Representation Learning via Semi-supervised Autoencoder for Multi-task Learning

Fuzhen Zhuang[1], Dan Luo[1,2], Xin Jin[1,2], Hui Xiong[3], Ping Luo[1], Qing He[1]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China; {zhuangfz, heq}@ics.ict.ac.cn, luop@ict.ac.cn.
[2]University of Chinese Academy of Sciences, Beijing 100049, China; {luod, jinx}@ics.ict.ac.cn.
[3]MSIS Department, Rutgers University; hxiong@rutgers.edu.

***Abstract*—Multi-task learning aims at learning multiple related but different tasks. In general, there are two ways for multi-task learning. One is to exploit the small set of labeled data from all tasks to learn a shared feature space for knowledge sharing. In this way, the focus is on the labeled training samples while the large amount of unlabeled data is not sufficiently considered. Another way has a focus on how to share model parameters among multiple tasks based on the original features space. Here, the question is whether it is possible to combine the advantages of both approaches and develop a method, which can simultaneously learn a shared subspace for multiple tasks and learn the prediction models in this subspace? To this end, in this paper, we propose a feature representation learning framework, which has the ability in combining the autoencoders, an effective way to learn good representation by using large amount of unlabeled data, and model parameter regularization methods into a unified model for multi-task learning. Specifically, all the tasks share the same encoding and decoding weights to find their latent feature representations, based on which a regularized multi-task softmax regression method is used to find a distinct prediction model for each task. Also, some commonalities are considered in the prediction models according to the relatedness of multiple tasks. There are several advantages of the proposed model: 1) it can make full use of large amount of unlabeled data from all the tasks to learn satisfying representations; 2) the learning of distinct prediction models can benefit from the success of autoencoder; 3) since we incorporate the labeled information into the softmax regression method, so the learning of feature representation is indeed in a semi-supervised manner. Therefore, our model is a semi-supervised autoencoder for multi-task learning (SAML for short). Finally, extensive experiments on three real-world data sets demonstrate the effectiveness of the proposed framework. Moreover, the feature representation obtained in this model can be used by other methods to obtain improved results.**

I. Introduction

In many practical situations, people need to solve a number of related tasks, and multi-task learning (MTL) [5]–[7], [15] is a good choice for these problems. It learns multiple related tasks together so as to improve the performance of each task relative to learning them separately. For example, webpage classification for webpages from Yahoo[1] and Open Directory Project[2] are two related text classification tasks. MTL methods can share knowledge among them to help learn better classifiers for each task. For face recognition problem, there are face image databases collected by different organizations and under different environmental conditions or have different poses. When each organization only have limited labeled data, it is desirable to utilize all databases to improve the generalization performance. Generally, there are two types of MTL methods. One kind is using the supervised information from all the tasks to help tasks do feature selection collectively or learn a shared feature space for knowledge sharing. The other type of methods focus on how to share model parameters among multiple tasks based on the original features.

Many works studied how to learn shared features among multiple tasks. The earliest MTL method [5] learns a shared hidden layer representation for different tasks, in which all the tasks share the same weights between the input layer and hidden layer, while having different weights between hidden layer and output layer. Multi-task feature learning learns a low-dimensional representation which is shared across a set of related tasks [2], [11]. The methods to learn predictive structures on hypothesis spaces from multiple learning tasks are also proposed in [1], [6]. These methods can learn some shared features or shared structures between different tasks, to facilitate information sharing among tasks. However, the above methods only use the supervised information from all the tasks to learn the features, while ignoring large amount of unlabeled data, which may also contain plenty of useful information.

The other type of methods to share knowledge among multiple tasks are by taking advantage of the commonness of the prediction model parameters. Supposing that all the tasks are similar, a regularization formulation is proposed for MTL [8] to make the model parameters of these tasks be similar. MTL can be modeled by stochastic process methods, such as [15], [19]. In this way, the correlation between multiple tasks' model parameters can be represented and learned. To deal with outlier tasks, a robust multi-task learning algorithm is proposed [7], which can utilize the similarity among similar tasks' models while reducing the harmful effects of outlier tasks. These methods share knowledge by placing a common prior on the model parameters of each task in hierarchical Bayesian models and explicitly share some model parameters or model structure among tasks. Most of these methods share model parameters in the original feature space. In this work, we focus on the parameter sharing in the learnt subspace by

[1]http://www.yahoo.com/
[2]http://www.dmoz.org/

1550-4786/15 $31.00 © 2015 IEEE
DOI 10.1109/ICDM.2015.22

semi-supervised autoencoders.

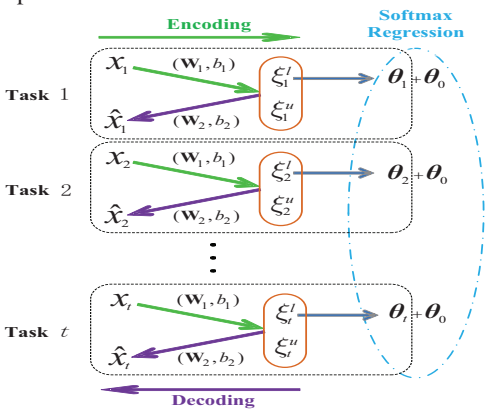


Fig. 1: The framework of SAML.

In fact, both feature representation learning and model parameter sharing are helpful for multi-task learning. Autoencoder has been proven to be an effective way to learn subspace only using the unlabeled data. In this work, we aim at discovering a new feature representation for multi-task learning by autoencoder, which is promoted by the model parameter sharing of regularization method. To this end, we combine autoencoder and multi-task softmax regression into a unified model. The proposed Semi-supervised Autoencoder for Multi-task Learning (SAML) is shown in Fig. 1. In Fig. 1, the encoding weights are shared by all tasks to map the data into common compressed expression, and for labeled and unlabeled data, their corresponding new representations are $\boldsymbol{\xi}_t^l$ and $\boldsymbol{\xi}_t^u$, respectively. With $\boldsymbol{\xi}_t^l$ in each task, a regularized multi-task softmax regression method is used to train a separate model for each task. The softmax regression model includes two parts, one is $\boldsymbol{\theta}_0$, which is shared by multiple tasks and the other is a task specific part $\boldsymbol{\theta}_t$ for each task t. The overall framework includes two parts, i.e., autoencoder and regularized softmax regression, and these two parts are optimized at the same time. The proposed framework not only minimizes the reconstruction error, but also aims at reducing the classification error of the softmax regression on the encoding output. This framework can do semi-supervised learning with only a small number of labeled data and plenty of unlabeled ones. It has several advantages. First, it can use autoencoder to discover the intrinsic knowledge of the training samples in the raw feature representation. Second, the small number of labeled training samples can also help autoencoder to learn the latent feature representation. Third, based on the regularization method, multiple tasks can share knowledge by model parameters in the learning process. Finally, the feature transformation process is nonlinear, i.e., the transformation from the input layer to hidden layer is nonlinear in autoencoder, which enhances the learning ability of the proposed framework.

II. Preliminary Knowledge

A. Autoencoder

The basic framework of autoencoder [3] is a neural network, which comprises an input layer, an output layer and at least one hidden layer. The aim of an autoencoder is to transform inputs into outputs with the least possible amount of deviation. So it is usually used as an information compressor and it contains the encoding and decoding processes. An single hidden layer autoencoder can be described by Fig. 2. As we can see, the input is encoded into a lower-dimension representation, then decoded into an output with the same size of input. Usually,

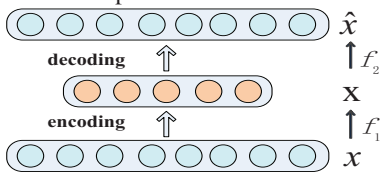


Fig. 2: Single hidden layer autoencoder.

f_1 and f_2 are nonlinear activation functions (sigmoid function is adopted in this paper). This process can be summarized as follows.

Given input $\boldsymbol{x}_i \in \mathbb{R}^{m\times 1}$, weight matrix $\boldsymbol{W}_1 \in \mathbb{R}^{k\times m}$, $\boldsymbol{W}_2 \in \mathbb{R}^{m\times k}$, and bias vector $\boldsymbol{b}_1 \in \mathbb{R}^{k\times 1}$, $\boldsymbol{b}_2 \in \mathbb{R}^{m\times 1}$,

$$\boldsymbol{\xi_i} = f(\boldsymbol{W}_1\boldsymbol{x_i} + \boldsymbol{b}_1),\ \hat{\boldsymbol{x}}_{\boldsymbol{i}} = f(\boldsymbol{W}_2\boldsymbol{\xi_i} + \boldsymbol{b}_2). \tag{1}$$

As the objective of autoencoder is to enforce the output $\hat{\boldsymbol{x}}_{\boldsymbol{i}}$ as close as possible to the input $\boldsymbol{x}_i$, we try to minimize the distance of the input and output. The goal of autoencoder is to minimize the reconstruction error using Euclidean distance,

$$\min_{\boldsymbol{W}_1,\boldsymbol{b}_1,\boldsymbol{W}_2,\boldsymbol{b}_2} \mathcal{J}_r = \sum_{i=1}^{n} ||\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x_i}||^2. \tag{2}$$

B. Softmax Regression

Softmax regression [9] is a generalization of logistic regression, which can handle multi-class classification problems, and the class label $y \in \{1, 2, ..., c\}$, where $c \geq 2$ is the number of classes. Given a test input $\boldsymbol{x}_i$, softmax regression can estimate the probability that $p(y_i = j|\boldsymbol{x}_i)$ for each value of $j = 1, \cdots, c$,

$$p(y_i = j|\boldsymbol{x}_i; \boldsymbol{\theta}) = \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}_i}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}_i}} \tag{3}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_c\}$ are the parameters of the model. The term $\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}_i}$ normalizes the distribution to guarantee the probability condition. In the special case where $c = 2$, the softmax regression reduces to logistic regression.

Given the training set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n, y_i \in \{1, 2, ..., c\}$, the solution of softmax regression can be derived by minimizing the following optimization problem,

$$\min_{\boldsymbol{\theta_1},\cdots,\boldsymbol{\theta_c}} (-\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} 1\{y_i = j\} \log \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}_i}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}_i}}), \tag{4}$$

where $1\{\cdot\}$ is the indicator function, which equals to 1 only when the value of the statement is true, otherwise 0. After training the model, the probability of a new instance $\boldsymbol{x}_i$ belonging to label j can be computed via Eq.(5), then $\boldsymbol{x}_i$ is labeled to the class which has the biggest conditional probability,

$$y_i = \arg\max_j \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}_i}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}_i}}. \tag{5}$$

C. Regularized Multi-Task Learning

Regularized multi-task learning [8] is a framework based on the minimization of regularization functions. It learns all tasks simultaneously, and gets the common sub-model shared by all tasks and specific sub-models owned by each task privately.

For the sake of brevity, we assume that the function $\hbar_t$ is the hyperplane for the t-th task, that is $\hbar_t(\boldsymbol{x}_{ti}) = \boldsymbol{w}_t^\top \boldsymbol{x}_{ti}$, where $\boldsymbol{x}_{ti} \in \mathbb{R}^{k\times 1}$. As the tasks are related, for every task $t \in \{1, ..., T\}$, $\boldsymbol{w_t}$ can be written by $\boldsymbol{w}_t = \boldsymbol{w}_0 + \boldsymbol{v}_t$. The vector $\boldsymbol{w}_0$ represents the common model parameter of all tasks, and the vectors $\boldsymbol{v}_t$ are different between tasks. When the tasks are similar to each other, $\boldsymbol{v}_t$ are "small". It means that all models $\boldsymbol{w}_t$ are close to some model $\boldsymbol{w}_0$. On the other hand, when $\boldsymbol{w}_0$ is extremely small, all models $\boldsymbol{w}_t$ are unrelated, then the framework is equal to training each task separately. Using L2-norm on each component of $\boldsymbol{w}_t$, we can get the framework as,

$$\min_{\boldsymbol{w}_0, \boldsymbol{v}_t, \varepsilon_t} J(\boldsymbol{w}_0, \boldsymbol{v}_t, \varepsilon_t) = \sum_{t=1}^{T} \varepsilon_t + \frac{\lambda_1}{T} \sum_{t=1}^{T} ||\boldsymbol{v}_t||^2 + \lambda_2 ||\boldsymbol{w}_0||^2, \tag{6}$$

where

$$\varepsilon_t = \ell(\boldsymbol{w}_t^\top \boldsymbol{x}_t), \quad \boldsymbol{x}_t = [\boldsymbol{x}_{t1}, \boldsymbol{x}_{t2}, \ldots, \boldsymbol{x}_{tn}]. \tag{7}$$

In Eq. (6), λ_1 and λ_2 are positive regularization parameters, and the loss ε_t measures the error that each final model $\boldsymbol{w}_t$ makes on all training data. Intuitively, if λ_1 is much larger than λ_2, it will result to making the models of all tasks to be the same model $\boldsymbol{w}_0$. If λ_2 is much larger than λ_1, it will result to making all tasks unrelated, which equals to training each task separately. To simplify the model, we choose L2-norm regularization in this work.

III. Representation Learning via Semi-supervised Autoencoder for Multi-task Learning

In this section, we first formalize the proposed framework, then describe the derivation of the solution in detail.

A. Framework Formalization

Given T task data sets with both labeled and unlabeled data, i.e., $\boldsymbol{D}_t = \{\boldsymbol{x}_{ti}, y_{ti}\}_{i=1}^{n_{tl}} + \{\boldsymbol{x}_{ti}\}_{i=1}^{n_{tu}}$ $(1 \le t \le T)$, with $\boldsymbol{x}_{ti} \in \mathbb{R}^{m\times 1}$, $y_{ti} \in \{1, 2, ..., c\}$, where n_{tl} is the number of labeled data in task t and n_{tu} is the corresponding number of unlabeled data. Actually, our framework is semi-supervised, which can effectively employ the small set of labeled data to facilitate the common knowledge sharing between tasks via the regularized multi-task framework, and use the autoencoder to find good representations.

As shown in Fig. 1, our proposed framework contains two components mainly. The first component is the autoencoder, which projects the input $\boldsymbol{x}$ into lower dimension $\boldsymbol{\xi}$. The second component is the softmax regression, which learns the model with $\boldsymbol{\xi}$ as its input. The proposed learning framework for multi-task learning can be formalized as follows,

$$\mathcal{J} = \sum_{t=1}^{T} \mathcal{J}_r(\boldsymbol{x}_t, \hat{\boldsymbol{x}}_t) + \alpha \sum_{t=1}^{T} \mathcal{L}(\boldsymbol{\xi}_t, \boldsymbol{\theta}_t) + \Omega_r + \Omega_l. \tag{8}$$

The first term $\mathcal{J}_r(\boldsymbol{x}_t, \hat{\boldsymbol{x}}_t)$ is the reconstruction error for both labeled and unlabeled data of task t where $\hat{\boldsymbol{x}}_{\boldsymbol{t}} \in \mathbb{R}^{m\times n_t}$, which can be defined as follows,

$$\mathcal{J}_r(\boldsymbol{x}_t, \hat{\boldsymbol{x}}_t) = \sum_{i=1}^{n_t} |\hat{\boldsymbol{x}}_{ti} - \boldsymbol{x}_{ti}|^2, \tag{9}$$

where

$$n_t = n_{tl} + n_{tu}, \boldsymbol{\xi}_{ti} = f(\boldsymbol{W}_1 \boldsymbol{x}_{ti} + \boldsymbol{b}_1), \hat{\boldsymbol{x}}_{ti} = f(\boldsymbol{W}_2 \boldsymbol{\xi}_{ti} + \boldsymbol{b}_2). \tag{10}$$

In this term, all the tasks share the same encoding and decoding weights, i.e., $\boldsymbol{W}_1$, $\boldsymbol{b}_1$, $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$. The hidden layer has k nodes ($k \le m$), and the weight matrixes $\boldsymbol{W}_1 \in \mathbb{R}^{k\times m}$ and $\boldsymbol{b}_1 \in \mathbb{R}^{k\times 1}$ connect the input layer and the hidden layer, while $\boldsymbol{W}_2 \in \mathbb{R}^{m\times k}$ and $\boldsymbol{b}_2 \in \mathbb{R}^{m\times 1}$ connect the hidden layer and the output. We try to minimize the reconstruction error of each instance, to ensure that the hidden layer output contains the same information as the input data. Since the tasks are related, we believe the collaborative autoencoders among all tasks can lead to better feature representations.

The second term $\mathcal{L}(\boldsymbol{\xi}_t, \boldsymbol{\theta}_t)$ is the optimization problem of softmax regression, in which we try to incorporate the labeled data of all tasks. The minimization of objective function is formalized as,

$$\mathcal{L}(\boldsymbol{\xi}_t, \boldsymbol{\theta}_t) = -\frac{1}{n_{tl}} \sum_{i=1}^{n_{tl}} \sum_{j=1}^{c} 1\{y_{ti} = j\} \log \frac{e^{\hat{\boldsymbol{\theta}}_{tj}^\top \boldsymbol{\xi}_{ti}}}{\sum_{p=1}^{c} e^{\hat{\boldsymbol{\theta}}_{tp}^\top \boldsymbol{\xi}_{ti}}}, \tag{11}$$

where $\hat{\boldsymbol{\theta}}_{tj} = \boldsymbol{\theta}_{tj} + \boldsymbol{\theta}_{0j}$. In this term, the input data $\boldsymbol{\xi}_{ti} \in \mathbb{R}^{k\times 1}$ is the output of the antoencoder hidden layer. The weight vectors $\boldsymbol{\theta}_{tj} \in \mathbb{R}^{k\times 1}$ and $\boldsymbol{\theta}_{0j} \in \mathbb{R}^{k\times 1}$ $(j \in \{1, ..., c\})$ are respectively the specific model parameters for each task and common model parameters shared by all tasks. To control the importance of $\boldsymbol{\theta}_{tj}$ and $\boldsymbol{\theta}_{0j}$, we can add the L2-norm regularization into the multi-task framework as

$$\Omega_l = \frac{\lambda_1}{T} \sum_{t=1}^{T} \sum_{j=1}^{c} ||\boldsymbol{\theta}_{tj}||^2 + \lambda_2 \sum_{j=1}^{c} ||\boldsymbol{\theta}_{0j}||^2, \tag{12}$$

where λ_1 and λ_2 are the trade-off parameters. Large value of λ_1 will lead to all tasks sharing the same model, while large value of λ_2 will result in the separate training of each task.

Also, to control the complexity of the autoencoder model and to improve its generalization ability, we add a weight decay term Ω_r in Eq. (8), which can be written as follows,

$$\Omega_r = \lambda_3 (||\boldsymbol{W}_1||^2 + ||\boldsymbol{b}_1||^2 + ||\boldsymbol{W}_2||^2 + ||\boldsymbol{b}_2||^2), \tag{13}$$

where λ_3 is trade-off parameter with small positive value for the whole framework.

B. Solution of the Proposed Framework

The optimization problem of our proposed framework is to minimize $\mathcal{J}$ (seen in Eq.(8)) as a function of $\boldsymbol{W}_1$, $\boldsymbol{b}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_2$, $\boldsymbol{\theta}_{tj}$ and $\boldsymbol{\theta}_{0j}$ $(j \in \{1, ..., c\})$. Obviously, it is an unconstrained optimization problem. To solve this problem, we adopt the gradient descent method to derive it. Due to the

space limitation, we only give the partial derivatives of $\boldsymbol{\theta}_{tj}$ and $\boldsymbol{\theta}_{0j}$ ($j \in \{1, ..., c\}$) in the following,

$$\frac{\partial J}{\partial \boldsymbol{\theta}_{tj}} = \alpha(-\frac{1}{n_{tl}} \sum_{i=1}^{n_{tl}} \sum_{j=1}^{c} 1\{y_{ti} = j\}(1 - \frac{e^{\hat{\boldsymbol{\theta}}_{tj}^{\top} \boldsymbol{\xi}_{ti}}}{\sum_{l=1}^{c} e^{\hat{\boldsymbol{\theta}}_{tl}^{\top} \boldsymbol{\xi}_{ti}}})\boldsymbol{\xi}_{ti})) + \frac{2\lambda_1}{T}\boldsymbol{\theta}_{tj}, \tag{14}$$

$$\frac{\partial J}{\partial \boldsymbol{\theta}_{0j}} = \alpha \sum_{t=1}^{T}(-\frac{1}{n_{tl}} \sum_{i=1}^{n_{tl}} \sum_{j=1}^{c} 1\{y_{ti} = j\}(1 - \frac{e^{\hat{\boldsymbol{\theta}}_{tj}^{\top} \boldsymbol{\xi}_{ti}}}{\sum_{l=1}^{c} e^{\hat{\boldsymbol{\theta}}_{tl}^{\top} \boldsymbol{\xi}_{ti}}})\boldsymbol{\xi}_{ti})) + 2\lambda_2 \boldsymbol{\theta}_{0j} \quad . \tag{15}$$

Based on the partial derivatives, we develop an alternately optimization algorithm to derive the solutions with the following rules,

$$\begin{aligned} \boldsymbol{W}_1 &\leftarrow \boldsymbol{W}_1 - \eta \frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1}, \quad \boldsymbol{b}_1 \leftarrow \boldsymbol{b}_1 - \eta \frac{\partial \mathcal{J}}{\partial \boldsymbol{b}_1}, \\ \boldsymbol{W}_2 &\leftarrow \boldsymbol{W}_2 - \eta \frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_2}, \quad \boldsymbol{b}_2 \leftarrow \boldsymbol{b}_2 - \eta \frac{\partial \mathcal{J}}{\partial \boldsymbol{b}_2}, \\ \boldsymbol{\theta}_{tj} &\leftarrow \boldsymbol{\theta}_{tj} - \eta \frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}_{tj}}, \quad \boldsymbol{\theta}_{0j} \leftarrow \boldsymbol{\theta}_{0j} - \eta \frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}_{0j}}, \end{aligned} \tag{16}$$

where η is the step length, which determines the speed of convergence.

Though the objective of the whole framework is not convex, we can obtain relatively good results through appropriate initiation of the variables. The experimental results also validate the effectiveness of the proposed solution. Specifically, we use Stacked AutoEncoder [4] to get the initial value of $\boldsymbol{W}_1$, $\boldsymbol{b}_1$, $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$, and we can get the output $\boldsymbol{\xi}$ of the hidden layer. Then Bayesian Multinomial Regression (BMR for short) [12] is adopted to obtain the initial value of $\boldsymbol{\theta}_{tj}$ and $\boldsymbol{\theta}_{0j}$ ($j \in \{1, ..., c\}$) with $\boldsymbol{\xi}$ as its input. That is, $\boldsymbol{\theta}_{tj}$ is the weight of the single task BMR model and $\boldsymbol{\theta}_{0j}$ is the average of all $\boldsymbol{\theta}_{tj}$.

C. Classifier Construction

After all parameters are learned, there are two ways to construct classifiers for all tasks. The first way is to use the softmax regression to predict the unlabeled data directly as described in Eq. (5) by the proposed framework. That is, for each instance, we can estimate the probability $P(y_{ti} = j|\boldsymbol{x}_{ti})$, and then assign the class label with the maximum probability. The second way is to apply standard classification algorithms, e.g., logistic regression (LR) [9] [16] to train a classifier for each task independently in the compressed low-dimensional space. These two methods are denoted as SAML1 and SAML2, respectively.

IV. Experimental Evaluation

A. Data sets and Preprocessing

The first two data sets are for binary classification, and the third one is for multi-class classification.

Corel Data Set[3] includes two different top categories, *flower* and *traffic*, which is used for transfer learning [22]. Each top category further consists of four subcategories. We use *flower* and *traffic* as positive and negative instances, respectively. To construct the multi-task learning classification problems, we randomly select one subcategory from *flower* and one from *traffic* to form a task. In this way, we can get four tasks with no repeat. Totally 24 (P_4^4) 4-task learning classification problems can be constructed from this data set. In each task, several instances are randomly selected to construct the labeled data set, and the rest are used for the unlabeled one.

ImageNet Data Set[4] contains five categories, i.e., *ambulance*, *taxi*, *jeep*, *minivan* and *scooter*, in which *scooter* is a big category. To construct the multi-task learning classification problems that all tasks are related, we randomly divide *scooter* into four subcategories. Then we regard *scooter* as negative and the other categories as positive. Similar with Corel data set, we can also construct 24 (P_4^4) 4-task learning classification problems.

Leaves Data Set [13] includes 100 plant species that are divided into 32 different genera, and each species has 16 instances. We choose three genera with more than four plant species to construct 4-task 3-class classification problems, and use 64 margin descriptor features to represent an instance. Similar with the construction method of the above two data sets, we can construct 576 ($P_4^4 \cdot P_4^4$) 4-task 3-class classification problems.

B. Compared Algorithms

We compare our multi-task learning algorithm SAML with the following baseline algorithms,

- The single task supervised classification algorithm Bayesian Multinomial Regression (BMR) [12];
- Robust Multi-Task Feature Learning (rMTFL) [10], which simultaneously captures a common set of features among relevant tasks and identifies outlier tasks;
- Robust Multi-Task Learning (RMTL) [7], which captures the task relationships using a low-rank structure, and simultaneously identifies the outlier tasks using a group-sparse structure;
- Clustered Multi-Task Learning (CMTL) [20], which captures structures by minimizing sum-of-square error (SSE) in K-means clustering.

Since our proposed model can find new feature representation in multi-task problems, so based on which BMR is used to train classifiers for each task, denoted as SAML2.

C. Implementation Details

We adopt the MALSAR package [21] for the implementation of our compared algorithms, i.e. rMTFL, RMTL and CMTL. Besides, we adopt BMR[5] as baseline, which is a robust algorithm based on Bayesian framework. Also, we use BMR for the initiation of the model parameters in our framework. In addition, BMR is used to build the single task classification model in SAML2. We use Stacked AutoEncoders to initiate the encoding and decoding weights of our framework.

[3] http://archive.ics.uci.edu/ml/datasets/Corel+Image+Features.

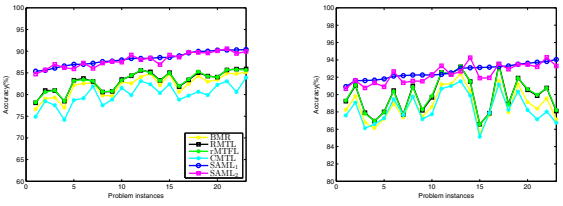
[4] http://www.image-net.org/download-features

[5] http://www.bayesianregression.com/bmr.html

After some preliminary experiments, we set $\lambda_1 = 0.0001$, $\lambda_2 = 0.0001$, $\lambda_3 = 0.0001$ for all three data sets, and $\alpha = 0.005$ for Coral and ImageNet data sets, and $\alpha = 0.002$ for Leaves data set. As the three data sets have different feature dimension originally, we set different values of k for each of them. We set $k = 30$ for Coral data set, set $k = 80$ for ImageNet data set and set $k = 10$ for Leaves data set. For the baselines, which can not handle multi-class classification problems, we adapt the compared multi-task baselines to handling multi-class problems via one-versus-all. The evaluation metric is the averaged prediction accuracy over all tasks.

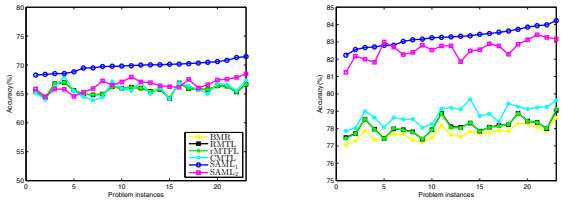
TABLE I: Comparisons among BMR, RMTL, rMTFL, CMTL, and SAML on Coral, ImageNet and Leaves data sets

		2	4	6	8	10	Avg
Corel	BMR	74.8	82.3	85.5	88.9	90.2	84.3
	RMTL	75.4	83.3	86.6	90.0	90.8	85.2
	rMTFL	75.4	83.3	86.6	90.0	90.9	85.2
	CMTL	74.6	80.2	84.9	88.5	89.3	83.5
	SAML1	**82.4**	**88.4**	**91.8**	**92.7**	93.6	**89.8**
	SAML2	81.5	88.1	91.2	92.5	**93.9**	89.4
		2	4	8	16	32	Avg
ImageNet	BMR	62.4	65.8	71.6	77.8	84.1	72.3
	RMTL	62.9	65.8	72.4	78.1	84.4	72.7
	rMTFL	62.9	65.8	72.4	78.1	84.4	72.7
	CMTL	63.6	65.8	73.3	78.8	85.6	73.4
	SAML1	**64.5**	**69.9**	**76.1**	**83.3**	**89.8**	**76.7**
	SAML2	61.7	66.6	72.9	82.6	89.4	74.6
		3	6	9	12	15	Avg
Leaves	BMR	79.9	88.9	91.9	93.7	92.7	89.4
	RMTL	80.6	89.3	91.6	92.8	92.3	89.3
	rMTFL	80.6	89.3	91.6	92.8	92.3	89.3
	CMTL	79.7	89.4	90.8	92.1	92.1	88.8
	SAML1	83.3	**95.2**	**95.3**	96.1	**94.8**	**92.9**
	SAML2	**83.5**	92.0	94.3	**96.1**	92.7	91.7



(a) Problems with 4 labeled samples (b) Problems with 8 labeled samples

Fig. 3: Comparisons among BMR, RMTL, rMTFL, CMTL, and SAML on Coral Data Set

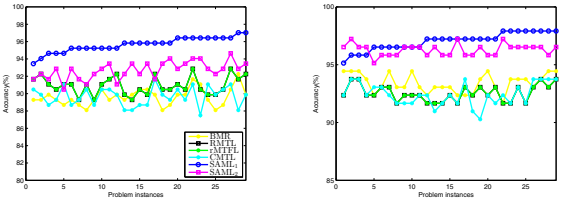


(a) Problems with 4 labeled samples (b) Problems with 8 labeled samples

Fig. 4: Comparisons among BMR, RMTL, rMTFL, CMTL, and SAML on ImageNet Data Set

D. Experimental Results

1) Comparison Results on Coral and ImageNet Data Sets: We first evaluate all the algorithms on the Coral and ImageNet data sets for binary classification. For both data sets, there are 24 constructed multi-task problems. Actually, the multi-task learning setting is a semi-supervised style, so we sample a small subset of data as labeled samples for each problem. Specifically, the numbers of labeled samples are 1, 2, 3, 4 and 5 for each class on Coral data set, and which are 1, 2, 4, 8 and 16 on ImageNet data set. We conduct 3 independent trials for each number of sampled labeled data, and then the average results are recorded. For the sake of succinctness, the detailed results of two labeled data sizes (i.e., 4, 8.) are shown in Fig. 3 and Fig. 4. In these figures, x-axis is the problem instance which demonstrates the 24 classification problems, and y-axis is the corresponding accuracy. From these results, we have the following insightful observations,



(a) Problems with 6 labeled samples (b) Problems with 12 labeled samples

Fig. 5: Comparisons among BMR, RMTL, rMTFL, CMTL, and SAML on Leaves Data Set

- As a single task algorithm, BMR can also get a relatively comparable performance with the baseline multi-task learning algorithms, which indicates the effectiveness of Bayesian Multinomial Regression. This inspires us to use it for the initialization of $\boldsymbol{\theta}_{0j}$ and $\boldsymbol{\theta}_{tj}$ $(1 \leq t \leq T)$.
- The output of the proposed framework SAML1 is significantly better than all the compared baselines, which demonstrates the effectiveness of considering both feature representation learning by autoencoder and model parameter sharing between multiple tasks simultaneously. Furthermore, SAML1 performs much stable than all the baselines.
- In most cases, SAML2 also outperforms the compared baselines, which shown that the feature representation obtained in our model can be used by other supervised algorithms to obtain improved performance.
- With the increasing number of labeled samples, all the algorithms obtain better results. It is worth mentioning that SAML1 not only achieves the best results but also performs well when there are only few labeled samples. This further validates the superiority of our model over the baselines, since in real-world applications the labeled data are always not easy to obtain.

In Table I, we also report the average results over 24 problems of each data set for all algorithms under different numbers of labeled samples. From these results in Table I, SAML1 also performs the best, which again validate the effectiveness of the proposed framework.

2) Comparison Results on Leaves Sets: To validate our framework can also directly deal with multi-task learning with multi-class problems, we evaluate all algorithms on the Leaves

data set. The numbers of sampled labeled samples are 3, 6, 9, 12 and 15, i.e., 1, 2, 3, 4 and 5 samples from each class, respectively. To clearly show the results, we randomly select 30 problems and their detailed results of two labeled data sizes (i.e., 6, 12.) are shown in Fig. 5. The average results over 576 problems are shown in Table I. We can obtain the similar observations as the Coral and ImageNet Data Sets from these results. Our model SAML1 again achieves the best results, and SAML2 can obtain improved results based on the output feature representation of our framework.

V. Related Works

Multi-task learning (MTL) conducts multiple related learning tasks simultaneously so that the useful information in one task can be used for other tasks. The earliest MTL method is based on neural networks, it let the neural networks for multiple tasks share the hidden layer, which means that all the tasks use the same hidden layer feature representation to learn their distinct classifiers [5]. There are also other methods to learn a low-dimensional representation which is shared across a set of multiple related tasks [2], [11]. These methods can learn some shared features between different tasks, which is used to facilitate information sharing among tasks. However, most algorithms are supervised methods and cannot take advantage of unlabeled data.

Since the multiple tasks are related, regularization method can be used to make the model parameters be similar [8]. The correlations between the model parameters of multiple tasks can be formulated in stochastic process models [15], [19]. In this way, the correlations can be learned and help the knowledge sharing among multiple tasks. To deal with outlier tasks, a robust multi-task learning algorithm is proposed [7]. It allows the prediction models of outlier tasks to be different from other tasks. These methods share knowledge by placing a common prior on the model parameters of each task in hierarchical Bayesian models and explicitly share some model parameters or model structure among tasks. But most of them share model parameters in the original feature space. In this paper, we try to share the model parameter based on the new feature representation.

Poultney *et al.* [14] described a novel unsupervised method with an energy-based Model for learning sparse, over complete features. In their model, the decoder produces accurate reconstructions of the patches, while the encoder provides a fast prediction of the code without the need for any particular preprocessing of the inputs. Denoising autoencoders [17] is to learn a more robust representation from an artificially corrupted input, and further Stacked denoising autoencoders [18] tries to learn useful representations through a deep network. However, these methods have not considered the label information and they are not designed for multi-task learning.

VI. Conclusion

In this paper, we propose a novel semi-supervised feature learning framework with few labeled data in each task for multi-task learning. In this framework, the well known representation learning model autoencoder is considered, and we propose a generalized logistic regression to incorporate the labeled information to supervise it. The proposed framework simultaneously learns the new feature representation and minimizes the prediction error on the labeled data. Moreover, the knowledge is shared among multiple tasks in two aspects, one is all tasks sharing the same encoding and decoding weights, and the other one is that they share the model parameter based on the latent feature space. Extensive experiments on three real-world image data sets demonstrate the effectiveness of the proposed framework.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 61175052, 61203297, 61473273, 61473274), National High-tech R&D Program of China (863 Program) (No.2014AA015105, 2013AA01A606), Science and Technology Planning Project of Guangdong Province, China (2015B010109005).

References

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 2005.
[2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *NIPS*, pages 41 – 48, 2007.
[3] Y. Bengio. Learning deep architectures for ai. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.
[4] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, U. D. Montral, and M. Qubec. Greedy layer-wise training of deep networks. In *NIPS*, 2007.
[5] R. Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.
[6] J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning shared structures from multiple tasks. In *26th ICML*, 2009.
[7] J. Chen, J. Zhou, and J. Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *SIGKDD*, 2011.
[8] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *10th ACM SIGKDD*, pages 109 – 117, 2004.
[9] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent, 2009.
[10] P. Gong, J. Ye, and C. Zhang. Robust multi-task feature learning. In *18th ACM SIGKDD*, pages 895–903, 2012.
[11] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *NIPS*, 2010.
[12] D. Madigan, A. Genkin, D. D. Lewis, E. G. D. D. Lewis, S. Argamon, D. Fradkin, L. Ye, and D. D. L. Consulting. Author identification on the large scale. In *Proc. of the Meeting of the Classification Society of North America*, 2005.
[13] C. Mallah and Orwell. Plant leaf classification using probabilistic integration of shape, texture and margin features. *Signal Processing, Pattern Recognition and Applications*, 2013.
[14] C. Poultney, S. Chopra, Y. L. Cun, et al. Efficient learning of sparse representations with an energy-based model. In *NIPS*, 2006.
[15] G. Skolidis and G. Sanguinetti. Bayesian multitask classification with gaussian process priors. *IEEE TNN*, 22(12):2011–2021, 2011.
[16] J. Snyman. *Practical mathematical optimization: an introduction to basic optimization theory and classical and new gradient-based algorithms*, volume 97. 2005.
[17] B. Vincent, Larochelle and Manzagol. Extracting and composing robust features with denoising autoencoders. In *25th ICML*, pages 1096–1103, 2008.
[18] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *JMLR*, 2010.
[19] S. Yu, V. Tresp, and K. Yu. Robust multi-task learning with t-processes. In *24th ICML*, volume 227, pages 1103 – 1110, 2007.
[20] J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. In *NIPS*, pages 702–710. 2011.
[21] J. Zhou, J. Chen, and J. Ye. *MALSAR: Multi-tAsk Learning via StructurAl Regularization*. Arizona State University, 2011.
[22] F. Zhuang, P. Luo, H. Xiong, Y. Xiong, Q. He, and Z. Shi. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE TKDE*, 22(12):1664–1678, 2010.