Extending B-CNN to Hierarchical Fine-Grained Visual Classification

Jeng Shih-Ying
Dept. of Electrical and Electronic Engineering
University of Hong Kong
jengtallis@hku.hk

Abstract

Machine learning has shown great success in visual classification. However, despite great progress in basic-level visual classification, FGVC (Fine-Grained Visual Classification) remains highly challenging. Although hierarchical classification claims beneficial restriction of error to sub-classes, as compared to flat classification, few studies explore machine learning approaches for hierarchical visual classification. The project explores the research gap by extending the novel B-CNN (Branch Convolutional Neural Network) [1] *architecture, which captures the hierarchical label tree structure of the dataset, to fine-grained visual classification. The BT-strategy (Brach Training strategy) presented by Zhu and Bain* [1] *for training B-CNN does not readily demonstrate empirical effectiveness when the B-CNN model is extended to fine-grained visual classification. A novel LBT-strategy (Loop Branch Training strategy), different from the BT-strategy presented by Zhu and Bain, is introduced for B-CNN on hierarchical fine-grained visual classification. The B-CNN architecture together with the LBT-strategy are shown to outperform the corresponding baseline CNN on fine-grained visual classification tasks with experiments on the benchmark dataset Leeds Butterfly* [2].

Keywords
CNN, B-CNN, FGVC, hierarchical classification.

1. Introduction

Psychologists suggest that human visual systems develop the ability to perform basic-level visual classification (e.g. pig vs. dog) well before the ability to perform fine-grained visual classification) (e.g. Belgian Malinois vs. German Shepherd Dog), or subordinate-level visual classification [3]. A similar trajectory in computer vision is seen. While great progress in basic-level visual classification has been made, fine-grained visual classification remains a challenging area [4].

FGVC (Fine-Grained Visual Classification) is a difficult task even for human beings [5] due to the small inter-class variance in the fine-grained dataset.

Small inter-class variance refers to that the difference between instances of difference classes is little, namely that different classes appear visually similar. For instance, different species of butterfly may appear visually similar, as illustrated in Figure 1.

Figure 1. FGVC difficulty: small inter-class variance.

The implication of small inter-class variance is that the intra-class variance, namely the variance within instances of the same class, could be potentially larger than inter-class variance, which further complicates the task [6]. The fine-grained labeling of images typically require teams of experts [7] or extensive crowd-sourcing pipelines [8]. Automated visual systems for such tasks could therefore be valuable in various fields of applications.

Since the remarkable success of AlexNet in the 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge) [9], which drew great attention of the computer vision community, a great amount of emphasis has been placed on machine learning methodologies. Nevertheless, machine learning for fine-grained visual recognition remains a field with great room for improvement [4].

Traditional machine learning image classifiers are single-layer flat classifiers which make predictions on a single level of classes [10], [11]. Such structural design reflects an assumption that all the classes are on the same level, and are equally difficult to classify [1]. However, an inherent

hierarchical data structure is seen in fine-grained datasets where the images are at subordinate levels. The assumption of classes at same level is therefore violated in fine-grained dataset. In addition, Zeiler and Fergus have shown that the lower layers in CNN (Convolutional Neural Network) captures lower level features of the images such as basic shapes whereas higher layers capture higher level features such as the pattern of a butterfly wing or the nose of a dog [12], which suggests that CNN models are intrinsically hierarchical. Therefore, it is reasonable to assume that CNN models with careful structural modification could better capture the hierarchical label tree structure of the data and make more informed predictions accordingly, which is potentially advantageous as the hierarchical prior could be used to guide the classification from coarse classes to fine-grained classes, restricting errors to fine level sub-classes.

Restriction of errors to fine level sub-classes is one significant advantage of hierarchical classification [1], which implies that hierarchical classifiers are more informative than single-level classifiers. Figure 2 illustrates the merit of restriction of error to fine level sub-classes with a hierarchical dog classifier.

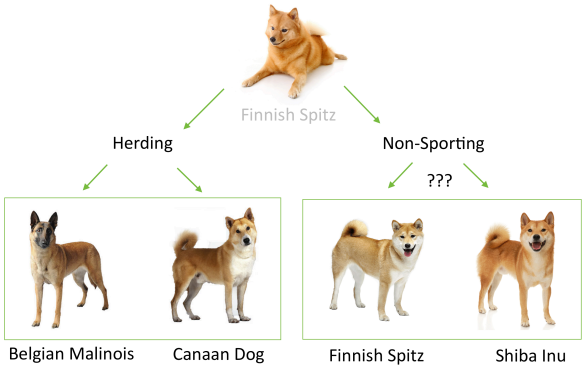


Figure 2. Benefit of Hierarchical classification: restriction of error to fine level sub-classes.

A flat classifier may confuse a Finnish Spitz with a Shiba Inu or a Canaan Dog. On the other hand, for a hierarchical classifier, if the super class non-sporting dog is correctly recognized, although it is still difficult to distinguish Finnish Spitz from Shiba Inu, the classifier will not misclassify a Finnish Spitz as the visually similar Canaan Dog, which is a herding dog.

Consequently, as a research area yet thoroughly explored, CNN based hierarchical classifier has a great potential of improving the performance in fine-grained visual classification tasks, which makes it worthy of research effort and attention.

This paper extends the B-CNN architecture from hierarchical basic-level visual classification to fine-grained visual classification and verifies that hierarchical classification demonstrates empirical benefit for not only basic level, but also fine-grained dataset.

The BT-strategy for training B-CNN models, presented in Zhu and Bain's work, does not prove empirical effectiveness as in basic level visual classification when extended to fine-grained visual classification. A novel training strategy, named the Loop Branch Training strategy (LBT-strategy), is introduced to overcome the issue and boost the performance of the B-CNN models.

The major contributions of this paper are summarized below. Firstly, this paper presents that the B-CNN model introduced by Zhu and Bain could be extended from basic-level visual classification to fine-grained visual classification. Secondly, a novel training strategy for B-CNN, named Loop Branch Training strategy (LBT-strategy), is introduced to boost the classification accuracy when B-CNN model is extended to fine-grained visual classification. Finally, the models and training strategies are validated on the benchmark Leeds Butterfly dataset with experimental results suggesting empirical benefits.

2. Related Work

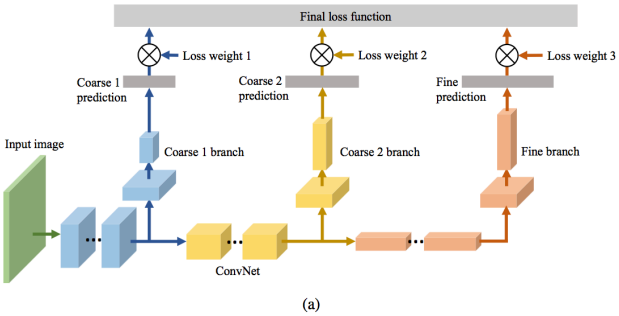
CNN based approaches have been successful in various computer vision problems, with visual classification as the most successful one [9].

Deepening the depth of CNN model within a certain boundary has been shown to boost the accuracy of the CNN classifiers. [13] Many studies make attempts to enhance the components of CNN classifiers by methods such as introducing novel activation functions [14], [15] and custom initialization strategies [16]. The focus is mainly on devising strategies to overcome the potential weakness inside existing CNN classifier models.

Another group of researchers place more emphasis on the fine-grained visual dataset. Identifying the difficulty of acquiring labeled fine-grained data, numerous attempts have been made on augmenting the dataset with external sources such as image search engines [17], [18]. There are also work on discriminative feature mining and attention models that lead the model to focus on discriminative candidate patches [4], [6]. The major emphasis is placed on the data which serves as the input to the CNN model by pre-processing features or data augmentation with external sources.

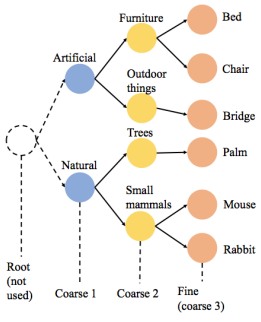
Few studies in the literature explore hierarchical machine learning classifiers. HD-CNN (Hierarchical Deep Convolutional Neural Network) [10] is a model proposed for large scale visual recognition problem in 2015. HD-CNN exploits the hierarchical structure of classes by embedding deep CNNs into a category hierarchy, which essentially requires multiple CNNs to capture the data hierarchy. HINET [19], proposed by Wu and Saito in 2017, is a hierarchical neural network classifier. HINET leverages the data hierarchy by modeling each class as a neuron. However, HINET is not a CNN image classifier.

B-CNN (Branch Convolutional Neural Network) [1], introduced by Zhu and Bain in October 2017, models the hierarchical label tree structure of the data in a single CNN model. Each level in the label tree of the data is captured by one branch in the B-CNN. Each output branch in B-CNN outputs prediction on the corresponding level of classes in the hierarchical label tree of the data. The architecture of a B-CNN is tailored to suit the corresponding hierarchy of the dataset, as shown in Figure 3.



(a)

Figure 3. (a) The Architecture of B-CNN [1].



(b)

Figure 3. (b) A sample hierarchical label tree [1].

Zhu and Bain have shown B-CNN's empirical merits on basic-level visual classification tasks [1], given the boosted accuracy on the classic MNIST, CIFAR-10, and CIFAR-100 datasets.

Different from the abovementioned studies, this paper aims to explore single-CNN-based image classifier with a hierarchical structure that captures the label tree hierarchy of the dataset and to extend such single-CNN-based hierarchical classifier beyond basic-level visual classification to the more challenging fine-grained visual classification.

3. Methodology

3.1. Image pre-processing

Many public image datasets, including the widely studied fine-grained visual classification datasets Leeds Butterfly [2] and Stanford Dogs [20] contain images of variable resolutions, scales, and object sizes, unlike the classic basic-level MNIST and CIFAR datasets. While the images in the dataset come in varying sizes, the standard CNN structure requires input images of a specified fixed size. The mismatch between the nature of the dataset and the CNN potentially impacts the performance of CNN on the desired task. Variance of the images in the dataset, including variance in resolutions, object size, and scale poses challenges to CNN classifiers [21]. The convolutional filters, which are tuned during the training process, deal with these variances to perform the desired task. Ultimately, convolutional filters in CNN could be categorized into two types, the scale-invariant filter which ignores task-irrelevant variances and the scale-variant filter which focuses only on task-relevant features at a specific scale [22]. Gluckman demonstrated that forcing filters to become scale-invariant may lead to the CNN missing image structures relevant to the task [22]. His findings suggest that object recognition benefit from scale-variant information. Furthermore, it is argued that the limit in the image resolutions causes a need for scale-variance [23]. The implication is that the recognition task could be highly different given the varying image resolutions.

Potential approaches to address the abovementioned mismatch between the dataset nature and the CNN fall into two types. The first type is to transform the images into a specified fixed size by means of image preprocessing. The second type requires modifying the architecture of standard CNN into tailored architecture such as the multi-scale CNN proposed by Noord and Postma [21].

As the major focus of the study is to model the data hierarchy within the CNN architecture instead of devising a CNN architecture that handles the variances in the input images, the potential uncertainty introduced by any architectural modification unrelated to the data hierarchy is

undesirable. Therefore, the first type of approaches, namely image preprocessing that transforms images into a specified square size, is adopted in the project.

Noting the influence of variances in image resolution, object size, and scale, different image preprocessing approaches were performed on the Leeds Butterfly dataset in the experiment with an aim to suggest a suitable configuration for the fine-grained visual classification task on the Leeds Butterfly dataset [2].

Potential solutions to transformation of images to the specified square size include cropping the larger images, padding the smaller images, and scaling the images. Minimal information loss and minimal distortion are desirable in the transformation process. Uniformly cropping the larger image to a smaller size may result in loss of discriminative features of the target object as a result of the varying resolutions, object sizes and scales in the images. Padding, which attaches additional pixels to the borders or the image, may also introduce potential bias and increase the variance in relative object sizes and scales especially as the resolutions of images could differ at hundreds to thousands order of magnitude. Naïve padding could undesirably result in the object of interest accounting for a tiny part (a few millionths) in the padded image, as figure 4 illustrates.

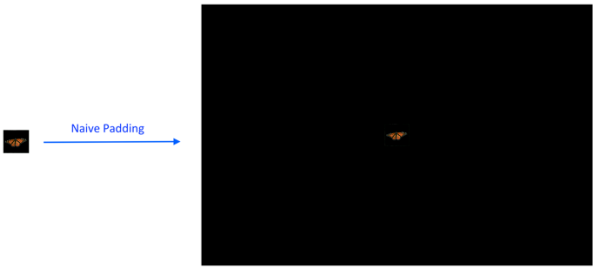


Figure 4. Disadvantage of naïve padding: the object of interest accounts for a tiny portion in the padded image.

At the same time, images of high resolutions also require more memory and longer training time for a CNN classifier model. Therefore, to ensure the space and time efficiency of the CNN model, the resolution of the input images should be controlled within a certain range.

Considering the abovementioned concerns, a combination of scaling and padding is proposed for transforming the images into a desirable fixed square size. To reduce the potential bias introduced in the padding process, the target objects are first segmented from the original image. The segmentation process extracts the target object out from the background with a segmentation mask and returns an image of the target object with black background, as shown in Figure 5.

Figure 5. Segmentation of target object.

After segmentation, scaling and padding are performed to transform the image into a desired square size. As the original images may not be square, directly scaling the image to the specified square size may change the ratio of the object. The common methods assume that the ratio need not be preserved for the CNN to perform visual classification as the CNN is supposed extract the distinctive features from the image. However, given the previous discussion on scale-invariant and scale-variant filters, preserving the object ratio is considered to be potentially advantageous.

Therefore, two different types of scaling are carried out on the dataset and the classification performance of classifiers on the two types of datasets are compared. The first type of scaling scale the segmented image into the specified square size in a brute force manner without preserving the ratio. The second type of scaling is ratio-preserving. The segmented image is first scaled until the longer side matches the specified length, and then black pixels are padded on the two sides of the shorter dimension to make the image square.

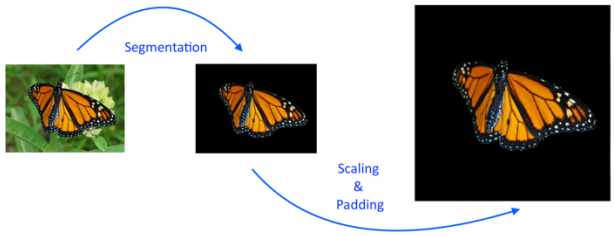


Figure 6. Transformation of images: segmentation, scaling and padding.

The complete transformation process includes segmentation, ratio-preserving scaling and padding, as illustrated in Figure 6.

On the other hand, there is a potential trade-off between image resolutions and training efficiency. Higher resolution images may contain distinctive features with better quality, which potentially results in better classification accuracy. On the other hand, the time and space efficiency of the training may be less as the larger memory is required for storing larger images, the computational time with increased parameters is

also longer. Experiments on datasets with different resolutions are carried out to investigate this trade-off.

3.2. Branch Convolutional Neural Network

The architecture of B-CNN is shown in Figure 3a with a corresponding data label tree in Figure 3b. The structure of B-CNN is tailored for the corresponding label tree of the dataset. The classification task provides the fine-grained level of classes, which are modeled as the leaf nodes in B-CNN. The fine-grained classes are clustered into super-classes in courser levels. The data label tree could be provided in the dataset, manually constructed, or generated with unsupervised learning approaches [1].

Following the terminology in Zhu and Bain's work, this paper uses level to refer to the levels in the data level tree, branch to refer to the output branch network of B-CNN, and layer to refer to the layers in the neural network.

B-CNN is built on top of existing traditional CNN models. B-CNN is any arbitrary multi-layer ConvNet with additional internal output branches. Each of the output branch outputs prediction for a corresponding level of classes in the data label tree. Each output branch contains fully connected (FC) layer and softmax layer to produce class prediction in one-hot representation. Zhu and Bain suggest that branches may consist of ConvNet in addition to fully-connected layers. For simplicity, this paper does not investigate the use of ConvNet in the branches.

Cross-entropy loss is used as the loss of each branch. The cross-entropy loss of the i^{th} sample in the mini-batch on the k^{th} level in the label tree is shown in equation (1).

$$L_{k,i} = -\log\left(\frac{e^{f^k_{y_i}}}{\sum_j e^{f^k_j}}\right) \quad (1)$$

where f_j denotes the j^{th} element in the class scores vector f from the model's last layer output.

A weighted summation of the prediction losses at each level of output is defined to be the loss function of B-CNN, as shown in equation (2) [1].

$$L_i = \sum_{k=1}^{K}\left(-A_k \cdot \log\left(\frac{e^{f^k_{y_i}}}{\sum_j e^{f^k_j}}\right)\right) \quad (2)$$

K denotes the number of levels in the label tree, and A_k denotes the loss weight of the kth level. The loss weight A_k determines how much influence a level has to the final loss weight L_i. Despite that the contribution of each branch is determined by the relative loss weight, a standard representation requires the loss weights to satisfy the conditions defined in (3) and (4) [1].

$$A_k \in [0,1] \quad (3)$$

$$\sum_{k=1}^{K}(A_k) = 1 \quad (4)$$

The implication of loss weight is that B-CNN is a super model of traditional CNN. For example, a B-CNN corresponding to a three-level data label tree has three branches of outputs. When the loss weight is set to [0,0,1], the B-CNN converges to a traditional CNN [1].

3.3. Loop Branch Training Strategy

The loss weight distribution determines the training focus of the B-CNN model by tuning the influence of each output branch. For instance, a weight distribution [0.98, 0.01, 0.01] of a three-branch network makes the B-CNN model focuses more on the coarse classes, or the low-level feature extractions and pays little attention to the finer classes.

The Branch Training strategy (BT-strategy) [1] proposed by Zhu and Bain exploits the loss weight distribution to change the focus, or the greatest loss weight, of training during the training process of the B-CNN model by modifying the loss weights. The setting of a focus guides the B-CNN model to pay more effort learning that particular level.

BT-strategy shifts the focus of loss weight distribution from coarse level to fine level once throughout the training process. The rationale behind BT-strategy is to make the classifier extract lower level features first and then fine tune parameters on higher level features. One benefit of the BT-strategy, Zhu and Bain suggested, is that the vanishing gradient problem could be prevented.

When the B-CNN model is extended to fine-grained dataset, more training epochs may be required and as the focus shifts from coarse to fine levels, the coarse level accuracy may go down in later part of the training as the loss of the coarse level plays a less influential role in the training process. As a result, the hierarchical prior used for guiding the classification is prone to degradation. Furthermore, the vanishing gradient problem may

re-emerge when the number of epochs trained with a heavy focus on the fine-grained level is large.

This paper proposes an alternative training strategy – the Loop Branch Training strategy (LBT-strategy) to tackle the issue discussed above. LBT-strategy also shifts the focus from course to fine level during the training process. However, instead of a single round of shifting the focus from coarse to fine, such shifting process is performed in a looping manner, namely that multiple cycles of shifting the focus is carried out during the training process. In LBT-strategy, each of the cycle of shifting the training focus is shorter in number of epochs compared to its counterpart in BT-strategy.

By looping LBT-strategy and shortening the cycle of shifting the training focus from coarse to fine, the prediction accuracy of the coarse level is maintained over the training process. Consequently, degradation of the hierarchical prior and re-emergence of vanishing gradient problem are prevented.

4. Experiments

4.1. Overview

The three major purposes of the experiments are described below. Firstly, the experiments aim to verify whether B-CNN could outperform traditional CNN when the visual classification task is extended from basic level visual classification to fine-grained visual classification. Secondly, the experiments investigate and compare the performance of models that adopt BT-strategy to models that adopt LBT-strategy on fine-grained visual classification. Lastly, the experiments compare the influence of different image pre-processing methods, namely non-ratio-preserving scaling and ratio-preserving scaling (as discussed in 3.1), and different image sizes on the models' performance.

Stochastic gradient descent (SGD) with a momentum at 0.9 is used as the optimizer in all the experiments. ReLU (rectified linear unit) is used in the activation layers in all models. The trainings of all models are performed on the 8-core Intel(R) Xeon(R) CPU on FloydHub (www.floydhub.com). The number of training epochs are limited to no more than 180 epochs. Given the purposes discussed above, the focus of the experiments is mainly on the comparison between the performance of B-CNN models with different training strategies and the performance of the baseline CNN models, rather than outperforming the state-of-the-art CNN models on the Leeds Butterfly dataset. Therefore, simple image preprocessing methods are used in the dataset and all models are trained without weight initialization from pre-trained models, data augmentation, or adoption of ensemble methods. The experiments are minimal yet sufficient to demonstrate the ideas of the paper.

4.2. Branch CNN architecture

A B-CNN consists of a baseline CNN and several branch networks. A 3-level label tree is used for the Leeds Butterfly dataset in the experiment. Therefore, a 3-branch B-CNN architecture is constructed. The architecture of the baseline CNN and the branch networks are presented below. Colored images are used in all experiments, therefore the input ore of shape (*size* x *size* x 3), where the 3 denotes the 3 color channels. The star symbol * and ** denote the junction of connections between the baseline CNN and the branch networks.

The baseline CNN architecture is similar to that of the VGG16 proposed by Simonyan and Zisserman [13], as shown in Table 1 below.

Table 1. Baseline CNN architecture. size in the input denote the size of the square images. x in the last FC layer denotes the number of fine-grained target classes.

Baseline CNN	
Layer	Component
Input (*size* x *size* x 3)	Input
(conv3-64) $_{x2}$	Block 1
maxpool-2	
(conv3-128) $_{x2}$	Block 2
maxpool-2 *	
(conv3-256) $_{x3}$	Block 3
maxpool-2 **	
(conv3-512) $_{x3}$	Block 4
maxpool-2	
Flatten	Fine-grained Branch
FC-4096	
FC-4096	
FC-*x*	
Softmax	Softmax

The baseline CNN uses 3 x 3 patch size filters with a stride of 1 pixel. The activation layers are ReLU (rectified linear unit) layers. Max-pooling are performed using a 2 x 2 window with a stride of 2. Batch normalization is used to accelerate the training process [24]. In order to prevent overfitting, dropout [25] regularization is adopted between fully-connected (FC) layers.

Table 2. Coarse branch network architecture.

Coarse Level Branches	
*Flatten	

FC-256	Coarse Branch 1
FC-256	
FC_{CB1}	
**Flatten	Coarse Branch 2
FC-512	
FC-512	
FC_{CB2}	

For a 3-branch B-CNN, there are two coarse level branches in addition to the fine-grained branch. The configurations of the coarse level output branches are illustrated in Table 2.

4.3. Loss Weight Modifier

The BT-strategy and the LBT-strategy described in 3.3. are implemented by the introduction of loss weight modifiers. The loss weight modifier modifies the loss weight A_k of each output branch at certain epochs during the training process. Two loss weight modifiers BT-modifier and LBT-modifier are implemented for BT-strategy and LBT-strategy, respectively.

The loss weight distribution for the 3-branch B-CNN is denoted as $[A_1, A_3, A_3]$ where A_1, A_2, A_3 refers to the loss weight of the coarse branch 1, the coarse branch 2, and the fine-grained branch, respectively. In the 3-branch B-CNN, the loss weights are first initialized to [0.98, 0.01, 0.01] for both modifiers. Table 3 shows the loss weight modification of LBT-modifier and BT modifier for trainings of 120 and 180 epochs.

Table 3. Loss weight modifications with LBT-modifier and BT-modifier on trainings of 120 and 180 epochs.

	LBT-modifier	BT-modifier
Epoch #	Loss weight distribution	
1	[0.98, 0.01, 0.01]	
8	[0.1, 0.8, 0.1]	
18	[0.1, 0.2, 0.7]	
28	[0, 0, 1]	
58	[0.98, 0.01, 0.01]	[0, 0, 1]
68	[0.1, 0.2, 0.7]	[0, 0, 1]
78	[0, 0, 1]	[0, 0, 1]

The LBT-modifier performs the focus shifting cycle twice throughout the training while the BT-modifier conducts the focus shifting cycle only once. It is worth noting that with the BT-modifier, the training of the B-CNN model converges to traditional CNN training since epoch 28, which is an early epoch for a training process with 120 or 180 epochs. The loss weight distribution remains to be [0, 0, 1] for a long period (from epoch 28 to the last epoch). As a result, the model is prone to degradation of the hierarchical prior and re-emergence of vanishing gradient problem, as discussed in 3.3. On the other hand, the LBT-modifier loops back the cycle of shifting loss weight distribution one more time from epoch 58 to 78, which prevents the two potential drawbacks.

4.4. Leeds Butterfly Data Label Tree

The hierarchical fine-grained visual classification experiments are carried out on the fine-grained dataset Leeds Butterfly [2]. The Leeds Butterfly dataset is a 10-class dataset without official data label tree which depicts the data hierarchy. As the B-CNN architecture is tailored for the data label tree, the construction of a data label tree for the dataset is necessary. Instead of supervised learning approach, manual construction is used in the experiment for lower complexity and lower likelihood of uncertainty in the tree quality as fine-grained labeling often requires domain-specific knowledge [7].

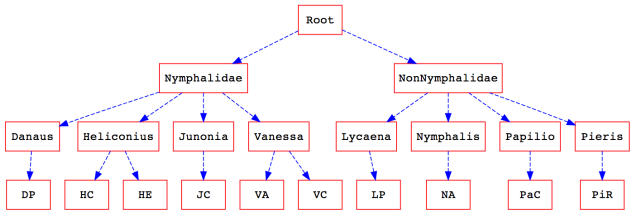


Figure 7. Manually constructed label tree for the Leeds Butterfly dataset. The root is not used as a level of label.

The construction of the label tree for the Leeds Butterfly dataset is based on taxonomic rank of the butterfly species. The root is not counted as a level in the label tree. The first coarse level corresponds to the family of the butterfly, which contains 2 classes, Nymphalidae and non-Nymphalidae. The second coarse level corresponds to the genus of the butterfly, containing 8 classes. The fine-grained level is the target level corresponding to the butterfly species, containing 10 target classes. The manually constructed label tree for the Leeds Butterfly dataset is shown in Figure 7.

4.5. Image Pre-processing

Two image processing approaches discussed in 3.1., namely segmentation followed by ratio-preserving scaling with padding and non-ratio-preserving scaling, are carried out on the Leeds Butterfly dataset.

Images of different resolutions, or different sizes, are also experimented to investigate the trade-

off between classification performance and time-space efficiency. The sizes tested are (128 x 128 x 3) and (84 x 84 x 3). Larger image sizes are not tested due to the concern of space and time efficiency.

5. Results and Discussion

5.1. B-CNN on Fine-grained Classification

Figure 8 shows the test accuracy of baseline CNN, B-CNN with BT-strategy, and B-CNN with LBT-strategy on the dataset with ratio-preserved images of size (128 x 128) over 120 training epochs. The red, blue, green lines correspond to the baseline CNN, B-CNN with BT-strategy, and B-CNN with LBT-strategy, respectively. It is clear from the figure that the B-CNN models are trained much slower than the baseline CNN at the beginning (the baseline CNN reaches a test accuracy greater than 60% before epoch 50 while the test accuracies of the B-CNNs are still below 20%). However, the B-CNNs speed up significantly at later epochs. A training process of 120 seems to result in immaturely trained B-CNNs, as the test accuracies of both B-CNNs still show an increasing trend at the end of the 120-th epoch. Therefore, although the baseline CNN outperforms B-CNNs at epoch 120, this result is not very conclusive.

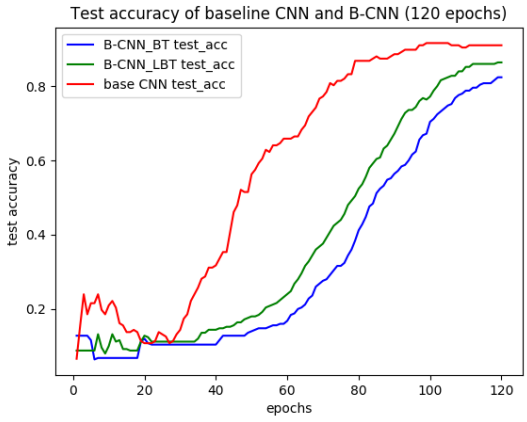


Figure 8. Test accuracy of baseline CNN, B-CNN with BT-strategy, and B-CNN with LBT-strategy on the dataset with ratio-preserved images of size (128 x 128) over 120 epochs.

Given the observation of immaturely trained B-CNNs, the number of training epochs is increased from 120 to 180. The test accuracy of baseline CNN, B-CNN with BT-strategy, and B-CNN with LBT-strategy on the dataset with ratio-preserved images of size (128 x 128) over 180 training epochs is shown in Figure 9. From Figure 9., it is seen that the B-CNN with LBT-strategy outperforms the baseline CNN after epoch 120, and achieves the higher test accuracy among all three models at the end of the 180-th epoch. However, the B-CNN with BT-strategy results in the poorest test accuracy at the end of the training. This is likely due to the disadvantage of a degradation in the hierarchical prior of BT-strategy in long training process.

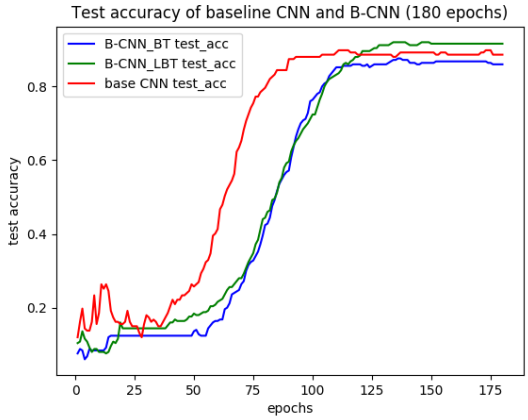


Figure 9. Test accuracy of baseline CNN, B-CNN with BT-strategy, and B-CNN with LBT-strategy on the dataset with ratio-preserved images of size (128 x 128) over 180 epochs.

Despite the inferior performance of B-CNN with BT-strategy, the experimental results suggest that B-CNN with LBT-strategy outperforms the baseline CNN. Therefore, it is concluded that B-CNN outperforms traditional CNN on fine-grained visual classification tasks.

The performance of the models trained on the dataset with ratio-preserved images of size (128 x 128) over 180 training epochs are summarized in Table 4, where the state-of-the-art performance from literature is also listed for comparison.

Table 4. Performance comparison between the tree models and state-of-the-art from the literature.

Approach	Accuracy (%)
Baseline CNN	88.62
B-CNN with BT-strategy	86.00
B-CNN with LBT-strategy	**91.59**
Goel & Banerjee CNN [26]	91.20
Mattos & Feis Feature Descriptors [27]	87.95
Goel & Banerjee Hierarchical Metric Learning [26]	94.60

Although the datasets and models in the experiments are minimal where the models are trained without weight initialization from pre-trained models or adoption of ensemble methods and the datasets are preprocessed with simple means without data augmentation, the performance

of the B-CNN model with LBT-strategy trained on ratio-preserved images of size (128 x 128) for 180 training epochs achieves a test accuracy of 91.59 %, which is not substandard and should be sufficient to demonstrate the idea of the paper.

5.2. LBT-strategy

To analyze the influence of the BT-strategy and LBT-strategy over the performance of the CNN, closer investigation on the training of B-CNN on the dataset with ratio-preserved images of size (128 x 128) over 180 training epochs is carried out.

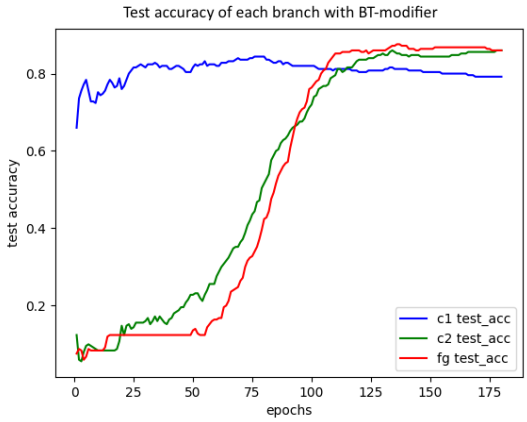


Figure 10. Test accuracy of each branch for B-CNN with BT-modifier on the dataset with ratio-preserved images of size (128 x 128).

Figure 10. shows the test accuracy of each output branch of the B-CNN with BT-modifier. Figure 11. shows the test accuracy of each output branch of the B-CNN with LBT-modifier. In both of the figures, the blue, green, red lines correspond to the test accuracy of the coarse branch 1, coarse branch 2, and the fine-grained branch, respectively.

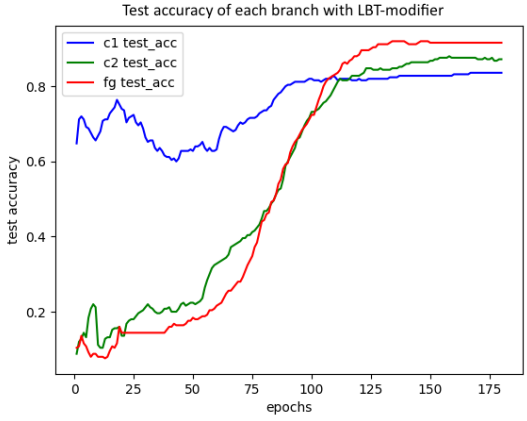


Figure 11. Test accuracy of each branch for B-CNN with LBT-modifier on the dataset with ratio-preserved images of size (128 x 128).

It could be seen from the blue line in Figure 10. that the test accuracy of the coarse branch 1 degrades in later stage of the training. Since the focus of the training is placed entirely on the fine-grained branch since epoch 28 in the BT-strategy, the accuracy for the coarse level decreases and causes a degradation of the hierarchical prior used to guide the classification. This reflects the problem of degradation of hierarchical prior discussed in 3.3.

On the contrary, the blue line in Figure 11. shows that the prediction accuracy for coarse branch 1 in B-CNN with LBT-strategy is maintained and improved in the later training process. Essentially, none of the test accuracies for all three branches decreases even in later part of the training. The implication is that LBT-strategy helps sustain a good hierarchical prior for guiding the classification and prevent the degradation of hierarchical prior problem seen in BT-strategy.

5.2. Image Ratio and Size

Figure 12. shows the performance of B-CNN models with different training strategies trained for 120 epochs on datasets of ratio-preserved and non-ratio-preserved images of size (128 x 128). Non-ratio-preserving scaling is shown to result in better performance of the B-CNN model.

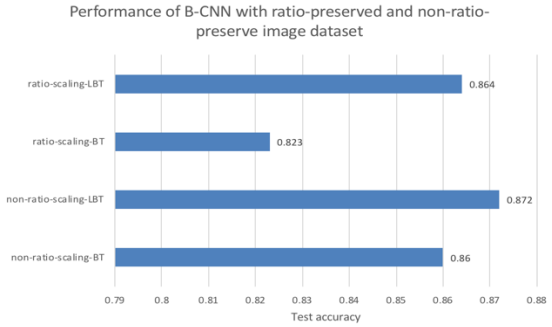


Figure 12. Performance of B-CNN models with different training strategies trained for 120 epochs on datasets of ratio-preserved and non-ratio-preserved images of size (128 x 128).

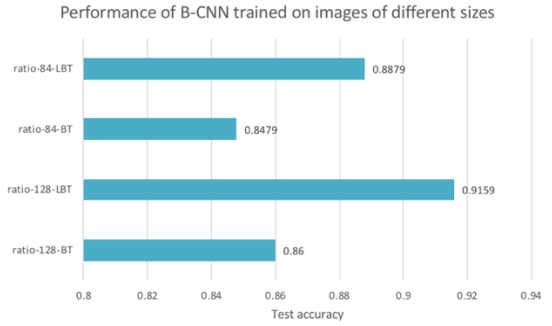


Figure 13. Performance of B-CNN models with different training strategies trained for 180 epochs on datasets of ratio-preserved images of size (128 x 128) and (84 x 84).

Figure 13. illustrates the Performance of B-CNNs with different training strategies trained for 180 epochs on datasets of ratio-preserved images of size (128 x 128) and (84 x 84). Images of size (128 x 128) result in better performance of the B-CNN than images of size (84 x 84).

Figure 14. shows the time and space efficiency of the B-CNN models trained on ratio-preserved images of size (128 x 128) and size (84 x 84) for 180 epochs in terms of number of epochs per millisecond and RAM usage.

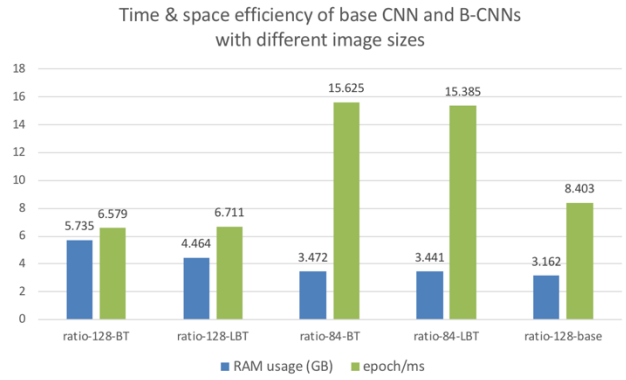


Figure 14. Time and space efficiency of CNN and B-CNN models trained on images.

Table 5. Time and space efficiency of CNN and B-CNN models in absolute and relative terms.

Model	sec / epoch	RAM usage (GB)	time efficiency (ratio to base)	space efficiency (ratio to base)
128-base	119	3.162	1	1
128-BT	152	5.735	0.783	0.551
128-LBT	149	4.464	0.800	0.708
84-BT	64	3.472	1.859	0.911
84-LBT	65	3.441	1.831	0.919

Table 5. shows the time and space efficiency of CNN and B-CNN models trained on ratio-preserved images of size (128 x 128) and size (84 x 84) for 180 epochs in absolute and relative terms.

In Figure 14, The B-CNN models trained on larger image uses more memory and takes more time for the training each epoch (trains more epoch in every millisecond). The trade-off between the time and space efficiency is clearly shown. However, mages of size (84 x 84) is too low in resolutions such that the classification accuracy is compromised. With 91.59 % test accuracy, 80.0 % time efficiency and 70.8 % space efficiency relative to the baseline, the B-CNN models with LBT-strategy trained on images of size (128 x 128) leads to good performance in terms of classification accuracy, time efficiency, and space efficiency.

5.3. Summary of Results

The conclusions drawn from the experimental results are summarized below. Firstly, the experiments verify that B-CNN could outperform traditional CNN on fine-grained visual classification tasks. Secondly, LBT-strategy are shown to accelerate the learning process and boost the performance of the model compared to the BT-strategy. Thirdly, the experimental results suggest that ratio-preserving scaling combined with padding does not have empirical benefit in comparison with non-ratio-preserving brute-force scaling for the Leeds Butterfly dataset. Images of size (128 x 128) is preferred to those of size (84 x 84). Finally, the B-CNN model with LBT-strategy shows acceptable cost-effectiveness in terms of time and space efficiency and the classification accuracy.

6. Conclusion

This paper verifies the B-CNN (Branch Convolutional Neural Network) proposed by Zhu and Bain [1] could be successfully extended from basic level visual classification to fine-grained visual classification and outperforms traditional CNN. A novel training strategy, the LBT-strategy (Loop Branch Training strategy) is proposed as an alternative for training B-CNN in longer training process. The LBT-strategy demonstrates empirical merits in the accelerating the training and boosting the performance by effectively preventing the degradation of hierarchical prior, which is a potential weakness of the BT-strategy. The paper also investigates the impacts of ratio-preserving scaling and non-ratio-preserving scaling on the performance of the B-CNNs. Experimental results suggest that non-ratio-preserving scaling is not inferior. Further studies on alternative training strategies for B-CNN and automatic construction of the data label tree may be investigated.

Acknowledgement

Sincere gratitude is expressed to the project supervisor Professor Edmund Y. Lam for his insightful advice and invaluable guidance.

References

[1] X. Zhu, M. B. preprint arXiv:1709.09890, and undefined 2017, "B-CNN: Branch Convolutional Neural Network for Hierarchical Classification," *arxiv.org*.

[2] J. Wang, K. Markert, and M. Everingham, "Learning models for object recognition from natural language descriptions," 2009.

[3] K. Johnson, A. E.-C. Development, and undefined 1998, "Effects of knowledge and development on subordinate level categorization," *Elsevier*.

[4] B. Yao, A. Khosla, L. F.-F.-C. V. and Pattern, and undefined 2011, "Combining randomization and discrimination for fine-grained image categorization," *ieeexplore.ieee.org*.

[5] P. Welinder, S. Branson, T. Mita, C. Wah, and F. Schroff, "Caltech-UCSD birds 200," 2010.

[6] T. Xiao, Y. Xu, K. Yang, … J. Z.-C. V. and, and undefined 2015, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," *openaccess.thecvf.com*.

[7] S. Maji, E. Rahtu, J. Kannala, … M. B. preprint arXiv, and undefined 2013, "Fine-grained visual classification of aircraft," *arxiv.org*.

[8] T. Berg, J. Liu, S. Lee, … M. A.-… V. and P., and undefined 2014, "Birdsnap: Large-scale fine-grained visual categorization of birds," *openaccess.thecvf.com*.

[9] A. Krizhevsky, I. Sutskever, G. H.-A. in neural, and undefined 2012, "Imagenet classification with deep convolutional neural networks," *papers.nips.cc*.

[10] Z. Yan, H. Zhang, … R. P.-P. of the, and undefined 2015, "HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition," *cv-foundation.org*.

[11] R. Babbar, I. Partalas, … E. G.-A. in N., and undefined 2013, "On flat versus hierarchical classification in large-scale taxonomies," *papers.nips.cc*.

[12] M. Zeiler, R. F.-E. conference on computer vision, and undefined 2014, "Visualizing and understanding convolutional networks," *Springer*.

[13] K. Simonyan, A. Z. preprint arXiv:1409.1556, and undefined 2014, "Very deep convolutional networks for large-scale image recognition," *arxiv.org*.

[14] D. Clevert, T. Unterthiner, S. H. preprint arXiv:1511.07289, and undefined 2015, "Fast and accurate deep network learning by exponential linear units (elus)," *arxiv.org*.

[15] I. Goodfellow, D. Warde-Farley, … M. M. preprint arXiv, and undefined 2013, "Maxout networks," *jmlr.org*.

[16] D. Mishkin and J. Matas, "All you need is a good init," Nov. 2015.

[17] S. Xie, T. Yang, X. Wang, Y. L.-CVPR2015, and undefined 2014, "Hyper-class augmented and regularized deep learning for fine-grained image classification," *cv-foundation.org*.

[18] J. Krause, B. Sapp, A. Howard, H. Zhou, A. T.-… on C. Vision, and undefined 2016, "The unreasonable effectiveness of noisy data for fine-grained recognition," *Springer*.

[19] Z. Wu, S. S. preprint arXiv:1705.11105, and undefined 2017, "HiNet: Hierarchical Classification with Neural Network," *arxiv.org*.

[20] A. Khosla, … N. J.-P. C. W., and undefined 2011, "Novel dataset for fine-grained image categorization: Stanford dogs," *people.csail.mit.edu*.

[21] N. van Noord, E. P.-P. Recognition, and undefined 2017, "Learning scale-variant and scale-invariant features for deep image classification," *Elsevier*.

[22] J. G.-C. V. and P. Recognition, undefined 2006, and undefined 2006, "Scale variant image pyramids," *ieeexplore.ieee.org*.

[23] D. Park, D. Ramanan, C. F.-E. conference on computer, and undefined 2010, "Multiresolution models for object detection," *Springer*.

[24] S. Ioffe, C. S. preprint arXiv:1502.03167, and undefined 2015, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arxiv.org*.

[25] G. Hinton, N. Srivastava, … A. K. preprint arXiv, and undefined 2012, "Improving neural networks by preventing co-adaptation of feature detectors," *arxiv.org*.

[26] A. Goel, B. B. preprint arXiv:1708.01494, and undefined 2017, "Hierarchical Metric Learning for Fine Grained Image Classification," *arxiv.org*.

[27] A. Mottos, R. F.-I. P. (ICIP), 2014 IEEE, and undefined 2014, "Fusing well-crafted feature descriptors for efficient fine-grained classification," *ieeexplore.ieee.org*.