$$p(\mathbf{x}|W) = \frac{1}{Z(W)} \exp^{-\mathrm{E}(\mathbf{x};W)} \quad (2)$$

where $Z(W) = \sum_{\mathbf{x}} \exp^{-\mathrm{E}}(\mathbf{x}; W)$ is a normalization constant. Computing the maximum likelihood is difficult as it involves solving the normalization constant, which is a sum of an exponential number of terms.

To learn these weights of each RBM layer and maximize the global energy function efficiently, the approximate maximum likelihood Contrastive Divergence (CD) [25] approach can be used which minimizes the difference of two Kullback-Leibler (KL) divergences given by:

$$\mathrm{CD} = \mathrm{KL}(\mathcal{N}(\mu_{t+1}, \Sigma_{t+1}) || \mathcal{N}(\mu_\infty, \Sigma_\infty)) - \mathrm{KL}(\mathcal{N}(\mu_t, \Sigma_t) || \mathcal{N}(\mu_\infty, \Sigma_\infty)) \quad (3)$$

$$\text{where } \mathrm{KL}(p||p_\infty) = \sum\nolimits_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{p(\mathbf{x}; W)}$$

where the weights W of n neurons follow a normal Gaussian distribution with mean vector $\mu \in \mathcal{R}^n$ and covariance matrix $\Sigma \in \mathcal{R}^{n \times n}$. In practice, this method employs each training sample as the starting state vector for the visible layer of an RBM. For subsequent steps, a reconstruction of the visible layer is produced using the layer weights and the state of hidden neurons. Now the hidden layer is updated using this reconstructed visible layer. If this Gibbs is repeated, after a sufficient number of times, it is possible to converge to the equilibrium.

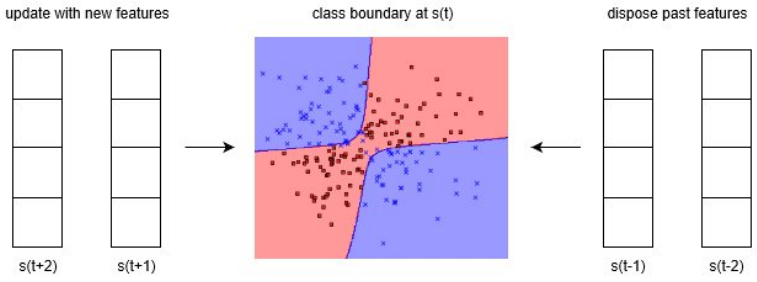


Fig. 3. Online Adaptation of Features in a Sentence. Here each the classification boundary is updated using each sentence $s(t)$ at time index t. Next, we dispose of the features for sentence before updating the model using the next sentence $s(t+1)$.

The 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:

$$\hat{h}_j = b_j + \sum_i v_i w_{ij}, \tag{4}$$

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:

$$h_j = \frac{1}{1 + e^{-\hat{h}_j}} \tag{5}$$

Similarly, in the next iteration, the binary state of each visible node v_i is reconstructed.

Lastly, the weights w_{ij} are updated as the difference between the original and reconstructed visible layer labeled as the vector $\boldsymbol{v}_{recon}$, using:

$$\triangle w_{ij} = \alpha(< v_i h_j >_{data} - < v_i h_j >_{recon}) \tag{6}$$

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 Eq 4. It is possible to create a Convolutional RBM (CRBM) naturally extending a traditional RBM in 2 dimensions using the convolution operation [26]. If we stack many layers of CRBMs, it is possible to create to a CDBN, where we simply partition the hidden layer into Z groups. The energy function of layer l is now a sum over the energy of individual blocks given by:

$$\mathrm{E}^l = -\sum_{z=1}^{\mathrm{Z}} \sum_{i,j}^{\substack{(\mathrm{L_x}-n_x+1),\\(\mathrm{L_y}-n_y+1)}} \sum_{r,s}^{n_x,n_y} v_{i+r-1,j+s-1} h_{ij}^z w_{rs}^l. \tag{7}$$

Lastly, after the convolution we do pooling to remove redundant or highly similar features. Fig. 2 illustrates CRBM how sentences are classified using convolution. Here each sentence is represented by a vector of length 300. The kernels have length 300 and width 3 to capture three consecutive words in a sentence. The CRBM is first pre-trained in an unsupervised manner followed by fine-tuning using supervised labels.

C. Recurrent Neural Networks

In the last few years, RNN in general received much attention due to their temporal memory and their ability in solving sequence problem. The CDBN described in the previous section will extract bi-grams and tri-grams in a single sentence. RNN will remember the polarity of the previous sentence while classifying the current sentence. This is because a product review is made up of many individual sentences of different polarity.

The standard RNN output, $\mathbf{y}(t)$, at time step t is calculated using the following equations:

$$\mathbf{y}(t) = f(W_{\mathrm{R}} \cdot \mathbf{h}(t-1) + W_l \cdot \mathbf{n_h}(t)) \tag{8}$$

where W_{R} is the interconnection matrix among hidden neurons $\mathbf{h}$, W_l is the weight matrix that connect hidden neurons to the input nodes $\mathbf{n_h}$ and f is the activation function. Final output of the RNN is a compact sentence representation n_r, which is independent of the number of n-grams present in the sentence itself. Fig. 2 illustrates how the CRBM features are used to train an RNN. The label for each sentence is used to predict the next using memory states in the RNN.

TABLE II
TOP BI-GRAMS IN KITCHEN AND ELECTRONICS DOMAIN

	Positive	Negative
Kitchen	perfect size	very disappointed
	good quality	poorly designed
	works great	dont buy
	cutting board	stopped working
Electronics	highly recommended	stopped working
	good quality	main problem
	love it	tech support
	works well	doesnt work

IV. CONVOLUTIONAL ONLINE LEARNING

In this section, we introduce the online learning model for classifying sentences. Next, we describe our proposed framework for reducing the dimensionality of input to online learning using deep learning. The resulting model is called convolutional online adaptation learning (COAL).

A. Online Learning for Sentences

The traditional machine learning model such as support vector machine (SVM) learns the classification boundary using all the training samples. Such a model over-fits to the training data, hence we need to retrain for each domain. Instead online learning recursively updates the classification boundary using a single sample each time. This is achieved by using a sampling approach to update the weights from a Gaussian distribution.

Fig. 3 illustrates the online adaptation of features in a sequence of sentences. The classification boundary after training with sentence $s(t)$ is also shown. The previous sentence in the sequence at $s(t-1)$ is discarded after training. The next sentence $s(t+1)$ will be used to update the classification boundary iteratively. For each sentence $s(t)$ in a sequence, a SVM uses the corresponding target labels $y(t) \in \{+ve, -ve\}$ to optimize a dual form objective function with both min and max terms:

$$\max_{\beta} \min_{\alpha} \frac{1}{2} \sum_{i=1}^{T} \sum_{j=1}^{T} \alpha_i \alpha_j y(i) y(j) \left(\beta K(x(i), x(j)) \right) - \sum_{i=1}^{T} \alpha_i, \text{s.t} \sum_{i=1}^{T} \alpha_i y(i) = 0,\ 0 \leq \alpha_i \leq C \forall i. \quad (9)$$

where the positive definite Gaussian kernel is denoted by $K(x(i), x(j))$ in each modality with a set of different parameters and α_i, b and $\beta \geq 0$ are coefficients to be learned simultaneously from the training data using quadratic programming.

Next, we consider online learning as a binary classification task. For the multi-class case, we adopt a one-vs-all strategy. The learner at time t learns from a sequence of training instances. A binary online classifier learns from a sequence of training instances, $\mathbf{x}_t$ is d dimensional input vector at time instance t and $y_t \in \{-1, +1\}$ is the true class label. Since online SVM is binary, we use a one-against-all strategy to deal with multi-class data. The most popular kernel is the Gaussian distribution for weights. Hence, we have to specify both the mean and the variance of the distribution.

We assume that the weights follow a normal Gaussian distribution where kernels are determined as $K(x(i), x(j)) \sim \mathcal{N}(\mu, \Sigma)$ then at each time point we have to update the parameters as follows:

$$\mu_{t+1} = \mu_t - \eta \Sigma_t \circ (-y_t \mathbf{x}_t) \quad (10)$$
$$\Sigma_{t+1} = \Sigma_t - \frac{\Sigma_t \circ \mathbf{x}_t \circ \mathbf{x}_t \circ \Sigma_t}{\gamma + (\mathbf{x}_t \circ \Sigma_t)^{\mathrm{T}} \mathbf{x}_t}$$

where $\circ$ is the element-wise multiplication operator and γ, η are positive parameter to avoid over-fitting. Instead of using the entire dataset $\mathbf{x}$ is used to compute the new distribution $\mathcal{N}(\mu_{t+1}, \Sigma_{t+1})$, here only a single new sample $\mathbf{x}_t$ is needed.

B. Online Domain Adaptation

When we update the online model with a single sample the model will become biased. Since, the context of words in different domains may be different we consider a prior word vector a sample from a particular domain and class.

Algorithm 1 COAL Training

```
Initialize : μ_1 = 0, Σ_1 = I
repeat
    Receive x_t
    if y_t ∈ {−1, +1} then
        Update weighs using Deep Learning Eq(6)
        Update weights using Online Learning Eq(10)
    else
        Update weighs using Deep Learning Eq(6)
    end if
until t⩽T
```

The word vectors for the top k bi-grams for each domain and each class are extracted from the available training data. Next, we introduce a modified update for the mean in Eq 10 as follows:

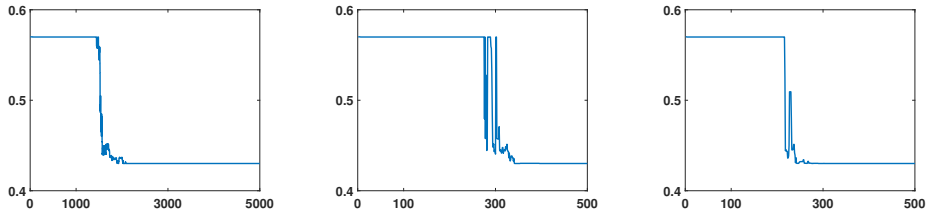
$$\mu_{t+1} = \mu_t - \eta \Sigma_t \circ (-y_t \mathbf{x}_t) + \hat{x}_a^y \quad (11)$$

where $\hat{x}_a^y$ is the word vectors for top bi-grams in domain a and for class label y. For example, the domain can be 'electronics' and the class label 'y' can be positive or negative review. Hence, Eq 10 shows the modified online learning update that uses the word vectors of bi-grams with high frequency as a prior. Table II illustrates the top bi-grams found in product reviews for 'kitchen' and 'electronics' domains. We can see that for positive and negative reviews slightly different words are commonly used. For example, 'electronics' may need 'tech support' and kitchen items have 'perfect size'.

For illustration, let us consider domain adaptation from kitchen to electronics domain. Here, the model is trained with a stream of product reviews from kitchen domain and tested on electronics domain. The traditional online SVM updates the classification boundary with each new sample. This is easily achieved by updating the mean and variance of the kernel function. Hence, we introduce a new domain specific prior during updating. Here, we use the word vectors of the bi-grams with high frequency in a specific domain and with a certain polarity. For example, in Table II, we see the bi-gram 'perfect size' frequently in positive kitchen reviews. This is not likely to be used in electronics. Similarly, for a negative review in electronics you will see the bi-gram 'tech support' very often.

Deep learning can update weights from both labeled and unlabeled samples. Hence, when receiving $\mathbf{x}_t$, deep learning uses contrastive divergence to update the weights if the label is unknown and back-propagation to update the weights if the label is known. Unlike online learning in regular supervised online learning, when receiving $\mathbf{x}_t$ COAL only updates weights if the label y_t is known. Otherwise, the algorithm ignores the sample and process the next sample.

To generate the final prediction based on the task aware n-grams embedding feature learned by the CDBN, we stack a SVM classifier on top of the network . In particular, we adopt the online SVM proposed in [27].



(a) (b) (c)

Fig. 4. Effect of aggressiveness parameter on learning (a) c=0.005 (b) c=0.05 (c) c=0.1

As shown in Fig. 2, a minimal CDBN with visible layer of $\mathrm{L_x} \times d$ nodes is constructed, where $\mathrm{L_x}$ is the max-length of the sentence and d is the word embedding dimension. Several hidden convolutional layers are applied. Such hidden layers filter the n-grams (here, we show bi-gram) generated form the sentence and are trained to maximize the log-likelihood of dataset. A hidden logistic layer of n_h neurons is used to flatten the embeddings generated by the convolutional kernels. These n_h features are used to train a RNN that models the dependencies between multiple sentences in a single product review. The last layer exploits the online SVM to generate sentiment aware embedding.

To determine the number of hidden layers in the CDBN, we compute the change in visible layer reconstruction error $\triangle\epsilon$ on the training samples. This is the root mean 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. Following [28], to determine the optimal number of hidden neurons in a single layer, we consider the number of components with eigenvalues above a threshold after during principal component analysis. Each hidden neuron in the final output layer corresponds to a particular sentiment class. The contrastive divergence approach samples features with high frequency into the upper layers, resulting in the formation of n-grams at hidden neurons in the first layer, bigger phrases at hidden neurons in second hidden layer and so on.

Algorithm 1, describes the COAL algorithm. We initialize the mean and covariance of the online SVM to 0 and I respectively, where I is the identity matrix. For each new data sample $\mathbf{x}_t$, if the label is known, we use online learning otherwise we only use deep learning to update the weights in an unsupervised manner.

V. Experiments

In this section, the proposed COAL (available on GitHub[1]) was applied to two real sentence classification problems in order to assess its efficacy. We have used pre-trained word vectors for English. We use line graphs to show the increase in accuracy of an online learning model as new samples appear. The results for other baselines are reported on the entire training data.

[1]http://github.com/senticnet/convolutional-online-adaptation-learning

We see that the proposed model first performs inferior to the baseline, however once sufficient training data is used it is able to outperform batch learning. Due to limited space we only provided comparison with the complete COAL model and the baselines. However, our experiments showed a significant reduction in accuracy if the prior word vectors are not used during online learning.

A. Parameter Settings

The experimental settings for SVM and CRBM are same as the baseline in order to allow comparison with their results. For example, word vector length is set to 300 and the learning rate is 0.01. For the RBF kernel parameter C in Eq 9 we notice that the model convergence is much faster with higher values of C until saturation. Fig. 4 shows that when we increase C from 0.005 to 0.05, the model needs only 300 samples to converge compared with 2000 samples. Further increase in C to 0.1 only marginally improves the convergence to 250 samples. Hence, we set C to 0.1 for all our experiments.

B. Multi-domain Sentiment Dataset

In this section, we verify the effectiveness of COAL in classifying subjective sentences using the multi-domain sentiment analysis dataset [23]. We first report the results on the binary problem of classifying reviews as positive (4 or 5) and negative (1 or 2). Following previous authors we removed the rating 3 reviews. This allowed us to directly compare with their results. The four domains consist of 'Books' (B), 'DVD' (D), 'Electronics' (E), and 'Kitchen' (K) reviews, where each domain contains 2000 reviews. Hence, as an illustration training data in the form of 1000 positive and 1000 negative reviews were taken. Each review was split into individual sentences resulting in over 10000 sentences for each domain.

We divided the reviews into train, validation and test sentences in the ratio of 60:20:20. Each review was split into a sequence of sentences. All sentences in a single review were assigned the same label corresponding to the star rating for the review. We discard reviews with rating 3 as they are ambiguous. We construct 12 cross-domain tasks of sentiment classification on this dataset. Here, 2000 reviews in one domain are the training data and 2000 reviews in a different domain are the test data. Each review is further split into individual sentences resulting in over 8000 training sentences.

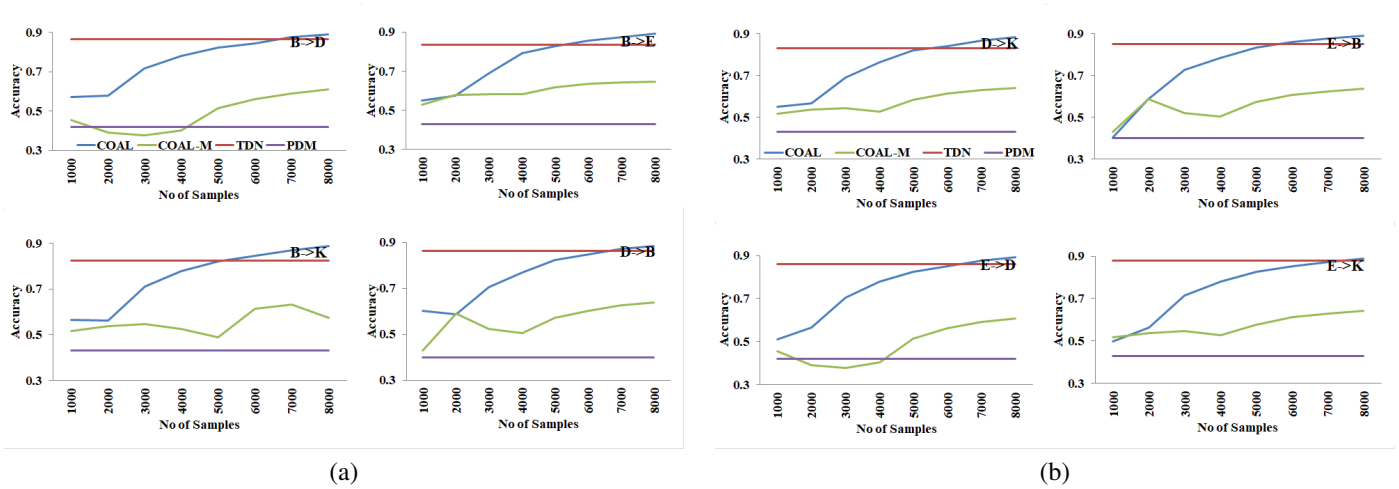


Fig. 5. (a) Accuracy of the 12 binary and multi-class transfer tasks on the multi-domain sentiment dataset (Tasks 1-4) (b) Tasks 5-8. Accuracy for TDN and PDM is using all 8000 samples.

Fig. 5 (a) and (b) and Fig. 6 (a) show the comparison for different methods and for different number of training sentences. We can see that accuracy increases with number of initial training samples and outperforms baseline transfer deep network (TDN) [29] in the range of [2-6]% when 8000 training samples are used. In TDN, the authors considered two parallel deep auto-encoders to learn transferable features and classification features. However, they do not use CDBNs, hence they are unable to capture the context of words. The highest improvement over baseline is for 'DVD→Electronics' (6%) and the lowest improvement is for 'Kitchen→Books'(2%). It is easy to see that the first pair of domains is very similar and the second pair is very dissimilar. Next, we consider the multi-class setting COAL-M, where each rating corresponds to a single class resulting in a 4-class problem.

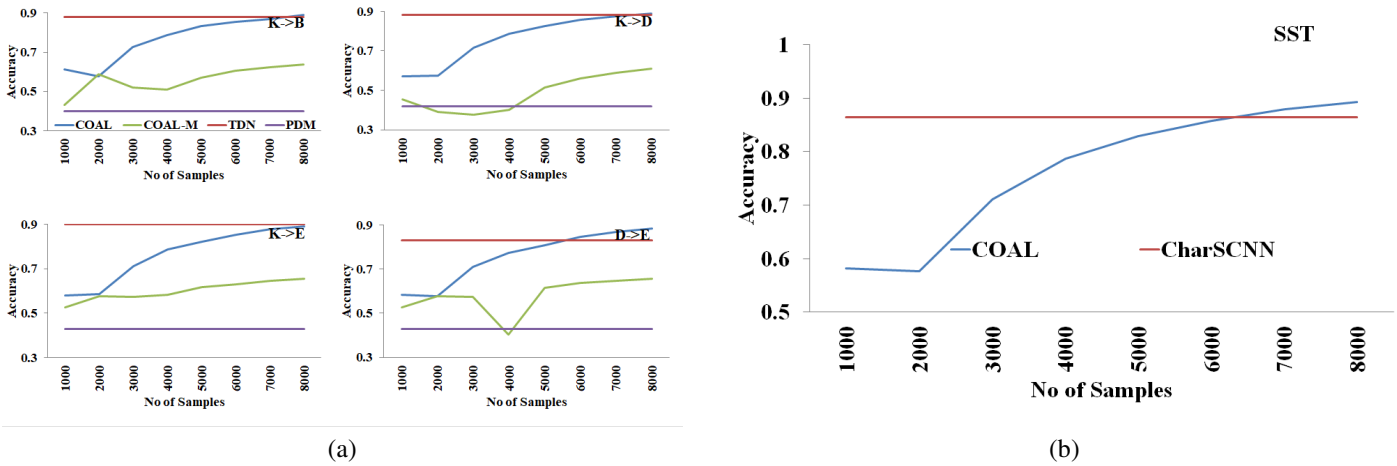
Fig. 5 (a) and (b) and Fig. 6 (a) show that accuracy of COAL-M increases with number of training sentences. We can see that accuracy increases with number of initial training samples and outperforms baselines Predictive Distribution Matching (PDM) [30] in the range of [5-20]% when 8000 training samples are used. This is because PDM uses a bag-of-words model that is unable to capture semantic context between words. Previous models for sequence of sentences assume that the data is stationary or the class labels of sentences are periodically repeating. However, in the real world this often untrue: class labels in a sequence of sentences changes dramatically with new product reviews. Online learning updates the parameters using each new sentence and hence is robust to such non-stationary data. For example, we see that the accuracy of the model COAL improves from 1000 to 8000 samples. The model learns from the new samples and is able to adapt the parameters. Even for the case of multi-class COAL-M the accuracy decreases slightly and improves as it sees new samples. In contrast, the baselines such as TDM and PDM assume that the parameters are constant and the accuracy does not change with increasing training data.

C. Short Text Dataset

Stanford Sentiment Treebank (SST) is a movie review dataset from Twitter. There is a need to better capture sentiment from short comments, such as Twitter data, which provide less overall signal per document. Here, we analyze performance on only positive and negative sentences, ignoring the neutral class. Following [22], the sentences in the treebank were split into a train (8544), validation (1101) and test splits (2210). Fig. 6 (b) shows the accuracy of COAL on SST dataset with increasing number of training samples. We can see that accuracy increases with number of initial training samples and outperforms baselines character-to-sentence CNN (CharSCNN) [26] by over 4% when 8000 training samples are used. CharSCNN combined character-level and word-level embedding to predict sentiment in short texts, however this is unable to capture the non-stationary data in online forums such as Twitter.

VI. Conclusion

To deal with the daily influx of new product reviews, we considered an online algorithm that learns incrementally from new reviews without the need for re-training. The features learned from CDBN are used to train the online classifier if the label is known. For new reviews without labels, the unsupervised contrastive-divergence approach is used to update the weights of the deep model. We evaluated our method on binary and multi-class sentiment prediction on the Twitter and Amazon dataset. The model is trained on short tweets or sentences in product reviews. It can be trained in one domain and used to predict sentiment in another domain. We could outperform the baseline in the range of $5-20\%$. The model was also robust to over-fitting compared to traditional deep learning models. Hence, the accuracy increases with the number of training samples.



(a) (b)

Fig. 6. (a) Accuracy of the 12 binary and multi-class transfer tasks on the multi-domain sentiment dataset (Tasks 9-12). Accuracy for TDN and PDM is using all 8000 samples. (b) Accuracy of the binary task on the short text sentiment dataset.

VII. Acknowledgements

This research is supported by the College of Science and Engineering at James Cook University.

References

[1] A. Picasso, S. Merello, Y. Ma, L. Oneto, and E. Cambria, "Technical analysis and sentiment embeddings for market trend prediction," *Expert Systems with Applications*, vol. 135, pp. 60–70, 2019.

[2] E. Cambria, M. Grassi, A. Hussain, and C. Havasi, "Sentic computing for social media marketing," *Multimedia Tools and Applications*, vol. 59, no. 2, pp. 557–577, 2012.

[3] E. Cambria, A. Hussain, T. Durrani, C. Havasi, C. Eckl, and J. Munro, "Sentic computing for patient centered applications," in *IEEE International Conference on Signal Processing*, 2010, pp. 1279–1282.

[4] Y. Ma, K. L. Nguyen, F. Xing, and E. Cambria, "A survey on empathetic dialogue systems," *Information Fusion*, vol. 64, pp. 50–70, 2020.

[5] E. Cambria, Y. Li, F. Xing, S. Poria, and K. Kwok, "SenticNet 6: Ensemble application of symbolic and subsymbolic AI for sentiment analysis," in *CIKM*, 2020, pp. 105–114.

[6] I. Chaturvedi, E. Ragusa, P. Gastaldo, R. Zunino, and E. Cambria, "Bayesian network based extreme learning machine for subjectivity detection," *Journal of The Franklin Institute*, vol. 355, no. 4, pp. 1780–1797, 2018.

[7] A. Hussain and E. Cambria, "Semi-supervised learning for big social data analysis," *Neurocomputing*, vol. 275, pp. 1662–1673, 2018.

[8] N. Ofek, S. Poria, L. Rokach, E. Cambria, A. Hussain, and A. Shabtai, "Unsupervised commonsense knowledge enrichment for domain-specific sentiment analysis," *Cognitive Computation*, vol. 8, no. 3, pp. 467–477, 2016.

[9] Y. Li, Q. Pan, S. Wang, T. Yang, and E. Cambria, "A generative model for category text generation," *Information Sciences*, vol. 450, pp. 301–315, 2018.

[10] W. Zhao, H. Peng, S. Eger, E. Cambria, and M. Yang, "Towards scalable and reliable capsule networks for challenging NLP applications," in *ACL*, 2019, pp. 1549–1559.

[11] E. Cambria, A. Hussain, C. Havasi, and C. Eckl, "Sentic computing: Exploitation of common sense for the development of emotion-sensitive systems," ser. LNCS. Springer, 2010, vol. 5967, pp. 148–156.

[12] E. Ragusa, P. Gastaldo, R. Zunino, M. J. Ferrarotti, W. Rocchia, and S. Decherchi, "Cognitive insights into sentic spaces using principal paths," *Cognitive Computation*, vol. 11, no. 5, pp. 656–675, 2019.

[13] I. Chaturvedi, R. Satapathy, S. Cavallari, and E. Cambria, "Fuzzy commonsense reasoning for multimodal sentiment analysis," *Pattern Recognition Letters*, vol. 125, pp. 264–270, 2019.

[14] N. Liang, G. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[15] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *ACL*, 2014, pp. 655–665.

[16] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *ICML*, 2008, pp. 160–167.

[17] Z. Chen, N. Ma, and B. Liu, "Lifelong learning for sentiment classification," in *ACL*, 2015, pp. 750–756.

[18] L. Xiao, "Dual averaging method for regularized stochastic learning and online optimization," in *NIPS*, 2009, pp. 2116–2124.

[19] L. Cheng and S. J. Pan, "Semi-supervised domain adaptation on manifolds," *IEEE TNNLS*, vol. 25, no. 12, pp. 2240–2249, 2014.

[20] Y. Yan, Q. Wu, M. Tan, M. K. Ng, H. Min, and I. W. Tsang, "Online heterogeneous transfer by hedge ensemble of offline and online decisions," *IEEE Trans. on Neural Networks*, vol. 29, no. 7, pp. 3252–3263, 2018.

[21] A. R. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[22] 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," in *EMNLP*, 2013.

[23] M. Dredze, K. Crammer, and F. Pereira, "Confidence-weighted linear classification," in *ICML*, 2008, pp. 264–271.

[24] F. Asja and I. Christian, "An introduction to restricted boltzmann machines," in *Iberoamerican Congress on Pattern Recognition*. Springer, 2012, pp. 14–36.

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

[26] C. N. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *COLING*. ACL, 2014, pp. 69–78.

[27] S. Hao, P. Zhao, J. Lu, S. C. H. Hoi, C. Miao, and C. Zhang, "SOAL: second-order online active learning," in *ICDM*, 2016, pp. 931–936.

[28] M. Tanaka and M. Okutomi, "A novel inference of a restricted boltzmann machine," in *ICPR*. IEEE Computer Society, 2014, pp. 1526–1531.

[29] M. Long, J. Wang, Y. Cao, J. Sun, and P. S. Yu, "Deep learning of transferable representation for scalable domain adaptation," *IEEE Trans. on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2027–2040, 2016.

[30] C. Seah, Y. Ong, and I. W. Tsang, "Combating negative transfer from predictive distribution differences," *IEEE Trans. on Cybernetics*, vol. 43, no. 4, pp. 1153–1165, 2013.