Room Cleanliness Detector Report

Yiqing Liu, Zhao Jin, JinWei Zhang

Room Cleanliness Detector Report

Yiqing Liu, Zhao Jin, JinWei Zhang

Abstract

Computer vision is a rising field in machine learning with various applications including image classification, image inquiry retrieval, and item detection. In our project, we are dealing with an image classification problem; specifically, we construct a model classifying the cleanness of the room. Several popular approaches for solving this kind of problem are the basic logistic regression model, KNN model, and more complex deep learning natural networks. In our experiment, we trained several models using the above approaches. We found out the baseline model in KNN has an accuracy of 80% on classifying the image, and carefully designed Convolutional Neural Network (CNN) results in a better performance (85% accuracy). Moreover, adopting the model based on advanced ResNet architecture results in the best performance (100% accuracy).

Keywords: **Computer Vision**, **Deep Learning**, **CNN**

1. Introduction

Computer vision is a field of science that studies how to make machines "see", referring to the usage of cameras and computers to identify, track and measure objects instead of human eyes, and further apply graphics processing. It enables image classification, which is utilized to solve visual recognition problems relating to object detection and image captioning. And with development of algorithms, data set and computation capacity, convolutional neural networks (CNN) becomes an important tool for object recognition.

Email addresses: `846051330@qq.com` (Yiqing Liu), `jinzhao0531@qq.com` (Zhao Jin), `2735074855@qq.com` (JinWei Zhang)

Aiming at construct a room cleanliness detector model utilizing deep neural network, which is able to output a probability of a room of messy or clean with input of a picture of the room, this project applies both deep learning algorithm and non-deep learning algorithm and compares the different performances of them. Deep learning framework like Pytorch is used and training skills such as dropout and data augmentation are applied in order to improve the model.

The rest of this report is constructed as: Section 2 introduces background of CNN, KNN and ResNet. Section 3 presents the methodology of the model, and Section 4 illustrates the experiments of the project. Finally in Section 5 the conclusion is drawn.

2. Background

1. **Background of Convolutional neural networks (CNN)**

   Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms, which reflect the behavior of the human brain, allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning. And convolutional neural networks (CNN) are one type of it often utilized for classification and computer vision tasks.

   Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are: convolution layer, pooling layer and fully connected layer. The convolution layer is the core building block of a CNN. In it a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. The pool layer (also known as subsampling) reduces the size and the number of parameters in the input. Although a lot of information is lost in the pool layer, it also has many benefits for CNN, helping reduce complexity, improve efficiency, and limit the risks of overfitting. And in a fully connected layer, each node in the output layer is directly connected to the node in the previous layer. It performs the classification task based on the elements extracted through the previous layer and their different filters.

2. **KNN and Logistic Regression**

   There are different algorithm of image classification, and KNN and logistic regression are two important part of it. To begin with K-Nearest Neighbor (KNN), one of the simplest machine learning algorithms, the idea of which is that if most of the K samples in the eigenspace that are most similar to the sample (the most adjacent samples in the eigenspace) belong to a certain category, then the sample also belongs to this category. And logistic regression is a generalized linear regression analysis model often used in data mining, automatic disease diagnosis, economic forecasting and other fields. Applying multinomial logistic regression, sotfmax classifier can interpret raw classifier scores as probabilities.

3. **ResNet**

   To solving the degradation problem occurring with the development of the CNN that when the network depth increases, the accuracy of the network becomes saturated and even decreases, ResNet is invented utilizing residual block, composed by F(x)+x.

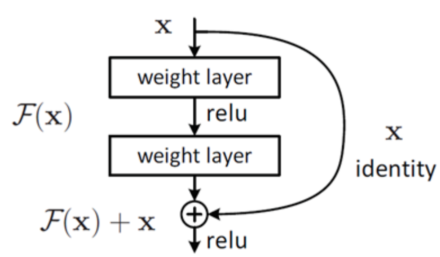

Figure 1: *Structure of residual block of ResNet*

With this special structure, the convolution layer learns new features based on the input features and thus has better performance, which is kind of like a shortcut in a circuit, so it is also called shortcut connection. The advantage of adding this type of connection is because if any layer hurt the performance of architecture then it will be skipped by regularization.

3. Our Method

Determining whether the room is clean or messy is a task of binary classification. Based on our intuition before, we would choose KNN algorithm or Logistic Regression classification for their simplicity and effectiveness. However, with the idea of deep neuron network, we can build a convolutional neuron network(CNN) model to train in our dataset and we expected to see a better performance over the dataset comparing with two algorithms stated before.

4. Experiments

Pytorch is a popular and beginner-friendly open source machine learning library, used for a wide range of applications such as Computer Vision. So for this project we used Pytorch to achieve our primary target.

We have figure dataset of different rooms, each of the figure is of size 299×299. And each figure is nicely labeled as 'clean' or 'messy' according to the file location path. So we can import the dataset and label easily into our Pytorch file. And the figures are already organised into training dataset and validation dataset, which we have 192 figures for training and 20 figures for validating. The example figures from the dataset is shown in Figure 2:

(a) *Clean Room* (b) *Messy Room*

Figure 2: *Figures of different room labeled as Clean or Messy*

For better performance of the training model, we first pre-processed the data. Data argumentation is used to prevent the case of over-fitting. We used random rotation, random crop and scales to 224×224 patch as well as horizontal flip. Figure 3 shows one of the example of figure with data argumentation:

Figure 3: *Pre-processed Room Figure*

Then we trained our Convolutional neural network model by Pytorch nn modules. We defined our Cnn model as sequence of three convolutional layers which consists of Con2d, Batch Normalization for limit over-fitting, ReLU for activation function and Maximum Pooling. Then used two fully-connected layers with dropout and final classifier.

As a result from this model, we achieved accuracy over the validation set around 85%. As shown in Figure 4 which produced 'probablities' of the rooms are labeled 'Messy'. But this result seems not as ideal as expected for deep neuron network over this small dataset.

0	tensor(0)	0.722008
1	tensor(0)	0.624029
2	tensor(0)	0.209312
3	tensor(0)	0.293367
4	tensor(0)	0.683587
5	tensor(0)	0.449437
6	tensor(0)	0.228189
7	tensor(0)	0.088631
8	tensor(0)	0.155525
9	tensor(0)	0.778073

10	tensor(1)	0.866424
11	tensor(1)	0.932162
12	tensor(1)	0.618290
13	tensor(1)	0.873948
14	tensor(1)	0.912651
15	tensor(1)	0.911762
16	tensor(1)	0.644107
17	tensor(1)	0.884319
18	tensor(1)	0.931003
19	tensor(1)	0.667457

Figure 4: *Table of Probabilities calculated by Softmax function of each room figure in validation set. First column is the index, second column stands for the true label of the figure with tensor(0) as 'Clean' and tensor(1) as 'Messy', and the probability of the figure label as 'Messy' from our model in the last column*

To evaluate our result, we compared with other machine learning algorithm, specifically KNN and Logistic Regression. It turns out the best model of regard those non deep learning models has an accuracy of 80% , which is worse than our CNN model.

Specifically, we used the pixel-wise feature first in training our KNN and Logistic Regression model. By flatting the multi-dimension array of the RGB info, we got a single dimension array with all pixel values. By training the high dimensional feature directly on the KNN and the Logistic Regression model, we got a validation accuracy of 55% on Logistic Regression and 60% on KNN with the best hyper-parameter setting.

	Training accuracy	Validation accuracy
C = 0.0001	1.0	0.50
C = 0.001	1.0	0.55
C = 0.01	1.0	0.55
C = 0.1	1.0	0.55
C = 1	1.0	0.50
C = 10	1.0	0.50

(a) *Logistic Regression*

	Training accuracy	Validation accuracy
k = 1	1.000000	0.60
k = 2	0.583333	0.50
k = 3	0.697917	0.55
k = 4	0.583333	0.55
k = 5	0.604167	0.55

(b) *KNN*

Figure 5: *Accuracy on baseline model*

To improve on our model, we did a feature engineering on the pixel data. We select a threshold 100, and return the feature with the average number of pixels that are greater than the threshold over three color, and then average row-wise and column-wise. By training on our improved model with the best hyper-parameter, we got a validation accuracy of 65% on Logistic Regression and 80% on KNN.

	Training accuracy	Validation accuracy
C = 0.0001	0.828125	0.60
C = 0.001	0.921875	0.60
C = 0.01	1.000000	0.60
C = 0.1	1.000000	0.65
C = 1	1.000000	0.65
C = 10	1.000000	0.65

(a) *Logistic Regression*

	Training accuracy	Validation accuracy
k = 1	1.000000	0.50
k = 2	0.802083	0.55
k = 3	0.807292	0.70
k = 4	0.781250	0.55
k = 5	0.781250	0.80

(b) *KNN*

Figure 6: *Accuracy on improved model*

Furthermore, we used ResNet18 and ResNet34 model pretrained on ImageNet. With normalization to the ImageNet mean and standard deviation, we reached 100% accuracy to classify the cleanliness of the room.

5. Conclusion

In our project, we designed a CNN model with three main blocks which each one contains a convolutional layer, a batch normal layer, a ReLU layer, and a max-pooling layer. We also used two fully connected layers in the end with the ReLU activation function and dropout method. By training this model, we got around 85% accuracy on our validation set. To understand our results, we compared the accuracy with baseline models. The best baseline model (KNN) with feature engineering could only achieve an accuracy of 80%. It turns out that the performance of our CNN model is acceptable.

Furthermore, to increase our classification accuracy, we employed the pre-trained model (ResNet18 and ResNet34). By training this model, the validation accuracy reached 100%.

By understanding models like ResNet can classify all images correctly, we would like to figure out what features in the ResNet make it successful and improve our model in the future.