A Weakly Supervised Fine Label Classifier Enhanced by Coarse Supervision

Fariborz Taherkhani, Hadi Kazemi, Ali Dabouei, Jeremy Dawson, Nasser M. Nasrabadi
Lane Department of Computer Science and Electrical Engineering
West Virginia University
{ft0009, hakazemi, ad0046}@mix.wvu.edu, {jeremy.dawson, nasser.nasrabadi}@mail.wvu.edu

Abstract

Objects are usually organized in a hierarchical structure in which each coarse category (e.g., big cat) corresponds to a superclass of several fine categories (e.g., cheetah, leopard). The objects grouped within the same coarse category, but in different fine categories, usually share a set of global features; however, these objects have distinctive local properties that characterize them at a fine level. This paper addresses the challenge of fine image classification in a weakly supervised fashion, whereby a subset of images is tagged by fine labels (i.e., fine images), while the remaining are tagged by coarse labels (i.e., coarse images). We propose a new deep model that leverages coarse images to improve the classification performance of fine images within the coarse category. Our model is an end-to-end framework consisting of a Convolutional Neural Network (CNN) which uses fine and coarse images to tune its parameters. The CNN outputs are then fanned out into two separate branches such that the first branch uses a supervised low rank self-expressive layer to project the CNN outputs to the low rank subspaces to capture the global structures for the coarse classification, while the other branch uses a supervised sparse self-expressive layer to project them to the sparse subspaces to capture the local structures for the fine classification. Our deep model uses coarse images in conjunction with fine images to jointly explore the low rank and sparse subspaces by sharing the network parameters during the training which causes the data obtained by the CNN to be well-projected to both sparse and low rank subspaces for classification.

1. Introduction

Over the past few years, CNNs have provided promising results in object recognition and other visual classification tasks [15, 20, 30]. Along with these developments, image sub-categorization has been used to increase performance of a wide variety of applications in computer vision, such as face recognition [39] and object detection [10]. However, training a CNN requires a vast amount of accurately annotated images [20]. Moreover, providing a sufficient amount of labeled images to train a CNN is labor intensive, time consuming, and usually requires expert knowledge to annotate them accurately, especially where the class of objects is too fine-grained [28, 6].

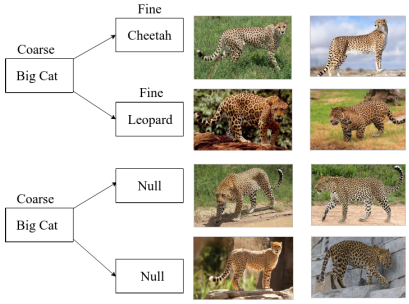


Figure 1. Hierarchical structure for weakly supervised learning.

Objects in the fine classes which are grouped in the same coarse category usually share a set of common visual features. These shared visual properties are typically the global structure underlying the objects which allow them to be categorized at a coarse level. However, these objects have distinctive local properties that are used to characterize them at a fine level. For example, consider the images in Fig. 1. These images are different species of big cats (e.g., cheetah, jaguar, leopard). We can see that all these animals have spots; this is a global feature that is common among them (i.e., commonalities). However, these animals have distinctive feline features (e.g., cheetahs have a "tear line" on their faces that leopards do not) which are specific to each species. These are the local features that are disjoint from the common features among these animals which are used for fine image classification.

The low rank and sparse representation of the high dimensional data is based on the fact that a correlation often exists among the data which belong to the same class such that a low rank subspace captures the global and smooth structures, while a sparse subspace captures the local structures and fine details underlying the data [3, 24, 8, 38].

In this work, we make a structural assumption about all the data points which are extracted from the CNN in our

2380-7504/19/$31.00 ©2019 IEEE
DOI 10.1109/ICCV.2019.00656

framework. This assumption is based on the fact that there is correlation among the data points which belong to the same class. This structural assumption causes all the data extracted from the CNN to exist in a union of low rank, or approximately low rank, subspaces. These subspaces in our model are explored by using a self-expressive property such that each data point can be represented as a linear combination of other samples in the same subspace. In addition to the low rankness of the data (i.e., low rank self-expressive), we also put one sparsity constraint on the number of data which are used to express a given image from a fine category (i.e., sparse self-expressive). This is because the differences between images in a fine category are very small, so we want any given data point from a fine category to be expressed as a linear combination of a very small number of similar data points. We use low rank and sparse subspaces to represent the coarse and fine concepts of the data to address the challenge of fine and coarse image classification. In our model, sparse and low rank subspaces are explored jointly by sharing the network parameters to take advantage of both the coarse and fine images during the training which causes the data extracted from the CNN to be well-projected to sparse and low rank subspaces for classification.

2. Related Work

2.1. Hierarchical Structure for Visual Recognition

The hierarchical structure between objects in most large-scale datasets, such as ImageNet, has been incorporated in deep models to learn each category of images in conjunction with the other categories to improve the overall recognition performance [12, 32, 5, 35, 28, 36]. Most of these methods such as [28], learn the shared and disjoint properties among the objects jointly by focusing on their commonalities and differences in a class hierarchy such that the shared properties discriminate the objects at coarse level of abstraction, while the disjoint properties characterize them at fine level of abstraction. For example, Srivastava *et al*. [32] create a class hierarchy and use a CNN model which transfers the knowledge between the classes to enhance the overall performance by using a small number of training samples.

In other scenarios, Xiao *et al*. [35] introduce a training method that expands a network hierarchically. Due to scalability constraints, the categories in this method are first grouped together based on their similarities, and then self-organized into two groups including coarse and fine levels.

Furthermore, Goo *et al*. [12] propose a method that uses the shared and specialized properties in a semantic hierarchical structure to learn improved discriminative CNN features. This method uses min and difference pooling to implement generalization and specialization layers. In another case, Guo *et al*. [13] introduce an end-to-end framework that integrates a CNN and a Recurrent Neural Network (RNN) for hierarchical categorization. In this method, goal of the CNN is to obtain discriminative features from the input images, while the goal of the RNN is to train the coarse and fine image classification jointly.

Among all of the aforementioned methods which use hierarchical structure to improve fine and coarse classification performance, there are only two methods [28, 13] that consider fine image classification in a weakly supervised fashion. Here, we revisit the hierarchical structure between the fine and coarse categories and propose a new deep model which uses the self-expressiveness property of the data with low rank and sparse representation to tackle the challenge of fine image classification in a weakly supervised fashion.

2.2. Low Rank and Sparse Representation

Low Rank and Sparse Representation (LRR, SR) methods are usually used in an unsupervised manner to capture low dimensional linear subspaces underlying the data [24, 9, 34, 7]. These subspaces usually have a self-expressive property, meaning that a sample taken from a single subspace can be expressed as a linear combination of other samples from the same subspace [7, 27]. Generally, in LRR and SR, an affinity matrix is constructed to measure the pairwise similarities between the data points. The LRR methods usually construct the affinity matrix such that it has the minimum possible rank, while SR methods construct the affinity matrix such that it has the minimum ℓ_1 norm.

There are also supervised versions of LRR [31, 33] and SR [18, 16, 26, 37, 11]. These methods sufficiently exploit the labeled data to learn a discriminative low rank and sparse representation for the data points. These methods incorporate label information as a constraint to guide the learning process for exploring a robust and discriminative subspace projection [23, 22]. In these methods, the data from different classes are well-separated after projection. Like the supervised methods, in this work we incorporate the label information by using a contrastive loss function [14] during the projection of the CNN outputs to the low rank and sparse subspaces to increase the class separability.

3. Preliminaries

3.1. Low Rank and Sparse Self-Expressive

Assume that the data points $\{x_1, x_2, ..., x_n\}$ are clean and sampled from multiple linear subspaces. A subspace is considered self-expressive if each data point from the subspace is expressed by a linear combination of the other data from the same subspace. By stacking all of the data points x_i in a column-wise fashion into a data matrix X, this property can be represented by a linear equation as follows: $X = XC$, where C is an affinity matrix which measures the pairwise similarity between all of the data points. Specifi-

cally, this idea is formulated by an optimization problem as:

$$\min_{C} ||C||_p \quad \text{s.t.} \quad X = XC, \quad \text{diag}(C) = 0, \tag{1}$$

where p is an arbitrary matrix norm. The diagonal constraint on C avoids trivial solutions such as identity matrix.

Various methods have been introduced to define an affinity matrix to explore a set of subspaces. All of these methods aim to find C such that $c_{ij} \neq 0$ if x_i and x_j are in the same subspace, and $c_{ij} = 0$ if they are in different subspaces. In the low rank representation [9, 24, 21], the nuclear norm of C is minimized in (1) instead of minimizing the rank of C, because rank minimization is an NP hard problem and the nuclear norm is the tightest convex relaxation to the rank [4]. Following the model [9] for LRR in general case which data may be contaminated by noise, the model searches for a clean dictionary (e.g., A), and then assumes that the data are obtained by adding noise or error term (e.g., E) to the clean dictionary (i.e., $X = A + E$). In this theoretically sound model, the error term for LRR is relaxed by a Frobenius norm as follows:

$$\min_{C} ||C||_* + \frac{\lambda}{2}||X - XC||_F^2, \tag{2}$$

where, $||.||_*$ is the nuclear norm and $||.||_F$ is the Frobenius norm, and λ is a hyper-parameter that balances the nuclear and the Frobenius norm. By denoting the SVD of X as $U\Sigma V$, the optimal solution of (2) is obtained by using:

$$\hat{C} = V_1(I - \frac{1}{\lambda}\Sigma_1^{-2})V_1^\top, \tag{3}$$

where $U = [U_1, U_2]$, $\Sigma = diag(\Sigma_1, \Sigma_2)$ and $V = [V_1, V_2]$. Matrices are partitioned according to the sets $\mathcal{I}_1 = \{i : \sigma_i > \frac{1}{\sqrt{\lambda}}\}$ and $\mathcal{I}_2 = \{i : \sigma_i \leq \frac{1}{\sqrt{\lambda}}\}$ [9].

In a sparse representation, however, each data point is expressed by the minimum possible number of the other data from the same subspace. Similar to the low rank representation case, we relax the optimization problem by replacing the ℓ_0 with the ℓ_1 norm, because the ℓ_0 norm is a combinatorial norm and ℓ_1 is the closest convex norm to the ℓ_0 norm. Thus, (1) for the sparse subspace, is as follows:

$$\min_{C} \quad ||C||_1 + \frac{\lambda}{2}||X - XC||_F^2, \quad \text{diag}(C) = 0. \tag{4}$$

This problem can be efficiently solved by Alternating Direction Method of Multipliers (ADMM) algorithm [2].

4. Supervised Self-Expressive Layers

Before explaining our entire framework in Section 5, in this section we describe our supervised low rank and sparse self-expressive layers in our framework, as shown in Fig. 2. These layers incorporate labels to explore sparse and low rank subspaces underlying the data. In this framework, the coarse properties of the data are represented by low rank subspaces, while the fine properties of the data are represented by sparse subspaces. Therefore, the supervised self-expressive layers in our framework have two main characteristics: 1) the low rank self-expressive layer learns the global and coarse concepts of the data, while the sparse self-expressive layer learns the local and fine concepts of the data, and 2) these layers incorporate the label information to increase the class separability by leveraging a contrastive loss function during the exploration of the subspaces.

Note that in the self-expressive layers, coarse concepts can be represented by samples from the fine categories. However, fine concepts may not be represented by coarse samples. For example, consider the coarse category of big cats including cheetah, leopard and jaguar subcategories. When we want to represent the "spots" which is a coarse and common concept among all of the samples from the big cats category, we use samples from all of the fine categories and other samples which are only labeled by big cat as all have this property. However, if we want to represent the "tear line on the face", which is a fine concept, we use the samples from the cheetah class, not samples from the leopard or jaguar classes, since the cheetahs have this property while leopards and jaguar do not. For this reason, we separate the CNN outputs into two different branches, where one of them is used for classifying the coarse concepts and the other one is used for classifying the fine concepts.

4.1. Supervised Low Rank Self-Expressive Layer

The main goal of the low rank self-expressive layer in our framework is to explore the subspaces that represent the global and coarse structures of the CNN outputs. Assume that $g(w_s, x_i)$ indicates the feature vector obtained by our CNN for sample x_i (i.e., output of fully connected layer in Fig. 2 for sample x_i), where w_s is parameters of the CNN. Let $G(w_s, X)$ represent the feature matrix constructed by stacking $[g(w_s, x_1), g(w_s, x_2), ..., g(w_s, x_n)]$ column-wise. The cost function $SLR(w_s, C_\mathcal{L}, X)$, used to train the supervised low rank self-expressive layer is as follows:

$$\begin{aligned} &||C_\mathcal{L}||_* + \frac{\lambda}{2}||G(w_s, X) - G(w_s, X)C_\mathcal{L}||_F^2 + \\ &\sum_{i=1}^{n}\sum_{j=1}^{n}(1 - y_{ij})||g(w_s, x_i) - g(w_s, x_j)||^2 + \\ &y_{ij}\ max\{0, (m - ||g(w_s, x_i) - g(w_s, x_j)||)\}^2, \end{aligned} \tag{5}$$

where, $C_\mathcal{L}$ is a low rank affinity matrix. The first two terms in (5) are the loss terms which are used to explore the low rank subspaces underlying the data matrix $G(w_s, X)$. $C_\mathcal{L}$ in (5) forces $G(w_s, X)$ to be projected into the low rank subspaces. In other words, $C_\mathcal{L}$ regularizes w_s to learn the low rank structures of features. This is

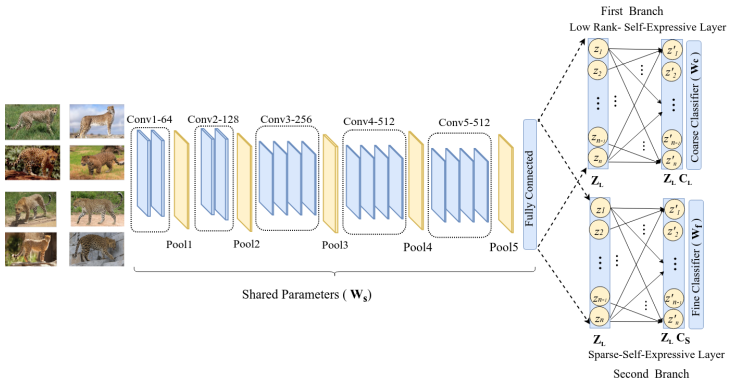


Figure 2. indicates our deep framework, the CNN output is fanned out into two separate branches for fine and coarse image classification.

because, based on the rank inequality of multiplying two matrices (i.e., $rank(AB) < min\{rank(A), rank(B)\}$), we can conclude that, by minimizing (5) with respect to w_s, $G(w_s, X)$ is projected into subspaces such that $rank(G(w_s, X)) < rank(C_{\mathcal{L}})$. Note that we simultaneously minimize $rank(C_{\mathcal{L}})$ by using the term $||C_{\mathcal{L}}||_*$ in (5) which causes to be reduced the rank of $G(w_s, X)$.

The third term in (5) is a contrastive loss, the goal of which is to bring $g(w_s, x_i)$ and $g(w_s, x_j)$ close to each other if x_i and x_j samples belong to the same coarse category, while pushing them away from each other if they belong to the different coarse categories. $y_{ij} = 0$ if x_i and x_j have the same coarse label, otherwise, $y_{ij} = 1$. Here, m is the margin used in the contrastive loss.

4.2. Supervised Sparse Self-Expressive Layer

The goal of the sparse self-expressive layer in our framework is to explore the subpaces which represent the local and fine structures of the CNN outputs. In the sparse self-expressive layer, we put a sparsity constraint on the number of similar data points which are used to express a given sample (x) from a fine category. The sparsity constraint prevents the samples (Z) within the same coarse category, but belonging to the different fine categories, from contributing to the expression of sample x. This is because we want to express the fine concept of x, and Z may not have the same fine concept. Indeed, we choose the sparse subspace as a solution to express the fine concepts because the differences between images in a fine category are very small, so we want any given data point from a fine category to be expressed as a linear combination of a very small number of similar data points. Eq. (6) shows the loss used to learn the sparse self-expressive layer. The first and second terms in (6) are the loss terms used to explore sparse subspaces, while the third term is the supervised loss used to increase the separability of the fine classes. Minimizing (6) with respect to w_s forces the feature matrix $G(w_s, X)$ to be projected in subspaces such that each sample $g(w_s, x_i)$ can be expressed by the minimum possible number of other samples from the same subspace. This is obtained by putting a constraint on the affinity matrix C_S so that it has a small ℓ_1 norm. $y'_{ij} = 0$ if x_i and x_j belong to the same fine category, otherwise $y'_{ij} = 1$. Our supervised sparse self-expressive layer cost function, $SS(w_s, C_S, X)$, is defined as follows:

$$
\begin{aligned}
&||C_{\mathcal{S}}||_1 + \frac{\lambda}{2}||G(w_s, X) - G(w_s, X)C_{\mathcal{S}}||_F^2 + \\
&\sum_{i=1}^{n}\sum_{j=1}^{n}(1 - y'_{ij})||g(w_s, x_i) - g(w_s, x_j)||^2 + \\
&y'_{ij}\ max\{0, (m - ||g(w_s, x_i) - g(w_s, x_j)||)\}^2,
\end{aligned}
\tag{6}
$$

where, $C_{\mathcal{S}}$ is a sparse affinity matrix. Note that in the sparse subspaces of our model, the structural assumption (i.e., low rankness) among the data points is preserved as well. This is because the two branches of the framework are trained jointly by sharing the parameters (w_s) during the training such that the rank of $G(w_s, X)$ is attempted to be minimized in the low rank self-expressive layer.

5. Fine-Coarse Label Classifier Framework

Fig. 2 illustrates our complete architecture. This architecture is an end-to-end framework consisting of a CNN whose output is fanned out into two separate branches. The first branch projects the CNN outputs into the low rank subspaces by using a supervised low rank self-expressive layer. The data points projected into the low rank subspaces are then classified into the coarse categories by using a softmax layer (w_c in Fig. 2). The second branch, however, projects the CNN outputs into the sparse subspaces by using a supervised sparse self-expressive layer. The data points projected into the sparse subspaces are then classified into the fine classes by using a softmax layer (w_f in Fig. 2).

Eq. (7) indicates the total loss function that our deep model uses for joint fine and coarse image classification. In this loss function, both the fine and coarse classification tasks share the parameters (w_s) to jointly explore the sparse and low rank subspaces by optimizing the Eq. (5) and Eq. (6) simultaneously during the training phase. In this frame-

work, coarse images contribute to the fine classification because images with the same coarse label, but belonging to different fine categories, still have a common global structure that can be used in conjunction with fine images to better tune the parameters of our model (w_s) during the training phase. In such a case, the coarse images affect the parameters such that the CNN outputs are well-projected into both low rank and sparse subspaces which contributes to the fine classification. The parameters which are optimized in our architecture are $\{w_c, w_f, w_s, C_{\mathcal{L}}, C_{\mathcal{S}}\}$. The total loss of the framework, $\mathcal{L}(.)$, is formulated as follows:

$$\begin{aligned} &\mathcal{L}(w_s, w_c, w_f, C_{\mathcal{L}}, C_{\mathcal{S}}, X) = \\ &\phi(SS(w_s, C_{\mathcal{S}}, X) + \gamma(SLR(w_s, C_{\mathcal{L}}, X)) + \\ &\mathcal{L}_c(w_c G(w_s, X), L_c) + \mathcal{L}_c(w_f G(w_s, X), L_f), \end{aligned} \tag{7}$$

where, $\mathcal{L}_c(.)$ is the softmax cross entropy loss function used for classification. The SLR and SS are the supervised sparse and low rank self-expressive layer loss functions defined in (5) and (6), respectively. L_c and L_f are the coarse and fine ground truth labels, respectively. $w_c G(w_s, X)$ and $w_f G(w_s, X)$ are the predicted coarse and fine labels by our model, respectively. ϕ and γ are the hyper-parameters that balance different loss terms during the training.

6. Experiments and Implementation Details

6.1. CNN Architecture

We use a VGG [30] architecture as shown in Fig. 2. We apply batch normalization [17] after each convolutional layer, and before performing the Rectified Linear Units (ReLU) activation function [20]. We use an Adam optimizer [19] with the default hyper-parameters values ($\epsilon = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) to train the parameters of the CNN. The batch size in all experiments is fixed to 128 and the framework is implemented in TensorFlow.

6.2. Training the Framework

We arrange a strategy which has two steps to train the parameters of the entire network: pre-training and fine-tuning. This strategy also prevents the trivial all-zero solution while minimizing the losses defined in (5) and (6). In the pre-training step, each component of the framework (i.e., CNN network (w_s), first branch (w_c, $C_{\mathcal{L}}$) and second branch (w_f, $C_{\mathcal{S}}$) is trained separately, while in the fine-tuning step, all of the components of the framework are trained jointly.

6.2.1 Pre-training Step

In the pre-training step, we first initialize the CNN parameters (w_s) by a VGG-Net pre-trained on a subset of the ImageNet 2010 [28]. In the next step of the pre-training step, we train the parameters of each branch separately. In each branch, there are two tasks that are learned successively: 1) projecting the CNN outputs (i.e., $G(w_s, X)$) into the low rank and sparse subspaces, and 2) classifying the projected data points into the coarse and fine categories. Therefore, in the first branch, the first task updates $C_{\mathcal{L}}$, and the second task updates the coarse classifier parameters (w_c). In the second branch, however, the first task updates $C_{\mathcal{S}}$, and the second task updates the fine classifier parameters (w_f).

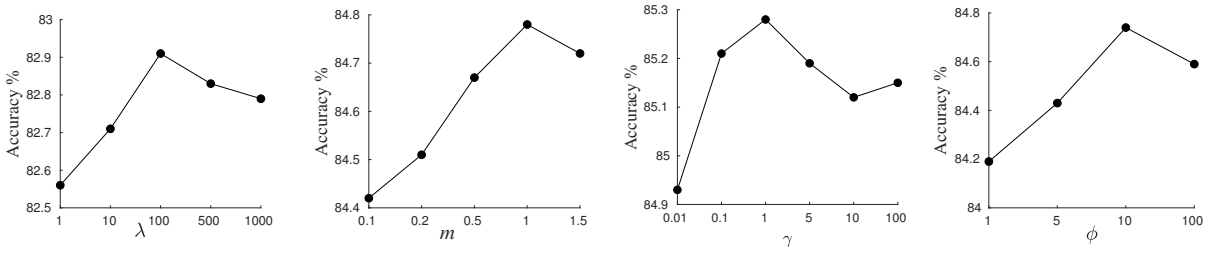
Note that $C_{\mathcal{L}}$ in the first branch and C_S in the second branch can be thought of as the parameters of an additional network layer (i.e., self-expressive layer as it is shown in Fig. 2), which allows us to find a solution for $C_{\mathcal{L}}$ and $C_{\mathcal{S}}$ in (5) and (6) by using back-propagation in the first and second branch, respectively. We also note that the self-expressive layers in our framework are not a fully connected layer. In the low rank-self-expressive layer, each data point ($z_i = g(w_s, x_i)$ in Fig. 2, where n is the number of training data) is only connected to all the data which belong to the same coarse category. Thus, data within the same coarse category construct a complete bipartite graph in the low rank-self-expressive layer. In the sparse-self-expressive layer, however, each data point attempts to be connected to only the minimum possible number of data points which belong to the same fine category. Thus, in this layer, data points within the same fine category construct a bipartite graph which is very sparse.

6.2.2 Fine-tuning Step

In the fine-tuning step, all parameters of the framework are updated jointly. The parameters of the model in this step have been initialized with a proper starting point obtained from the pre-training step. In this step, we aim to minimize the total loss function in (7), all at once. Since the framework is trained in batch mode through several epochs, we optimize (7) with respect to $C_{\mathcal{L}}$ and $C_{\mathcal{S}}$ parameters only after each epoch is finished, where we have observed all the training data, because $C_{\mathcal{L}}$ and $C_{\mathcal{S}}$ need to access all the data points when they project the data points into the subspaces with the self-expressive property. Therefore, we train the framework in batch mode and optimize (7) with respect to all the parameters excluding $C_{\mathcal{L}}$ and $C_{\mathcal{S}}$. Then, at the end of each epoch, we minimize the total loss with respect to $C_{\mathcal{L}}$ and $C_{\mathcal{S}}$ (i.e., this step is similar to the alternative minimization algorithm, where we optimize one parameter alternatively by fixing all other parameters). Note that during the training, we store the CNN features of the images. This process avoids the need to re-calculate those features (i.e., $G(w_s, X)$), when we update $C_{\mathcal{L}}$ and $C_{\mathcal{S}}$ parameters.

6.3. Dataset and Experimental Setup

We arrange our experimental setup as described in [28] and [13]. We perform our experiments on a subset of the ImageNet 2010. This subset consists of the classes from the ImageNet 2010 which have only one parent class. Among



(a) (b) (c) (d)

Figure 3. Hyper-parameters tuning: accuracy of the model on the validation set, where $|S_{Coarse}| = |S_{fine}| = 0.5|S|$

these classes, the parent classes form the coarse categories, S_{coarse}, while their corresponding children contain the fine categories, S_{fine}. Using this setup, we have $|S_{coarse}| = 143$ coarse classes, and $|S_{fine}| = 387$ fine classes. Since in this work, we consider fine image classification in a weakly supervised fashion, the original training, validation, and test sets of the ImageNet 2010 dataset are truncated to S_{fine}. The truncated training set contains 487K images in which there are between 1.4K and 9.8K images for each coarse category, and between 668 and 2.4K images for each fine category. For the validation and the test sets, there are 50 and 150 images per fine category, respectively. In all of our experiments, we report the performance of the model by using the top-one average accuracy, as reported in [28] and [13]. We randomly divide the truncated training set, S, into two disjoint subsets for each fine category. In this case, the first subset of images, S_{coarse}, have only the coarse labels, while the second subset of images, S_{fine}, have fine labels as well. More details about the classes, training, validation and test sets are found here [1].

6.4. Hyper-parameters Tuning

We adjust the hyper-parameters of the model based on a range of the values that provide the best accuracy on the validation set. We set $|S_{coarse}| = |S_{fine}| = 0.5|S|$ in the training set during the tuning of the hyper-parameters. The hyper-parameters of our model are $\{\gamma, \phi, \lambda, m\}$ defined in (7) in which m and λ are used in the $SS(.)$ and $SLR(.)$ functions. Fig. 3 indicates the accuracy of the model on the validation set in tuning the hyper-parameters. Fig. 3(a) indicates the performance of the model by setting $\lambda = \{1, 10, 100, 500, 1000\}$, the results indicate that $\lambda = 100$ provides the best accuracy. When choosing m to be $\{0.2, 0.5, 1, 1.5\}$, as shown in Fig. 3(b), the best value for the margin, m, is 1. When setting γ to be $\{0.01, 0.1, 1, 5, 10, 100\}$ and ϕ to be $\{1, 5, 10, 100\}$, as shown in Fig. 3(c) and Fig. 3(d), values of $\gamma = 1$ and $\phi = 10$ provide the best accuracy.

[1] http://www.vision.ee.ethz.ch/datasets_extra/mristin/ristin_et_al_cvpr15_data.zip

6.5. Evaluating the Model in Different Cases

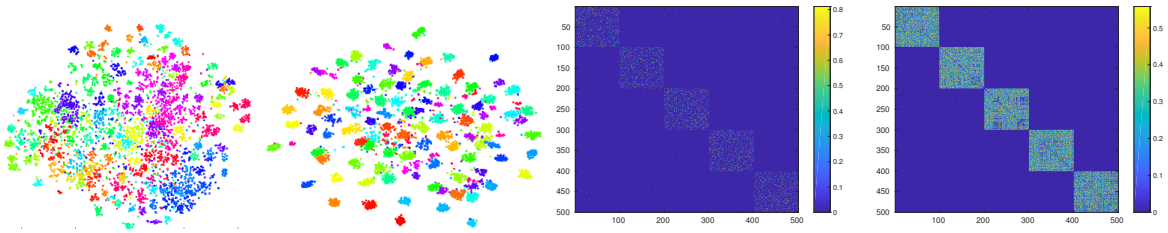
In this section, we evaluate performance of our model-Deep Fine Classifier (DFC), in five different cases. DFC-S is the case where we train each component of the framework 'separately' (pre-training step defined in the Section 6.2.1). However, in this case we ignore the contrastive loss in (5) and (6) during exploring the subspaces. DFC-S-CL is the case where we add contrastive loss to the DFC-S case.

DFC- J is the case where we train the entire framework 'jointly' by using the loss (7), but ignore the contrastive loss in (5) and (6). DFC-J-CL is the case where we add contrastive loss to the DFC- J case, in which we train the entire framework 'jointly' by considering the contrastive loss during the exploration of subspaces in (5) and (6). Finally, Base-J-CL is the case where we remove the sparse and low rank constraints (i.e., we remove the first two terms from (5) and (6)) and then train the network using the loss (7).

6.5.1 Supervised Sparse and Low Rank Layers

In a supervised version of exploring low rank and sparse subspaces, we incorporated the label by using a contrastive loss function. To see the effectiveness of this process in our model, we compare DFC on the testing set for two pairs of different cases (i.e., {DFC-S, DFC-S-CL}, and {DFC-J, DFC-J-CL}). Table. 1 indicates the performance of the model in different cases on the testing set where we fix $|S_{coarse}| = 0.5|S|$ and set $|S_{fine}|$ to be $0.1|S|$, $0.2|S|$ and $0.5|S|$, respectively. Table. 2 shows the accuracy of the model on the testing set where we fix $|S_{fine}| = 0.5|S|$ and set $|S_{coarse}|$ to be $0.1|S|$, $0.2|S|$ and $0.5|S|$, respectively.

By comparing DFC-J with DFC-J-CL in Table. 1 and Table. 2, we observe that including contrastive loss during the exploration of the subspaces, in either configuration of splitting training data to the coarse and fine labels, increases our model performance. We can see that fine label classifier performance in our model is improved by 1.05% on average (Table. 1) when we fix $|S_{coarse}| = 0.5|S|$ and vary the size of $|S_{fine}|$, while it is improved by 0.83% on average (Table. 2) when we fix $|S_{fine}| = 0.5|S|$ and vary the size



(a) Separate-training (b) Join-training (c) Sparse (d) Low rank

Figure 4. Feature visualization (a) and (b); visualization of affinity matrices for coarse categories (c) and (d).

Methods	$0.1\|S\|$	$0.2\|S\|$	$0.5\|S\|$
Base-J-CL	69.19	70.51	72.64
DFC-S	72.84	74.46	77.58
DFC-S-CL	74.32	76.16	79.41
DFC-J	76.23	77.39	80.12
DFC-J-CL	76.89	78.83	81.17

Table 1. Comparing performance of our model in different scenarios, we fix $|S_{coarse}| = 0.5|S|$, and we set $|S_{fine}|$ to be $0.1|S|$, $0.2|S|$ and $0.5|S|$.

Methods	$0.1\|S\|$	$0.2\|S\|$	$0.5\|S\|$
Base-J-CL	71.12	71.49	72.64
DFC-S	76.52	76.98	77.58
DFC-S-CL	77.48	77.81	79.41
DFC-J	78.63	78.91	80.12
DFC-J-CL	79.15	79.84	81.17

Table 2. Comparing performance of our model in different scenarios, we fix $|S_{fine}| = 0.5|S|$, and we set $|S_{coarse}|$ to be $0.1|S|$, $0.2|S|$ and $0.5|S|$.

of $|S_{coarse}|$. Moreover, by comparing DFC-S with DFC-S-CL, the average improvement of the model in Table. 1 and Table. 2 are 1.67% and 1.20%, respectively. Finally, the comparison of results between Base-J-CL and DFC-J-CL in Table. 1 and Table. 2 shows that the LR and sparse constraints have a significant impact on our model.

6.5.2 Effectiveness of Joint Training

In this section, we show the effectiveness of joint-training for our framework. The results reported in Table. 1 and Table. 2 show that training the model jointly (i.e., DFC-J-CL case) significantly improves the overall performance of the framework in comparison to the case where we train our model components separately (i.e., DFC-S-CL case). These improvements, on average, are 2.33% and 1.82% in Table. 1 and Table. 2, respectively.

6.6. Impact of Fine and Coarse Samples in Training

Here we investigate the effect of fine images during the training. Moreover, we demonstrate how coarse images improve the model performance by fixing the fine image set size during the training. Table. 1 shows the case in which we fix the coarse image set size (i.e.,$|S_{coarse}| = 0.5|S|$) while we increase the fine image set size ($|S_{fine}|$) during the training. As shown in Table. 1, the overall performance of our entire framework (i.e., DFC-J-CL) increases by 1.94% and 2.34 % as we increase the size of $|S_{fine}|$ from $0.1|S| \rightarrow 0.2|S|$ and $0.2|S| \rightarrow 0.5|S|$, respectively. Table. 2 shows the effectiveness of the incorporation of coarse images into our model. Table. 2 shows the case where we fix the fine images set size (i.e.,$|S_{fine}| = 0.5|S|$) while we increase the coarse image set size ($|S_{coarse}|$) during the training. As shown in Table. 1, the overall performance of our model (i.e., DFC-J-CL) increases by 0.69% and 1.33 % as we increase the size of $|S_{coarse}|$ from $0.1|S| \rightarrow 0.2|S|$ and $0.2|S| \rightarrow 0.5|S|$, respectively.

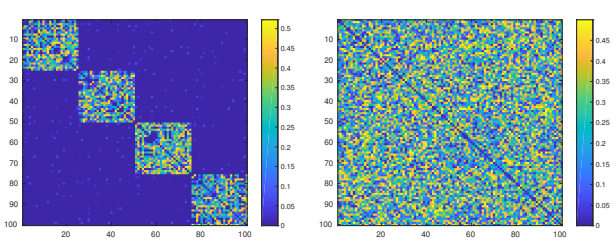
Size	$0.1\|S\|$	$0.2\|S\|$	$0.5\|S\|$
DFC-J-CL-Base	73.92 ± 0.16	76.31 ± 0.21	79.06 ± 0.14
DFC-J-CL	76.89 ± 0.24	78.83 ± 0.18	81.17 ± 0.11

Table 3. The model accuracy by using and not using coarse data.

We also evaluated our entire model (i.e., DFC-J-CL) for a naive baseline called (DFC-J-CL-Base) where we use two branches, one for fine and the other one for coarse classification. However, in this case during the training, we ignore coarse images which are not tagged by fine labels and we only use images which have both the fine and coarse labels. We set $|S_{fine}|$ to be $0.1|S|$, $0.2|S|$ and $0.5|S|$, and ran this baseline. By comparing this baseline with DFC-J-CL in Table. 3, we observe that our model improves fine classification on average by 2.97 %, 2.52 %, and 2.11 % when we use coarse labeled images with missing fine labels. Note that the DFC-J-CL-Base and DFC-J-CL will be the same case if we use 100% of images with fine and coarse labels and the accuracy of our model in this case is $85.67 \pm 0.26\%$.

6.7. Comparison

We compare our deep fine label classifier with NN-H-RNCMF [28] and its baseline, RNCMF, and RNN [13], which also attempt to enhance fine classification by leveraging the coarse images in hierarchical structure. Moreover, we compare our model with DMSC [1] as a baseline, where a similar sparse and low-rank approach was employed for a clustering task. For this method, we modified it to our task. Specifically, the encoder in this model takes unimodal data as the input data in our case are RGB images. All of the hyper-parameters in these methods are chosen based on the



(a) Sparse (b) Low rank

Figure 5. Affinity matrices for fine categories.

Methods	$0.1\|S\|$	$0.2\|S\|$	$0.5\|S\|$
RNCMF [28]	68.49	70.49	73.07
NN-H RNCMF [28]	69.95	71.41	73.43
DMSC [1]	72.97	74.49	76.72
RNN [13]	74.26	75.64	77.12
DFC-J-CL	76.89	78.83	81.17

Table 4. Comparing our method with others by fixing $|S_{coarse}| = 0.5|S|$, and setting $|S_{fine}|$ to be $0.1|S|$, $0.2|S|$ and $0.5|S|$.

authors suggestion. All of these methods use the features extracted from the VGGNet pre-trained on the subset of ImageNet dataset [28]. We are consistent with these methods regarding the amount of coarsely and finely labeled images during the training. We set $|S_{coarse}| = 0.5|S|$, and change $|S_{fine}|$ to $\{0.1|S|, 0.2|S|, 0.5|S|\}$. The results in Table. 4 show that DFC-J-CL outperforms other methods in all configurations, indicating its great potential to leverage coarse images for improving fine image classification. In further study, we compared our method in a case where all the fine images are available during the training. In this configuration, our model accuracy is **85.67%**, while the accuracy of the methods including the baseline VGG fine-tuned on the ImageNet2010, NN-H-RNCMF, and RNN are 76.01%, 74.18%, and 82%, respectively as reported in [28, 13].

6.8. Further Analysis

Inspired by [29], we use a Grad-CAM to expose the implicit attention of our model on the images during the classification. We observe that our model is triggered by semantic regions of the images for fine image classification. By using the Grad-CAM, we can see that our fine label classifier provides a "visual explanation" for the decision that it makes during the classification. Fig. 6 indicates the Grad-CAM in our model for 'bear', 'fox', 'wolf' and 'spider' species. It shows that our model makes decisions by using the appropriate regions of the images to classify these species.

Furthermore, we used T-SNE [25] to visualize the CNN features for training in two cases where we train the model separately and jointly. Fig. 4(a) illustrates the case where we train our model separately, while Fig. 4(b) shows the case where we train the model jointly. Fig. 4(a) and Fig. 4(b) shows that the training data in joint-training are better separated than separate-training case.

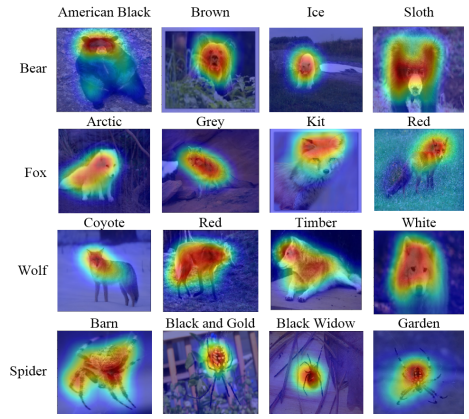


Figure 6. Example of Grad-CAM for classifying different species.

Moreover, we visualized the affinity matrices of sparse and low rank subspaces for coarse categories. We selected 100 samples of five classes including 'bear', 'fox', 'wolf' , 'spider' and 'grouse' with the same coarse labels. Fig. 4(c) shows the sparse affinity matrix, $C_{\mathcal{S}}$, and Fig. 4(d) indicates the low rank affinity matrix, $C_{\mathcal{L}}$, for the coarse categories. Fig. 4(c) shows that affinity matrix, $C_{\mathcal{S}}$, is much sparser than affinity matrix, $C_{\mathcal{L}}$. This means that the fine data points are expressed by few number of samples, while the coarse data are expressed by more samples in the same subspace.

In further study, we visualized the affinity matrices of sparse and low rank subspaces for fine categories. We picked 100 samples from the 'fox' category by choosing 25 samples per each fine category including 'artic', 'grey', 'kit' , and 'red'. Fig. 5(a) shows the sparse affinity matrix, $C_{\mathcal{S}}$, and Fig. 5(b) indicates the low rank affinity matrix, $C_{\mathcal{L}}$, for the fine categories. Fig. 5(b) shows that almost all samples contribute to express a given sample in the low rank subspace, while Fig. 5(a) shows that these samples are grouped to four categories such that each sample can be expressed by the other samples from the same group.

7. Conclusion

We proposed a novel CNN model that uses coarse images to improve weakly supervised fine image classification performance. Our model represents coarse and fine concepts of the images in low rank and sparse self-expressive subspaces such that the sparse and low subspaces are used to classify images at fine and coarse levels of abstraction, respectively. In our model, the sparse and low rank subspaces are explored jointly by sharing the parameters to use coarse images in conjunction with fine images during the training, which causes the data obtained by the CNN to be well-projected into sparse and low rank subspaces for classification. The experimental results show the great potential of our model for using coarse images to improve weakly supervised fine classification. Moreover, the results indicate the superiority of our model in comparison to the other methods which also use coarse images for enhancing weakly supervised fine classification.

References

[1] Mahdi Abavisani and Vishal M Patel. Deep multimodal subspace clustering networks. *IEEE Journal of Selected Topics in Signal Processing*, 12(6):1601–1614, 2018. 7, 8

[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011. 3

[3] Maria Brbić and Ivica Kopriva. Multi-view low-rank sparse subspace clustering. *Pattern Recognition*, 73:247–258, 2018. 1

[4] Emmanuel J Candes. The restricted isometry property and its implications for compressed sensing. *Comptes rendus mathematique*, 346(9-10):589–592, 2008. 3

[5] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In *European conference on computer vision*, pages 48–64. Springer, 2014. 2

[6] Jia Deng, Jonathan Krause, and Li Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2013. 1

[7] Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2790–2797. IEEE, 2009. 2

[8] Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013. 1

[9] Paolo Favaro, René Vidal, and Avinash Ravichandran. A closed form solution to robust subspace estimation and clustering. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1801–1807. IEEE, 2011. 2, 3

[10] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010. 1

[11] Yuan Gao, Jiayi Ma, and Alan L Yuille. Semi-supervised sparse representation based classification for face recognition with insufficient labeled samples. *IEEE Transactions on Image Processing*, 26(5):2545–2560, 2017. 2

[12] Wonjoon Goo, Juyong Kim, Gunhee Kim, and Sung Ju Hwang. Taxonomy-regularized semantic deep convolutional neural networks. In *European Conference on Computer Vision*, pages 86–101. Springer, 2016. 2

[13] Yanming Guo, Yu Liu, Erwin M Bakker, Yuanhao Guo, and Michael S Lew. Cnn-rnn: a large-scale hierarchical image classification framework. *Multimedia Tools and Applications*, pages 1–21, 2018. 2, 5, 6, 7, 8

[14] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[16] Jin Huang, Feiping Nie, Heng Huang, and Chris HQ Ding. Supervised and projected sparse coding for image classification. In *AAAI*, 2013. 2

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 5

[18] Xudong Jiang and Jian Lai. Sparse and dense hybrid representation via dictionary decomposition for face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (1):1–1, 2015. 2

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. 1, 5

[21] Chun-Guang Li and Rene Vidal. Structured sparse subspace clustering: A unified optimization framework. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 277–286, 2015. 3

[22] Ping Li, Jun Yu, Meng Wang, Luming Zhang, Deng Cai, and Xuelong Li. Constrained low-rank learning using least squares-based regularization. *IEEE transactions on cybernetics*, 2016. 2

[23] Sheng Li and Yun Fu. Robust subspace discovery through supervised low-rank constraints. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 163–171. SIAM, 2014. 2

[24] Guangcan Liu, Zhouchen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 663–670, 2010. 1, 2, 3

[25] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 8

[26] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R Bach. Supervised dictionary learning. In *Advances in neural information processing systems*, pages 1033–1040, 2009. 2

[27] Shankar R Rao, Roberto Tron, René Vidal, and Yi Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. 2008. 2

[28] Marko Ristin, Juergen Gall, Matthieu Guillaumin, and Luc Van Gool. From categories to subcategories: large-scale image classification with partial class label refinement. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 231–239, 2015. 1, 2, 5, 6, 7, 8

[29] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017. 8

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 5

[31] Farzad Siyahjani, Ranya Almohsen, Sinan Sabri, and Gianfranco Doretto. A supervised low-rank method for learning invariant subspaces. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4220–4228, 2015. 2

[32] Nitish Srivastava and Ruslan R Salakhutdinov. Discriminative transfer learning with tree-based priors. In *Advances in Neural Information Processing Systems*, pages 2094–2102, 2013. 2

[33] JianWen Tao, Dawei Song, Shiting Wen, and Wenjun Hu. Robust multi-source adaptation visual classification using supervised low-rank representation. *Pattern Recognition*, 61:47–65, 2017. 2

[34] René Vidal and Paolo Favaro. Low rank subspace clustering (lrsc). *Pattern Recognition Letters*, 43:47–61, 2014. 2

[35] Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, Yuxin Peng, and Zheng Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 177–186. ACM, 2014. 2

[36] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 2740–2748, 2015. 2

[37] Jianchao Yang, Jiangping Wang, and Thomas Huang. Learning the sparse representation for classification. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1–6. IEEE, 2011. 2

[38] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7370–7379, 2017. 1

[39] Manli Zhu and Aleix M Martinez. Subclass discriminant analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1274–1286, 2006. 1