Deep Multimodality Model for Multi-task Multi-view Learning

Lecheng Zheng [*] Yu Cheng [†] Jingrui He[*]

Abstract

Many real-world problems exhibit the coexistence of multiple types of heterogeneity, such as view heterogeneity (i.e., multi-view property) and task heterogeneity (i.e., multi-task property). For example, in an image classification problem containing multiple poses of the same object, each pose can be considered as one view, and the detection of each type of object can be treated as one task. Furthermore, in some problems, the data type of multiple views might be different. In a web classification problem, for instance, we might be provided an image and text mixed data set, where the web pages are characterized by both images and texts. A common strategy to solve this kind of problem is to leverage the consistency of views and the relatedness of tasks to build the prediction model. In the context of deep neural network, multi-task relatedness is usually realized by grouping tasks at each layer, while multi-view consistency is usually enforced by finding the maximal correlation coefficient between views. However, there is no existing deep learning algorithm that jointly models task and view dual heterogeneity, particularly for a data set with multiple modalities (text and image mixed data set or text and video mixed data set, etc.). In this paper, we bridge this gap by proposing a deep multi-task multi-view learning framework that learns a deep representation for such dual-heterogeneity problems. Empirical studies on multiple real-world data sets demonstrate the effectiveness of our proposed Deep-MTMV algorithm.

Keywords:
Multi-view Learning, Multi-task Learning, Deep learning

1 Introduction

In contrast to the single view or single task in a traditional classification setting, it is usually the case that many real-world problems have multiple views or multiple tasks or both of them. For example, in web classification problems, each web page can be characterized by multiple sources, including web title, web links, content in the web, etc. Each source can be considered as one view and usually contains the complementary information to each other. In image classification problems, the classifiers could learn to distinguish the domestic animals from the wild animals and also to classify the object in the image to be a cat or a dog, while the different views could be the distinct poses of the same animal.

Up until now, some researchers have proposed a variety of techniques to model a single type of heterogeneity. For example, in multi-view learning, [25] proposed an undirected graphical model to minimize the disagreement between multi-view classifiers; [29] follows the principles of view consistency by regularizing the prediction tensor. in multi-task learning, the intuition is that tasks usually share the same structure, such as the tree structure in [11], the clustered structure in [9] , etc. However, for the real-world problems that exhibit view and task dual heterogeneity, only making use of the techniques from multi-view learning or from multi-task learning is not able to achieve the optimal performance. To address this problem, [7] proposed a graph-based framework for multi-task multi-view learning (M2TV) that models both types of heterogeneity to help classify the unlabeled data. [17] proposed multilinear factorization machines, which can capture the relationships between multiple tasks with multiple views by constructing the task-view shared multilinear structures and learn the task-specific feature map. Despite the fact that these algorithms can deal with textual data very well, they fail to capture the spatial information of image data by just vectorizing the images.

Recently, deep learning techniques have been successfully applied to model various types of data, such as image data [15, 18] and text data [12, 20] with significantly improved performance and important features extracted in an automatic way. For example, in deep multi-view learning, the authors of [4] showed that a common feature representation of different views can be created by minimizing the loss in this unified feature space; in deep multi-task learning, a cross-stitch network proposed in [20] aimed to find the relatedness of two tasks in almost each hidden layer. However, to the best of our knowledge, there does not exist a deep learning algorithm for modeling view and task dual heterogeneity. In other words, existing deep neural network structures only take into consideration task or view heterogeneity, and cannot be naturally extended to model both types.

To bridge this gap, we propose a deep multi-task multi-view learning framework that can model multi-modality data. The key idea is that for different views, we construct a different neural network with one unit per layer at the beginning based on the data type (see Figure 1 for the architecture of the proposed model), and the complementary and consen-

[*] Arizona State University.Email: {lzheng43, jingrui.he}@asu.edu
[†] Microsoft AI & Research.Email: yu.cheng@microsoft.com

sus principles between these views are enforced by adding a regularization layer to constrain the output of multiple neural networks to be consistent. To integrate the output of these neural networks for multi-modality data, the weight of each view is automatically learned in the regularization layer and these weights are used to measure the contribution of each view to the final output. For different tasks, we group related tasks or attribute classifiers starting from the output layer to the input layer based on the similarity among tasks. Combining these two aspects, we propose an iterative algorithm to obtain the optimal estimates of the model parameters. Our main contributions are summarized below:

- A novel deep heterogeneous learning framework addressing task and view dual heterogeneity;
- A generalized deep learning framework for modeling multi-modality data;
- A regularization layer designed to maximize the consistency of multiple views;
- Experimental results on several data sets demonstrating the effectiveness of the proposed framework.

The rest of this paper is organized as follows. A brief review of the related work is discussed in Section 2. Then we introduce our proposed framework for deep multi-view multi-task learning in Section 3. In Section 4, we evaluate our framework on multiple data sets. Finally, we conclude the paper in Section 5.

2 Related Work

In this section, we briefly review the related work on multi-view learning, multi-task learning, multi-view multi-task learning, as well as convolutional neural network (CNN).

2.1 Multi-view Learning Multi-view learning has been studied for decades. [22] proposed Co-regularization method to jointly regularize two Reproducing Kernel Hilbert Space H^1 and H^2. [1] proposed Deep Canonical Correlation Analysis, which aims to find two deep networks such that the output layers of two networks are maximally correlated. In addition, multi-view Clustering (MVC) is another popular method used in unsupervised and semi-supervised learning and it aims to find several clusters such that similar data points are assigned to the same cluster and dissimilar data points are assigned to the different cluster by combining information from multiple views. [13] proposed a co-regularized multi-view clustering method by minimizing the disagreement between any pair of views. In this paper, we consider different tasks or attributes classifiers as data points and group these tasks by implementing multi-view clustering approach based on the similarities between tasks.

2.2 Multi-task Learning In parameter-based multi-task learning, task clustering approach and task relation learning approach are the most common strategies used to group tasks [24, 26]. The authors of [28] proposed a multi-task learning algorithm called CMTL, which assumes that each task can learn equally well from any other task. In feature-based multi-task learning, it assumes that different tasks share the same feature representation derived from the original feature under the regularization framework [2]. In deep multi-task learning, [18] used top-down layer-wise widening method to split one unit layer into several branches and group tasks in this layer based on the affinity of tasks. [23] proposed the tasks-constrained deep convolutional network method to formulate a task-constrained loss function, back-propagate the errors of related tasks jointly, and thus, improve the generalization of landmark detection. In our paper, we combine the relatedness of tasks from multiple views to determine how tasks are clustered in a more precise way.

2.3 Multi-view Multi-task Learning To cope with some real-world problems involving multiple views and multiple tasks, some researchers proposed to jointly model the two types of data heterogeneity. For example, [14] proposed spatio-temporal multi-task multi-view learning framework to predict the urban water quality, which fuses the heterogeneous data by penalizing the disagreement among different views and capture the spatial correlation among tasks by a graph Laplacian regularizer; [30] seeks to find a weight tensor to represent the worker's behaviors across multiple tasks by exploiting the structured information. [10] proposed a method to learn the feature transformation for different views by classical linear discriminant analysis [6] and explore the shared task-specific structure for different tasks. However, most of these methods are only good at dealing with one type of data and they might deteriorate with another type of data. For example, the approach proposed in [7] has good performance for text data but it ignores the spatial information of image data by just vectorizing the images. In contrast, in this paper, we construct different types of neural networks for distinct data types, utilize the complementary information among different views, and exploit the relatedness of tasks to improve the performance of our proposed method.

2.4 Convolutional Neural Network Two types of CNN are widely used for two types of data, including two-dimensional CNN for image data and one-dimensional CNN for text data. VGG-16 [21] is one of the most famous two-dimensional CNN (2d CNN) architectures that are widely used to solve image classification problems. Different from two-dimensional CNN, [12] proposed a model based on one-dimensional convolutional neural network (1d CNN) for sentence classification. At first, word2vec [19] is applied to find word embedding and each word has its own word vector in the feature space with dimensionality $R^{1\times d}$. Then, each documentation can form a matrix $M \in R^{k\times d}$ by concatenating words in the documentation together, where k is the maximal number of words in all documentations. N-grams are realized by training the different sizes of kernels. For example, a kernel $K \in R^{2\times d}$ can extract a bi-gram. In this paper, we use 1d CNN to extract N-grams from text data and 2d CNN for image data.

3 The Proposed Deep Multi-view Multi-task Learning

In this section, we introduce our proposed deep framework for multi-view multi-task learning , which is able to simultaneously address multi-modality data.

3.1 Preliminaries In this subsection, we briefly review the existing work of [18], which paves the way to our proposed framework. More specifically, the authors of [18] proposed an adaptive layer-wise widening model to automatically learn a multi-task architecture based on a thin version of the VGG-16 network [21]. The core procedure is to incrementally widen the layers with d branches by grouping the tasks based on the affinities of tasks, where d is the number of clusters. The authors defined A to be the affinity matrix, i, j the task index, E the expectation, and k, l the branch index. The error margin is defined as $m_i^n = |t_i^n - s_i^n|$, where t_i^n is the binary label for task i at example n and s_i^n is the prediction. The affinity of each pair of tasks i, j is defined as $A(i,j) = P(e_i^n = 1, e_j^n = 1) + P(e_i^n = 0, e_j^n = 0) = E\{e_i^n e_j^n + (1-e_i^n)(1-e_j^n)\}$, where e_i^n is an indicator variable for task i at example n. The indicator variable is set to be 1 if m_i^n is greater than the average error margin $E\{m_i\}$. To compute the affinity of two branches k, l connecting to the current layer, the authors denoted i_k and j_l as the i^{th} and j^{th} tasks in k and l branches respectively. The affinity of two branches is defined by $A(k,l) = mean_{ik}(\min_{jl} A(i_k, j_l))$ and $A(l,k) = mean_{jl}(\min_{ik} A(i_k, j_l))$. The final branch affinity score A^f is the average of two affinities k and l:

$$A^f(k,l) = (A(k,l) + A(l,k))/2. \tag{3.1}$$

After getting the affinity matrix A^f, the authors performed spectral clustering to obtain a grouping function $g_d : [c] \to [d]$, which means c old branches can be assigned to d clusters. In order to determine the optimal number of branches, the authors minimized the following loss function:

$$L^l(g_d) = (d-1)L_0 2^{p_t} + \alpha L_s(g_d) \tag{3.2}$$

where the first part is a penalty term for creating branches at layer l, the second part is the penalty for separation defined as: $L_s^i(g_d) = 1 - mean_{k\in g^{-1}(i)}(\min_{l\in g^{-1}(i)} A_f(k,l))$ and α is a positive parameter. In our proposed method, we use the same method to approximate the similarities of tasks, but we target the more complex scenario with multiple views instead of a single view. In addition, different from the the fully adaptive layer-wise widening model whose input data is limited to a single modality (i.e., image data), our model is able to handle multi-modality data, such as text, image, video, etc.

3.2 Deep MTMV Now we are ready to introduce our proposed framework. The main idea of the proposed model is to utilize the label information from the training data as well as the consistency among different views to help classify the test data. Suppose that the data set has m views and T tasks. We denote $\mathcal{D_L} = \{X_i^j, Y^j\}$ as the training data set, where $X_i^j \in R^{n_j \times d_i}$ corresponds to the feature matrix of the i^{th} view and the j^{th} task, $Y^j \in R^{n_j\times 1}$ consists of the class label of the j^{th} task, n_j is sample size of the j^{th} task and d_i is the dimensionality of the feature space in the i^{th} view. We denote H to be a feature mapping of a neural network and $H_i(W_i, X_i^j, b_i)$ to be the output of the i^{th} view shared by T tasks at the beginning with weights W_i and biases b_i. To combine the feature mappings of multiple views, we denote F to be the view fusion of multiple neural networks, which consists of several fully connected layers, and $F(\forall i, H_i(W_i, X_i^j, b_i)$[1] outputs the label in the last fully connected layer (Details will be discussed in the Section 3.3). Figure 1 provides the architecture of the framework, where we assume that the input consists of both image data and text data, although the proposed framework can be naturally generalized to handle additional data types. In this case, we construct two neural networks for two types of data, and $H_1(W_1, X_1^j, b_1)$ and $H_2(W_2, X_2^j, b_2)$ are the feature mappings of 2d CNN and 1d CNN (or Long Short-Term Memory, namely LSTM), respectively. $F(\forall i, H_i(W_i, X_i^j, b_i))$ combines the output of $H_1(W_1, X_1^j, b_1)$ and $H_2(W_2, X_2^j, b_2)$, and outputs the label. Generally speaking, suppose that the data set has m views and T tasks.

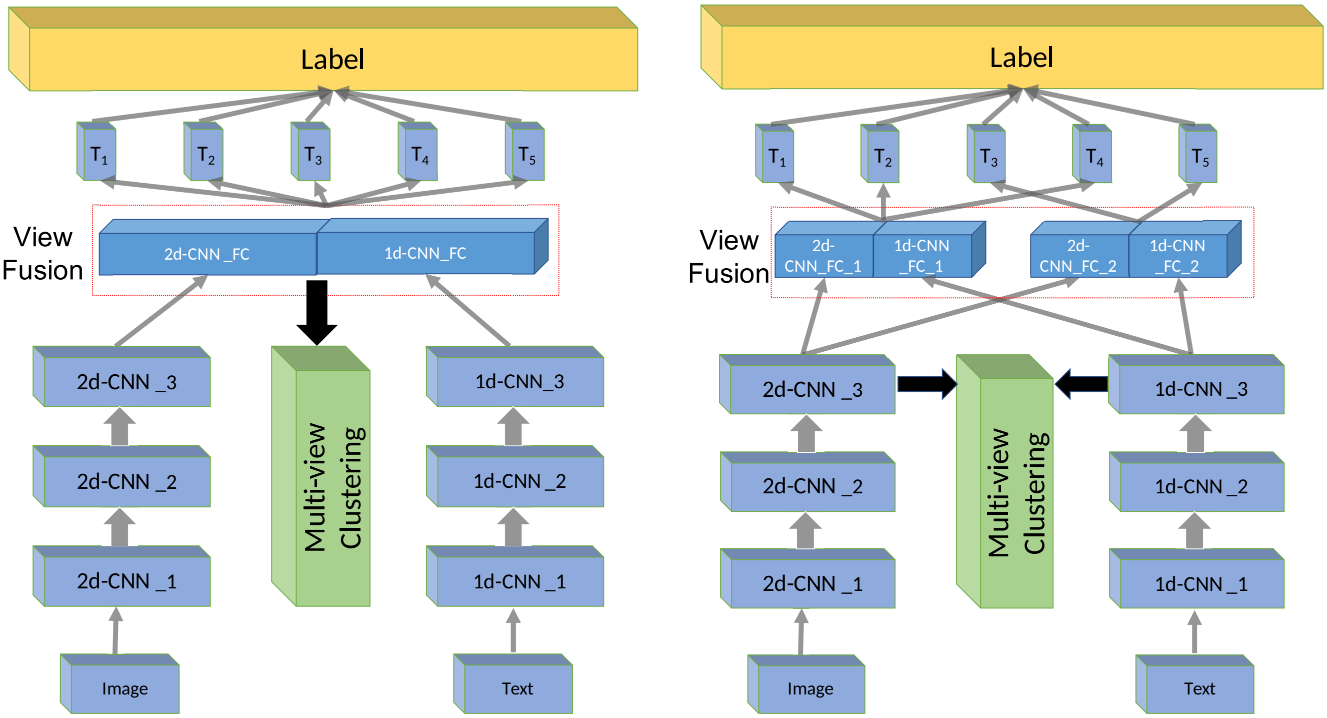
The cost function of our algorithm can be written as:

$$J = \frac{1}{2}\sum_j^T \left\| F_j(\forall i, H_i(W_i, X_i^j, b_i)) - Y_j \right\|_2^2 + \sum_{i=1}^m \lambda_i \left\| W_i \right\|_2 \tag{3.3}$$

where λ_i is a positive parameter. The objective of the proposed model is to fuse multiple views together, and to learn the weights of different views automatically. The relatedness of tasks is exploited to improve the performance by applying the multi-view clustering method.

3.3 Regularization Layer In multi-view learning, co-training [3] is a commonly used method by utilizing the con-

[1] we slightly abuse the notation and it will be explained in later section.



(a) Round 1

(b) Round 2

Figure 1: Suppose we are provided an image and text mixed data set with five tasks. We construct a 2d convolutional neural network (2d CNN) for image data and 1d CNN (or LSTM) for text data. In round 1, we aim to find the relatedness of tasks and multi-view clustering method is applied to decide how many branches we need to create and how to group tasks. After we train two neural networks with two types of data, we compute the similarities of tasks and update the structure in the next round. At the beginning of round 2, we decompose the split layer 2d-CNN_FC into 2d-CNN_FC_1 and 2d-CNN_FC_2, and the split layer 1d-CNN_FC into 1d-CNN_FC_1, 1d-CNN_FC_2, respectively. The size of 2d-CNN_FC_1, 2d-CNN_FC_2 are still the same as the size of 2d-CNN_FC. Then, we assign tasks T_1, T_2, T_4 to 2d-CNN_FC_1 and 1d-CNN_FC_1, and tasks T_3, T_5 to 2d-CNN_FC_2 and 1d-CNN_FC_2 based on the clustering results. The filters or the kernels at the newly created branches (2d-CNN_FC_2 and 1d-CNN_FC_2) are initialized by directly copying from 2d-CNN_FC or 1d-CNN_FC. Next, we aim to find the similarities of the branches (the first branch: 2d-CNN_FC_1 and 1d-CNN_FC_1, the second branch: 2d-CNN_FC_2 and 1d-CNN_FC_2) and split 2d-CNN_3 and 1d-CNN_3 to create more branches by repeating these procedures.

sistency principle to maximize the mutual agreement of label predicted by distinct views. In co-training, one uses multiple classifiers to predict the labels of the unlabeled data, adds the top k confident unlabeled data into the training data set and repeats this procedure until all the unlabeled data has been added into the training set. However, in some situations, we cannot assume that each view has equivalent contribution to the prediction. Take web classification problem as an example, multiple views in this classification problem are the content of the web page, the title of the web page and the link within the web page. Obviously, the content of the web page contributes most to the prediction, while the title of the web page and the link within the web page have less contribution to the prediction. If the prediction made by the classifier trained on either the title of the web page or the link within the web page is used to label the unlabeled data, the prediction may not be as accurate as the prediction made by the classifier trained on the content of the web page. Therefore, assuming the equivalent contribution may result in a worse performance in some scenarios like this.

To overcome this issue, we proposed multi-view fusion to automatically learn the weight of each view that contributes to the prediction. Given m views, we have

$$F(H) = \sigma(W_F, H, b_F) \tag{3.4}$$

where $H = H_1(W_1, X_1^j, b_1) \oplus H_2(W_2, X_2^j, b_2)... \oplus H_m(W_m, X_m^j, b_m)$, the symbol $\oplus$ means concatenation, $\sigma(\cdot)$ is an activation function, and W_F and b_F are the weight and bias, respectively. In this equation, $H_i(W_i, X_i^j, b_i)$ can be considered as a feature mapping of the i^{th} view and the expectation of the weights determines how many percentages each view contributes to the prediction of the training data.

These weights are also used to determine which view is the centroid view when we apply the centroid multi-view clustering method to group similar tasks in the next subsection.

3.4 Layer Widening and Task Clustering Similar to the structure proposed in [18], our training algorithm consists of a procedure to widen the layers of neural networks in order to explore the relatedness of multiple tasks and to group similar tasks into the same clusters based on the similarity among tasks. Most deep models assume that the tasks share the same parameters at the first several layers and have their own parameters at the following layers [26]. In our case, we have multiple neural networks to start with, each of which is associated with a single view. Furthermore, it is usually the case that each neural network ends up with its own distinct structure when we update its structure separately by grouping tasks based on their similarities within a single view. For example, suppose that both task A and task B are assigned to the same cluster in the first neural network (i.e., the first view), but in the second neural network (i.e., the second view), task A and task B may be assigned to two different clusters. This phenomenon contradicts our intuition that these neural networks created by multiple views should have the identical layer structure.

To address this problem, motivated by [20], we propose to insert a cross-stitch network into our architecture, in order to learn task relatedness inside the hidden layers of the multiple neural networks for multiple views. In this way, we obtain a unified task grouping informed by multiple views instead of potentially inconsistent groupings from different views. More specifically, in our proposed model, we consider each task as a data point. After the training stage, we compute the task-similarity matrices and estimate the affinity of the tasks in m views according to equation 3.1. To update the structure of neural networks, we create d branches for a unit layer based on equation 3.2, initialize the weight of these newly created branches by directly copying from the split layer, and link the old branches from the previous layer to these newly created branches based on the result of the following clustering method. Because the weight of one view might be higher than other views, centroid-based co-regularized multi-view spectral clustering approach [13] is used to assign similar tasks to the same group and dissimilar tasks to different groups in each round. The intuition is that the underlying clustering would assign the corresponding task in each view to the same cluster [13]. Given m views, T tasks, and k clusters, we have

$$\max_{U_i} \sum_{i=1}^{m} (U_i' L_i U_i) + \sum_{i=1}^{m} \lambda_i tr(U_i U_i' U_c U_c') \quad (3.5)$$
$$s.t \quad U_i' U_i = I, 1 \leq i \leq m, U_c' U_c = I$$

where $L_i \in R^{T \times T}$ is the graph Laplacian of the i^{th} view , $U_i \in R^{T \times k}$ consists of the eigenvectors of the i^{th} view, $U_c \in R^{T \times k}$ consists of the eigenvectors from the most important view and λ_i is the weight of the i^{th} view. The normalized graph Laplacian L_i of the i^{th} view is defined as $L_i = D^{-\frac{1}{2}} A_i^f D^{-\frac{1}{2}}$, where A_i^f is the affinity matrix for the i^{th} view based on equation 3.1 and $D \in R^{T \times T}$ is diagonal matrix with $D_{\alpha,\alpha}$ to be the sum of the α^{th} row of A_i^f. The detailed approach to solve this optimal problem can be found in [13], and is omitted here for brevity. The optimal solution determines how tasks are clustered and how a layer is split.

Algorithm 1 Deep-MTMV

```
Input: The initialized model $M$, the total number of round $R$, the training data set $\mathcal{D}_{\mathcal{L}} = \{X_i^j, Y^j\}$ and the number of branches $b$.
Output: The well-trained model $M$.
Initialization: Load pre-trained model or randomly initialize the weights and biases of the model, and set t to be 0.
while $t \leq R$ and $b > 1$ do
    Step 1: Train the model $M$ with training data $\mathcal{D}_{\mathcal{L}}$.
    Step 2: Compute the affinity matrices about the tasks similarities for $m$ views based on equation 3.1.
    Step 3: Determine the number of clusters by multi-view clustering method based on equation 3.2 and 3.5.
    Step 4: Create branches and widen layers for $M$ based on the results of multi-view clustering.
    Step 5: $b \leftarrow$ the number of branches in the current layer.
    Step 6: $t \leftarrow t + 1$.
end
Train the model $M$ until convergence.
```

In addition, we can naturally extend this multi-view clustering method to accommodate the scenario where some views may be missing for some tasks as in [27]. Although for the missing views, the corresponding entries of some tasks in the affinity matrices would be unavailable, these missing similarities can be estimated by the corresponding similarities in other views. In Section 3.3, our model automatically learns the weights of different views, with which the missing entries of the affinity matrices can be approximated by averaging the similarities from the available views. Besides, the learned weights of different views can also be used to set the parameters λ_i during the multi-view clustering process.

3.5 Multimodality Model for text and image mixed data set As mentioned before, our proposed framework is able to take as input multimodality data. Next, we use text and image mixed data set to illustrate the key idea. Given two sources of data: image data and text data, we build a convolutional neural network for image data, and a 1d convolutional neural network [12] (or Long Short-Term Memory) for text data. Notice that the specific choice of the neural network for each data modality is orthogonal to the proposed framework. Furthermore, the text data is pre-processed by word2vec algorithm [19] to extract word embeddings as the input of 1d CNN. The vital features, such as unigram, bi-gram, and tri-gram, are extracted by the filters of CNN with different size, such as $K_1 \in R^{1\times d}$, $K_2 \in R^{2\times d}$, and $K_3 \in R^{3\times d}$, where d is the dimensionality of word2vec embeddings. At first, two neural networks are trained separately, and then the feature mappings extracted by the two neural networks are appended in the fully connected layers to predict the labels of test data. In addition to the data sets containing the same data type, such as CelebA [16], WebKB, we will present experimental results on a real-world data set, FamousFood, which contains two types of data, image and text, to evaluate the performance of our proposed framework.

3.6 The Proposed Algorithm Our proposed algorithm is presented in Algorithm 1. It takes a initialized model (which is obtained based on the initialization algorithm in [18] , training data, the number of branches, and the total number of rounds as inputs, and outputs the well-trained model. The algorithm works as follows. We first construct a neural network structure for each view at the beginning, train m neural networks with the training data after initialization and fuse these neural networks in the regularization layer to get a final model. Then, we compute the affinity matrices about the tasks similarities. After the number of clusters is determined by minimizing the loss of the multi-view clustering method, we create new branches and assign the similar tasks into the same branch and dissimilar tasks into the different branches. When the number of round reaches its maximal or the branches cannot be split, then stop updating the structure and train the model until convergence.

4 Experimental Results

In this section, we demonstrate the performance of our proposed Deep-MTMV algorithm in terms of effectiveness by comparing with state-of-the-art methods.

4.1 Data sets In this paper, we evaluate our proposed algorithm on the following data sets:

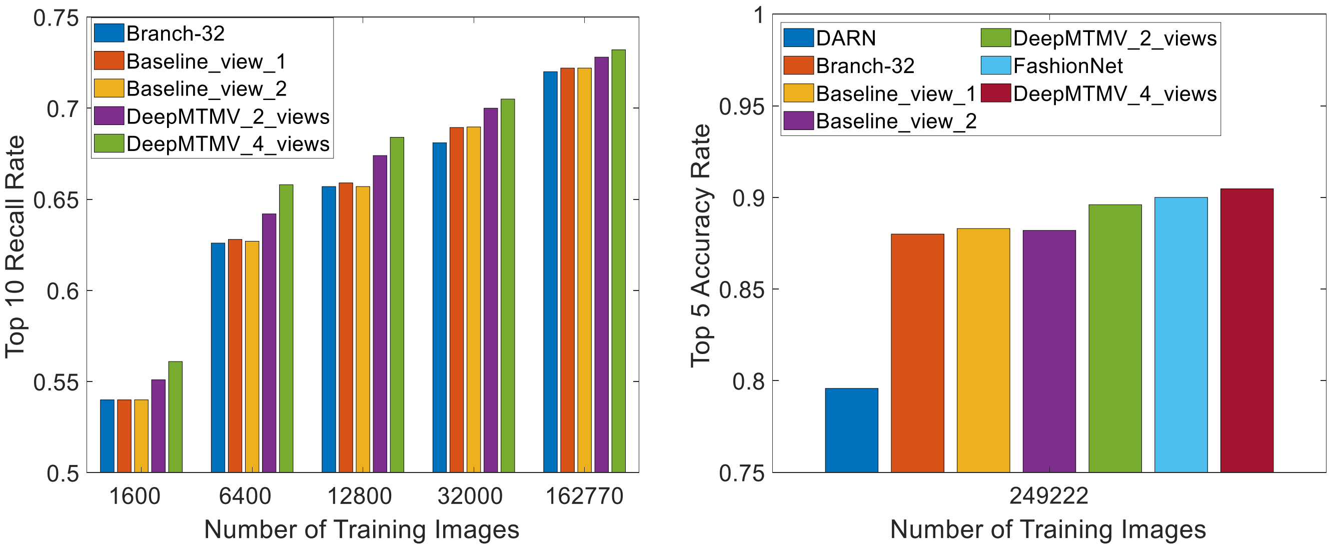
- **CelebA [16]**: It is composed of 202,599 images of celebrities, and 40 labeled facial attributes. Each attribute, such as black eye, brown eye, bald, is considered as one task in this classification problem. We extract two views or four views from each image in a way mentioned below. In our setting, we have 40 different tasks for 40 attributes and 2 (or 4) views.
- **Deepfashion [15]**: It consists of 50 categories and more than 289,222 images of clothes. Each category, such as hoodie and ramper, is considered as one task in this classification problem. We extract two views or four views from each image in a way mentioned below. In this setting, we have 50 different tasks for 50 different categories and 2 (or 4) views.
- **WebKB**: This is a textual data set, which consists of over 4000 web pages from 4 universities and includes 3 views, including the content of the web page, the title of the web page and the links within the web page. In our setting, each university is treated as a task and our goal is to classify each web page as course or non-course.
- **FamousFood**: In this data set, the images of famous food and the text of food description are crawled from the online photo sharing website Flickr. This data set contains 4 types of foods, which fall into 2 categories (sweet food or fast food), and each is considered as a task in our setting. Two different data source are image and the related text. For each food, it contains more than 450 images on average. In our setting, we have 6 (4 types of foods and 2 food categories) tasks and 2 views.

Pre-module	**Post-module**	**P value**
Branch-32	Deep-MTMV_4_views	1.01E-17
Baseline_view_1	Deep-MTMV_4_views	4.33E-56
Baseline_view_2	Deep-MTMV_4_views	9.24E-61
Deep-MTMV_2_views	Deep-MTMV_4_views	8.66E-10

Table 1: Student T Test with 95% Confidence Level

4.2 View extraction for two image data sets (CelebA and Deepfashion) In our experiments, we extract two views from a single image by splitting the width of each single image into two sets of indices: the even indices and the odd indices. Keeping the height of each image unchanged and combining all odd (even) indices together, we form the first (second) view. The way to get the four views is similar to the way to get the two views. Instead of only splitting the width of a single image into four views, we divide the single image into two parts both vertically and horizontally. By selecting the odd indices of the width and the odd indices of the height, we get the first view (and we can get the other three views in a similar way). The reason why we split the image this way is that we want to keep the views from overlapping.

4.3 Comparison methods In our experiments, we compare the performance of the following methods: (1). Our baseline model trained with view 1; (2). Our baseline model trained with view 2; (3). Branch-32 [18] (CelebA, DeepFashion and FamousFood data set); (4). FashionNet [15] (DeepFashion data set); (5). DARN [8] (DeepFashion data set);



(a) Top 10 recall for CelebA

(b) Top 5 accuracy for DeepFashion

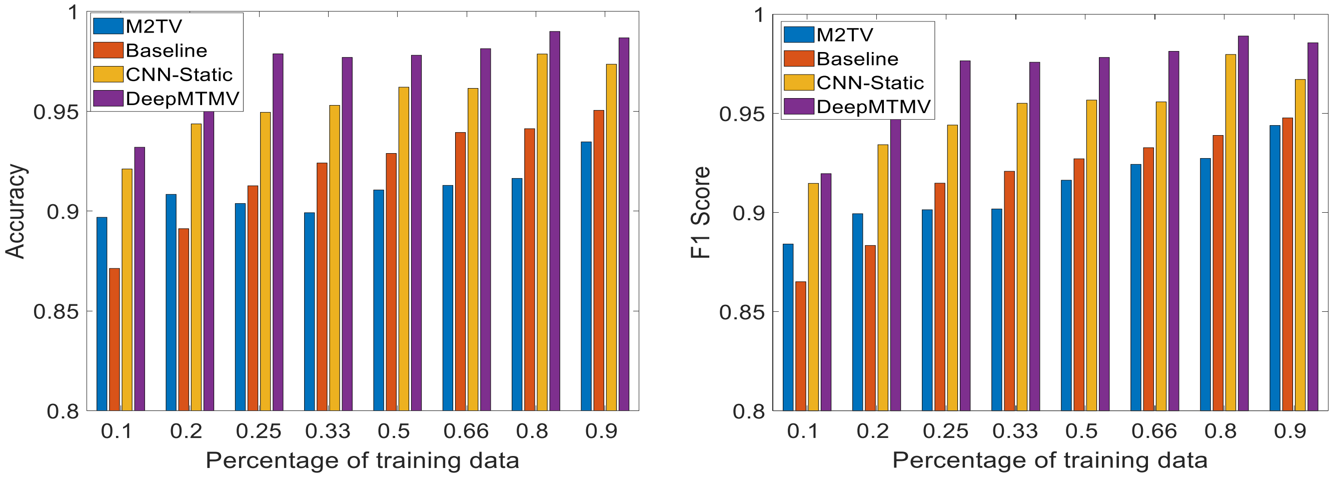
Figure 2: Effectiveness Analysis (Best viewed in color)

(6). M2TV [7] (WebKB data set and FamousFood data set); (7). CNN-Static [12] (WebKB data set).

Notice that some of these methods are best suited for certain data modalities. For example, Branch-32, FashionNet are designed to deal with image data; and M2TV, CNN-Static are only good at coping with text data. Therefore, we omit the results of these methods on the non-applicable data sets.

4.4 Two images data sets: CelebA and Deepfashion The labels of two images data set are attributes of object or person in the image. The comparison results conducted on the two data sets in terms of the top 10 recall rate and top 5 accuracy rate metrics are shown in Figure 2 (a) and 2 (b) respectively. In both figures, the x-axis is the number of training images. The y-axis in Figure 2 (a) is the top 10 recall rate for CelebA data set and the y-axis in Figure 2 (b) is the top 5 accuracy rate for DeepFashion data set. These two figures show that our algorithm outperforms the others with respect to these two evaluation metrics. From the two figures, we can observe that the results of Deep-MTMV with four views outperforms Deep-MTMV with two views. Our intuition of this observation is that four views of Deep-MTMV model can preserve more spatial information than Deep-MTMV model with two views. Moreover, to further prove that our proposed algorithm leads to significant improvement, we conduct the paired student t test, which is shown in Table 1. We compared our methods with Branch-32 method and we found that the p value is 1.01E-17, which indicates that our model does lead to significant improvements over other methods on average. In addition, the p value of the paired student t test on Deep-MTMV with two views and Deep-MTMV with four views indicates that the more views we have, the more spatial information of image we can preserve, and thus a better performance.

4.5 Text data set: Webkb Next, we test the performance of our proposed model on WebKB data set, and the goal is to classify each web page as course or non-course. The baseline method is a simple version of our proposed method, which is trained on a single view, i.e., the content of web page. To test the performance of CNN-Static, we concatenate three views together to be the input of this model. The comparison results in terms of the accuracy and the F1 score are shown in Figure 3 (a) and Figure 3 (b), respectively. The x-axis in these two figures is the percentage of training data and the y-axis is accuracy in Figure 3 (a) and F1 score in Figure 3 (b), respectively. These two figures show that our proposed model is better than the others with respect to both evaluation metrics. From these figures, we observed that the accuracy and F1 score of our proposed model can be as high as 92%, even if only 10% of training data is provided. When more than 80% of training data is given, the accuracy rate and F1 score reach 99%. In this experiment, we also evaluate the weight of each view that contributes to the final prediction. After the model is well-trained, the weight of the content of web page is around 0.0561, compared with 0.0501 and 0.0498 for the rest two views, which is consistent with our expectation that the web content is a little bit important than the link and the title.



(a) Accuracy for WebKB

(b) F1 score for WebKB

Figure 3: Results of WebKB (Best viewed in color)

Model	Accuracy of food prediction	Accuracy of category prediction	F1 score
Branch-32	59.71%	90.64%	55.18%
M2TV	48.75%	69.44%	49.18%
Deep-MTMV	71.22%	94.60%	71.96%

Table 2: Results for FamousFood data set

4.6 Text and image mixed data set: FamousFood Finally, we evaluate our model on text and image mixed data set. Due to the limitation of the compared models, Branch-32 only trains on the single view (image); to reduce the feature dimension for M2TV, the size of each image is re-sized from 224x224 to 50x50 and then each image is converted to a vector. The comparison results in terms of the accuracy and the F1 score are shown in Table 2. We measure the accuracy of type of food prediction, the accuracy of food category prediction, and the macro F1 score. This table shows that our proposed model outperforms the others with respect to these evaluation metrics. From this table, we observed that the accuracy of food prediction reaches 71.22% compared with 59.71% achieved by Branch-32 and 48.75% achieved by M2TV. The worse performance of M2TV for this data set might be due to the fact that M2TV fails to capture the spatial information of images, while Branch-32 cannot utilize the complementary information from another view to further improve the performance.

5 Conclusion

In this paper, we propose a deep multi-task multi-view learning framework, i.e., Deep-MTMV. It trains multiple neural networks, automatically learns the weight of the different views that contribute to the prediction in the regularization layer, groups similar tasks together based on the relatedness of tasks, and classifies the test data with a high accuracy. To the best of our knowledge, the proposed framework is the first deep model for jointly addressing task and view dual heterogeneity, particularly for a data set with multiple modalities. Furthermore, we generalize the proposed Deep-MTMV algorithm to solve multiple real image and text classification problems by (1) utilizing the complementary principle and the consensus principle of multiple views, and (2) learning the relatedness of tasks in each layer of the deep networks. Finally, we compare our algorithm with state-of-the-art techniques, and conduct experiments on multiple real-world data sets to demonstrate that our algorithm leads to statistically significant improvements in the performance. Applying our approach to other applications [5] is one of the future work.

Acknowledgement

This work is supported by the National Science Foundation under Grant No. IIS-1552654, Grant No. IIS-1813464 and Grant No. CNS-1629888, the U.S. Department of Homeland Security under Grant Award Number 17STQAC00001-02-00, and an IBM Faculty Award. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

References

[1] G. Andrew, R. Arora, J. A. Bilmes, and K. Livescu. Deep canonical correlation analysis. In *Proceedings of ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 1247–1255, 2013.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Proceedings of NIPS, Vancouver, Canada, De-*

cember 4-7, 2006, pages 41–48, 2006.

[3] A. Blum and T. M. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of COLT, Madison, Wisconsin, USA, 1998.*, pages 92–100, 1998.

[4] S. Chang, W. Han, J. Tang, G. Qi, C. C. Aggarwal, and T. S. Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of SIGKDD, Sydney, Australia, August 10-13, 2015*, pages 119–128, 2015.

[5] Y. Cheng, Q. Fan, S. Pankanti, and A. Choudhary. Temporal sequence modeling for video event detection. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2235–2242, 2014.

[6] K. Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[7] J. He and R. Lawrence. A graphbased framework for multitask multi-view learning. In *Proceedings of ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 25–32, 2011.

[8] J. Huang, R. S. Feris, Q. Chen, and S. Yan. Cross-domain image retrieval with a dual attribute-aware ranking network. In *2015 IEEE ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1062–1070, 2015.

[9] L. Jacob, F. R. Bach, and J. Vert. Clustered multi-task learning: A convex formulation. In *Proceedings of NIPS, Vancouver, Canada*, pages 745–752, 2008.

[10] X. Jin, F. Zhuang, H. Xiong, C. Du, P. Luo, and Q. He. Multi-task multi-view learning for heterogeneous tasks. In *Proceedings of CIKM 2014, Shanghai, China*, pages 441–450, 2014.

[11] S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of ICML, June 21-24, 2010, Haifa, Israel*, pages 543–550, 2010.

[12] Y. Kim. Convolutional neural networks for sentence classification. pages 1746–1751, 2014.

[13] A. Kumar, P. Rai, and H. D. III. Co-regularized multi-view spectral clustering. In *Proceedings of NIPS 2011, Granada, Spain.*, pages 1413–1421, 2011.

[14] Y. Liu, Y. Zheng, Y. Liang, S. Liu, and D. S. Rosenblum. Urban water quality prediction based on multi-task multi-view learning. pages 2576–2581, 2016.

[15] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the CVPR*, pages 1096–1104, 2016.

[16] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *2015 IEEE ICCV, Santiago, Chile, 2015*, pages 3730–3738, 2015.

[17] C. Lu, L. He, W. Shao, B. Cao, and P. S. Yu. Multilinear factorization machines for multi-task multi-view learning. In *Proceedings of WSDM 2017, Cambridge, United Kingdom, 2017*, pages 701–709, 2017.

[18] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. S. Feris. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. pages 1131–1140, 2017.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013.

[20] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert. Cross-stitch networks for multi-task learning. In *2016 IEEE CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3994–4003, 2016.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[22] V. Sindhwani, P. Niyogi, and M. Belkin. A co-regularization approach to semi-supervised learning with multiple views. In *Proceedings of ICML*, pages 74–79, 2005.

[23] L. Trottier, P. Giguère, and B. Chaib-draa. Multi-task learning by deep collaboration and application in facial landmark detection. volume abs/1711.00111, 2017.

[24] J. Wang, Y. Cheng, and R. S. Feris. Walk and learn: Facial attribute representation learning from egocentric video and contextual data. In *CVPR*, pages 2295–2304. IEEE Computer Society, 2016.

[25] S. Yu, B. Krishnapuram, R. Rosales, H. Steck, and R. B. Rao. Bayesian co-training. In *Proceedings of NIPS, Vancouver, Canada*, pages 1665–1672, 2007.

[26] Y. Zhang and Q. Yang. A survey on multi-task learning. *CoRR*, abs/1707.08114, 2017.

[27] D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *Proceedings of ICML, Corvallis, Oregon, USA, 2007*, pages 1159–1166, 2007.

[28] J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 702–710, 2011.

[29] Y. Zhou and J. He. A randomized approach for crowdsourcing in the presence of multiple views. In *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017*, pages 685–694, 2017.

[30] Y. Zhou, L. Ying, and J. He. Multic^2: an optimization framework for learning from task and worker dual heterogeneity. In *Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, Texas, USA, April 27-29, 2017.*, pages 579–587, 2017.