Deep Learning for Practical Image Recognition: Case Study on Kaggle Competitions

Xulei Yang*
Institute for Infocomm Research
yang_xulei@i2r.a-star.edu.sg

Zeng Zeng
Institute for Infocomm Research
zengz@i2r.a-star.edu.sg

Sin G. Teo
Institute for Infocomm Research
teosg@i2r.a-star.edu.sg

Li Wang
Institute for Infocomm Research
wang_li@i2r.a-star.edu.sg

Vijay Chandrasekhar
Institute for Infocomm Research
vijay@i2r.a-star.edu.sg

Steven Hoi
Singapore Management University
chhoi@smu.edu.sg

ABSTRACT

In past years, deep convolutional neural networks (DCNN) have achieved big successes in image classification and object detection, as demonstrated on ImageNet in academic field. However, There are some unique practical challenges remain for real-world image recognition applications, e.g., small size of the objects, imbalanced data distributions, limited labeled data samples, etc. In this work, we are making efforts to deal with these challenges through a computational framework by incorporating latest developments in deep learning. In terms of two-stage detection scheme, pseudo labeling, data augmentation, cross-validation and ensemble learning, the proposed framework aims to achieve better performances for practical image recognition applications as compared to using standard deep learning methods. The proposed framework has recently been deployed as the key kernel for several image recognition competitions organized by Kaggle. The performance is promising as our final private scores were ranked 4 out of 2293 teams for fish recognition on the challenge "The Nature Conservancy Fisheries Monitoring" and 3 out of 834 teams for cervix recognition on the challenge "Intel & MobileODT Cervical Cancer Screening", and several others. We believe that by sharing the solutions, we can further promote the applications of deep learning techniques.

KEYWORDS

Image recognition, deep learning, objection detection, and image classification

ACM Reference Format:
Xulei Yang*, Zeng Zeng, Sin G. Teo, Li Wang, Vijay Chandrasekhar, and Steven Hoi. 2018. Deep Learning for Practical Image Recognition: Case Study on Kaggle Competitions. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3219819.3219907

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '18, August 19–23, 2018, London, United Kingdom
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3219907

1 INTRODUCTION

Deep Neural Networks (DNNs) have emerged as powerful machine learning models that exhibit major differences from traditional approaches for image classification [13, 23]. DNNs with deep architectures have the capacity to learn complex models and allow for learning powerful object representations without the need to handle designed features. This has been empirically demonstrated on the ImageNet classification task across thousands of classes.

Compared to image classification, object detection is a more challenging task that requires more sophisticated methods to solve. In this context, we focus not only on classifying images, but also on estimating the classes and locations of objects within the images precisely. Object detection is one of the hardest problems in computer vision and data engineering. Owing to the improvements in object representations and machine learning methods, many major advances in object detection have been achieved, like Faster R-CNN [20], which has achieved excellent object detection accuracy by using DNNs to classify object proposals.

In real world applications, it may be required to classify a given image based on the object(s) contained within that image. For example, we want to classify an image into different categories based on the fish types within the image, as shown in Fig.1. There are some unique practical challenges for this kinds of image recognition problems: Firstly, the objects are very small as compared to the background. Standard CNN based methods like ResNet [7] and Faster R-CNN [20] may learn the feature of the boats (background) but not the fishes (objects). Therefore, it will fail when presented with images containing new boats. Secondly, imbalanced data sets exist widely in real world and they have been providing great challenges for classification tasks. As a result, the CNN models might be biased towards majority classes with large training samples, such that might have trouble classifying those classes with very few training samples. Thirdly, in real-world applications, getting data is expensive that involves time-consuming and labour-intensive process, e.g., ground truth have to be labeled and confirmed by multiple experts in the domain. How to achieve good performance with very limited training dataset remains a big challenge in both academic and industry.

In this paper, we address the aforementioned challenges by presenting a computational framework based-on the latest developments in deep learning. Firstly, we propose two-stage detection scheme to handle small object recognition. The framework combines the advantages of both object detection and image classification methods. We first use state-of-the-art object detection method

Figure 1: Example of image recognition based on the object types within the image. The object in this example is the fish indicated by red rectangle.

like Faster R-CNN [20] to identify the location of the object within the full image. Based on this localization, a small Region of Interest (ROI) is cropped from the image. Finally, deep convolutional networks, like ResNet [7], are employed to perform classification on the cropped image. Secondly, Data augmentation and cross validation are used to deal with imbalanced dataset problem. We oversample the rare samples by horizontal flipping, slight shifting and rotation, as well as adding color variance. We also use stratified K-fold to split the training set into different folds for cross validation so that the model is able to "see" enough of the rare classes for each fold. In such a way, the negative effect of data imbalance is reduced to a minimum. Lastly, we use the pseudo labeling approach to utilize the unlabeled data in the testing dataset. We first train a model based on the training dataset. The model is then used to perform prediction on the testing dataset. After that, the most confident samples from the testing dataset are added into the training dataset. We again train a new model with larger training dataset. The scheme can be iteratively processed to generate more training samples.

The main contributions of this paper are three folds: First, we propose a deep learning framework to handle several unique challenges for practical image recognition applications, e.g., small size of objects, imbalanced data distributions, and limited labeled training samples. Second, the proposed framework have been deployed for several image recognition competitions organized by Kaggle. The performance is promising as our final scores are ranked top 1% in the private leaderboard for all the competitions. Lastly, we publicly share the source codes of the implementation of our case studies for fish recognition on the Kaggle challenge "The Nature Conservancy Fisheries Monitoring" [3], as well as cervix recognition on the Kaggle challenge "Intel & MobileODT Cervical Cancer Screening" [2]. The interested readers may re-use the source codes for their own image recognition applications. The rest of this paper is organized as follows: Section 2 reviews related works of object detection and image classification methods. Our proposed framework for image recognition is presented in Section 3. Case studies of our proposed framework on two Kaggle image recognition tasks are illustrated in Section 4. Lastly, the conclusion is given in Section 5.

2 RELATED WORK

2.1 Image Classification Methods

In recent years, deep convolutional neural network architectures (DCNN), specifically, VGG [24], Inception [26], and ResNet [7] have been widely used for image classification tasks and seen great success in many computer vision challenges like ImageNet. More recently published deep neural networks include DenseNet [9] and Dual Path Networks [1]. The interested readers may refer to [17] for a comprehensive review of DCNN for image classification.

VGG. VGG network architecture is introduced by Simonyan and Zisserman [24]. This network is characterized by its simplicity, using only 3×3 convolutional layers stacked on top of each other in increasing depth. The reduction of volume size is handled by max pooling. Two fully-connected layers, each with 4, 096 nodes are then followed by a softmax classifier. Due to its depth and number of fully-connected nodes, VGG is over 533 MB for VGG16 and 574 MB for VGG19. This makes deploying VGG a tiresome task. However, VGG is still popularly used in many deep learning image classification problems due to its simplicity.

Inception. The "Inception" micro-architecture was first introduced by Szegedy et al. [26]. The goal of the inception module is to act as a "multi-level feature extractor" by computing 1×1, 3×3, and 5×5 convolutions within the same module of the network. The outputs of these filters are then stacked along the channel dimension before being fed into the next layer in the network. The origin of this architecture is titled as GoogLeNet, but after subsequent improvements, it is simply called Inception VN where N refers to the version number put out by Google, such as Inception V3 [28] and Inception V4 [25].

ResNet. First introduced by He et. al. [7], the ResNet architecture has become a seminal work, demonstrating that extremely deep networks can be trained using standard stochastic gradient descent (SGD) and a reasonable initialization function, through the use of residual modules. Further accuracy improvement can be obtained by updating the residual module to use identity mappings, as demonstrated in their follow-up publication [8]. Even though ResNet is much deeper than VGG, the model size is actually substantially smaller due to the usage of global average pooling rather than fully-connected layers: this reduces the model size to 102 MB for ResNet50.

2.2 Object Detection Methods

Significant progress has been achieved in recent years on object detection due to the development of convolutional neural networks (CNNs). Among them, Faster R-CNN [20], YOLO (You Only Look Once) [18], and SSD (Single Shot Multibox Detector) [15] are the most well-known. Other modern convolutional object detectors include R-FCN (region-based fully convolutional networks) [4] and multibox [27]. The interested readers may refer to [10] for the speed/accuracy trade-offs for various object detectors.

Faster R-CNN. Faster R-CNN is an improved version of Regional CNN (R-CNN) [22] and its descendant Fast R-CNN [21]. The basic idea is to break down object detection into two separate stages. In the first stage, regions within the image that are likely to contain

ROIs are identified. In the second stage, a convolutional neural network runs on each proposed region, and outputs the object category score and the corresponding bounding box coordinates that may contain objects. Faster R-CNN is the state-of-the-art model with the best mAP scores on the VOC and COCO benchmark datasets. However, with a framerate of 7 fps, Faster R-CNN is the slowest amongst other state-of-the-art models such as YOLO and SSD. The latest descendant in this family is called Mask R-CNN [12], which extends such object detection techniques to provide pixel level segmentation.

YOLO. It first partitions the raw input image into $N \times N$ squared regions. Then it fits a convolutional neural network directly on the input image and output M sets of object confidence scores and bounding box coordinates, with M depending on N. The entire model is trained end-to-end. Since YOLO takes raw image as the input and output confidence scores and bounding box coordinates directly with a single pass, it is considerably faster than Faster R-CNN, which requires multiple stages for training and inference. However, it is significantly less accurate than Faster R-CNN, and is particularly poor at localizing small objects. YOLO V2 [19] has made several small but important changes inspired by Faster R-CNN to improve the detection accuracy.

SSD. It is another state-of-the-art object detection model that is known to have good trade-off between speed and accuracy. Similar to YOLO, it requires only a single step for training and inference, and the entire model is trained in end-to-end style. The major contribution of SSD as compared to other models is that it makes use of feature maps of different scales to make predictions. In contrast, Faster R-CNN and YOLO base their predictions only on a single set of feature maps located at some selected intermediate level (e.g., "conv5") down the convolutional layers hierarchy. Theoretically speaking, using feature maps of varying scales can lead to better detection quality of objects with different sizes. Similar to YOLO, SSD is also suffering from small object detection problem. Feature-Fused SSD [6] is one of the latest research work to improve the performance of SSD on small object localization.

3 THE PROPOSED FRAMEWORK

Many real world applications, e.g., the two Kaggle image recognition competitions studied in Section 4, involve in the task of classifying a given image based on the objects contained within that image. The deep learning methods discussed in the last Section, either object detection or image classification, can be directly applied for this task. Their performance, however, may not meet the desired requirements, due to some unique challenges related to practical image recognition as discussed in Section 1, i.e., small size of the objects, imbalanced data distributions, limited labeled data samples. In this section, we propose a computational framework, which consists of two-stage detection scheme, pseudo labeling, data augmentation, cross-validation and ensemble learning, to tackle the above challenging issues.

Overview. The overall workflow of the proposed framework is illustrated in Fig. 2, which consists of two major successive tasks, i.e., object localization and ROI classification, followed by a stacking process to ensemble the results from various models. Fig. 3 shows one of the examples that uses our proposed framework to perform fish recognition from a fishing boat image. The detailed diagram of the framework is shown in Fig. 4. Fig. 4-a shows the diagram for object localization task, where various object detection models are trained on original input images to detect the objects within the image. Each model outputs one bounding box with maximum probability value. Simple average or major voting schemes are then used to select and crop the overlapped region as the final ROI image. In the case when the object detection models cannot find any object within the image, the image is labeled using the classification models directly trained on full images. Similarly, Fig. 4-b displays the diagram for ROI classification task, where various image classification models are trained on cropped ROI images from object localization. K-fold cross validation is applied here to train each single model, and the probability vector of the training set and testing set from each model is then feed into the stacking block for level-2 ensemble. Coming to the stacking process in Fig. 4-c, we are using hill-climbing or simple averaging methods to determine the linear combination weights, then apply the weights to the testing results to get the final image recognition result.

Two-stage Detection Scheme for Small Objects. Real-world image recognition applications, like the case demonstrated in Fig. 3, may involve small object recognition in large background. On one hand, image classification based methods like ResNet [7] will take the whole image as the input and extract different layers of features to classify the image. However, this method will learn the feature of the boats but not the fishes. Therefore, it will fail when presented with images containing new boats. On the other hand, object detection methods like Faster R-CNN [20] will locate and classify the fishes within the image. However, this method may mis-classify the fish types due to the small size of the objects in low resolution images.

In our proposed computational framework, as shown in Fig. 2 and Fig. 4, we deploy a two-stage detection scheme to cater for small object recognition. The scheme combines the advantages of both object detection methods (Fig. 4-a) and image classification methods (Fig. 4-b). The basic idea is to develop an automated ways to detect and crop out the objects of interest from the images, then apply image augmentation to the cropped regions, and lastly perform image classification on the augmented ROIs. In such a way, the proposed framework might more focus on the object itself rather than the background, such that perform much better than standard one-stage deep learning approach on small object recognition within a given large image.

Data Augmentation & Stratified K-fold for Imbalanced Data. Many real world image recognition datasets are imbalanced which they pose a great challenge to machine learning especially in a classification task. Both Kaggle image recognition competitions studied in Section 4 provide the imbalanced datasets. For example, in the nature conservancy fisheries monitoring, the training dataset provides 1719 ALB fish but only 67 LAG fish. Even this imbalanced dataset can be used to train a model. However, it can be biased and inaccurate with imbalanced class distribution, in other words, the model may tend to mis-classify the rare classes with less samples into the majority classes with large samples.

Figure 2: Overall diagram of the proposed image recognition framework

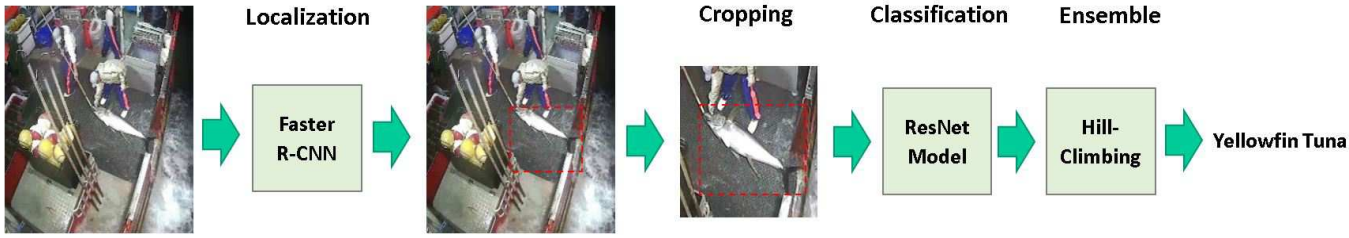

Figure 3: A sample demonstration using the proposed framework to perform fish classification

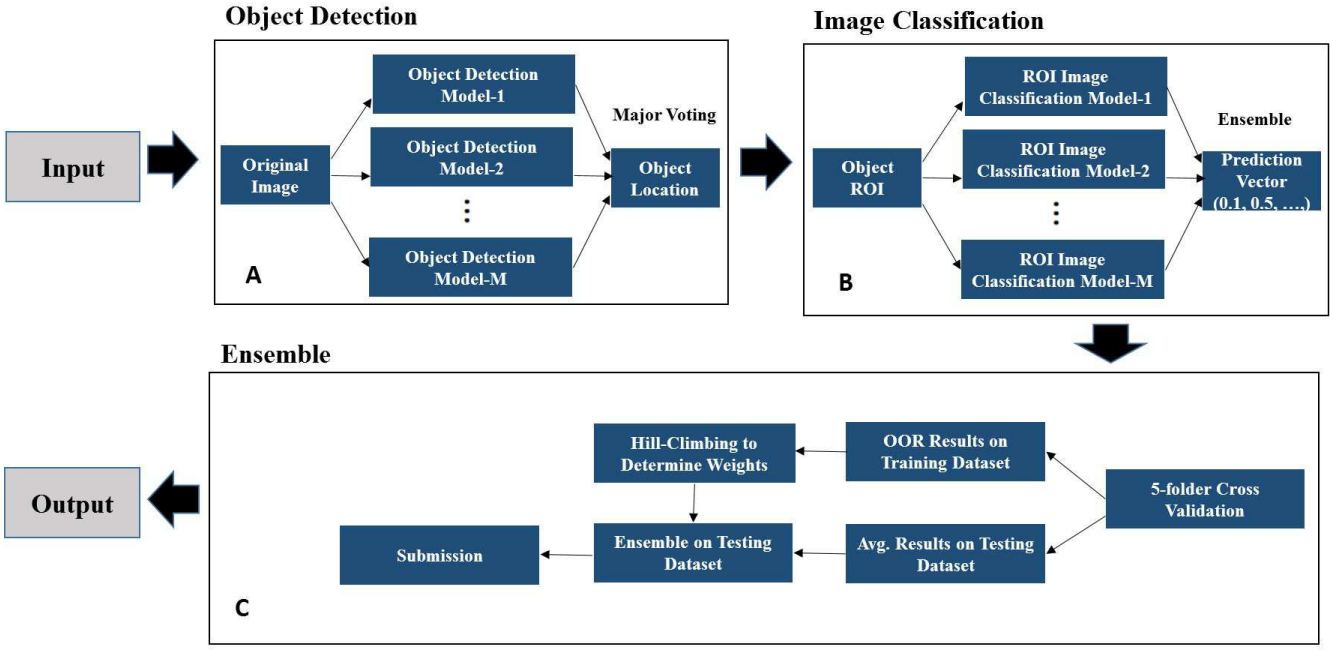

Figure 4: The three main steps of the proposed framework for image recognition based on Deep Learning Techniques

To solve this problem, we deploy data augmentation and cross validation that can help to increase accuracy of the classifier. Our method basically consists of i) oversampling the rare samples by horizontal flipping, slight shifting and rotation, and ii) oversampling the rare samples by changing color channel variance. We also split the training dataset into different folds during cross validation so as each fold contains enough the classes that are not represented equally in the dataset(e.g., Stratified K-fold). In such a way, our method can reduce the negative effect of data imbalance to a minimum.

Pseudo Labeling & Data Augmentation for Limited Training Samples. In many machine learning applications, getting data is expensive that it involves time-consuming and labour-intensive process, e.g., ground truth have to be manually labeled and then they will be confirmed by multiple experts in the domain. Using the limited data to train a good machine learning model is still an open but challenging problem. In both Kaggle image recognition competitions studied in Section 4, the number of training samples is only one-forth to one-third of the number of testing samples, e.g., 3792 versus 13153 in the nature conservancy fisheries monitoring.

So a question comes very naturally: can we leverage un-labeled data to further improve the performances of machine learning models?

We use the pseudo labeling approach to utilize the unlabeled data in the testing dataset to improve the performance of deep neural models. We first train a model based on original training dataset, then perform prediction on the testing dataset to select the most confident samples and add them into original training dataset. We again train a new model with enlarged training samples. The experiment results demonstrate that the pseudo labeling approach can slightly improve the overall performance of our proposed framework.

Dealing with Overfitting Issues. The goal of a deep learning model is to generalize the training data to any data from the problem domain. This allows us to make predictions on unseen data by the model in the future. In both Kaggle image recognition competitions studied in Section 4, samples in training dataset and public testing set are much smaller than the samples in private testing dataset. As a result, the model might be overfitted to training set and public testing set. As a result, the model might perform quite bad on private testing set though it achieve good results on public testing set.

We first deploy k-folder ($k = 10$ in fish recognition and $k = 5$ in cervix recognition) cross-validation approach to minimize the overfitting problem in both the competitions. The output of the model is the average of the k-trained sub-models. This model generalization approach increases our competition score significantly in both the leaderboards.

We also perform data augmentation to relief overfitting problem. To build a good image classifier using the limited training data, image augmentation is a method that can help to boost the performance of deep neural networks. We use the combination of multiple processing, e.g., random rotation, shifts, shear, scale and flips, to create artificial images. We also implemented mean-variance normalization, color space transformation and elastic transformation, to enhance the image augmentation. Our experiment results showed that data augmentation is one of the important methods to achieve the good performance in many image classifiers. Different domain applications with the limited training data can adopt different data augmentation strategies to build good image classifiers.

4 CASE STUDIES

Our proposed image recognition framework has been trained and evaluated using the image sets from various image recognition applications. Specifically, in this paper, we use two case studies on Kaggle image recognition competitions: "The Nature Conservancy Fisheries Monitoring" [3] and "Intel & MobileODT Cervical Cancer Screening" [2] in this Section. Both challenges are two-stage competitions. In the first phase, participants train models and submit to a small/temporary leaderboard. In the second phase, they use the same models developed in the first phase to predict on an unseen test set. The spirit of having a second stage is to prevent hand labeling and leaderboard probing of the test data. In order to achieve this, participants must upload their source codes or models with fixed parameters ahead of the second stage dataset is released. These "codes" or "models" will be examined by Kaggle and the competition host to determine the eligibility to win the competition.

4.1 Problem Description

The Nature Conservancy Fisheries Monitoring.

The main source of protein for nearly half of the world population depends on seafood. The Nature Conservancy reports that 60% of world's tuna is caught in the Western and Central Pacific. The fishing activity contains many illegal, unreported, and unregulated fishing practices that affect the balance of marine ecosystem, global seafood supplies, and local livelihood to some extent.

Many existing electronic fisheries monitoring systems work well and can be deployed easily. However, they generate monitoring data which are expensive to process manually in term of time and cost. Therefore, the Conservancy seeks to use cameras that can dramatically scale the fishing monitoring activities.

In this competition, the task is to develop an algorithm to automatically detect and classify species of the catch from fishing boats that it may shorten the video review process time (Fig. 5).

Figure 5: Image samples for fish recognition competition. Picture is taken from [3]

Two major advantages of faster review with reliable data are easy to i) reallocate human capital to management, and ii) enforce fishing activities. They bring a positive impact on environment of our planet.

The Conservancy provides three datasets, i.e., training dataset contains 3, 792 images, stage-1 testing dataset contains 1, 000 images, and stage-2 testing dataset contains 12, 153 images. The images are taken from fixed cameras mounted on fish boats. The goal is to develop a model that can detect and classify species of fish into 8 different classes such as i) Albacore tuna, ii) Bigeye tuna, iii) Yellowfin tuna, iv) Mahi Mahi, v) Opah, vi) Sharks, vii) Other (fish can not categorized into above the 6 classes, and viii) No Fish. For illustration, Fig.6 shows the pictures of fishes from the first six types. We assume that each image only belongs to the 8 classes. However, some given images show more than one class of fish, i.e., fish within one of above classes and some other small fish. The small fish are used as bait that are not counted in this competition.

Intel & MobileODT Cervical Cancer Screening.

Cervical cancer can be prevented for women if its pre-cancerous stage is identified, and effective and life-saving treatment is carried out. However, one of the main challenges in cervical cancer treatment is to find and determine the appropriate method due to varying physiological difference in women. Women who suffer cervical cancer cannot receive appropriate treatment in rural areas. Even worse, many of them receive wrong treatments that can result in high cost and risk their lives.

Figure 6: Illustration of six fish types. Picture is taken from [3]

Intel & MobileODT work together to improve the existing Quality Assurance workflow that can help rural healthcare providers to make better treatment decisions for cervical cancer. One improvement of the workflow is to allow real-time determinations of the cancer treatment based on woman cervix type (Fig. 7). In this competition, the task is to develop an algorithm that can correctly identify woman cervix type based on the given images. This algorithm can help to reduce cancer mistreatment and allow healthcare providers to refer some cases that need further advanced treatment.

Figure 7: Image samples for cervix recognition competition. Picture taken from [11]

Intel & MobileODT provide three datasets in this competition, i.e., training dataset contains 1,466 images, stage-1 testing dataset contains 512 images, and stage-2 testing dataset contains 3,506 images. The images are taken under various illumination, optical filtering, magnification, etc. The target is to classify a given image into three cervix categories: type-1, type-2, and type-3. Most cervical cancers begin in the cells of the transformation zone. Different transformation zone locations can be used to determine different types of cervix cancer, as shown in Fig. 8. Normally, cervix types-2 and 3 may include hidden lesions that require different treatments.

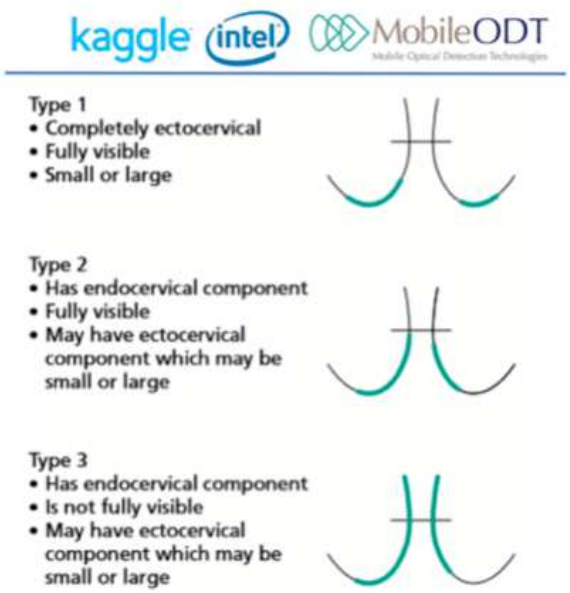

Figure 8: Illustration of three cervix types. Picture is taken from [2]

4.2 Our Solutions

The Nature Conservancy Fisheries Monitoring. Basically, the solution can be divided into three stages as follows: i) first one is to carry out classification by VGG16 and ResNet50 directly on full images, ii) the second is to detect and crop fish by Faster R-CNN and SSD, and iii) the last stage is to apply ResNet50 on detected fish region of interests (ROIs). The final output is the ensemble of the results from both stage-1 and stage-3. We detail each stage in the following.

Stage-1 Classification on original fish images. This stage can be divided into 5 steps. We run step by step as follows.

Step 1. Preprocessing - we use different data augmentation in terms of Keras ImageDataGenerator, deal with the training sample imbalance by horizontally flipping the images on the fish types with fewer samples, and increase the sample number for limited fish types. The original images are resized to 270×480 (height/width ratio is set to $9:16$). We add artificial borders to the image that its ratio is close to $9:16$. Distorting the original relative scale should be avoided that it is not helped in fish detection and classification.

Step 2. Model training - we train 10 VGG16 and 10 ResNet50 models on the original training dataset and then test them on the public testing dataset. After that, we ensemble results from VGG16 and ResNet50 respectively. The best loss result $0.9x$ from VGG16 is slightly worse than $0.8x$ of ResNet50 on the public testing dataset. So we choose ResNet50 in the sequential steps.

Step 3. Data filtering - we then proceed to clean the training dataset by manually checking and removing the mislabelled samples. As a result, the number of the original training samples is reduced from 3,792 to 3,702. We further remove 32 uncertain samples from 3,702 which are grouped into "Other" class. At the end of Step 3, we generate a new training dataset that contains 3,670 samples.

Step 4. Re-train - we train 10 ResNet50 models on the new training dataset with 3,670 samples from the Step 3, then tested on public testing dataset, and ensemble all results from the models and obtain the best loss result $0.7x$. We denote the best result from this step as Rst_1.

Step 5. Move to next stage-2 - we run many trials in this step. At the end of the trials, the best result is still around 0.7x. Therefore, we move to the next step, fish detection.

Stage-2 Fish detection by using Faster R-CNN and SSD. This stage can be divided into 5 steps. We run step by step as follows.

Step 1. Generating label text files: we borrow the annotation from the Kaggle forum discussion to generate the bounding box requested by Faster R-CNN and SSD for object detection.

Step 2. First object detection models: we train various Faster R-CNN and SSD models using the annotated fish dataset. Our goal in Step 2 is to perform fish detection only. Therefore, we configure one class (fish) only and then use the pre-trained VGG16 ImageNet model as base networks to train the models.

Step 3. Cropping fish region of interests (ROIs): we use the trained object detection models for fish detection on public testing image set, and then crop fish ROIs using the bounding box with the highest probability value. Out of 1000 given images, a total of fish ROIs is 789.

Step 4. Second object detection models: we use another set of annotations (we manually generate segmentation masks for each of the fish on the original fish training dataset). In a similar way, we also train another set of Faster R-CNN and SSD models, and then crop another set of 804 fish ROIs.

Step 5. Final output: we run two sets of trained object detection models on the same testing image set using the following condition: if majority of the models detect fish on the given image, then select the bounding box with the highest probability value, and output the fish ROIs. At the end of Step 5, the final output of ROIs number is about 746.

Stage-3 Classification on cropped fish ROIs. This stage can be divided into 3 steps. We run step by step as follows.

Step 1. Training: we train 10 ResNet50 models on annotated fish ROIs. In a similar way, we use same data augmentation and training strategy as in the stage-1.

Step 2. Testing: we test the trained models on the cropped fish ROIs from the public testing dataset, then ensemble and output the classification probability results, denoted as Rst_2.

Step 3. Merging: We use the result from Rst_1 on a given image without any fish detection. Otherwise we combine the weighted results of both Rst_1 of the Stage-1 and Rst_2 of the Stage-3. The best loss result $0.6x$ is obtained on public testing dataset.

Intel & MobileODT Cervical Cancer Screening. The overall workflow of our solution is depicted in Fig. 4. The solution can be divided into two stages as follows: i) first is to perform cervical detection using YOLO and faster R-CNN models on full images, and ii) second is to apply various image classifiers on cropped cervical ROIs. We detail each stage in the following.

Stage-1 Cervix detection on original image. This stage can be divided into 5 steps. We run step by step as follows.

Step 1. Generating label text files: the bounding box requested by YOLO and faster R-CNN for object detection is generated. All the images from all the image sets are squared size by bordering the shorter side of the images with black pixels.

Step 2. Model training: Various YOLO and faster R-CNN models are trained using the annotated cervical datasets with different number of anchors.

Step 3. Crop cervix ROIs: The trained models perform cervix detection on both training and public testing image sets, and cervical ROIs are cropped using the bounding box with the highest probability value from the above models.

Step 4. Re-train and cropping: The cropped cervical ROIs from training set are used as refined annotations to re-train various models and crop new ROIs by repeating Steps 2 and 3 above.

Step 5. Resize ROIs: In this step, all the cropped ROIs are resized to 224×224 for the next stage. In the end of this step, a total of 1466 and 512 ROIs are generated from training and public testing set, respectively.

Stage-2 Cervix classification on cropped ROIs. This stage can be divided into 3 steps. We run step by step as follows.

Step 1. Pre-processing: Image preprocessing is performed in this step, e.g., different data augmentation in terms of Keras ImageDataGenerator, such as rotation, flipping, and shifting. The data imbalance can be reduced using horizontal flipping of the images.

Step 2. Model training: Various CNNs (VGG16, VGG19, Inception V3, and ResNet50) are used to train models using the cropped ROIs from the training dataset, and then test on the cropped ROIs from public testing dataset.

Step 3. The final result is an ensemble of the probability values from various CNNs, where the ensemble weights are determined by using hill-climbing optimization.

4.3 Numerical Results

Evaluation Metric. Both the competitions involves in multi-class image recognition tasks. Each image has been labeled with one true class. We need to submit a set of predicted probabilities on every image. The performance of the proposed framework applying on the competitions is evaluated using the definition of log loss for multi-class image recognition problem as follows.

$$\log loss = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij}\log(p_{ij}), \tag{1}$$

where n is the number of images in the test dataset, m is the number of image class labels, y_{ij} is 1 if observation i belongs to Class j and 0 otherwise, and p_{ij} is the predicted probability that the observation i belongs to Class j. The smaller the log loss, the better the performance of our framework can achieve.

The Nature Conservancy Fisheries Monitoring. In this competition, we performe the fish localization by using Faster R-CNN and SSD, and then determine the fish location using the results from the localization models with top probabilities. Subsequently, we performe fish classification using VGGNet16 and ResNet50 models. In the last step, we obtain a final output by assembling the classification probabilities from each of the deep CNN models.

The final output is evaluated using the multi-class *log* loss based on Eq. 1. Table 1 shows that our proposed framework scores 0.64 and 1.19 on public leaderboard and private leaderboard, respectively, which are much better than the values of 0.72 and 1.89 by image classifiers only. In this competition, we are ranked 4 out of 2, 293 teams.

Method	Standard Image Classifier	Our Proposed Framework
Score	1.89/0.72	1.19/0.64

Table 1: Performance based on the Kaggle competition on "The Nature Conservancy Fisheries Monitoring" using evaluation metric Eqn. 1 on private/public leaderboard.

Intel & MobileODT Cervical Cancer Screening. In this competition, we first perform the cervical region of interest (ROI) detection using Faster R-CNN and YOLO, then determined the cervix location using the results from the localization models via a major voting scheme. Subsequently, we classify cervix types using various deep learning models such as VGGNet, Xception, Inception V3, and ResNet50. Finally, we use hill-climbing to determine the linear combination weights for the final ensemble as our output.

The final output is evaluated using the multi-class *log* loss based on Eq. 1. Table 2 compares the scores using image classifiers only and our proposed framework. Though the score 0.427 of image classifier on public leaderboard is better than 0.458 of our proposed framework. But the former seems overfitting to the stage-1 testing set, the score on private leaderbaord increases to 0.873, which is much worse than 0.808 of our proposed framework. The final score 0.808 of our proposed framework is ranked 3 out of 834 teams.

Method	Standard Image Classifier	Our Proposed Framework
Score	0.873 / 0.427	0.808 / 0.458

Table 2: Performance based on the Kaggle competition on "Intel & MobileODT Cervical Cancer Screening" using evaluation metric Eqn. 1 on private/public leaderboard.

5 CONCLUSIONS AND FUTURE WORKS

In this paper, we propose a computational deep learning framework used for image recognition, specifically on image classification of object types within the image. The proposed framework involves the latest developments in deep learning network architectures for object detection and image classification. The effectiveness of the proposed framework has been demonstrated by the promising performance on various Kaggle image recognition competitions.

To continuously enhance the performance of the proposed framework for image recognition applications, several directions are worthy of further investigations. Firstly, we are working on how to leverage un-labeled data to further improve performances. One of the possible solutions is to perform semi-supervised learning by Generative Adversarial Networks (GAN)[5]. Promising results have been reported in several research works like [16]. Secondly, from the experiments, we found out that most of the classification failures are caused by the hard samples or classes. For example, some classes in fish recognition competitions, say, "ALB", "BET" and "YFT", are extremely similar to each other. It is very hard for the model even human to correctly classify them. We are looking for some solutions like focal loss [14] to deal with this issue. Deep learning grows very fast, recently, more elegant deep network architectures have been presented for classification and regression problems, such as Inception V4 [25], DenseNet [9], Dual Path Networks [1] and many more. We are planing to integrate these new CNN architectures into our proposed framework to further increase the performance.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Russ Wolfinger, Mr. Dmytro Poplavskiy, Mr. Gilberto Titericz, and Mr. Joseph Chui for their insightful discussions and suggestions on the Kaggle competitions presented in this study.

REFERENCES

[1] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. 2017. Dual path networks. In *Advances in Neural Information Processing Systems*. 4470–4478.
[2] Kaggle Competition. 2017. Intel & MobileODT Cervical Cancer Screening. https://www.kaggle.com/c/intel-mobileodt-cervical-cancer-screening
[3] Kaggle Competition. 2017. The Nature Conservancy Fisheries Monitoring. https://www.kaggle.com/c/the-nature-conservancy-fisheries-monitoring
[4] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. 2016. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*. 379–387.
[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.
[6] Cao Guimei, Xie Xuemei, Yang Wenzhe, Liao Quan, Shi Guangming, and Jinjian Wu. 2017. Feature-fused SSD: fast detection for small objects. In *Ninth International Conference on Graphic and Image Processing*. 106151E.
[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer*

vision and pattern recognition. 770–778.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European Conference on Computer Vision*. Springer, 630–645.

[9] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2016. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2261–2269.

[10] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. 2017. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3296–3297.

[11] Health Information. 2015. Approaches of Cervical Cancer. http://magickeepers.blogspot.sg/2015/12/wise-woman-approaches-of-cervical-cancer.html

[12] He Kaiming, Gkioxari Georgia, Dollar Piotr, and Girshick Ross. 2017. Mask R-CNN. In *Proceedings of IEEE Conference on Computer Vision*. 2961–2969.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *arXiv:1708.02002*.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*. Springer, 21–37.

[16] Augustus Odena. 2016. Semi-supervised learning with generative adversarial networks. In *arXiv:1606.01583*.

[17] Waseem Rawat and Zenghui Wang. 2017. Deep convolutional neural networks for image classification: A comprehensive review. In *Neural computation*. MIT Press, 2352–2449.

[18] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 779–788.

[19] Joseph Redmon and Ali Farhadi. 2016. YOLO9000: better, faster, stronger. In *arXiv:1612.08242*.

[20] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.

[21] Girshick Ross. 2015. Fast R-CNN. In *arXiv:1504.08083*.

[22] Girshick Ross, Donahue Jeff, and Malik Trevor, Darrell andJitendra. 2016. Region-based convolutional networks for accurate object Detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2016), 142–158.

[23] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural networks*, 85–117.

[24] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. In *arXiv:1409.1556*.

[25] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning.. In *AAAI*. 4278–4284.

[26] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[27] Christian Szegedy, Scott Reed, Dumitru Erhan, Dragomir Anguelov, and Sergey Ioffe. 2014. Scalable, high-quality object detection. In *arXiv:1412.1441*.

[28] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2818–2826.