NEURAL DISCRIMINANT ANALYSIS FOR FINE-GRAINED CLASSIFICATION

Mai Lan Ha, Volker Blanz

Department of Electrical Engineering and Computer Science
School of Science and Technology
University of Siegen

ABSTRACT

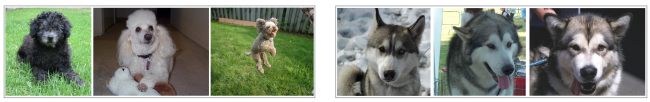
Feature learning is an essential process in image and object classification, even more so for fine-grained classification where objects have a similar global structure and subtle differences in local parts. Inspired by Linear Discriminant Analysis (LDA), we propose a two-phase optimization to transform deep features from their original space to a lower dimension space using neural networks with two primary goals: minimizing variances within each class and maximizing pairwise distances between features from different classes. The approach produces more discriminative features that lead to improvements in classification performance. We evaluate our method on four well-known fine-grained classification datasets. Our optimization leads to significantly better classification accuracy (up to 6.4% increase from the baseline). The standard deviations of the accuracy over multiple training runs show that our approach yields more consistent and reliable results than transfer learning.

***Index Terms*—** LDA, NDA, discriminant analysis, feature transformation, fine-grained classification

1. INTRODUCTION

The task of Fine-Grained Visual Classification (FGVC) is to classify sub-classes under one common super-class. Examples are recognizing different sub-species of birds [1, 2], different models and manufactures of cars [3], sub-types of flowers [4], etc. On the one hand, it is challenging because the sub-classes share the same visual structures and appearance; the differences are very subtle. In many cases, it requires domain experts to distinguish and label these sub-classes by recognizing their discriminative features on specific parts of the objects. Therefore, it is also a great challenge to obtain large-scale datasets for FGVC. On the other hand, the intra-class variance can be visually higher than the inter-class variance. Such cases can be seen in different colors and poses of objects in the same sub-class [1] (Figure 1).

FGVC methods, especially those that use Deep Convolutional Neural Networks (DCNNs), aim to learn discriminative features that can address the subtle differences among classes and the high variance within each class, which are the specific challenging aspects of FGVC. One recent approach that produces state-of-the-art results is to extract subsets of the ImageNet dataset [5] that are visually similar and relevant to fine-grained classification classes and combine them with other datasets to do transfer learning and fine-tuning [6, 7]. These methods are data-driven and take advantage of DCNNs. While the large-scale data empowers this domain adaptation and transfer learning approach, the task of learning discriminative features for FGVC is left to the DCNNs.



(a) (b)

Fig. 1: An example of high intra-class variance in (a) and low inter-class variance in (b). In (a): the miniature poodles (same class) have different colors and poses. In (b): images are from three different classes (Siberian Husky, Malamute, and Eskimo) that are difficult to distinguish.

Inspired by LDA [8], we take on the method's objectives and combine it with the power of neural networks to improve discriminative features. Our goal is to optimize the learned features produced by transfer learning or domain adaptation such that they become more discriminative and therefore lead to better classification results. This optimization focuses on solving these three criteria: (i) Reducing intra-class variance by minimizing the total distance between features in the same class to their means. (ii) Increasing inter-class variance by transforming the feature space such that features between two classes in the target space are moved further apart. (iii) Improving classification accuracy.

We implement all those three criteria using a small neural network with a two-phase optimization approach called Neural Discriminant Analysis (NDA). The first phase of the NDA optimization is to pull feature points within each class to the class center (class mean). In the second phase, we construct a Siamese network that uses the NDA network as a shared base. The Siamese optimization will pull pairs of features of the same class together and push pairs of features that belong to different classes apart. Finally, we train a classification layer

[1] Subsequently, we will use "classes" instead of "sub-classes" for short.

on the features produced by the NDA network. We evaluate our NDA network on four different FGVC datasets. Our results are better than the state-of-the-art on three datasets by a large margin and in a close match with the last dataset. The improvements are up to 5.4% over the state-of-the-art results and up to 6.4% compared to the baseline. We also compute standard deviations on the accuracy results of classification with and without NDA optimization. The statistics show that NDA optimization yields a more stable accuracy across different rounds of training with random initializations.

Our contribution to the FGVC field is an effective and useful two-phase Neural Discriminant Analysis (NDA) optimization that is implemented using a neural network with a few fully connected layers. The NDA optimization helps to transform the original features into a set of new features that are more discriminative. Our proposed optimization leads to improving fine-grained classification performances. As a small independent component, NDA can be easily added to the last feature layer of an existing classification DCNN to form an end-to-end network.

2. RELATED WORK

In FGVC, experts distinguish sub-classes (or classes) based on specific parts of the objects. Therefore, a straight-forward approach is to learn features of object parts [9, 10, 11]. This approach often requires heavy part annotations from domain experts, and therefore it is not very easy to extend to large-scale datasets. Another set of works focus on visual attention on discriminative regions [12, 13, 14, 15]. Different pooling techniques have also been developed such as bilinear pooling to study the interactions of sets of local features [16, 17, 18]. Besides the high intra-class and low inter-class variance challenge, FGVC also faces a problem from small scale datasets. In order to address this issue, researchers work on different strategies to collect more relevant images to enrich the datasets [7, 19] or employ human in the loop to bootstrap datasets [20, 21]. Cui *et al*.[6] and Zhang *et al*.[7] use subsets of ImageNet [5] that are visually similar to FGVC classes, in combination with iNat [22] or L-Bird [23] dataset to do transfer learning and fine-tuning. These methods hold state-of-the-art results in several benchmark datasets.

From high-level features produced by pre-trained DCNNs for classification,we take a step further by transforming deep features in order to move instances among classes apart and bring instances from the same class closer together. This objective is similar to Linear Discriminant Analysis (LDA). However, we implement it using a neural network architecture, thus the term Neural Discriminant Analysis (NDA).

3. NEURAL DISCRIMINANT ANALYSIS (NDA)

Similar to LDA, NDA operates on feature spaces. Features for image classifications can be obtained from various types of network models. We opt to use pre-existing transfer learning networks from [6] to extract classification features and use them as pre-optimized data for our method.

3.1. Feature discriminant analysis

Let f be the pre-optimized features that are extracted from a pre-trained classification network and $\mathcal{F}$ be the transformed features of f.

$$\mathcal{F} = \mathcal{N}(f), \tag{1}$$

where $\mathcal{N}$ is a general function for feature transformation and optional dimensionality reduction. The optimization objectives for $\mathcal{N}$ are the following:

- **Maximizing the fine-grained classification results:** Let $\mathcal{N}^1$ is the classification function to classify feature $\mathcal{F}$. Let the classification results be $q(x) = \mathcal{N}^1(\mathcal{F})$. Maximizing classification results is equivalent to minimizing the categorical cross entropy loss:

$$\mathcal{H}(p,q) = -\sum_{\forall x} p(x) log(q(x)), \tag{2}$$

  where $p(x)$ is the classification ground-truth.

- **Minimizing intra-class variance:** This is to minimize the total distances for all the feature points $\mathcal{F}_j^i$ of class i to its mean value, calculated as below for n classes:

$$\mathcal{L}_{Mean} = \sum_{i=1}^{n} \sum_{j} \left(\mathcal{F}_j^i - \bar{\mathcal{F}}^i\right) \tag{3}$$

  where $\bar{\mathcal{F}}^i$ are the mean features of class i.

- **Maximizing inter-class variance:** For this optimization, we propose to use pairs of images. If a pair of images belong to the same class, we want to reduce the distance between the corresponding image features; otherwise, we want to increase the distance between them. The effect is to push features from different classes apart while keeping features within the same class close to each other. Let $y = 0$ if two features $\mathcal{F}_k$ and $\mathcal{F}_l$ are from the same class and $y = 1$ if they are from different classes. The optimization for inter-class variance minimizes the following function:

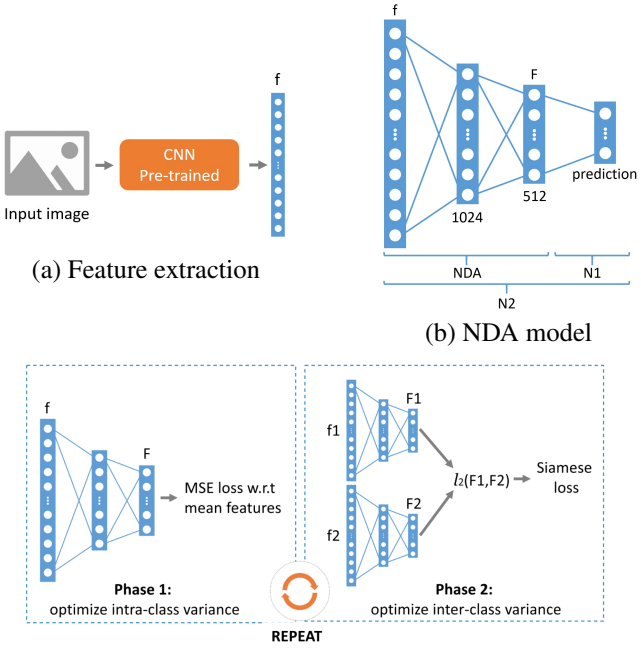
$$\mathcal{L}_{Siamese} = (1-y)(1-e^{-d}) + y * e^{-d} \tag{4}$$

$$d = l_2(\mathcal{F}_k, \mathcal{F}_l) \tag{5}$$

  d: Euclidean distance between two features $\mathcal{F}_k$ and $\mathcal{F}_l$.

When $y = 0$, Eq. 4, which is the Siamese loss, is also optimized for intra-class distance, but this is a weak constraint compared to Eq. 3. In the Siamese loss, the optimization is done pairwise and batch-based. As a result, feature points move relative to each other. On the other hand, the Eq. 3 is optimized such that all the feature points move to the pre-calculated class mean. The superior performance of NDA over Siamese optimization is presented in Table 2.

3.2. Discriminant analysis with neural networks



(a) Feature extraction

(b) NDA model

(c) NDA two-phase optimization

Fig. 2: An overview of the NDA model and its optimization.

The feature transformation function $\mathcal{N}$ in Eq. 1 is implemented as a neural network (NDA) that step-by-step reduces the dimension of the pre-optimized features. To satisfy the objective in Eq. 2, which is maximizing the classification results, we first append a single-layer neural network $\mathcal{N}^1$ at the end of the NDA network to train for classification using categorical cross-entropy loss. The result is a classification neural network $\mathcal{N}^2 = \mathcal{N}^1(\mathcal{N})$ (Fig. 2(b)). This first step of training is to make sure the dimension reduction of the NDA network $\mathcal{N}$ can maintain the classification accuracy from the original features.

The optimizations for inter-class and intra-class variance should be done jointly. We develop a two-phase optimization for each epoch of training. In the first phase, we optimize for the Eq. 3. All the features f_j^i are fed forward to the NDA network, forming features $\mathcal{F}_j^i$. The mean features $\bar{\mathcal{F}}^i$ are computed for every class i. We then use the Mean Squared Error loss to minimize the distances between all features $\mathcal{F}_j^i$ of the same class i to their mean feature $\bar{\mathcal{F}}^i$. In the second phase, we optimize for the inter-class variance. We use the NDA network as a base to form a Siamese network $\mathcal{S}$. The Siamese network $\mathcal{S}$ has two input features f_k and f_l and outputs the Euclidean distance between $\mathcal{F}_k$ and $\mathcal{F}_l$. If f_k and f_l are in the same class, the output distance is reduced, otherwise, increased. The loss function for the Siamese network $\mathcal{S}$ is implemented as in Eq. 4. Finally, we retrain the 1-layer prediction network $\mathcal{N}^1$ on the transformed features $\mathcal{F}$ produced by the NDA network $\mathcal{N}$ for fine-grained classification. The accuracy is improved thanks to the NDA optimized features.

Our NDA optimization alternates between Eq. 3 (minimize the mean variance) and Eq. 4 (optimize the Siamese loss) in each epoch. The two losses are not combined into one single optimization. Thus, no loss weights are required.

4. EXPERIMENTS

We evaluate our proposed method on the following FGVC datasets: CUB-200-2011 [1] that has 200 types of birds, Flower-102 [4] with 102 types of flowers, Stanford-Cars [3] that has 196 types of cars and NABirds [2] with 555 classes of birds. We use Inception-ResNet-V2, Inception-ResNet-V2-SE and Inception-V3-iNat models from Cui *et al.*[6] as base networks to compare across all datasets.

Implementation: Input images of dimensions 448×448 pixels are used for all the experiments. The NDA network $\mathcal{N}$ is a two-layer neural network with an input feature size of $2,048$ or $1,536$, depending on the networks that are used to extract the features. The first hidden layer has $1,024$ nodes followed by a ReLU activation, and the last layer has 512 nodes, also with ReLU activation. We add 0.35 and 0.25 dropout for each layer, respectively. The classification network $\mathcal{N}^1$ contains only a single prediction layer with a softmax activation. By concatenating $\mathcal{N}^1$ after $\mathcal{N}$, we have a neural network $\mathcal{N}^2 = \mathcal{N}^1(\mathcal{N})$. In the initial training, we train $\mathcal{N}^2$ to reach the reported accuracy in [6]. The number of training epochs can range from 100 to $1,000$, depending on the datasets and the types of pre-optimized features. NDA is optimized using 20 to 30 epochs for all the experiments. SGD optimizer with learning rate 10^{-5} is used for minimizing variance to class means, and RMSprop optimizer with learning rate 10^{-4} is used for the Siamese optimization. For each epoch, we re-compute the mean for each class and re-generate input data pairs for the Siamese optimization.

Data sampling: After each epoch, we re-sample pairs of data for the Siamese loss optimization in Eq. 4. For each image in a class, we randomly select one image from the same class and k images from different classes. The intra / inter ratio is therefore $1:k$, where $k \geq 1$. We chose $k = 2$, which leads to good results and surpasses the state-of-the-art. Increasing k will significantly increase the training time. In training, the data is shuffled randomly.

Finally, we re-train the single layer prediction network $\mathcal{N}^1$. It usually takes half the number of epochs of the transfer learning for the training to converge. We compare our results with the state-of-the-art in Table 1. We use features extracted from Inception-ResNet-V2, Inception-ResNet-V2-SE and Inception-V3-iNat models from [6]. The best performances of transfer learning networks in [6] are from Inception-V3 and Inception-ResNet-V2-SE. Even though the features from Inception-ResNet-V2 transfer learning do not produce the best results on their own, we are still able to top the state-of-the-art in a majority of the evaluated datasets.

Accuracy: Our method obtains the state-of-the-art in 3

Method	CUB-200	Stanford-Cars	Flower-102	NABirds
DLA [24]	85.1	94.1	-	-
Object-part-Attention [12]	85.8	92.2	97.1	-
Pairwise-Confusion [25]	86.9	92.9	91.4	82.8
Multi-Attention [14]	86.5	93.0	-	-
HBP [17]	87.1	93.7	-	-
DCL [11]	87.8	94.5	-	-
TASN [15]	87.9	93.8	-	-
Ge *et al.*[26]	**90.4**	-	-	-
Best of Transfer Learning [6] (baseline)	89.6	93.5	**97.7**	87.9
NDA (Inception-V3-iNat) (ours)	87.4	**99.9**	95.5	83.9
NDA (Inc-Res-V2) (ours)	**90.1**	97.4	**97.7**	88.4
NDA (Inc-Res-V2-SE) (ours)	89.7	**99.9**	**97.7**	**89.5**

Table 1: Comparison of results with the state-of-the-art. The features used in our methods are from transfer learning networks in [6]. The best results are reported for Transfer Learning [6] which are from their Inception-V3 or Inception-ResNet-v2-SE transfer learning networks and used as baselines. All of our results are averaged over 10 training runs with random initializations. "Inc-Res" is short for Inception-ResNet.

Method	CUB-200	Cars	Flower	NABirds
Siamese	89.7	96.5	97.5	88.2
NDA (ours)	**90.1**	**97.4**	**97.7**	**88.4**

Table 2: Comparison of results for NDA optimization and Siamese network (without Mean Loss) using Inc-Res-V2 model. All of our results are averaged over 10 training runs. The comparison in Table 1 shows that all the methods in this field compete at 0.1%, so the improvement of NDA over Siamese is substantial.

Model	Method	CUB-200	Cars	Flower
Inception-	TL	0.74	2.53	0.22
ResNet-V2	NDA	**0.28**	**1.80**	**0.10**
Inception-	TL	0.69	3.88	**0.13**
ResNet-V2-SE	NDA	**0.17**	**0.00**	**0.13**
Inception-	TL	3.96	8.32	0.50
V3-iNat	NDA	**0.27**	**0.00**	**0.23**

Table 3: Training stability: standard deviations of accuracy over 10 training runs for different types of networks. TL is short for transfer learning. The standard deviations for classification using NDA optimization are consistently smaller than those of classification without NDA optimization. It shows that the NDA optimization produces more stable results across different training runs.

datasets: Flower-102 [4], Stanford-Cars [3] and NABirds [2], and close to the best performance on CUB-200-2011 [1]. Our method surpasses the state-of-the-art on Stanford-Cars dataset by 5.4%, on NABirds dataset by 1.6%. Compared to the baselines, NDA is superior by 0.5%, 6.4% and 1.6% on CUB-200, Stanford-Cars and NABirds datasets, respectively. It is worth to take note that FGVC is a competitive field, all the methods compete with each other at 0.1% improvement. Therefore, the improvement from NDA is significant.

Reliability: We compute and report standard deviations of the accuracy across 10 runs per dataset per network in Table 3. The standard deviations are consistently lower in all NDA optimization results compared to transfer learning. It shows that the NDA optimization transforms the features such that the classification becomes more stable and reliable.

Two-phase NDA optimization: Our NDA optimization proves its effectiveness throughout various datasets (Table 1). If we use only one phase of the NDA optimization, such as Siamese Loss alone, the performance drops (Table 2). Without the Mean Loss, it lacks a strong and explicit constraint for intra-class optimization. Without Siamese Loss, there is no inter-class optimization. After completing the feature-based optimization, we can add the NDA neural network after the last feature layer of the base network that was used for feature extraction to make it a complete end-to-end model for prediction or testing. The end-to-end model is the advantage of NDA over the classical LDA. With LDA, we cannot integrate the feature transformation directly to DCNNs; it needs an additional classification using SVM or a neural network.

5. CONCLUSION

Inspired by the objectives of Linear Discriminant Analysis (LDA) and making use of the power of deep learning and neural networks, we propose a Neural Discriminant Analysis (NDA) optimization that is useful for Fine-Grained Visual Classification (FGVC). It is a two-phase optimization that minimizes the intra-class variance and maximizes the inter-class variance in the deep feature domain. We obtain the state-of-the-art accuracy on several popular FGVC datasets with a large margin. The analysis also shows that our optimization produces more stable and reliable results. Furthermore, the NDA model can be used on its own in the deep feature domain or as a plug-in component to existing DCNNs.

6. REFERENCES

[1] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," Tech. Rep. CNS-TR-2011-001, California Institute of Technology, 2011.

[2] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie, "Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection," in *CVPR*, 2015, pp. 595–604.

[3] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *CVPR*, 2015, pp. 3973–3981.

[4] M. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics Image Processing*, 2008, pp. 722–729.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR*, 2009.

[6] Y. Cui, Y. Song, C. Sun, A. Howard, and S. Belongie, "Large scale fine-grained categorization and domain-specific transfer learning," in *CVPR*, 2018.

[7] Y. Zhang, H. Tang, and K. Jia, "Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data," in *ECCV*, 2018.

[8] A. J. Izenman, *Linear Discriminant Analysis*, chapter 8, pp. 237–280, Springer, New York, NY, 2013.

[9] S. Huang, Z. Xu, D. Tao, and Y. Zhang, "Part-stacked CNN for fine-grained visual categorization," in *CVPR*, 2016.

[10] H. Zhang, T. Xu, M. Elhoseiny, X. Huang, S. Zhang, A. Elgammal, and D. Metaxas, "SPDA-CNN: Unifying semantic part detection and abstraction for fine-grained recognition," in *CVPR*, 2016, pp. 1143–1152.

[11] Y. Chen, Y. Bai, W. Zhang, and T. Mei, "Destruction and construction learning for fine-grained image recognition," in *CVPR*, 2019.

[12] Y. Peng, X. He, and J. Zhao, "Object-part attention model for fine-grained image classification," *IEEE Trans. Image Processing (TIP)*, vol. 27, no. 3, pp. 1487–1500, 2018.

[13] Z. Yang, T. Luo, D. Wang, Z. Hu, J. Gao, and L. Wang, "Learning to navigate for fine-grained classification," in *ECCV*, 2018.

[14] M. Sun, Y. Yuan, F. Zhou, and E. Ding, "Multi-attention multi-class constraint for fine-grained image recognition," in *ECCV*, 2018.

[15] H. Zheng, J. Fu, Z.J. Zha, and J. Luo, "Looking for the devil in the details: Learning trilinear attention sampling network for fine-grained image recognition," in *CVPR*, 2019.

[16] Y. Cui, F. Zhou, J. Wang, X. Liu, Y. Lin, and S. Belongie, "Kernel pooling for convolutional neural networks," in *CVPR*, 2017, pp. 3049–3058.

[17] C. Yu, X. Zhao, Q. Zheng, P. Zhang, and X. You, "Hierarchical bilinear pooling for fine-grained visual recognition," in *ECCV*, 2018.

[18] X. Wei, Y. Zhang, Y. Gong, J. Zhang, and N. Zheng, "Grassmann pooling as compact homogeneous bilinear pooling for fine-grained visual classification," in *ECCV*, 2018.

[19] Z. Xu, S. Huang, Y. Zhang, and D. Tao, "Webly-supervised fine-grained visual categorization via deep domain adaptation," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 40, no. 5, pp. 1100–1113, 2018.

[20] Y. Cui, F. Zhou, Y. Lin, and S. Belongie, "Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop," in *CVPR*, 2016.

[21] J. Deng, J. Krause, M. Stark, and L. Fei-Fei, "Leveraging the wisdom of the crowd for fine-grained recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 38, no. 4, pp. 666–676, 2016.

[22] G. V. Horn, O. M. Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. Belongie, "The inaturalist species classification and detection dataset," in *CVPR*, 2018, pp. 8769–8778.

[23] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei, "The unreasonable effectiveness of noisy data for fine-grained recognition," in *ECCV*, 2016, vol. 9907, pp. 301–320.

[24] F. Yu, D. Wang, and T. Darrell, "Deep layer aggregation," *CVPR*, pp. 2403–2412, 2018.

[25] A. Dubey, O. Gupta, P. Guo, R. Raskar, R. Farrell, and N. Naik, "Pairwise confusion for fine-grained visual classification," in *ECCV*, 2018.

[26] W. Ge, X. Lin, and Y. Yu, "Weakly supervised complementary parts models for fine-grained image classification from the bottom up," in *CVPR*, 2019.