IBM GRAND CHALLENG

GROUP 6: JING WANG & HE ZHANG & YANG MA

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

In this report, we explore the use of neural networks to classify images captured by web cams into rain/no rain. The dataset is composed of web images of Dutch, Belgian and London Urban areas provided by Weather Underground website. Every day there are incremental images added to the dataset with the weather condition information assigned to each of them. Neural networks are used to learn the hidden patterns such that the rain/no rain label is predicted automatically by the learned model.

One of the challenges of the project arises from the poor quality of the images. The raindrops in most of the rain images are not observable and it is difficult to determine what features can best distinguish rain/no rain. Here, We use both the regular neural networks and the convolutional neural networks(CNNs) to predict the label. The performance of the regular networks depends much on the features extracted from the images. Thus, two popular descriptors used for object detection, histogram of oriented gradients (HOG) and Scale-invariant feature transform (or SIFT), are applied to extract features. Despite its similarity to the regular networks, convolution layers in CNNs contain learnable filters such that networks can self-learn the features.

In Section 2, the previous work addressing this problem is presented. Then, the feature extraction techniques and the fundamentals of CNNs are discussed in details in Section 3. Section 4 shows the design of the regular networks and CNNs as well, while 5 explains the experiment design and performance metrics. At last, the results and analysis are presented in Section 6 followed by the conclusion of the project.

2 LITERATURE REVIEW

Traditional rain detection mainly focuses on the segmentation and the removal of rain streaks in images. While rain removal in image sequences has been intensively investigated, the problem of rain removal from a single image has been discussed far less often. Single-frame based rain removal approach is more challenging due to the lack of temporal information. Fu et al.[1] are among the first to propose a rain removal framework by formulating it as an image decomposition problem based on morphological component analysis (MCA). More recently, Kim et al. [2] used an adaptive nonlocal means filter for deraining. However, none of these approaches show good performance in the web cam images dataset, given that the rain streaks in most of the images are barely notable. Thus, what we are looking for is a descriptor more suitable for extracting features that can best distinguish rainy images from the non-rainy ones.

One of the most widely used feature descriptor in computer vision and image processing for object detection is the histogram of oriented gradients (HOG). The HOG descriptor was found to be particularly suitable for human detection even with a large range of pose variations[3]. Later, a complete method for pedestrian detection applied to infrared images was introduced[4]. In addition to human detection, results show that the HOG descriptors can also be applied to objects such as cars, buses and animals[5]. Given its advantages such as operating on local cells of color images and invariant to geometric and photometric transformations, The HOG descriptor

is expected to extract raindrops, puddles, as well as the features of clouds and sky in rainy days.

Scale-invariant feature transform (SIFT) is another popular algorithm to detect local features in images. The SIFT feature descriptor is robust for invariance to scaling, orientation and illumination changes[13][14][15]. A number of variations of SIFT are derived to improve the performance. SURF speeds up feature computation and matching by using integral images and increases the robustness of the descriptor[16]. PCA-SIFT is another variant of SIFT that reduces the dimensions of the descriptor with PCA[17]. Color SIFT is a colored local invariant feature descriptor robust to color and photometrical variations. Since the conventional SIFT is designed mainly for gray images, Color SIFT retrieves the color information in object detection[18].

Unlike a typical object detection problem, the raindrops in most of the raining images in the dataset are barely noticeable, making it difficult to determine what features can best describe the rainy days. Convolutional Neural Networks (CNNs) is one of the most powerful approaches for image recognition which requires little pre-processing or human effort in designing features. The convolution layers in a CNN network are responsible for learning filters such that the features are learned by the network itself[6]. Ciresan et al.[7] achieved near-human performance on the very competitive MNIST handwriting benchmark using deep convolutional neural networks. In 2012, the AlexNet, developed by Alex Krizhevsky et al., significantly outperformed the second runner-up in the ImageNet ILSVRC challenge. The AlexNet successfully popularized convolutional neural networks by stacking more than one convolutional layers on top of each other[8]. The ILSVRC 2014 winner, GoogLeNet, developed an Inception Module that significantly improved utilization of the computing resources inside the network. The runner-up in ILSVRC 2014 was the VGGNet which increased the depth to 16-19 weight layers, showing the depth of the network is critical for good performance. Many deep learning framework[10].

3 METHODS

In this section, we will discuss two learning methods to classify rain/no rain images: traditional feedforward neural networks and Convolutional Neural Networks. The inputs of traditional feedforward neural networks are the features extracted by computer vision and image processing algorithms. Thus, an effective feature extraction technique is crucial for the neural networks to have good performance. In our report, two widely used descriptors in object detection, HOG and SIFT, are adopted. CNNs are similar to regular neural networks but introduce the convolutional layers whose parameters consist of a set of learnable filters. Such filters save the human efforts to extract features but dramatically increase the number of parameters in the networks.

3.1 Feature Extraction

The rain/no rain classification problem is not as simple as detection of raindrops, since they are barely noticeable in many rain images partially because of the low resolution. The possible useful features could also be the puddles, color of sky, blurriness of the image and so on. HOG and SIFT are two descriptors that show great effectiveness in object detection.

HISTOGRAM OF ORIENTED GRADIENTS(HOG) The HOG descriptor counts occurrences of gradient orientation on a dense grid of local cells. In this way, the object can be described by a distribution of gradients and edge directions. Such computation of intensity gradients might also reflect the blurriness of the images. One of the greatest advantages of HOG is its invariance to geometric and photometric transformations, permitting the changing shapes of raindrops to be ignored. Although it is not invariant to object orientation, fortunately it is not a problem for objects like raindrops, puddles or clouds.

SCALE-INVARIANT FEATURE TRANSFORM (SIFT) SIFT extracts keypoints of objects from a set of reference images. Such keypoint locations are found at particular scales and orientations are assigned to them. Hence, the invariance to image location, scale and rotation is ensured, which makes us less worried about the different locations and scales of raindrops, puddles or clouds. After extracting the SIFT keypoints from the reference images, The next step is called Bag of Words(BoW). In this step, we use k-means to cluster these keypoints into a "codebook" and then each image in training set will match this codebook through a Euclidean-distance based nearest neighbor approach. To avoid sparse distance matrix and make the approach more robust, a "soft" assignment is used such that each keypoint is assigned to 8% of the number of the total codebook. Such BoW method is capable of learning multiple instances such as raindrops, dark sky and et al. However, SIFT has its defects in only accepting grayscale images. It means the learning process does not consider the influence of the color of images. We believe for rain/no rain classification problem, color is an important feature that can help distinguish the weather conditions. Thus, a Color SIFT(CSIFT) method is used to improve the SIFT descriptor by taking color into account.

3.2 Feed-forward Neural Networks

Multi-layer neural networks has great power of representation and is capable of approximating non-linear functions of the inputs. It makes even a regular feed-forward neural networks powerful in learning process. It is used to train the features extracted from images using techniques discussed in Section 3.1. However, neural networks has overfitting issue that needs to be carefully dealt with.

3.3 Convolutional Neural Networks

Recently, CNNs gains great popularity due to its good performance in image recognition. The learnable filters in the networks make it independent of human efforts in image feature extraction. This could be very useful in our problem because: 1) it is difficult to determine what features can best distinguish the rain images in our dataset; 2) there is no existing work to design features particularly for weather conditions. CNNs enables the networks to learn useful filters(kernels). Architecture of CNNs have three types of layers: Convolutional layer, Pooling layer and Fully-Connected layer. Convolutional layer consists of the learnable filters and takes the heaviest computation. During the forward pass, each filter is convolved across the input volume through a dot product computation. The stack of convolutional layers on top of each other may improve the performance at the cost of increasing parameters. The Pooling layer to reduce the amount of parameters

and computation in the network while Fully-Connected layer is exactly the same as the hidden layer in regular neural networks. Since we only have a medium size dataset, the architecture needs to be carefully designed to have enough representation power and meanwhile less number of parameters to avoid overfitting.

4 SYSTEM DESIGN

4.1 Feed-forward Neural Network

A feed-forward neural network is an artificial neural network wherein connections between the units do not form a cycle. The feed-forward neural network was the first and simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes and to the output nodes.

So we take this simplest neural network model as our baseline, and also using it to test the performance of several feature extraction techniques on the given dataset.

4.1.1 *Input and output*

For this simple feed-forward model, we tested different types of input. The two feature extraction techniques we used are HOG and SIFT.

The HOG features are widely use for object detection. HOG decomposes an image into small squared cells, computes an histogram of oriented gradients in each cell, normalizes the result using a block-wise pattern, and return a descriptor for each cell. By using [8,8] cellsize, We extracted more than 30,000 HOG features per picture. So the method of using HOG extracting features will yield an input dimension of around 30,000.

The SIFT bundles a feature detector and a feature descriptor. The detector extracts from an image a number of frames (attributed regions) in a way which is consistent with (some) variations of the illumination, viewpoint and other viewing conditions. The descriptor associates to the regions a signature which identifies their appearance compactly and robustly. The Bag of Words approach is then applied to these extracted keypoints by first clustering them into a codebook and then encoding each image over the codebook. Thus, the dimensions of input after applying the SIFT depends on the size of codebook. 1,000 words in the codebook, for instance, means 1000 dimensions of features per image.

As for the output, our goal is to detect whether it's rainy or not at the moment the given webcam image is captured. So it becomes a binary classification problem, with 0 standing for no rain and 1 standing for rain.

4.1.2 *Techniques*

While the simplest a single-layer network can only approximate linear functions, multi-layer networks have much more representation power. Caffe, a deep learning framework, is used to build our custom networks for reasons of its convenience to use and more importantly, its state-of-the-art techniques. As a regular feed-forward network, we only use the Fully-Connected layer provided by Caffe, which is identical to the hidden layer.

The commonly used activation functions include sigmoid, tanh, ReLU and so on. While sigmoid has been frequently used historically, it is found

to be easily saturate at either tail of 0 or 1 and as a result, will "kill" the gradients such that the network barely learns. ReLU, which computes $f(x) = \max(0, x)$, has gained popularity In recent years due to its quick convergence of stochastic gradient descent(SGD). It is also our choice of activation function.

The larger networks have more representation power but also very easily to overfit. To achieve a better generalization on test set, it is preferred to use larger networks with techniques such as L2 regularization and Dropout than the small networks. Dropout technique is especially effective and simple way to overcome overfitting, which keeps a neuron active with some probability p or setting it to zero otherwise[11].

Multi-layer networks use a variety of learning techniques, the most popular being back-propagation. SGD is the most widely used method for updating weights. The derivative of the error function with respect to the network weights is calculated, and the weights are then changed such that the error decreases (thus going downhill on the surface of the error function). There are many approaches to accelerate convergence such as Momentum update and batch learning.

4.1.3 *Parameter Tuning*

Although multi-layer feed-forward neural network is a relatively simple model, there are still some parameters worth tuning to find model with best performance.

HOG PARAMETER There are few parameters needed to tune for HOG descriptor. The optimal cell size was found to be 8x8 or 16x16 pixels with 9 histogram channels. Our experiment shows that cell size of 8x8 is enough to present the oriented gradients of the whole image.

SIFT PARAMETER The key parameters to tune for SIFT descriptor is the capacity of codebook and the encoding method. Capacity of codebook can vary from hundreds to thousands. Image encoding usually uses either hard assignment or soft assignment. Hard assignment matches each descriptor to its nearest neighbor while soft assignment matches N nearest neighbors with decreasing weights. To achieve a more robust system, we use soft assignment in our experiment.

AMOUNT OF LAYERS In the model of networks, the amount of hidden layers could be set in need. Since most networks cannot benefit much from increasing the amount of layers(4,5,6-layer), we only trained networks with 1 or 2 hidden layers with different sizes.

AMOUNT OF NEURONS For larger neural network, unsuitable number of neuron units will cause over-fitting (if too many) or under-fitting (if too small). In order to get better performance, we need to tune the number of neurons for a model. The HOG feature has input dimensions more than 30,000, so the we trained a relatively larger number of neurons per layer. For the SIFT feature, since its input dimensions is only around 1,000, the capacity of each layer is smaller.

DROPOUT Dropout is an extremely effective, simple and recently introduced regularization technique by Srivastava et al [11] that complements the other methods (L1, L2, maxnorm). During training, Dropout can be in-

terpreted as sampling a Neural Network within the full Neural Network, and only updating the parameters of the sampled network based on the input data. During testing there is no dropout applied (more about ensembles in the next section).

SOLVER The solver orchestrates model optimization by coordinating the network's forward inference and backward gradients to form parameter updates that attempt to improve the loss[12]. The responsibilities of learning are divided between the Solver for overseeing the optimization and generating parameter updates and the Net for yielding loss and gradients. We trained the networks with different solvers listed below.

1. Stochastic Gradient Descent (type: "SGD"),
2. AdaDelta (type: "AdaDelta"),
3. Adaptive Gradient (type: "AdaGrad"),
4. Nesterov's Accelerated Gradient (type: "Nesterov") and
5. RMSprop (type: "RMSProp")

4.2 Convolutional Neural Networks

Our deep CNNs is implemented on Caffe[12]. One of the biggest challenges we are facing is the number of images in the dataset. Deep networks like CNNs contains millions of parameters and thus, requires a large number of training images. In this section, we presents two method used to train CNNs: transfer learning and custom architecture of CNNs.

Considering there are less images to train, we first finetune the Caffenet structure. Input the of train set contains shuffled images and mean value of images. Output is the predicted label. Caffenet has five convolution layers and following three full connected layers. To decrease the probability of over-fitting, we change to 3 layers of convolution layer and remain the full connected layer.

4.2.1 *Input and output*

Input of our deep learning are shuffled images in *imdb* format. Several image pre-processing approaches are used before feeding the images into the first convolution layer. First, the images are center cropped and resized to 256by256 pixels. Then the mean value of images are calculated and substracted from the original pixel values. The output is the predicted label of either 0 or 1 indicating rain/no rain.

4.2.2 *Techniques*

IMAGE AUGMENTATION Image augmentation technique is widely used in deep learning to increase the number of total images fed into the networks. We use a simple augmentation approach by center cropping the images and producing mirror images.

FINE-TUNE Fine-tuning an existing CNNs is very common in practice when the dataset is not sufficiently large. Since the learnable filters in the bottom layers are usually very general, it is feasible to only tune the parameter in the top layers to suit a new dataset. The Caffenet architecture(shown in figure 2) is a implementation of the AlexNet which is the runner-up in the ImageNet ILSVRC challenge in 2012. Its great success is the most important

Figure 1: Caffenet architecture

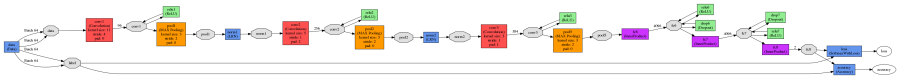


Figure 2: modified architecture

reason we decide to fine-tune this architecture. The original Caffenet has 1000 sorts, but we only need a binary classification. Thus, the Caffenet is customized by changing the top layer into a 4096 inputs and 2 outputs layer. Moreover, the two most important factors that influence the performance are the size of the new dataset (small or big), and its similarity to the original dataset. Considering that our dataset is of medium size and different from the original one, we decided to experiment on fine-tuning methods. We first increase the learning rate of the top layer by 10 times higher than others to get a rapid learning. Then, the second method that fine tunes the last convolution layer and all the fully connected layers was experimented.

CUSTOM ARCHITECTURE Our dataset has around only 20,000 images in total and is very easily to overfit if using the existing architecture like CaffeNet. The architecture needs to be further compacted to produce a reasonably smaller networks. Considering that our images are of low resolution, we decided to remove the last 2 convolution layers and their corresponding pooling layers in CaffeNet. The rest of the architecture is the same as the CaffeNet. The custom architecture contains 3 convolution layers, 3 pooling layers and 3 hidden layers, which is shown in figure 2.

The first convolutional layer filters the 224*224*3 input image with 96 kernels of size 11*11*3 with a stride of 4 pixels. The second convolutional layer takes as input the output of the first convolutional layer and filters it with 256 kernels of size 5*5*48. The fifth convolutional layer has 256 kernels of size 3*3*192. Full connected layers are same as Alex net with each of 4096 neurons.

5 EXPERIMENTAL SET-UP

5.1 Dataset manipulation

Since the original dataset contains a large number of noise and ambiguous images, using the original one without pre-processing will lead to a degraded performance. We manually removes the noise such as indoor images and also a lot of ambiguous images which cannot be classified even by humans. The derived new dataset contains around 7000 rain and 30,000 no rain images.

5.2 Imbalanced dataset

Our dataset is imbalanced with much more no rain images than rain images. If using such dataset, the resulting training accuracy will be high but the rain images are almost undetectable. Thus, the training set needs to be

balanced by sampling techniques to guarantee a good performance. Different ratios between the number of rain and no rain images in the training set are experimented to find out the optimal sampling ratio. The results will be discussed in Section 6.

5.3 Performance metrics

To make the result more robust, we use *k*-fold cross validation to train the dataset. The validation set accounts for nearly 10% of the images and the rest are the training set. Note that when using Caffe to train neural networks, a portion of the training images will be used as validation images to monitor the learning process. We use about 10% of the images in the training set for this purpose.

For learning on imbalanced dataset, accuracy is not to measure the performance. Metrics such as recall, precision and F1 score are more suitable in such situation. Precision measures result relevancy, while recall measures how many truly relevant results are returned. In other words, high precision means less no rain images are assigned to rain label, and high recall means less rain images are assigned to no rain label. Additionally, basic metrics such as ROC curve are also used for performance evaluation.

5.4 Caffe setup

We use Caffe to train both the regular feed-forward neural networks and the convolutional neural networks. The regular feed-forward neural networks only use CPU to train, while GPU mode is enabled to train CNNs due to its intensive computation.

When training CNNs, the batch size of 64 and 30,000 iterations are set, leading to around 90 epochs (30000*64/20000). The fine-tuning model is a pretrained CaffeNet model.

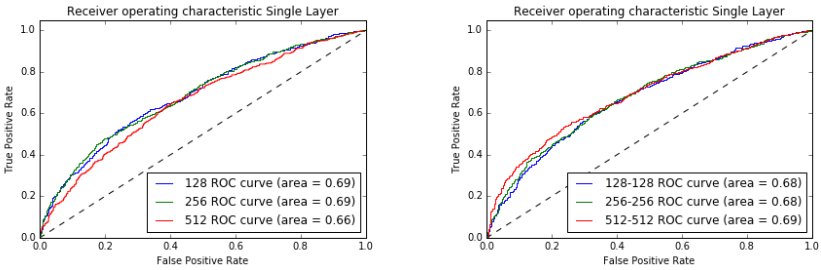
6 RESULTS AND DISCUSSION

6.1 Feed-forward Neural Network

6.1.1 *HOG Training*

Different model set-ups were tested to compare the performance. The experiment result shows a suitable sampling ratio of rain/no rain images is 1:2.5. Adding more no rain images in the training set makes the minority class(rain) underrepresented. The validation set has the same number of rain and no rain images(ratio of 1:1). With 10-fold cross validation applied, the performance is averaged on 10 runs. The metrics used to measure the performance are ROC curves, precision, recall and F1 score.

The precision, recall and F1 score on validation set were calculated for different model size. Based on the input feature dimensions(>30,000), 128, 256, 512 neurons with 1 or 2 hidden layers were selected to build the networks. Table 1 on the next page shows the results for different network architectures. Some architectures show imbalanced performance that is good at no rain detection but bad at rain detection, or vice versa. For example, 512-512 has very high recall when no rain is regarded as positive, but low when rain is regarded as positive. We are looking for an architecture that improves the



(a) Single layer with different size. **(b)** Double layers with different size.

Figure 3: ROC curves of different network models.

performance of rain detection and meanwhile, not degrades no rain detection. Thus, an architecture of one or two hidden layers of 256 neurons(256 or 256-256) has the highest F1 score(0.63) with rain regarded as positive and hence, has the best performance.

Figure 3 shows the ROC curve for different network sizes. The x and y axis represent the true-positive (rain classified as rain) rate and false-positive (no rain classified as rain) respectively. The AUC provides a useful metric to compare different tests. Whereas an AUC value close to 1 indicates an excellent diagnostic test, a curve that lies close to the diagonal (AUC = 0.5) has no information content and therefore no diagnostic utility. In Figure 3a, one-layer architecture with 256 neurons has the largest AUC value, indicating a least possibility of false alarm. Similarly, Figure 3b shows that with two-layer architecture with 256-256 neurons has better performance.

Table 1: Relationship between Network Size and Performance

	No Rain			Rain		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
128	0.56	0.94	0.7	0.79	0.24	0.36
256	0.54	0.84	0.7	0.72	0.42	0.53
512	0.64	0.53	0.58	0.59	0.69	0.63
128-128	0.58	0.86	0.69	0.71	0.34	0.46
256-256	0.6	0.8	0.69	0.68	0.45	0.54
512-512	0.56	0.95	0.71	0.83	0.22	0.35

The experiment results applying drop-out technique with different dropout rate are shown in Table 2 on the next page. while 512-512 architecture has the worst F1 score(regarding rain as positive), drop-out is applied to see whether it can improve the performance. It can be seen that F1 score is much improved especially with drop-out rate of 0.5. It proves the effectiveness of drop-out to overcome overfitting when the network is large.

Dropout scheme is also applied to the other architectures such as 256-256. It turns out the performance is not notably improved. Therefore, for a smaller network, drop-out might not be necessary.

At last, we tested different solvers provided by Caffe on the 256-256 architecture, shown in Table 3 on the following page. It seems that these solvers are not as good as the SGD method. Moreover, it should be noted that SGD is also the fastest among all to converge. AdaGrad and RMSProp almost triple the amount of training time.

Table 2: Relationship between Dropout Rate and Performance (For 512-512)

	No Rain			Rain		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
0.75	0.59	0.88	0.7	0.74	0.35	0.48
0.5	0.63	0.67	0.65	0.63	0.58	0.61
0.25	0.59	0.83	0.69	0.7	0.41	0.52

Table 3: Performance of different solver modes (for 256-256)

	No Rain			Rain		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
AdaDelta	0.54	0.94	0.69	0.74	0.16	0.27
AdaGrad	0.57	0.9	0.7	0.74	0.31	0.43
Nesterov	0.58	0.83	0.68	0.68	0.36	0.47
RMSProp	0.57	0.87	0.69	0.72	0.33	0.45

In conclusion, the feed-forward network of 256 or 256-256 trained by SGD are the best models for the HOG feature. They both achieve F1 score of around 0.54 when rain is regarded as positive and of 0.7 when no rain is regarded as positive.

6.1.2 *SIFT Training*

For the SIFT training, the input size is equal to the number of words in the codebook extracted from the training set. The number of words is a few thousands in our experiment and thus, requires only a small networks. At the beginning, we used SIFT descriptors to learn a codebook, but the result is not promising(almost all the rain images are classified as no rain). One of the most important reasons is that SIFT only accepts grayscale images and as a result, loses color information.

To overcome the drawback of SIFT, we used Color SIFT instead so that we can extract features from RGB images. For the reason of a small network, only one hidden layer is used and the number of neuron between 20 to 40 is enough. Different sizes of the codebook are experimented and soft assignment is used to encode images.

Figure 4 on the next page shows the ROC result, in which we see that a larger size of codebook containing 1500 words significantly outperforms the others. It means the number of words lower than 1000 cannot sufficiently represent the features of images.

More detailed metrics are shown in Table 4, indicating that the performance of Color SIFT is not comparable to HOG. It may be because it is difficult to learn useful keypoints in our dataset or our reference dataset to learn a codebook is not representative enough.

Table 4: Performance of Color Sift with different amount of words (For 20)

	No Rain			Rain		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
1500 words	0.74	0.92	0.82	0.71	0.39	0.50
1000 words	0.67	0.81	0.73	0.37	0.22	0.28
700 words	0.68	0.79	0.73	0.40	0.27	0.32

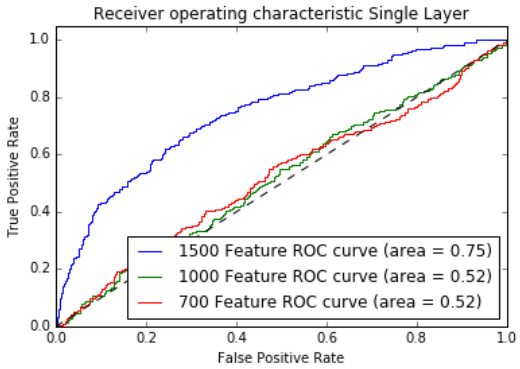


Figure 4: ROC plot for different codebook size on FF net of size 10

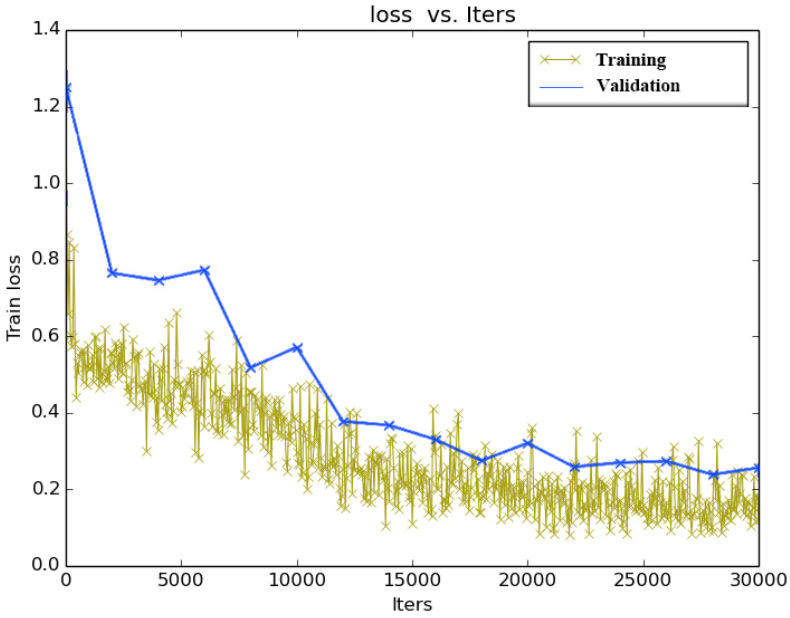


Figure 5: Training loss

6.2 Convolutional Neural Networks

6.2.1 *custom architecture*

As already discussed in 4.2.2, our custom architecture is built on the CaffeNet. However, we reduced two convolution layers to reduce parameters in the network. Due to the low resolution images in our dataset, it is feasible to do so.

Three different sampling ratios of rain/no rain images are compared to find out the one leading to the best performance. The three training set are of 10800 no rain/ 6000 rain(1.8:1), 13800 no rain/ 6000 rain(2.3:1) and 18000 no rain/ 6000 rain(3:1) images respectively. Figure 5 shows the loss on training set and validation set as a function of iterations with sampling ratio 3:1. The loss rate starts high but gradually converges to around 0.2 after 30,000 iterations.

Table 5: Performance between different no rain/rain ratio

	No Rain			Rain		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
1.8:1	0.78	0.51	0.62	0.63	0.85	0.72
2.3:1	0.74	0.65	0.69	0.68	0.76	0.72
3:1	0.67	0.77	0.72	0.72	0.61	0.66

Table 6: Performance of finetune Caffenet

	No Rain			Rain		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
fine-tune	0.60	0.88	0.71	0.79	0.34	0.46

The result of our custom CNNs is shown in table 5. Validation set contains around 800 no rain images and 800 rain images. We can see from the result that if the number of no rain images is not sufficiently large compared to rain images, the learned model shows a tendency to classify no rain images into rain. As a result, false alarm is dramatically increased, which is not acceptable in practice. By increasing the ratio of no rain/rain images to 3:1, we achieve a desirable performance with F1 score of 0.66 when rain is regarded as positive and of 0.72 when no rain is regarded as positive. Such performance outperforms both HOG and SIFT.

6.2.2 *fine-tune*

Fine-tune is done based on a pretrained CaffeNet, in which the learning rate in every layer is 0.001. Firstly, we chose to only fine-tune the top layer. The training took 30,000 iteration with 13,800 no rain and 6000 rain images. The learning rate is set to be 0.01 in the top layer and 0.001 in the rest of the layers. In this way, the trained weights in the bottom layers stay stable while those in the top layers are tuned more actively to suit the new dataset.

Table 6 shows the fine-tune results. It can be seen that the performance is worse than our custom architecture, primarily because the learned filters in convolution layers do not fit well in our new dataset.

We also experimented on fine-tuning not only the top layer, but also the last convolution layer and all the fully connected layers. However, we encountered serious overfitting problem, which probably because our dataset is not sufficiently large.

7 CONCLUSION

In this report, we explored neural-network-based learning methods to classify web cam images into rain/no rain label. We first removed the noise and ambiguous images in the original dataset and then, operated the regular neural networks and the convolutional neural networks on the dataset. The input of the regular neural networks is the feature extracted from HOG and SIFT descriptors respectively. Although CNNs do not need any feature extraction approaches, it is easy to overfit when the dataset is not sufficiently large. Thus, both custom architecture and fine-tuning method are experimented. The main conclusions are drawn as follows:

1. A suitable sampling ratio is critical for both the regular neural networks and CNNs to have good performance. The optimal sampling ratio found in our experiments is around 2.5 ~ 3.

2. HOG achieves a better performance than SIFT in our experiment. Besides, the performance of the conventional SIFT descriptors is undesirable while the Color SIFT improves the performance a lot.

3. Our custom architecture of CNNs which contains 3 convolution layers has the best F1 score under sampling ratio of 3:1. However, the result of fine-tuning method is either worse or showing overfitting.

REFERENCES

[1] Fu Y H, Kang L W, Lin C W, et al. Single-frame-based rain removal via image decomposition[C]//2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2011: 1453-1456.

[2] Kim J H, Lee C, Sim J Y, et al. Single-image deraining using an adaptive nonlocal means filter[C]//2013 IEEE International Conference on Image Processing. IEEE, 2013: 914-917.

[3] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). IEEE, 2005, 1: 886-893.

[4] Suard F, Rakotomamonjy A, Bensrhair A, et al. Pedestrian detection using infrared images and histograms of oriented gradients[C]//2006 IEEE Intelligent Vehicles Symposium. IEEE, 2006: 206-212.

[5] Dalal N, Triggs B. Object detection using histograms of oriented gradients[C]//Pascal VOC Workshop, ECCV. 2006.

[6] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[7] Ciregan D, Meier U, Schmidhuber J. Multi-column deep neural networks for image classification[C]//Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012: 3642-3649.

[8] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[9] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 1-9.

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

[11] Srivastava N, Hinton G E, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. Journal of Machine Learning Research, 2014, 15(1): 1929-1958.

[12] Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional architecture for fast feature embedding[C]//Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014: 675-678.

[13] Lindeberg T. Scale invariant feature transform[J]. Scholarpedia, 2012, 7(5): 10491.

[14] Lowe D G. Object recognition from local scale-invariant features[C]//Computer vision, 1999. The proceedings of the seventh IEEE international conference on. Ieee, 1999, 2: 1150-1157.

[15] Lowe D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60(2): 91-110.

[16] Bay H, Tuytelaars T, Van Gool L. Surf: Speeded up robust features[C]//European conference on computer vision. Springer Berlin Heidelberg, 2006: 404-417.

[17] Ke Y, Sukthankar R. PCA-SIFT: A more distinctive representation for local image descriptors[C]//Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. IEEE, 2004, 2: II-506-II-513 Vol. 2.

[18] Abdel-Hakim A E, Farag A A. CSIFT: A SIFT descriptor with color invariant characteristics[C]//2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). IEEE, 2006, 2: 1978-1983.