ABSTRACT

Multi-task learning (MTL) is a framework that enforces multiple learning tasks to share their knowledge to improve their generalization abilities. While shallow multi-task learning can learn task relations, it can only handle pre-defined features. Modern deep multi-task learning can jointly learn latent features and task sharing, but they are obscure in task relation. Also, they pre-define which layers and neurons should share across tasks and cannot learn adaptively. To address these challenges, this paper proposes a new multi-task learning framework that jointly learns latent features and explicit task relations by complementing the strength of existing shallow and deep multitask learning scenarios. Specifically, we propose to model the task relation as the similarity between tasks’ input gradients, with a theoretical analysis of their equivalency. In addition, we innovatively propose a multi-task learning objective that explicitly learns task relations by a new regularizer. Theoretical analysis shows that the generalizability error has been reduced thanks to the proposed regularizer. Extensive experiments on several multi-task learning and image classification benchmarks demonstrate the proposed method’s effectiveness, efficiency as well as reasonableness in the learned task relation patterns.

CCS CONCEPTS

- Computing methodologies → Multi-task learning. Regularization.

KEYWORDS

Multi-task Learning, Task Relation, Saliency Detection

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD ’22, August 14–18, 2022, Washington, DC, USA.
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9385-0/22/08...
https://doi.org/10.1145/3534678.3539442

1 INTRODUCTION

Multi-task learning (MTL, [5]) is an important research domain based on the idea that the performance of one task can be improved using related tasks as inductive bias. While traditional shallow MTL methods can fit the models for individual tasks and learn task relations, they do not focus on generating features from scratch and instead rely on pre-defined and explicit features [42, 46]. More recently, deep representation learning empowers MTL to go “deep” by equipping it with the capacity to generate features while fitting the tasks’ predictive models. Deep MTL is usually categorized according to the ways of correlating tasks’ models into two major types: hard-parameter sharing and soft-parameter sharing. Hard-parameter sharing methods [26, 48] essentially hard-code which part of neurons or layers to share and which part does not for different tasks instead of doing it adaptively. Moreover, they usually share the layers for representation learning (e.g., convolutional layers) but not those for decision making (e.g., fully-connected layers for classification). On the other hand, soft-parameter sharing methods [8, 30] do not require to hard-code the sharing pattern but instead build individual models for each task and “softly” regularize the relatedness among them. Hence, soft-parameter sharing has better flexibility in learning the task relation, while may not be efficient since its model parameters increase linearly with the number of tasks. Hard-parameter sharing, by contrast, is more “concise” but requires pre-define which parts are shared or not.

Therefore, although MTL is a long-lasting research domain, it remains a highly challenging and open domain that requires significantly more efforts to address challenges such as the trade-off between model flexibility and conciseness of hard- and soft-parameter sharing mentioned above. Although more recently, there have come a few attempts trying to alleviate the dilemma, such as those regularizing task relationships in task-specific layers in hard-parameter sharing to achieve knowledge transfer in unshared layers [26] and those adaptively learning which part to share or not by methods like branching [27] or Neural Architecture Search [40], the research frontiers still suffer from several critical bottlenecks, including (1) Difficulty in regularizing deep non-linear functions of different tasks. Adaptively learning task relation requires regularizing different tasks’ predictive functions, which, however, are much harder to achieve for nonlinear-nonparametric functions since it requires regularizing in the whole continuous domain of input. To work around it, existing works [26, 39] typically resort to a reduced problem which is to regularize the neural network parameters. Notice that this reduction deviates from the original problem and is over-restricted. For example, first, two neural networks with different permutations of latent neurons can represent the same function. Moreover, even if they have different architectures, they can still possibly represent the same function [22]. This gap deteriorates the model’s generalizability and effectiveness. (2) Lack of interpretability in joint feature generation and task relation learning. Despite incapability of generating features, shallow MTL
enjoys good interpretability since they learn explicit task correlations via how the hand-crafted features are utilized. However, in deep MTL, the generated features do not have explicit meaning and how the black-box models relate to each other is highly obscure. It is imperative yet challenging to increase the interpretability of both generated features and task relation. (3) Difficulty in theoretical analysis. While there are fruitful theoretical analyses on shallow MTL, such as on generalization error [3] and conditions for regularized MTL algorithms to satisfy representer theorems [2], similar analyses meet strong hurdles to be extended to deep MTL due to the difficulty in reasoning about neural networks whose feature space is given by layer-wise embeddings [43]. It is crucial to enhance the theoretical analyses on the model capacity and theoretical relation among different deep MTL models.

This paper proposes a new Saliency-Regularized Deep Multi-task Learning (SRDML) framework to solve the challenges mentioned above. First, we reconsider the feature weights in traditional linear multitask learning as the input gradient and then generalize the feature learning into the non-linear situation by borrowing the notion of saliency detection. Second, we recast the task relation problem as the similarity among saliency regions across tasks so as to regularize and infer the task relation. Third, to validate our hypothesis, we have given a theoretical analysis of their equivalence. Meanwhile, we also provide theoretical analysis on how the proposed regularization helps reduce the generalization error. Finally, we demonstrate our model’s effectiveness and efficiency on synthetic and multiple large-scale real-world datasets under comparison with various baselines.

2 RELATED WORK

Multi-task learning (MTL). Readers may refer to [7, 46] for a more comprehensive survey on MTL. Before the popularity of deep learning, traditional MTL usually focuses on hand-crafted features and can be generally divided into two categories: 1) multi-task feature learning, which aims to learn a shared/similar feature selection, latent space, or model parameters [1, 10]. 2) multi-task relation learning, which aims to quantify task relatedness via task clustering [18, 21] or task co-variance [12, 47]. However, they rely on hand-crafted features and the separation of feature generation and task learning may result in sub-optimal performance.

More recently, MTL takes advantage of the advancement of deep neural networks which can directly take raw, complex data such as images, audio, texts, and spatial-temporal data [14, 15, 49, 50] and learn in an end-to-end manner. Deep MTL integrates feature generation and task learning and simultaneously learns both of them [4]. In this domain, hard-parameter sharing [31, 48] requires to hard-code which part of the network is shared and which is not. Existing work usually shares the lower-level layers for representation learning (e.g., convolutions) while make higher-level layers (e.g., those for classification) separated across tasks. Some existing works extend hard-parameter sharing by considering Neural Architecture Search [9] like [27], [40], and [17]. Soft-parameter sharing based method has better flexibility where each task has its own models and regularization is used to enforce task relatedness by aligning their model parameters [8, 30]. To achieve both hard-parameter sharing’s conciseness and efficiency and soft-parameter sharing’s flexibility, some recent work [26, 39] shares the representation learning layers while exploits task relations in task-specific layers.

Saliency detection. Saliency detection is to identify the most important and informative part of input features. It has been applied to various domains including CV [13, 16], NLP [23, 32], etc. The salience map approach is exemplified by [45] to test a network with portions of the input occluded to create a map showing which parts of the data actually have an influence on the network output. In [36], a salience map can be created by directly computing the input gradient. Since such derivatives can miss important aspects of the information that flows through a network, a number of other approaches have been designed to propagate quantities other than gradients through the network. In CV domain, Class Activation Mapping (CAM, [52]) modifies image classification CNN architectures by replacing fully-connected layers with convolutional layers and global average pooling, thus achieving class-specific feature maps. Grad-CAM [34] generalizes CAM by visualizing the linear combination of the last convolutional layer’s feature map activations and label-specific weights, which are calculated by the gradient of prediction score w.r.t the feature map activations. Grad-CAM invokes different versions of backpropagation and/or activation, which results in aesthetically pleasing, heuristic explanations of image saliency. While there exist some other saliency-based methods along this research line including Guided Propagation [38], Deconvolutional Network [45], etc, they are designed only for specific architectures like ReLU Network for Guided Propagation.

3 PROPOSED METHOD

In this section, we introduce our proposed Saliency-regularized Deep Multi-task Learning (SRDML) method. We first review the pros and cons for each MTL method and describe our main motivation, then formally introduce our model and its objective function.

3.1 Problem Formulation

Consider a multi-task learning problem with $T$ tasks such that a dataset $\{ X, Y_1, Y_2, \ldots, Y_T \}$ is given with i.i.d training samples $X_i = \{ x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)} \}$, $Y_i = \{ y_1^{(t)}, y_2^{(t)}, \ldots, y_N^{(t)} \}$, where $n$ is the sample size and $(x_i^{(t)}, y_i^{(t)})$ is a pair of input and label such that $x_i^{(t)} \in X$ and $y_i^{(t)} \in \mathbb{R}$, $\forall i = 1, 2, \ldots, n$ and $t = 1, 2, \ldots, T$.

Given a predictor $g$ which factorizes as $g = f \circ h$, where “$\circ$” stands for functional composition. The function $h : X \rightarrow \mathbb{R}^K$ is called the feature or representation extraction part and is shared for all tasks, while $f : \mathbb{R}^K \rightarrow \mathbb{R}$ is a function defined on $\mathbb{R}^K$, a predictor
works with different architectures or neuron permutations could
weights in shallow MTL as
work: 1). Directly regularizing the prediction function of different
tasks is extremely hard. Existing work considered a reduced
in deep MTL due to the non-linear and non-parametric functions of
interpretability. This goal poses significant challenges to existing
problem by regularizing the feature weights of different
function family of functions with kth-order continuous derivatives for any
non-negative integer k. Given \( f_1, f_2 \in \mathcal{F} \), we have:
\[
f_1 = f_2 \quad \text{if and only if} \quad f'_1(x) = f'_2(x), \quad \forall x \in X \quad (1)
\]

Theorem 1. Define \( \mathcal{F} := \{ f \in \mathcal{C}^k : f(0) = 0 \} \), where \( \mathcal{C}^k \) is the
family of functions with kth-order continuous derivatives for any
non-negative integer k. Given \( f_1, f_2 \in \mathcal{F} \), we have:
\[
f_1 = f_2 \quad \text{if and only if} \quad f'_1(x) = f'_2(x), \quad \forall x \in X \quad (1)
\]

Our analysis above allows us to regularize the prediction functions
of different tasks in the functional space instead of parameter space.
The assumption over function family \( \mathcal{F} := \{ f \in \mathcal{C}^1 : f(0) = 0 \} \) is reasonable in practice since an all-zero input x simply corresponds to a "black" picture, and for any tasks we assume a black picture contains no useful information and should be classified as the negative sample (i.e., ground-truth label should be 0).

Key Merit 2: Inspiration from saliency map and enhancement of interpretability. Evaluating task relation with derivative similarity has justification from a saliency perspective. Saliency is a derivative of the prediction score w.r.t. input features, and it denotes the semantic features that influence the prediction most. In addition, similar tasks tend to have similar saliency, while dissimilar tasks tend to have dissimilar saliency. As shown in Figure 2, we enforce higher-level semantic features as saliency.

Many previous work have asserted that deeper representations in a CNN capture higher-level visual constructs [4]. Furthermore, convolutional layers naturally retain spatial information which is lost in fully connected layers, so we expect the last convolutional layers to have the best compromise between high-level semantics and detailed spatial information. By following a recent work called Grad-CAM [34], we use the gradient information flowing into the last convolutional layer of the CNN to capture the saliency map to each neuron for a particular task or class of interests.

3.3 Objective Function

We first give a formal definition of saliency. For example, in computer vision, given an input image \( I \), a classification ConvNet \( f \)
where \( W = \sum_{1 \leq i < j \leq T} \omega_{ij} \). Thanks to our normalization trick, the overall objective of SRDML is differentiable and can be trained in an end-to-end manner. We use standard gradient descent (e.g., Adam [19]) to solve Eq. 4, where we aim to learn multiple tasks and the task relationship simultaneously. Although the normalization trick introduced in Eq. 4 no longer guarantees that the hard constraint of the lower bound of all \( \omega_{ij} \) can be strictly satisfied, our empirical results show that our normalization trick works well in practice and SRDML can capture reasonable task relationship by optimizing Eq. 4 with finetuned hyperparameters.

A general overview of SRDML architecture can be found in Figure 3. First, the input image is fed into a shared feature extractor, which is implemented by a sequence of convolutional layers. Right after the feature extraction process, we obtain a set of flattened feature maps (shown as the blue bar in Figure 3), which contains high-level semantic information with respect to the original image [34]. On top of the feature map, each task-specific head will first calculate the saliency map with respect to its own prediction. Based on the saliency map for all the tasks, the task similarity can be calculated via some distance measure. Note that our overall framework is differentiable and can be trained in an end-to-end manner.

Last, how to share the convolutional layers is orthogonal to the focus of our paper because our SRDML focuses on task-specific layers instead of representation learning layers. This also implies whichever the best choice for convolutional layer sharing strategy can be utilized to work for our model. Our empirical results also demonstrated the reasonableness of the sharing policy that we used in this paper.

4 THEORETICAL ANALYSIS

In this section, we present the theoretical analyses of our SRDML model. First, we prove that our proposed regularizer can help reduce the generalization error. Second, we formally analyze the relation between SRDML and other MTL methods. We put all formal proofs in the appendix due to the limited space.

4.1 Generalization Error Bound

Here we show the generalization bound of our model. Our main contribution here is we proved that our proposed regularization term can help reduce the generalization error.

For simpler notation, define

\[
F_{(a)} := \{ f \in F^T : \forall 1 \leq i < j \leq T, x \in X, \quad dist(\nabla_x f_i, \nabla_x f_j) \leq \epsilon_{ij}, \quad \sum_{1 \leq i < j \leq T} \epsilon_{ij} \leq \alpha \} \tag{5}
\]

where \( f = (f_1, f_2, \cdots, f_T) \) is the vectorization of each task’s function, and \( \{\epsilon_{ij}\}_{1 \leq i < j \leq T} \) is a set of global slack variables. Hence, the optimization problem of Eq. 2 can be simplified as

\[
\min_{h, f_1, \cdots, f_T, \omega \geq 0} \frac{1}{n} \sum_{t=1}^{T} \sum_{i=1}^{n} L_t(f_t(h(x_i)), y_i^{(t)}) \tag{6}
\]

Before introducing the theorem, we make the following standard assumptions over the loss function:
ASSUMPTION 1 ([29]). The loss function \( \mathcal{L} \) has values in \([0, 1]\) and has Lipschitz constant \( l \) in the first argument, i.e.: (1) \( \mathcal{L}(y, y') \in [0, 1] \) (2) \( \mathcal{L}(y, y') \leq y, \forall y' \).

Different Lipschitz constants can be absorbed in the scaling of the predictors and different ranges than \([0, 1]\) can be handled by a simple scaling of our results.

**Definition 1 (Expected risk, Empirical risk).** Given any set of function \( h, f_1, \cdots, f_T \), we denote the expected risk as:

\[
\mathbb{E}(h, f_1, \cdots, f_T) := \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}(X_t, Y_t) - \mu_t \mathcal{L}_t(f_t(h(X_t)), Y_t)
\]

Given the data \( Z = (X, Y) \), the empirical risk is defined as:

\[
\hat{\mathcal{E}}(h, f_1, \cdots, f_T|Z) := \frac{1}{T} \sum_{t=1}^{T} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_t(f_t(h(x_i)), y_i(t))
\]

**Definition 2 (Global optimal solution, Optimized solution).** Denote \((h^*, f^*)\) as the global optimal solution of the expected risk:

\[
(h^*, f^*) := \arg \min_{h \in \mathcal{H}, f \in \mathcal{F}(a)} \mathbb{E}(h, f_1, \cdots, f_T)
\]

and \((\hat{h}, \hat{f})\) as the optimized solution by minimizing the empirical risk:

\[
(\hat{h}, \hat{f}) := \arg \min_{h \in \mathcal{H}, f \in \mathcal{F}(a)} \hat{\mathcal{E}}(h, f_1, \cdots, f_T|Z)
\]

The following theorem provides theoretical guarantee of our proposed method’s generalizability.

**Theorem 2 (Generalization Error).** Let \( \delta > 0 \) and \( \mu_1, \mu_2, \cdots, \mu_T \) be the probability measure on \( X \times \mathbb{R} \). With probability of at least \( 1 - \delta \) in the draw of \( Z = (X, Y) \sim \prod_{t=1}^{T} P_{X_t Y_t} \), we have:

\[
\mathbb{E}(\hat{h}, \hat{f}) - \mathbb{E}(h^*, f^*) \leq c_1 \mathcal{L}(h(X)) + c_2 B \frac{\lambda_{\min}^{-1}}{n \sqrt{T}} \mathcal{L}(h(X)) + c_3 B \frac{\sqrt{\ln(4/\delta)}}{n \sqrt{T}}
\]

where \( c_1, c_2 \) are universal constants, \( \mathcal{L}(h(X)) \) is the Gaussian average defined as \( \mathcal{L}(h(X)) = \mathbb{E}[\sup_{\rho \in \mathcal{P}} \mathbb{E}[\sum_{i=1}^{n} \mathcal{L}_t(h(X_i))]_1] \), where \( (\mathcal{L}_t) \) is i.i.d standard normal variables. \( L \) is the Laplacian matrix of graph with \( T \) vertices and edge-weights \( (x_i) \) s.t. \( \sum_{i=1}^{T} \mathcal{L}(x_i) \) is its smallest non-zero eigenvalue. \( B \) is any positive value that satisfies that the condition \( \sum_{i=1}^{T} \mathcal{L}(x_i) \cdot \text{dist}^2(\nabla h_{f_i}, \nabla h_{f_j}) \leq B^2 \).

Some remarks over Theorem 2: 1. The first term of the bound can be interpreted as the cost of estimating the shared representation learning function \( h \in \mathcal{H}. \) This term is typically of order \( \frac{1}{n} \).

2. The last term contains the confidence parameter. According to [29] the constant \( c_1 \) is pretty large, so the last term typically makes limited contribution in the bound.

3. The second or middle term contains the cost of estimating task-specific predictors \( f \in \mathcal{F}. \) and this term is typically of order \( \frac{1}{\sqrt{\mathcal{L}}} \). Here the positive constant \( B \) provides important insights into the relationship between our proposed regularizer and the error bound. The smaller our regularization term becomes, the smaller values \( B \) could take and in turn reduces the second term in the bound. In general, our generalization error result bounds the gap between the test error of the model trained from finite samples and that trained from infinite data, namely the theoretically optimal model/solution. In other words, Theorem 2 provides theoretical guarantee for our performance on actual test set.

### 4.2 Relation with Other MTL Frameworks

In this section, we mathematically elucidate the relation and difference between our proposed SRDML and other MTL methods, i.e., shallow MTL and deep MTL. Proof can be found in the appendix.

**Natural generalization of shallow MTL.** Following [46], traditional multi-task learning methods (i.e., linear model based MTL) can be generally classified into two categories: multi-task feature learning and multi-task relation learning, with objective function \( \tilde{w}_{T, B} L(W, b) + \lambda_{\mathcal{T}}/2 \cdot \text{tr}(W^T \Sigma^{-1} W) \) and \( \tilde{w}_{T, B} L(W, b) + \lambda_{\mathcal{T}}/2 \cdot \text{tr}(W^T W^T) \), where \( \Theta \) and \( \Sigma \) models the covariance between different features and tasks, respectively. For any regularization-based shallow MTL defined as above, it can be formulated as a special case under the general framework of SRDML, with identity feature extraction function \( h \), linear task-specific function \( f \) and the corresponding regularizer on the input gradients.

**Relation with deep MTL.** Define two hyperparameters: 1. The coefficient of regularizer in SRDML \( \lambda \) and 2. the number of layers \( T \) before which the model is shared cross tasks. When \( \lambda \) equals 0 and \( T \) is greater than 1 and less than \( L \) (total number of layers), SRDML degenerates to hard-parameter sharing. On the other hand, when \( T \) equals to 1 and \( \lambda \) is greater than 0, our SRDML is equivalent to soft-parameter sharing. Hence, both hard-parameter sharing and soft-parameter sharing can be formally formulated as special cases of our proposed SRDML framework.

### 5 EXPERIMENTS

In this section, we validate SRDML on synthetic and real-world datasets against multiple methods, on various aspects including performance, sensitivity, qualitative analysis and ablation study. The experiments were performed on a 64-bit machine with 4-core Intel Xeon W-2123 @ 3.60GHz, 32GB memory and NVIDIA Quadro RTX 5000. Code available at https://github.com/BaiTheBest/SRDML.

#### 5.1 Experimental Settings

**Controlled Synthetic Dataset.** We first check the validity of SRDML on a controlled regression synthetic dataset. We generate \( T \) tasks \((T = 12)\) and for each task \( t \) we generate \( m \) samples \((m = 100)\). The input data \( X_t \in \mathbb{R}^{m \times d} \) for each task \( t \) is generated from \( X_t \sim N(y_t, 1) \) with mean vector \( y_t \) and identity covariance matrix \( I \). Next, we generate feature weight \( W \) by the following steps: 1) Generate two base feature weights. As shown in Figure 4a, the first base feature weight (on the LHS column) corresponds to \( w_1 = (1; 0)^T \) and the second base feature weight (on the RHS column) corresponds to \( w_2 = (0; 1)^T \), where 1 and 0 each denotes a 10-dimensional all-one and all-zero vector respectively. In this way, \( w_1 \) and \( w_2 \) can simulate two different regions in the input.
X since the regions zeroed out by \( \mathbf{w} \) will not be helpful in corresponding tasks. 2) Generate task specific feature weight. Based on \( \mathbf{w}_1 \) and \( \mathbf{w}_2 \), we further consider creating different levels of saliency by multiplying the base feature weights by some magnitude parameter. Here we select 3 different magnitude parameters to create different levels of saliency for each base feature weight, and for each level of saliency we create two tasks which are basically twin tasks. For example, in Figure 4a, task 1 and task 2 are twin tasks which share the same level of saliency, and the lightest blue color means they are generated by the lowest magnitude parameter. We denote each generated task-specific feature weight as \( \mathbf{w}_i \), \( i \in \{1, 2, \ldots, T\} \). The aforementioned logistics are basically symmetric for \( \mathbf{w}_1 \) and \( \mathbf{w}_2 \). 3) Add noise and create labels. We first inject some noise into each task’s feature weight by randomly flipping the sign of the value in some positions of each \( \mathbf{w}_i \). The proportion of the flipped positions is controlled to guarantee the overall pattern can be well kept. Then, we generate the label for each task by \( \mathbf{Y}_i = \mathbf{X}_i \cdot \mathbf{w}_i + \epsilon_i \), where \( \epsilon_i \sim \mathcal{N}(0, 0.1 \cdot \mathbf{I}) \) is random normal noise.

**Real-world Dataset.** We evaluate the proposed method on 3 real-world benchmarks with varying number of tasks and difficulty, including: multi-task version of CIFAR-10 [20] (CIFAR-MTL), a modified version of CelebA [25] and a modified version of MS-COCO [24]. To follow our model’s assumption, all tasks are image classification ones. For CIFAR-MTL, we follow existing work [33] to create one task for each of the 10 classes in original CIFAR-10 dataset. There are 10 binary classification tasks with 2k training samples and 1k testing samples per task. CelebA has 200 thousand images of celebrity faces and each image is labeled with 40 facial attributes. We follow existing work [51] to select 16 attributes more related to face appearance and ignore attributes around decoration such as eyeglasses and hat for our experiments. We randomly selected 30k training samples and include whole validation and test set. For MS-COCO we select 19 types of objects and remove those with too sparse labels. We include all images that contain at least two of the 19 types of objects and randomly split them into training and testing set by half. All results are reported on the test set. For hyperparameter tuning of our method, without further specification, we applied grid search on the range of \( \{10^{-3}, 5 \cdot 10^{-3}, \ldots, 0.5, 1\} \) for the regularization coefficient.

**Comparison Methods** We compare SRDML with various existing methods, including two baselines, three shallow and five deep state-of-the-art MTL methods:

- **Practical Baselines:** 1) *Single Task Learning (STL)* is to train a separate predictor for each task independently. 2) *Hard Parameter Sharing (Hard-Share)* considers a shared representation learning backbone (e.g., convolutional layers in CNN) and task-specific prediction head.

- **Shallow MTL Methods:** 1) *Lasso* is an \( \ell_1 \)-norm regularized method which introduce sparsity into the model to reduce model complexity and feature learning, and that the parameter controlling the sparsity is shared among all tasks. 2) *Joint Feature Learning (JFL)* [10] assumes the tasks share a set of common features that represent the relatedness of multiple tasks. 3) *Robust Multi-task Learning (RMTL)* [6] method assumes that some tasks are more relevant than others. It assumes that the model \( W \) can be decomposed into a low rank structure \( L \) that captures task-relatedness and a group-sparse structure \( S \) that detects outliers.

- **Deep MTL Methods:** *Multilinear Relationship Networks (MRNs)* places a tensor normal prior on task-specific layers of the deep multi-task learning model [26]. 2) *Multi-gate Mixture-of-Experts (MMoE)* [28] adapt the Mixture-of-Experts (MoE) structure to multi-task learning by sharing the expert submodels across all tasks, while also having a gating network. 3) *Progressive Layered Extraction (PLE)* [41] separates shared components and task-specific components explicitly and adopts a progressive routing mechanism to extract and separate deeper semantic knowledge gradually, improving efficiency of joint representation learning and information routing across tasks in a general setup. 4) *Multi-task Learning as Multi-Objective Optimization (MGDA-UB)* [35] considers multi-task learning from optimization perspective by using Pareto optimality and Multiple Gradient Descent Algorithm. 5) *Gradient Surgery for Multi-task Learning (PCGrad)* [44] aims to solve the problem of gradient interference by gradient surgery, which is basically by gradient projection to make sure the gradients of different tasks have direction smaller than 90°.

Figure 4: Experimental results on synthetic dataset. (a): Ground-truth of each task’s feature weight. (b): Task relation learned by our proposed SRDML. Tasks from different bases show strong independency (as in dark purple), tasks from the same bases show clear similarities (as in light green), and each pair of twin tasks show very strong similarities (as in yellow). (c): The performance improvement of SRDML over single task learning in RMSE (blue bar) and MAE (green bar). As shown, SRDML model generally outperforms STL on the synthetic dataset by a large margin. (d): Sensitivity analysis on regularization coefficient.
Table 2: Performance (%) on real-world large-scale multi-task learning datasets. Our proposed SRMTL outperforms most comparison methods on all three datasets. Bold and underline score are for the best and second best methods, respectively.

| Model          | CIFAR-MTL | CelebA     | COCO      |
|----------------|------------|------------|-----------|
|                | Accuracy  | AUC | Precision | Recall | Accuracy  | AUC | Precision | Recall | Accuracy  | AUC | Precision | Recall |
| STL            | 92.65     | 66.20      | 71.32      | 69.83   | 86.83     | 90.96 | 70.53      | 60.39   | 79.23     | 62.91 | 32.23      | 27.04   |
| Hard-Share     | 94.70     | 95.56      | 76.30      | 72.28   | 89.24     | 91.38 | 71.40      | 58.84   | 85.11     | 73.68 | 37.43      | 19.84   |
| Lasso          | 91.48     | 86.64      | 68.90      | 24.74   | 76.55     | 66.69 | 37.38      | 36.62   | 78.36     | 64.40 | 28.53      | 28.61   |
| L21            | 91.50     | 87.58      | 68.01      | 29.32   | 76.09     | 66.12 | 37.11      | 36.13   | 75.07     | 65.02 | 28.95      | 27.34   |
| RMTL           | 92.28     | 85.65      | 61.34      | 28.15   | 73.52     | 66.99 | 37.48      | 36.74   | 76.87     | 65.01 | 29.28      | 28.43   |
| MRN            | 94.51     | 96.67      | 79.94      | 76.95   | 89.35     | 91.54 | 71.51      | 64.64   | 85.13     | 75.88 | 32.73      | 25.89   |
| MMoE           | 93.53     | 93.17      | 73.42      | 69.32   | 77.57     | 67.84 | 68.79      | 58.92   | 81.20     | 62.37 | 33.08      | 26.14   |
| PLE            | 94.01     | 93.32      | 75.26      | 70.15   | 83.21     | 69.32 | 70.03      | 59.72   | 82.53     | 63.42 | 35.27      | 27.53   |
| MGDA-UB        | 90.74     | 84.38      | 57.80      | 24.10   | 90.03     | 92.92 | 73.42      | 62.65   | 84.51     | 73.68 | 36.17      | 16.08   |
| PCGrad         | 95.11     | 96.69      | 79.03      | 74.82   | 90.11     | 92.87 | 73.51      | 62.92   | 85.42     | 74.39 | 34.52      | 25.26   |
| SRDML          | 95.82     | 96.43      | 81.22      | 75.93   | 90.15     | 92.95 | 73.87      | 64.91   | 85.68     | 76.77 | 35.82      | 28.73   |
| SRDML (w/. PCGrad) | 96.03   | 96.72      | 82.59      | 77.01   | 90.26     | 93.01 | 73.93      | 65.30   | 85.87     | 78.38 | 36.14      | 30.02   |

Since PCGrad targets gradient interference, it is only applied onto the shared layers of each model to avoid the contradiction of each task’s gradients. Specifically, PCGrad is applied onto the shared convolutional layers.

Implementation Details. All shallow MTL methods are implemented according to standard package MALSAR [53]. Deep MTL methods and our SRDML are built based on VGG-16 [37], which is a very popular architecture in computer vision. The convolutional layers are followed by one fully connected layer with 128 hidden neurons and one classification layer for our SRDML. Each model is trained by Adam [19]. For PCGrad, due to the fact that it is a gradient surgery method which is model-agnostic and can be applied onto any deep MTL method, we report its performance by combining it with the best baseline on each real-world dataset (i.e., Hard-Share on CIFAR-MTL, MGDA-UB on CelebA, MRN on COCO). In addition, we also consider applying PCGrad onto our own method SRDML, resulting in two versions of our method, namely SRDML and SRDML with PCGrad.

5.2 Experimental Results

Effectiveness on controlled synthetic dataset. The empirical results on the regression synthetic dataset demonstrate that our model can generally outperform single task learning and is capable to capture the ground-truth task relations. Quantitative evaluation in Figure 4c shows that SRDML can outperform single task learning in general, which can be attributed to the effective knowledge sharing between task-specific layers. In addition, the task relationship pattern (i.e., $w_{ij}$ in Eq. 4) learned by SRDML as shown in Figure 4b is accurate and reasonable, since tasks belong to different bases and only differ in the magnitude.

Sensitivity analysis. The sensitivity of hyperparameter $\lambda$ in SRDML on synthetic dataset is shown in Figure 4d. As can be seen, the optimal value for $\lambda$ is around 0.5 measured by RMSE. The general “U” shape is potentially reasonable because as $\lambda$ goes to infinity the regularization term would dominate the overall objective function while too small $\lambda$ will reduce the functionality of the regularizer and finally degenerate to single task learning.

Effectiveness on real-world datasets.

- CIFAR-MTL: Table 2 shows the performance results of our proposed SRDML and other baselines on CIFAR-MTL dataset. We can make the following observations from the results. 1) Deep multi-task learning models generally outperform shallow ones by a great margin, which confirms the importance of learning the deep representation features as well as the shared policy of feature extraction part which allows knowledge transfer across tasks. 2) Our proposed SRDML outperforms baselines in majority of metrics and achieves comparable performance in rest. 3) Combining with PCGrad can further improve the performance of SRDML due to the mitigated negative transfer in the shared convolutional layers by gradient surgery of PCGrad.

- CelebA: In this case, we tackle a larger and more challenging benchmark, where we tailored the dataset to contain 16 binary classification tasks with each one corresponding to a certain human’s facial feature. As shown in Table 2, our model outperforms all comparison methods in majority of metrics, which is attributed to the potential fact that the salient regions in some tasks are close to those in the related tasks. For example, there are two tasks to classify whether a celebrity’s beard is goatee or mustache, respectively. For both tasks the salient regions are highly overlapped around the mouth area (as can be seen in Section “Saliency map visualization” in appendix) so enforcing similar input gradients around the mouth area could improve the knowledge transfer and achieve better performance.

- COCO: To evaluate our model under various settings, we consider COCO which contains different types of objects like human, animals, vehicles, furniture, etc, and each type object has varying rate of occurrence. In Table 2, we report the task-average classification error with lower values indicating better performance. As shown in Table 2, our proposed SRDML outperforms all the baselines by a great margin. This experiment also validates the effectiveness of our model when the number of tasks is relatively large and the image context is complicated. Moreover, MMoE and PLE perform generally not quite well probably due to the fact...
Table 3: Sensitivity analysis on regularizer coefficient when tasks are contradicting. Our regularizer coefficient can adaptively reduce to zero and avoid negative transfer.

| λ    | 1   | 0.1  | 0.01 | 0.001 | 0  |
|------|-----|------|------|-------|----|
| RMSE | 2.726 | 1.550 | 1.405 | 1.393 | 1.392 |
| MAE  | 2.198 | 1.260 | 1.127 | 1.127 | 1.126 |

Adaptive regularizer on contradicting tasks. In this section, we conducted another sensitivity analysis when all tasks compete (we generate such synthetic dataset by following a similar procedure introduced in Section 5.1), and the results in Table 3 demonstrate the efficacy of our regularization term, which can adaptively decrease the task-similarity weight to zero and avoid competition.

6 CONCLUSION

Learning interpretable task relations is challenging in multi-task learning problem. In this paper, we proposed Saliency-regularized Deep Multi-task Learning (SRDML) framework, which regularizes the input gradient of different tasks by saliency and achieves good task relation interpretability. Instead of regularizing parameters like existing work, we directly regularize in functional space, which allows better expressiveness. Theoretical analyses show that our regularizer can help reduce the generalization error. Experiments on multiple synthetic and real-world datasets demonstrate the effect and efficiency of our methods in various metrics, compared with several comparison methods and baselines. The reasonableness of the task relation learned by SRDML is also validated on different challenging real-world datasets.

ACKNOWLEDGEMENT

This work was supported by the National Science Foundation (NSF) Grant No. 1755850, No. 1841520, No. 2007716, No. 2007976, No. 1942594, No. 1907805, a Jeffress Memorial Trust Award, Amazon Research Award, NVIDIA GPU Grant, and Design Knowledge Company (subcontract number: 10827.002.120.04).
REFERENCES

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2008. Convex multi-task feature learning. Machine learning 73, 3 (2008), 243–272.

[2] Andreas Argyriou, Charles A Micchelli, Massimiliano Pontil, and Yiming Ying, 2007. A spectral regularization framework for multi-task feature learning. In NIPS, Vol. 1290. Citéseer, 1296.

[3] Jonathan Baxter. 2000. A model of inductive bias learning. Journal of artificial intelligence research 12 (2000), 149–198.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence 35, 8 (2013), 1798–1828.

[5] Rich Caruana. 1997. Multitask learning. Machine learning 28, 1 (1997), 41–75.

[6] Jianhui Chen, Jiayu Zhou, and Jieping Ye. 2011. Integrating low-rank and group-sparse structures for robust multi-task learning. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, 42–50.

[7] Michael Crawford. 2020. Multi-Task Learning with Deep Neural Networks: A Survey. arXiv preprint arXiv:2009.09796 (2020).

[8] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (volume 2: short papers). 845–850.

[9] Thomas Elsken, Jan Hendrik Metzen, Frank Hutter, et al. 2019. Neural architecture search: A survey. J. Mach. Learn. Res. 20, 55 (2019), 1–21.

[10] An Evgeniou and Massimiliano Pontil. 2007. Multi-task feature learning. Advances in neural information processing systems 19 (2007), 41.

[11] Theodoros Evgeniou, Charles A Micchelli, Massimiliano Pontil, and John Shawe-Taylor. 2005. Learning multiple tasks with kernel methods. Journal of machine learning research 6, 4 (2005).

[12] Theodoros Evgeniou and Massimiliano Pontil. 2004. Regularized multi-task learning. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 109–117.

[13] Yuyang Gao, Tong Sun, Guanghai Bai, Carrie Gu, Sungsoo Hong, and Liang Zhao. 2022. RES: A Robust Framework for Guiding Visual Explanation. In Proceedings of the 28th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM.

[14] Yuyang Gao and Liang Zhao. 2018. Incomplete label multi-task ordinal regression for spatial event scale forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32.

[15] Yuyang Gao, Liang Zhao, Lingfei Wu, Yangfan Ye, Hui Xiong, and Chaowei Yang. 2019. Incomplete label multi-task deep learning for spatio-temporal event subtype forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33. 3638–3646.

[16] Stan Goffman, Lili Zelink-Manor, and Ayellet Tal. 2011. Context-aware saliency detection. IEEE transactions on pattern analysis and machine intelligence 34, 10 (2011), 1915–1926.

[17] Pengcheng Gao, Chen-Yu Lee, and Daniel Ulbricht. 2020. Learning to branch for multi-task learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34.

[18] Wanli Ouyang, Xiao Chu, and Xiaogang Wang. 2014. Multi-source deep learning and machine intelligence research. Journal of artificial landmark detection by deep multi-task learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33.

[19] Yifan Sun, Jyunyong Liu, Meng Zhao, and Xudong Gong. 2020. Progressive layer extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In Fourteenth ACM Conference on Recommender Systems. 269–278.

[20] Johnny Torres, Guanghai Bai, Juxiiang Yang, Liang Zhao, Carmen Vaca, and Cristina Abril. 2021. Sign-regularized Multi-task Learning. arXiv preprint arXiv:2102.11191 (2021).

[21] Sen Wu, Hongyang R Zhang, and Christopher Ré. 2020. Understanding and improving information transfer in multi-task learning. arXiv preprint arXiv:2005.09944 (2020).

[22] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. arXiv preprint arXiv:2001.06782 (2020).

[23] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In European conference on computer vision. Springer, 818–833.

[24] Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. IEEE Transactions on Knowledge and Data Engineering (2021).

[25] Yu Zhang and Dit-Yan Yeung. 2012. A convex formulation for learning task relatedness in multi-task learning for images in context. In Proceedings of the 2019 on International Conference on Multimedia Retrieval. 78–86.

[26] Xi Meng Sun, Rameswar Panda, Rogerio Feris, and Kate Saenko. 2019. Adashare: Learning what to share for efficient deep multi-task learning. arXiv preprint arXiv:1911.12423 (2019).

[27] Hongyang Tan, Junning Liu, Meng Zhao, and Xudong Gong. 2020. Progressive relatedness in multi-task learning with applications in image retrieval. arXiv preprint arXiv:1911.12423 (2019).

[28] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. 2016. The benefit of multitask representation learning. Journal of Machine Learning Research 17, 81 (2016), 1–32.

[29] Jianhui Chen, Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 1930–1939.
Appendix

In this appendix, we describe detailed experimental setup, additional experimental results, and complete proofs. Our code is available at https://github.com/BaiTheBest/SRDML. Please note that the code is subject to reorganization to improve the readability.

A THEORETICAL PROOF

In this section, we provide the formal proof for all the theories presented in Saliency-regularized Deep Multi-task Learning paper.

A.1 Proof of Theorem 1

**Proof.** Suppose \( X \subseteq \mathbb{R}^{K} \) is an open set and \( f_{1}, f_{2} : X \rightarrow \mathbb{R} \), where both functions are differentiable and equal to zero at the origin.

\( \Rightarrow \): This direction is obvious, since two exactly the same functions will have the same gradient at any input point.

\( \Leftarrow \): Given \( \nabla f_{1}(x) = \nabla f_{2}(x) \), we know that

\[
\frac{\partial f_{1}}{\partial x_{k}} = \frac{\partial f_{2}}{\partial x_{k}}, \quad k = 1, 2, \ldots, K, \quad x \in X
\]  

(12)

For arbitrary \( k \), by \( \frac{\partial f_{1}}{\partial x_{k}} = \frac{\partial f_{2}}{\partial x_{k}} \), we know that

\[
\exists c_{k}(x_{1}, \ldots, x_{k}, \ldots), \quad s.t., \quad f_{1} = f_{2} + c_{k}
\]  

(13)

Meanwhile, notice \( \forall k \neq k \), \( \frac{\partial c_{k}}{\partial x_{k}} = 0 \) (otherwise, contradiction!) Hence, \( dc_{k} = 0 \) and we know \( c_{k} \) is a constant. Also, the value of \( c_{k} \) does not depend on \( k \) since for all \( k, l \), we have \( f_{1} - f_{2} = c_{k} = c_{l} \), thus there exists a constant \( c \) such that \( f_{1} = f_{2} + c. \) Finally, by the boundary condition that \( f_{1}(0) = f_{2}(0) = 0 \), we know that \( c = 0 \), i.e., \( f_{1} = f_{2} \), which finishes the proof. \( \square \)

A.2 Proof of Theorem 2

In this section, we provide the proof of our model’s generalization error bound. First, we introduce some definitions and lemmas which will be continuously used, and at the end of this section we present the proof for Theorem 2.

In general, we will use \( Y \) to denote a generic vector of i.i.d standard normal variables, whose dimension will be clear in context. In addition, without further specification, we will use \( K, T, n \) to denote the (flattened) dimension of the output space from the feature extraction function \( h \), number of tasks, and number of training samples, respectively. We denote the representation class for task-specific function \( \hat{f} \) and representation extraction function \( h \) as \( \mathcal{F} \) and \( \mathcal{H} \), respectively. Two hypothesis classes here can be very general, and the only assumption here is that \( \forall f \in \mathcal{F}, f \) has Lipschitz constant at most \( L \), for any positive \( L \).

**Definition 3.** Given a set \( V \subseteq \mathbb{R}^{n} \), define the Gaussian average of \( V \) as

\[
G(V) := \mathbb{E} \sup_{\gamma, o} = \mathbb{B} \sup_{\gamma, o} \sum_{i=1}^{n} \gamma_{i} o_{i}
\]

(14)

As mentioned in section 3.1 in main paper, we denote the feature representation learning part as function \( h \in \mathcal{H} \). As we will see later, the complexity of representation class \( \mathcal{H} \) is important in our proof for the error bound, so we define a measure of its complexity by Gaussian average.

**Definition 4.** Given observed input data \( X \in \mathcal{X}^{Tn} \), define a random set \( \mathcal{H}(X) \subseteq \mathbb{R}^{K} \) by

\[
\mathcal{H}(X) := \{ h_{k}(x_{i}) : h \in \mathcal{H} \}.
\]  

(15)

The Gaussian average over \( \mathcal{H}(X) \) can be defined accordingly as

\[
G(\mathcal{H}(X)) = \mathbb{E}[\sup_{h \in \mathcal{H}} \sum_{k=1}^{K}\sum_{i=1}^{n} h_{k}(x_{i})|x_{i}]
\]  

(16)

The following lemmas are useful in our proof later, and we introduce them here in advance.

**Lemma 1.** \( \forall A, C \subseteq \mathbb{R}^{m \times n} \) and \( B \in \mathbb{R}^{m \times m} \),

\[
tr(AB) = \sum_{i,j} B_{ij} \sum_{k,l} A_{ik} C_{jk}.
\]  

(17)

**Lemma 2.** Suppose \( X \subseteq \mathbb{R}^{K} \) is an open set, and two differentiable functions \( f_{1}, f_{2} : X \rightarrow \mathbb{R} \), \( x \in X \), if \( \exists B > 0, \quad s.t \parallel \nabla f_{1}(x) - \nabla f_{2}(x) \parallel < B \) then

\[
\lim_{\Delta x \rightarrow 0} \left| \frac{f_{1}(x + \Delta x) - f_{1}(x)}{\parallel \Delta x \parallel} - \frac{f_{2}(x + \Delta x) - f_{2}(x)}{\parallel \Delta x \parallel} \right| < B.
\]

(19)

Given everything above, we can prove our Theorem 2. However, the formal proof of Theorem 2 is quite complicated and due to the limited space of appendix here, we decide to put the formal proof for our Theorem 2 into the link: https://drive.google.com/file/d/1Mtb5zPfIP9P931V5vgsXUXU07uYjfi0Ad/view?usp=sharing. Please refer to the link for our formal proof.

A.3 Proof of Section 4.2

Natural generalization of shallow MTL

**Proof.** Basically, when the feature extraction function \( h \) is identity function and each task-specific function \( f_{t}, t = 1, 2, \ldots, T \) are linear functions, we know for any input \( x \in \mathcal{X} \),

\[
h(x) = x, \quad \nabla f_{t}(x) = w_{t}, \quad \forall t
\]

(20)

where \( w_{t} \) is the model parameter of linear model \( f_{t} \). Hence, denote \( W = [w_{1}; w_{2}; \ldots; w_{T}] \) and take the \( dist() \) function in Eq. 3 to be inner product, by Lemma 1 we have

\[
\sum_{i,j} \omega_{ij} \cdot dist(\nabla f_{i}(x), \nabla f_{j}(x)) = \sum_{i,j} \omega_{ij} \cdot \langle w_{i}, w_{j} \rangle = tr(W^{T}W)
\]

(21)

where \( \Omega = (\omega_{ij}) \). Let \( \Omega \) to be either \( \Theta^{-1} \) or \( \Sigma^{-1} \) as in section 4.2 can finish the proof. \( \square \)

Relation with deep MTL

**Proof.** First, we define two hyperparameters:

- \( \lambda \): The coefficient of our regularizer in SRDML
- \( \ell \): The index of the layer before which the model is shared across different tasks.
We add an experiment on our method with normalizing the input without normalization) on ALL 3 real-world dataset we used in our original paper. As shown in the Table 5, adding normalization did not obviously change the performance in task-average classification error. The task-average classification error decreased by \(<0.2\%\) on CIFAR-MTL and increased by around 0.1\% on CelebA and COCO.

One explanation is, for similar tasks like “Black hair” and “Brown hair” both generally overlap around the hair related tasks. For example, the saliency map generated for “Black hair” and “Brown hair” both generally overlap around the hair region of the woman, and the saliency map generated for three types of beard all overlap around the mouth and beard region of the man. Notice that the quality of saliency itself is not the main focus of this paper, but instead we are more interested in the task relation induced by the saliency map similarity (i.e., saliency across tasks).

In this section, we provide more remarks on our main theorem, namely Theorem 2, for better understanding.

Remark 1. The equation above bounds the gap between the test error of the model trained from finite samples and that trained from infinite data, namely the theoretically optimal model/solution. In other words, Theorem 2 provides theoretical guarantee for our performance on actual test error.

Remark 2. In Eq. 2- 6, we assume all tasks share the same set of X which is a very common case in Multi-task Learning on image dataset. Theorem 2 does not need different tasks to have different X(t), since \(\mu_1 = \mu_2 = \cdots = \mu_T = \mu\) is a special case of the version in Theorem 2. Our current assumption is actually a more general one and can handle the case in Eq. 2- 6.

### B ADDITIONAL DETAILS ON SYNTHETIC DATASET GENERATION

“What are base feature weights?”. Since we want to generate tasks with different level of similarity in our synthetic dataset, we achieved it by controlling the similarity in the feature weight (i.e., \(w\)) of different tasks. The base feature weights \(w_1\) and \(w_2\) are basically two vectors (with length equal to number of features) for generating the feature weight vectors for all the tasks. We call them “base” feature weight because they serve as the base vector or unit vector for generating all the tasks’ feature weights. In addition, \(w_1\) and \(w_2\) are orthogonal and each has length 1 in any dimension.

“How are base feature weights used?”. The base feature weights are used to generate each task’s feature weight in the following steps: 1) We choose which base the current task belongs to. In our setting, we chose the first half of tasks to belong to the first base (i.e., \(w_1\)) and the second half of tasks to belong to the second base (i.e., \(w_2\)). Since two bases are orthogonal, they can actually simulate two non-overlapping regions in pictures which means tasks from different bases should not be similar while those from the same base should be similar since, they focus on the same region. 2) Within each base, we multiply the base vector (i.e., \(w_1\) and \(w_2\)) by some positive integers to generate the actual feature weight for the tasks. For example, we multiply \(w_1\) by integer 1, 2 and 3 to generate the feature weight vectors for the first half of tasks.

### C NORMALIZATION ON INPUT GRADIENT

We add an experiment on our method with normalizing the input gradients and compare its results with our original method (i.e., without normalization) on ALL 3 real-world dataset we used in our original paper. As shown in the Table 5, adding normalization did not obviously change the performance in task-average classification error. The task-average classification error decreased by \(<0.2\%\) on CIFAR-MTL and increased by around 0.1\% on CelebA and COCO.

One explanation is, for similar tasks like “Black hair” and “Brown hair” both generally overlap around the hair related tasks. For example, the saliency map generated for “Black hair” and “Brown hair” both generally overlap around the hair region of the woman, and the saliency map generated for three types of beard all overlap around the mouth and beard region of the man. Notice that the quality of saliency itself is not the main focus of this paper, but instead we are more interested in the task relation induced by the saliency map similarity (i.e., saliency across tasks).

|          | CIFAR-MTL | CelebA | MS-COCO |
|----------|-----------|--------|---------|
| SRDML w/o normalization | 4.18      | 9.91   | 14.32   |
| SRDML w/ normalization   | 4.02      | 10.03  | 14.41   |

### D ADDITIONAL QUALITATIVE ANALYSIS

We also conduct a set of qualitative analysis experiment on the saliency map generated by our proposed SRDML on similar or related tasks. As can be seen in Figure 6 and Figure 7, our proposed SRDML can generate saliency map focusing on similar regions for related tasks. For example, the saliency map generated for “Black hair” and “Brown hair” both generally overlap around the hair region of the woman, and the saliency map generated for three types of beard all overlap around the mouth and beard region of the man. Notice that the quality of saliency itself is not the main focus of this paper, but instead we are more interested in the task relation induced by the saliency map similarity (i.e., saliency across tasks).

### E ADDITIONAL REMARKS ON THEOREM 2

In this section, we provide more remarks on our main theorem, namely Theorem 2, for better understanding.

Remark 1. The equation above bounds the gap between the test error of the model trained from finite samples and that trained from infinite data, namely the theoretically optimal model/solution. In other words, Theorem 2 provides theoretical guarantee for our performance on actual test error.

Remark 2. In Eq. 2- 6, we assume all tasks share the same set of X which is a very common case in Multi-task Learning on image dataset. Theorem 2 does not need different tasks to have different X(t), since \(\mu_1 = \mu_2 = \cdots = \mu_T = \mu\) is a special case of the version in Theorem 2. Our current assumption is actually a more general one and can handle the case in Eq. 2- 6.