Collaboratively Weighting Deep and Classic Representation via l_2 Regularization for Image Classification

Shaoning Zeng[a,1,2], Bob Zhang[b,1], Yanghao Zhang[c,3] and Jianping Gou[d,4]

[1] Department of Computer and Information Science, University of Macau,
[2] School of Information Science and Technology, Huizhou University,
[3] Electronics and Computer Science, University of Southampton,
[4] School of Computer Science and Communication Engineering, Jiangsu University,

Abstract

Deep convolutional neural networks provide a powerful feature learning capability for image classification. The deep image features can be utilized to deal with many image understanding tasks like image classification and object recognition. However, the robustness obtained in one dataset can be hardly reproduced in the other domain, which leads to inefficient models far from state-of-the-art. We propose a deep collaborative weight-based classification (DeepCWC) method to resolve this problem, by providing a novel option to fully take advantage of deep features in classic machine learning. It firstly performs the l_2-norm based collaborative representation on the original images, as well as the deep features extracted by deep CNN models. Then, two distance vectors, obtained based on the pair of linear representations, are fused together via a novel collaborative weight. This collaborative weight enables deep and classic representations to weigh each other. We observed the complementarity between two representations in a series of experiments on 10 facial and object datasets. The proposed DeepCWC produces very promising classification results, and outperforms many other benchmark methods, especially the ones claimed for Fashion-MNIST. The code is going to be published in our public repository[1].

1 Introduction

Machine learning methods have been applied to deal with various multi-media and computer vision tasks. Traditionally, linear models such as sparse representation (SR) [30] and collaborative representation (CR) [44] have drawn much attention and gained promising results in image classification. Lately, nonlinear deep learning [12] models, e.g., ResNet [8] and VGG [24], have produced state-of-the-art results in many image-based tasks, including face recognition, object detection, video tracking, etc. Linear sparse models can be utilized to improve deep neural networks [34]. On the other hand, more and more conventional methods took deep features as input to gain more promising classification results [2, 42, 43]. However, recent studies showed that deep features from neural networks are usually designed for SVM-like classifiers [13]. Using deep features as sole input in non-SVM classical models could be dubious. For this problem, we believe that the technique of linearly representing images can be applied to enhance nonlinear deep models.

Deep learning shows a very strong capacity to learn discriminative image features. CNN features off-the-shelf [23] were demonstrated to be powerful for recognition tasks. Learning deep features can help to obtain state-of-the-art results for different tasks like face recognition [29], scene recognition [46], person re-identification [32] and general image classification [22, 26]. The good news is that many conventional machine learning methods can also learn credible features from images. In recent years, Sparse Representation (SR) [30] via l_1 regularization has shown huge potential in feature extraction and image classification. On the other hand, l_2 regularization-based Collaborative Representation (CR) [44] can also build a similarly robust linear model. The l_2 regularization inside CR helps to create an equally discriminative but faster sparse representation [37]. According to our observation, sparseness plays an important role in both linear and nonlinear models. It is likely for these two paradigms, deep and classic representations, to generate a new representation learning model when collaborating with each other.

In this paper, we propose a Collaborative Weight-based Classification method that brings deep and classic non-deep representation together, to implement a more promising im-

[1] https://github.com/zengsn/research

age classification. We name it DeepCWC for short. The contribution of this work includes: 1) proposing a new classifier to integrate features from linear and nonlinear models, 2) giving an analysis on how black-box deep features work in a sparse classification model, 3) conducting image classification experiments on different CNN models and convolutional layers inside them, to demonstrate the performance of DeepCWC in a consistent and comprehensive way. The proposed method produces promising results on face and object recognition. In particular, it ranks first in recognition (97.66%) on the Fashion-MNIST dataset.

2 Related Work

The root inspiration comes from the popular deep residual network (ResNet) [8]. ResNet keeps an identity map learned from the last layer, and applies it to next layer of learning. Then, it constructs a new building block $y = \mathscr{F}(x, \{W_i\}) + x$, as shown in Fig. 1(a). This explicitly allows these layers to fit a residual mapping, so as to make it easier for the residual to be zero (sparse) than to fit an identity mapping by a stack of nonlinear layers. In this way, the discrimination learned in the previous layers will be propagated layer-by-layer. Also, there would be a linear transformation between some layers, if two connected blocks have a different dimension, as shown in Fig. 1(b). Denote the linear projection as W_s, where the building blocks become $y = \mathscr{F}(x, \{W_i\}) + W_s x$.

ResNet attracts great attention and progresses observably [9], while our main focus is the way how it utilizes the prior information. It is possible to include sparse learning, as prior information, in deep neural networks as well. For example, grouping multiple sparse regularizations for simultaneously optimizing deep neural networks [21]. Wen et al. proposed to learn a structured sparsity in deep neural networks to regularize the inside structures (i.e., filters, channels, filter shapes, and layer depth) [28]. Afterwards, a fixed linear sparse filter can be cascaded with a thresholding nonlinearity to maximize sparsity in deep neural networks [34]. It becomes an emerging trend to utilize linear sparse models to collaborate with nonlinear neural networks.

As shown in Fig. 1(c), our idea has a similar structure following the building block of ResNet. The key is fusing the identity map learned from l_2-norm collaborative representation [44] to the result after deep residual learning. To simplify the implementation structure, the linear model is not injected into the building block of the neural network. Instead, it performs on the classifier. There are several reasons for this structure. Firstly, it helps to avoid overheads in the training process of the neural networks. Furthermore, it creates a more general structure that can be easily extended to other types of neural networks, which are not limited to ResNet. For example, we also implement this in Inception [25] and VGG [24], which will be demonstrated in Sec. 4. The idea behind pairing nonlinear deep learning with an additional linear representation is to make the network more capable in different classification tasks.

However, the usage of the prior learned information is different in our implementation. ResNet adds up the learned x in model training, while the proposed DeepCWC will introduce a collaborative weight in classification, which is obtained by an element-wise multiplication instead of addition. The next section explains the detailed implementation.

3 Deep collaborative weight-based classification

The key idea in Deep Collaborative Weight-based Classification is straightforward: using the model of Collaborative Representation (CR) [44] to learn a classifier from the original images and deep features in pairs. CR is based on l_2 normalization and emphasizes the collaboration among all samples in the representation. However, more and more evidence points to the fact that the collaboration requires help from the sparseness in the representation to maintain a high level of performance [1]. We believe that using deep features is one of the possible solutions.

3.1 Pair of residual learning

In CR based classification (CRC), the role of collaboration among classes is stressed, rather than sparsity in the representation, when representing a test sample. Let A denote the training samples selected from all C classes, while y is the test sample. Both A and y will be normalized to have l_2-norm. The representation of y by A can be denoted as an approximate linear problem $y \approx A\alpha$, where $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_C]$ is the representation coefficient to be solved.

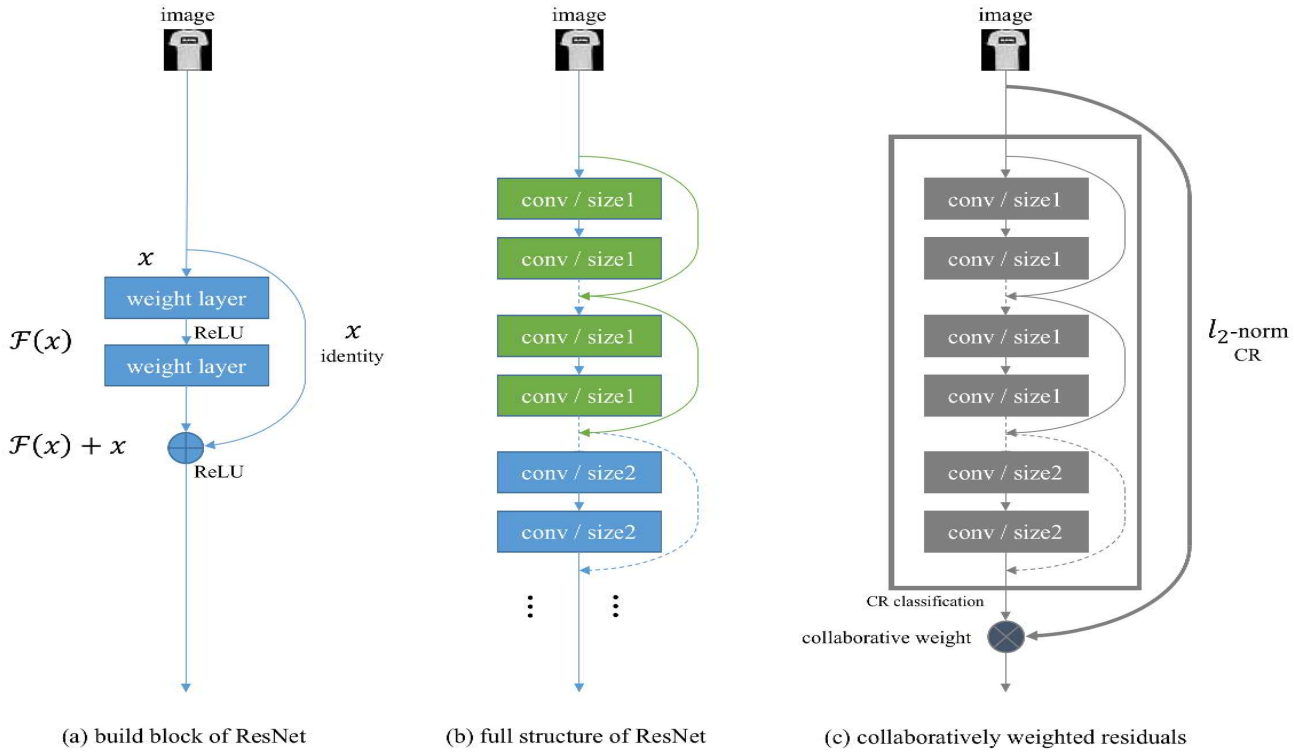
First of all, a regularized least square method [43] is used to solve the problem and perform the collaborative representation of the original image sample as follows

$$(\hat{\alpha}) = \arg\min_{\alpha}\{\left\|y - A \cdot \alpha\right\|_2^2 + \lambda \left\|\alpha\right\|_2^2\}, \qquad (1)$$

where λ is the regularization parameter, which introduces a certain number of "sparsity" to the solution. The solution of this linear problem by using regularized least square can be derived as

$$\hat{\alpha} = (A^T A + \lambda \cdot I)^{-1} A^T y. \qquad (2)$$

Let $P = (A^T A + \lambda \cdot I)^{-1} A^T$, such that P is a projection matrix that can be pre-solved and independent of the test sample y. The projection makes CR much faster than the conventional SR. It is noted that this operation may not



(a) build block of ResNet (b) full structure of ResNet (c) collaboratively weighted residuals

Figure 1. Collaboratively weighted residuals of CR and ResNet models.

fit in the memory of a large-scale dataset [38]. In our implementation, we used an incremental strategy [33, 20] to deal with this problem, despite the fact that there are other potential solutions, e.g., dictionary learning [2, 35].

From this, we can obtain the coefficient vector $\hat{\alpha_i}$ related with the ith class. Typically, in SRC the coefficient is utilized to solve the representation residual of a specific class by $\left|\left|y - A_i \cdot \hat{\alpha_i}\right|\right|_2$. Besides this, in the original CRC implementation, the l_2-norm of the sparse coefficient $\left|\left|\hat{\alpha_i}\right|\right|_2$ was also added to obtain more discrimination when performing classification [44]. Finally, the residual is obtained by

$$res_{img} = \left|\left|y - A_i \cdot \hat{\alpha_i}\right|\right|_2. \tag{3}$$

At the same time, a side-by-side CR is performed on deep features. The so-called deep features are specific to the layer in the deep model. Theoretically, any layer can be utilized to extract deep features. For example, the $global\_pool$ layer [8, 9, 25] in the ResNet and Inception models. Here, we denote the deep features from one specific layer (j) of the neural networks for all training samples as

$$B = fea_{cnn}(A, layer_j), \tag{4}$$

and therefore the query sample becomes

$$y_{cnn} = fea_{cnn}(y, layer_j). \tag{5}$$

The size of the feature set is determined by the design of the specific layer, where it is normally mismatched with the size of the original images. It is hard to simply perform integration on the feature level. Therefore, fusion of the feature pairs will be performed on the residuals, which have the size same as the class number.

Then, the representation coefficient obtained from the deep features by a similar CR process is

$$\hat{\beta} = (B^T B + \lambda \cdot I)^{-1} B^T y_{cnn}. \tag{6}$$

After that, the residual between the query feature set y_{cnn} and each class of training feature sets can be solved with the same method as Eq. (3)

$$res_{cnn} = \left|\left|y - B_i \cdot \hat{\beta}_i\right|\right|_2. \tag{7}$$

3.2 Fusion based on collaboratively weighting

Our proposed method manages to retrieve this part of the missed information via a novel fusion operation. The fusion is performed on two residuals, since both have an equal dimension depending on the number of classes. Therefore, fusion on the residuals is not only straightforward, but also faster.

Let us denote the residuals solved from two groups of samples as
$res_{img} = [d_{img,1}, d_{img,2}, \ldots, d_{img,C}]$ and $res_{cnn} = [d_{cnn,1}, d_{cnn,2}, \ldots, d_{cnn,C}]$, where C is the number of classes in the dataset. Then, the fusion via the collaborative weight is performed on the residual vector via an element-wise multiplication,

$$res_{fusion} = res_{img} \odot res_{cnn}, \tag{8}$$

where the residual entry related with the ith class is calculated by the collaborative weight. This weight means that each entry in the residual vector is assigned a weight solved by the collaborative representation of the original images. The information carried by this weight compensates the missing part of the abstract higher layers in a neural network. In this way, we obtain the final fusion residual. Although additional or weighted averages are a more common approach to perform fusion in many other methods [40, 42, 27], they require a set of fine-tuned factors to obtain a good result. What is more, we observed a descending accuracy when adding up two residuals.

Finally, we classify the test sample to a class with minimal residual as follows

$$identity(y) = \arg\min_i (res_{fusion,i}). \tag{9}$$

The idea of our collaborative weight is simplistic and intuitive. The collaborative weights are determined by the relative contribution of each class from the original samples. Each residual of the deep features is overlapped by a weight solved using the collaborative representation of the original samples, in order to integrate its contribution.

3.3 Why deep features works in CR

To answer this question, we first need to answer another question: What is the relationship between collaboration and sparsity? Many had tried to give an answer with some considering sparsity as being more important [30, 6]. On the other hand, others insist that collaboration matters more [44, 18]. However, as for the rest, they treat collaboration and sparsity as equal factors [41, 40, 1]. So far, the last viewpoint well explains our proposed collaboratively weighting deep and non-deep representation.

The subspace occupied by the columns of a sparse dictionary Φ can be denoted as a set of Ψ. Fig. 2 shows the geometrical illustration of this subspace. A test sample y can be approximately represented by the columns of Φ, and the error is $\epsilon = y - \tilde{y}$. In addition, vector $\tilde{y}$ can be decomposed to ξ_i and $\tilde{\xi}_i$, as depicted in Fig. 2 (a1). According to [44], the angles β and γ together decide the robustness of the CR model. However, Fig. 2 (a2) shows that it is likely to have more than one right answer, which is depicted as the circle. The distances of pz and qz are the same. Therefore, CR by itself without considering sparsity may not be robust enough [1].

Fig. 2 (b1) and (b2) show how the sparsity can help in the CR model. In these two cases, where (b1) $\xi_i \neq \tilde{\xi}_i$ and (b2) $\xi_i = \tilde{\xi}_i$, a and b are two paths to points p and q. The distances are also the same $||\epsilon_i||_2 = ||\epsilon_j||_2$ in these two instances, while $a \neq b$ in (b1) and $a = b$ in (b2). This means that the class-specific residuals are equal in both cases, but construction of vectors ξ_i and ξ_j may be different. The components consisting of the path depict the sparsity in the representation, where fewer steps of ξ_i indicates a sparser collaborative representation coefficient α. Using sparser features in CR can help to produce a more robust classification, which is the very reason why DeepCWC works.

The black box deep model provides an unpredictable sparsity in the deep features. Currently, we can only accept this fact according to the largely contracted dimension of the deep feature set, with feature learning being the most powerful characteristic of deep learning. As shown in Fig. 2 (c), the effort of CR on deep features can be painted as a random and unpredictable curve between o and p. This can be treated as potentially the most efficient path and is also the result observed in the experiments.

3.4 Why the fusion is positive

When the deep features are ready and fit well in the CR model, the next problem is how to consolidate both of them into an united set. This is where the collaborative weight works. Previous work showed that well constructing the residuals is helpful to generate a robust sparse model, i.e., the two-phase sparse representation model [36]. To illustrate the impact of the weight, we captured some runtime data from our experiments, which is plotted in Fig. 3.

The purpose of the collaborative weight is to expand the more promising residuals, while restricting the other ones. The target class is selected by a final minimization, hence, we look for the smallest values. For example, the residuals of classes 1-10 are shown on the $ResNet\_v1\_101$ model in Fig. 3 (a). The correct class label is 1, where the distances of CRC on images and deep features are both below 1 (0.72). However, the minimal values of them are 0.65 and 0.62, respectively. This could lead to a wrong classification result (Class 4 and 10). After the fusion, the resultant residual becomes even smaller, resulting in the correct class being chosen (Class 1). This ensures that the classification will not be affected by other nearby classes. The same phenomenon can be found on other models, which are annotated in Fig. 3 (b) - (f). On the other hand, the classes with a larger distance value, e.g., $d_i > 1.0$, the fusion will make it much larger ($m * n > 1.0$, if $m > 1.0$ and $n > 1.0$). This in turn helps to avoid negative results.

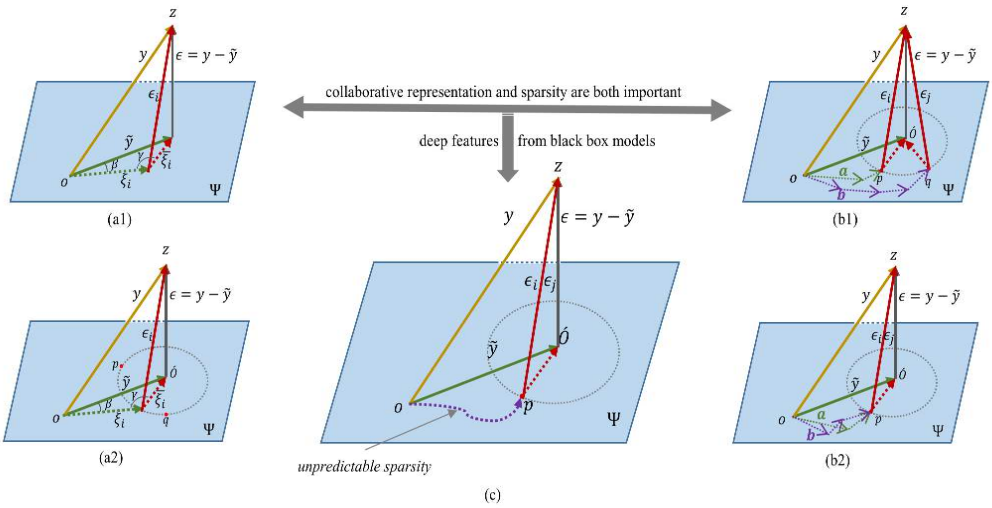


Figure 2. The sparsity from deep features in the CR model.

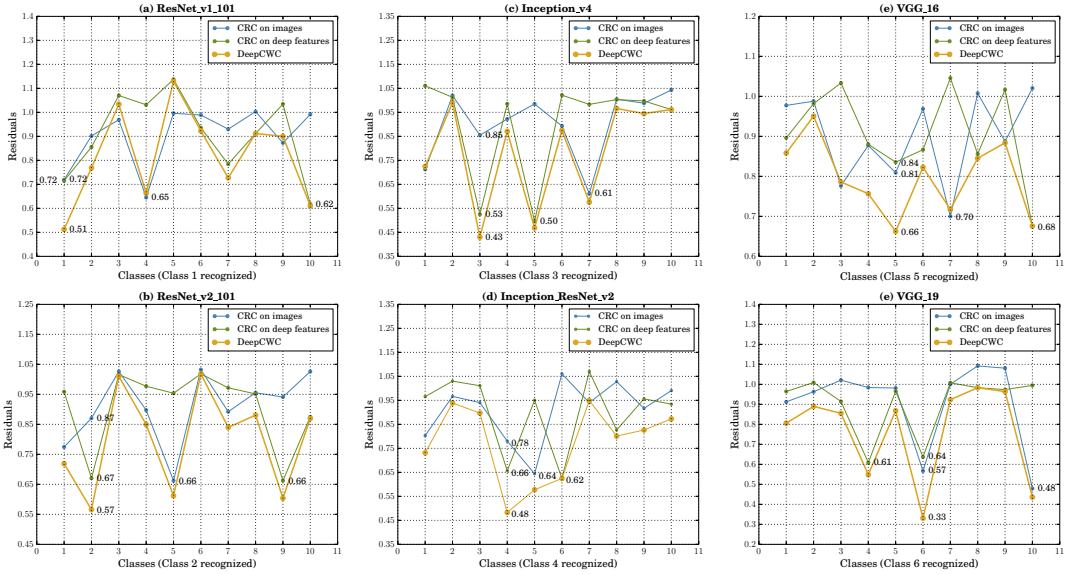


Figure 3. The residuals from the pair of CR representation and the final fusion.

Another point to note in the fusion is that the parameters do not need to be tuned. This is not like other conventional fusion schemas, which usually introduces one or two fusion factors [40, 42, 27], where more parameters call for more tuning. The fusion result is determined by the residuals themselves. We barely need to look for an optimal parameter and maintains the effectiveness of DeepCWC.

4 Experimental results

This section describes the experiments to demonstrate the robustness and performance of the proposed method. First of all, six facial datasets, including FERET [19], MUCT [16], Yale B [7], Georgia Tech (GT) [4], AR [15], and ORL [3], are selected to evaluate the performance of face recognition. These experiments were conducted due to the fact that CRC is usually applied to face recognition. Secondly, another set of experiments had been performed on some object datasets, including a leaf dataset Flavia [26], and three object datasets CIFAR-10 [10], Fashion-MNIST [31] and COIL-100 [17], which are often utilized to evaluate deep learning methods.

Also, in order to evaluate the robustness after introducing the collaborative weight between the original images and the deep features, we extracted the deep features using multiple state-of-the-art deep CNN models, including ResNet_v1_101, ResNet_v2_101, Inception_v4, Inception_ResNet_v2, VGG_16 and VGG_19. All of these are trained previously on the ImageNet dataset [5] in Google TensorFlow[2]. Our assumption is the proposed DeepCWC works on different deep CNN models. The feature extraction is performed on the TensorFlow-Slim library. Besides this, another goal is to investigate which layer of features in a CNN model are more suitable for collaborative weight. Based on the considerations, we conducted a set of relevant experiments and obtained the following results.

4.1 Experiments for face recognition

We ran experiments on six popular benchmark facial datasets. These datasets are relatively small. The smaller datasets do not contain enough samples to train a robust model by CNN, but we can extract the deep features using pre-trained deep CNN models. Our goal in this group of experiments is to compare the classification result between our proposed method and state-of-the-art methods. The best results are shown in Fig. 4 (a).

It is clear that the proposed DeepCWC method outputs a higher recognition accuracy than normal CRC, CRC using deep features and other state-of-the-art methods, no matter which CNN model is used to extract the deep features. It is uncertain that using deep features would generate a higher recognition accuracy than using the original images. For example, when utilizing the ResNet_v1_101 model to extract features of the AR dataset, CRC performs better on original images than deep features, as shown in Fig. 4 (a). And this is also true in some other experimental cases, which shows one of the limitations of a typical deep learning method. This is the very reason why we proposed the DeepCWC.

No matter which dataset, performing fusion of two feature sets based on the collaborative weight generates a higher recognition accuracy. Even in cases that merely use deep features without collaborative weight, e.g., on FERET, MUCT, Yale B, GT, AR and ORL, the proposed DeepCWC helps the recognition by fusing features from the original images and the CNN models.

4.2 Experiments for object recognition

The next set of experiments were performed on some leaf and object datasets, including Flavia (leaf), COIL-100, CIFAR, and Fashion-MNIST. The results are consistent on all datasets, as shown in Fig. 4 (b). Incorporating the deep features learned by ResNet_v1_101, the recognition accuracy (yellow) is much higher than the result on the original images (blue), except the result on the Flavia. However, DeepCWC further pushes the recognition up to an even higher level, and the improvements are stable on all datasets.

The highest accuracy is obtained on the COIL-100 dataset when using the first 60 samples in each class (83%) as the training samples. Deep features are beneficial to classification on this dataset, where DeepCRC (up to 98.83%) outperformed CRC (only 69.0%) by over 30%. That being said, the proposed DeepCWC still produces the highest accuracy of 99.42%, which reached a state-of-the-art level in recognition. On the Flavia leaf dataset, the improvement generated by collaborative weight is remarkable, though the accuracy is relatively lower, as shown in Fig. 4 (b). The results on the CIFAR-10 dataset get an improvement as well. And the improvement (the column Impr) is calculated by the rate of the accuracy from the DeepCWC over the higher one between CRC on images and deep features, and the improvements on the Flavia and Fashion-MNIST are up to 21.01% and 12.41%, respectively.

4.3 Experiments on different layers

Two versions of the ResNet pre-trained models, ResNet_v1_101 [8] and ResNet_v2_101 [9], are tested in this set of experiments. As described above, we borrowed a similar architecture idea from the deep residual network, as shown in Fig. 1. For this reason, we design the first implementation of DeepCWC based on ResNet. There are 101 layers in the network, and we evaluate the proposed

[2]https://github.com/tensorflow/models/tree/master/research/slim#Pretrained

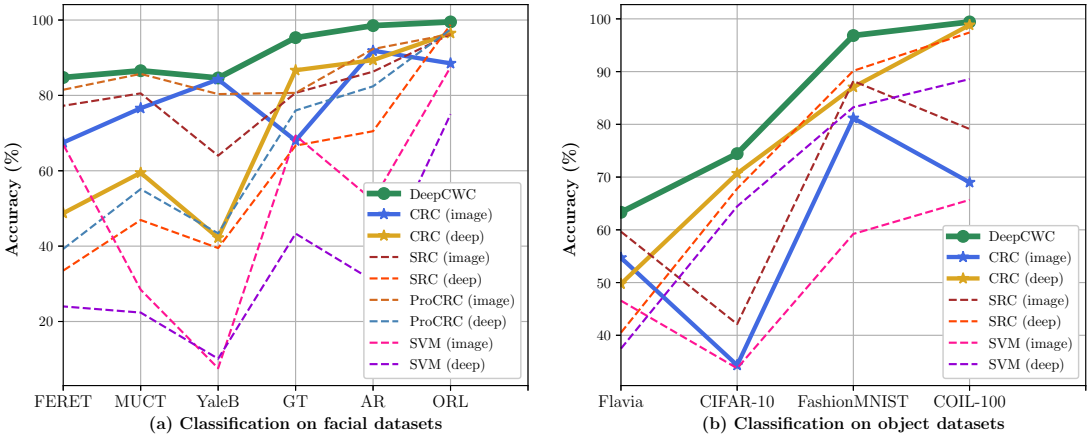


Figure 4. Face and object recognition accuracy comparison.

method on the features obtained from three layers, which are $global\_pool$, $logits$[3] and $spatial\_squeeze$[4] from shallower to deeper. The layer $spatial\_squeeze$ is the layer before the last convolutional layer, while the $logits$ layer is before $spatial\_squeeze$. These two layers have the same feature set size (1000). However, the $global\_pool$ layer is before this layer and has a larger size of 2048. The classification results are demonstrated in Fig. 5 (a) and (b).

In both models, performing CRC on deep features obtained from each layer produces a higher accuracy than using the original images. DeepCWC generates an even higher accuracy than both of these. We observe exactly the same result (97.26%) in both $logits$ and $spatial\_squeeze$ layers. For classification, the feature maps of the last convolutional layer are fed into fully connected layers followed by a softmax logistic regression layer [11]. The global average pooling [14] is introduced to avoid overfitting in the fully connected layers. Using the features maps captured from the $global\_pool$ layer produces a slightly higher accuracy (97.36%). Every result reaches a state-of-the-art level, and is higher than all current implementations (See subsection 4.4). It is noted that the computation time increases due to a larger size (double) of the feature maps from layer $global\_pool$. Therefore, the $logits$ (or $spatial\_squeeze$) layer should be a better choice when considering the balance between accuracy and speed.

To investigate the performance when using a different CNN model, two Inception models are evaluated in a similar way. They are Inception_v4 and Inception_ResNet_v2 models pre-trained on ImageNet. Besides the $global\_pool$ layer, the $Logits$ and $AuxLogits$ layers are also utilized to extract deep features, before being fed into to the linear CR model. A set of similar results are observed in the experiments, as shown in Fig. 5 (c) and (d).

DeepCWC achieves an accuracy over 97% on deep features obtained from three layers. The highest result (97.24%) is the one with the largest size (2048) from the $global\_pool$ layer. In this case, the $AuxLogits$ layer, with a smaller size (1001) than $global\_pool$, produced an approximately equal accuracy of 97.23%. The result from $Logits$ is close to this. In fact, all of the results in DeepCWC for the three cases are stable and close to each other.

The last set of models are of the VGG implementation. We chose VGG-16 and VGG-19 models, and utilized the feature maps from their $fc6$, $fc7$ and $fc8$ layers. The size of both $fc6$ and $fc7$ is 4096, while the last $fc8$ layer has a smaller size of 1000. The largest feature set in this group of experiments produced the most promising classification results. What is more, the trend is consistent with before. As shown in Fig. 5 (e) and (f), the highest accuracy is up to 97.66%, using the shallower $fc6$ layer in VGG-16, which is also the most promising result we obtained using this dataset and in all cases. The results achieved by VGG-19 are slightly lower than this, but higher than the other cases. The larger feature size helps to produce a more accurate classification.

4.4 Comparison to the state-of-the-arts on Fashion-MNIST

The results obtained on the pre-trained models (over 97%) are all state-of-the-art, as shown in in Tab. 1. According to the description of the current methods, all of them are tuned and trained on the Fashion-MNIST dataset locally. Also, most of them applied one or two preprocessing tech-

[3] $resnet\_v1\_101/logits$ or $resnet\_v2\_101/logits$.

[4] $resnet\_v1\_101/spatial\_squeeze$ or $resnet\_v2\_101/spatial\_squeeze$.

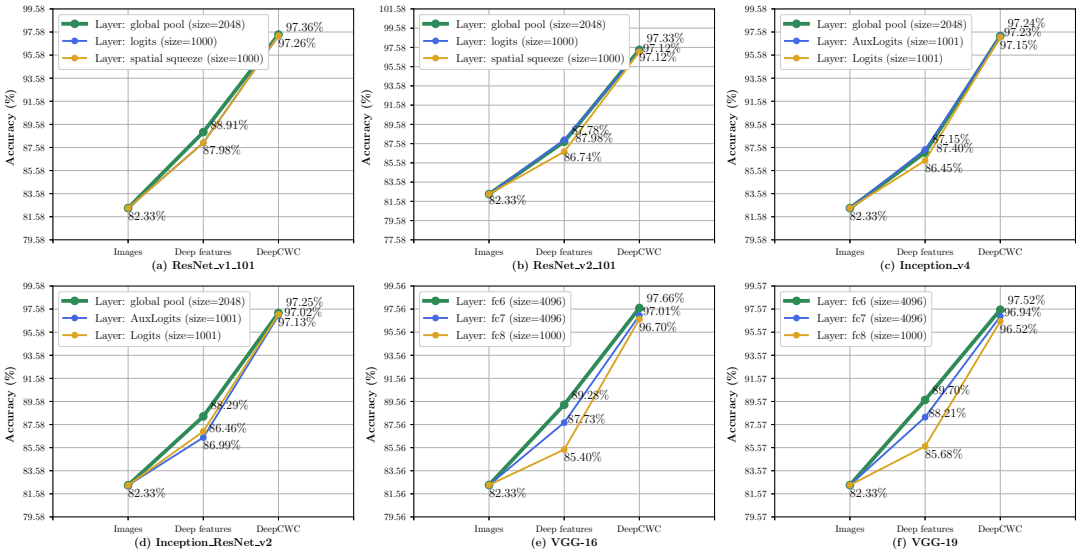


Figure 5. Accuracies with different layers of six deep models.

Table 1. State-of-the-art accuracies on the Fashion-MNIST dataset.

Deep Model (∗)	Preprocessing	Highest Accuracy
CapsNet	None	90.6%
VGG16 26M parameters	None	93.5%
GoogleNet with cross-entropy loss	None	93.7%
MobileNet	Yes	95.0%
DenseNet-BC 768K params	Yes	95.4%
Dual path wide resnet 28-10	Yes	95.7%
WRN-28-10 + Random Erasing	Yes	96.3%
WRN40-4 8.9M params	Yes	96.7%
Ours		
DeepCWC with Inception_v4	None (global pool)	97.24%
DeepCWC with Inception_RN_v2	Yes (global pool)	97.25%
DeepCWC with ResNet_v2_101	Yes (global pool)	97.33%
DeepCWC with ResNet_v1_101	None (global pool)	97.36%
DeepCWC with VGG_19	None (fc6)	97.52%
DeepCWC with VGG_16	None (fc6)	**97.66**%

∗ Data claimed in https://github.com/zalandoresearch/fashion-mnist

∗ CapsNet in https://github.com/naturomics/CapsNet-Tensorflow#results

Table 2. Classification time and speed on different models.

Model	Layer	Feature Size	Time (sec)	Speed (sec/image)
VGG-16	/fc6	4096	16342.347	0.233
VGG-16	/fc8	1000	6826.632	0.098
VGG-19	/fc6	4096	16567.730	0.237
VGG-19	/fc8	1000	6526.632	0.093
Inception_v4	/Logits	1001	7177.612	0.103
Inception_v4	/global_pool	1536	8626.181	0.123
Inception_RN_v2	/Logits	1001	6734.609	0.096
Inception_RN_v2	/global_pool	1536	7884.362	0.113
ResNet_v1_101	/global_pool	2048	9108.256	0.207
ResNet_v1_101	/logits	1000	7302.812	0.104
ResNet_v2_101	/global_pool	2048	8373.421	0.120
ResNet_v2_101	/logits	1000	7254.657	0.104

niques, and used the same deep neutral network architecture, e.g., VGG, ResNet, etc. Previously, the most promising accuracy was obtained by the Wide Residual Networks (WRN) model [39], both of which applied the standard preprocessing (mean/std subtraction/division) and augmentation (random crops/horizontal flips). For example, one used the random erasing technique [45] and produced an accuracy of 96.3% [5], while the other one with 96.7% accuracy had 8.9 M parameters and utilized freezing layers [6].

Compared to current state-of-the-art methods, the proposed DeepCWC produces a higher result using multiple CNN models. The classification accuracy ranges from 97.24% to 97.66%, which are all higher than previous methods. The highest accuracy is generated on VGG-16 from the $fc6$ layer with a size of 4096.

4.5 Discussion

Our experimental machine was configured with the following hardware, including an Intel® Core™ i7-7820X CPU@3.60GHz x 16, 64 GB RAM, 1.3 TB SSD and one NVIDIA TITAN Xp GPU. The code was run on TensorFlow 1.6, MATLAB R2016 and Ubuntu 16.04 OS. The recorded time consumption of each experimental case is shown in Tab. 2.

This time includes the whole training and testing of both the original images and deep features in CR, but does not count the time for feature extraction by the pre-trained models. Therefore, the speed (seconds per sample) is calculated by dividing the total time by the size of dataset (70000). Considering that the running state of the machine may fluctuate, the speed is between 0.1 - 0.2 seconds per sample. Furthermore, the following can be discussed about the proposed method.

[5] https://github.com/zhunzhong07/Random-Erasing

[6] https://github.com/ajbrock/FreezeOut

Linear representation such as CR can improve deep neural networks based representation learning. Even using a pre-trained model, the proposed DeepCWC achieved a state-of-the-art classification result on the Fashion-MNIST dataset. The accuracy and performance outperformed current popular methods as well. This gives us a clue that linear methods have a new way to cooperate with nonlinear models.

The collaborative weight of two diverse representations help produce an accurate classifier. Currently, more work is focusing on the neural network architecture and/or parameter tuning. However, the proposed DeepCWC neither pays much attention to deep learning model itself, nor tunes any parameters. Fusing multiple representations creates a robust classifier that works well on multiple deep learning models. The results are all at a state-of-the-art level.

The global average pool layer shows an effective capacity to extract discriminative features. Global average pooling was proposed to enforce the learning of the class level feature maps [14]. The experiments on layer analysis showed that using features extracted from the global average pool layer can produce a higher accuracy. This is true in the Inception and ResNet models, as shown in Fig. 4.3 (a) and (d), and Tab. 1. That being said, it needs more computation due to the relatively larger size of the feature set from this layer.

Multiple layers in a deep CNN model show an effective capacity to extract discriminative features, including the global average pooling layer [14] and the fully connected layer. This is confirmed in the experiments, as shown in Fig. 4.3, and Tab. 1. However, the size of feature map decides the time consumption.

The proposed DeepCWC takes the NO. 1 position in current benchmark rank of Fashion-MNIST. The most promising result is obtained on the VGG-16 model, which

outperforms current leaders mainly using the WRN model.

5 Conclusions

We propose a deep collaborative weight-based classification (DeepCWC) method. It first performs the linear representation on original images and deep features, extracted from nonlinear neural networks. Then, both of them collaboratively weight each other to build a strong discriminative classifier. The method is extensively evaluated using multiple popular deep CNN models, like ResNet, Inception, and VGG. The experimental results are promising on more than one layers in these neural networks, with most of the results belonging to a state-of-the-art level.

The l_2-norm based CR model is chosen as the linear constraint in this work to enhance the classification based on pre-trained CNN models. However, there are still some questions, for example, whether there are other linear models (like sparse representation, dictionary learning, etc.), more suitable for the same task, or whether it can bring one more step of break-through when applied on locally trained and tuned CNN models. We will keep working on these open topics in the future.

6 The Acknowledgements

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] N. Akhtar, F. Shafait, and A. Mian. Efficient classification with sparsity augmented collaborative representation. *Pattern Recognition*, 65:136–145, 2017.

[2] S. Cai, L. Zhang, W. Zuo, and X. Feng. A probabilistic collaborative representation based approach for pattern classification. In *Computer Vision and Pattern Recognition*, pages 2950–2959, 2016.

[3] A. L. Cambridge. The orl database of faces. `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`. Online; accessed 12-Octobor-2017.

[4] L. Chen, H. Man, and A. V. Nefian. Face recognition based on multi-class mapping of fisher scores. *Pattern Recognition*, 38(6):799–811, 2005.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[6] W. Deng, J. Hu, and J. Guo. In defense of sparsity based face recognition. In *Computer vision and pattern recognition (cvpr), 2013 ieee conference on*, pages 399–406. IEEE, 2013.

[7] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE transactions on pattern analysis and machine intelligence*, 23(6):643–660, 2001.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. IEEE, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[10] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[13] Y. Li, P. Richtarik, L. Ding, and X. Gao. On the decision boundary of deep neural networks. *arXiv preprint arXiv:1808.05385*, 2018.

[14] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[15] A. M. Martinez. The ar face database. *CVC Technical Report*, 24, 1998.

[16] S. Milborrow, J. Morkel, and F. Nicolls. The MUCT Landmarked Face Database. *Pattern Recognition Association of South Africa*, 2010. `http://www.milbo.org/muct`.

[17] S. Nayar, S. Nene, and H. Murase. Columbia object image library (coil 100). *Department of Comp. Science, Columbia University, Tech. Rep. CUCS-006-96*, 1996.

[18] X. Peng, L. Zhang, Z. Yi, and K. K. Tan. Learning locality-constrained collaborative representation for robust face recognition. *Pattern Recognition*, 47(9):2794–2806, 2014.

[19] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss. The feret evaluation methodology for face-recognition algorithms. *IEEE Transactions on pattern analysis and machine intelligence*, 22(10):1090–1104, 2000.

[20] M. Ristin, M. Guillaumin, J. Gall, and L. V. Gool. Incremental learning of ncm forests for large-scale image classification. In *Computer Vision and Pattern Recognition*, pages 3654–3661, 2014.

[21] S. Scardapane, D. Comminiello, A. Hussain, and A. Uncini. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.

[22] L. Shao, Z. Cai, L. Liu, and K. Lu. Performance evaluation of deep feature learning for rgb-d image/video classification. *Information Sciences*, 385:266–283, 2017.

[23] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[24] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[25] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inceptionv4, inceptionresnet and the impact of residual connections on learning. In *AAAI*, pages 4278–4284, 2017.

[26] A. Valada, L. Spinello, and W. Burgard. Deep feature learning for acoustics-based terrain classification. In *Robotics Research*, pages 21–37. Springer, 2018.

[27] J. Wen, B. Zhang, Y. Xu, J. Yang, and N. Han. Adaptive weighted nonnegative low-rank representation. *Pattern Recognition*, 81:326–340, 2018.

[28] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2074–2082, 2016.

[29] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.

[30] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.

[31] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[32] T. Xiao, H. Li, W. Ouyang, and X. Wang. Learning deep feature representations with domain guided dropout for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1249–1258, 2016.

[33] T. Xiao, J. Zhang, K. Yang, Y. Peng, and Z. Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *ACM International Conference on Multimedia*, pages 177–186, 2014.

[34] B. Xin, Y. Wang, W. Gao, D. Wipf, and B. Wang. Maximal sparsity with deep networks? In *Advances in Neural Information Processing Systems*, pages 4340–4348, 2016.

[35] Y. Xu, Z. Li, B. Zhang, J. Yang, and J. You. Sample diversity, representation effectiveness and robust dictionary learning for face recognition. *Information Sciences*, 375(C):171–182, 2017.

[36] Y. Xu, D. Zhang, J. Yang, and J.-Y. Yang. A two-phase test sample sparse representation method for use with face recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(9):1255–1262, 2011.

[37] Y. Xu, Z. Zhong, J. Yang, J. You, and D. Zhang. A new discriminative sparse representation method for robust face recognition via $l\_\{2\}$ regularization. *IEEE transactions on neural networks and learning systems*, 28(10):2233–2242, 2017.

[38] H. F. Yu, C. J. Hsieh, K. W. Chang, and C. J. Lin. Large linear classification when data cannot fit in memory. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(4):23, 2012.

[39] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[40] S. Zeng, J. Gou, and X. Yang. Improving sparsity of coefficients for robust sparse and collaborative representation-based image classification. *Neural Computing & Applications*, pages 1–14, 2017.

[41] S. Zeng, X. Yang, and J. Gou. Multiplication fusion of sparse and collaborative representation for robust face recognition. *Multimedia Tools and Applications*, 76(20):20889–20907, 2017.

[42] S. Zeng, B. Zhang, and Y. Du. Joint distances by sparse representation and locality-constrained dictionary learning for robust leaf recognition. *Computers and Electronics in Agriculture*, 142:563–571, 2017.

[43] S. Zeng, B. Zhang, Y. Lan, and J. Gou. Robust collaborative representation-based classification via regularization of truncated total least squares. *Neural Computing and Applications*, pages 1–9, 2018.

[44] L. Zhang, M. Yang, and X. Feng. Sparse representation or collaborative representation: Which helps face recognition? In *IEEE International Conference on Computer Vision*, pages 471–478, 2012.

[45] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.

[46] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.