Room Cleanliness Detection Using Deep Learning Network

1st Xuanmao Huang
School of Information Science and Technology
Shanghaitech University
xuanmao259@126.com

***Abstract*—The goal of this project is to evaluate the messy probability of a given room picture. For an input image, our model will output a messy probability, indicating how messy the room is. We tune parameters for our models to achieve the best test probability compared to ground truth. We show how different models perform on this task, providing training time, test accuracy, etc.**

***Index Terms*—SVM, CNN, ResNet**

I. Introduction

Room cleanliness detection can be modelled as a typical computer vision task. In this project, we view this problem as a binary classification. For an input image with 3 RGB channels, we will determine whether it is dirty or tidy. To achieve that, 3 models are implemented, including SVM, CNN and ResNet. We compare the performance of our models based on training time, evaluation results and parameter tuning.

II. Method Review and Experimental Results

A. SVM

1) Method Overview: For Support Vector Machine(SVM) to apply on image-like input, we need to convert the image matrix into a flat vector form. And for each image, we have an associated label: clean or dirty. Thus we can use SVM to tackle this problem. Denote input as: $X = \{x_1, ..., x_n\}$, output as $Y = \{y_1, ..., y_n\}$ where n is the number of all data points. SVM aims to find a hyperplane to correctly classify the two sets. The goal function is given by:

$$\begin{aligned} &argmin_{w,b}\frac{1}{2}||w||_2^2 \\ &\text{s.t. } y_i(w^T x_i + b) \geq 1 \end{aligned}$$

This optimization problem can be transformed into dual form by KKT constraints. The dual form is given by:

$$\begin{aligned} argmin_\alpha \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j K(x_i \cdot x_j) - \sum_{i=1}^{n}\alpha_i \\ \text{s.t. } 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{n} y_i \alpha_i = 0 \end{aligned}$$

The $K(x_i \cdot x_j)$ is a kernel function. In this project, we implement three kernel modes: 'linear', 'rbf' and 'poly'.

2) Training phase: To minimize the dual form, the core idea is to use alternating minimization. Each time we only update one pair of (α_i, α_j) and set others as constants. We update this pair accoding to the following rule:
1. For a pair (α_i, α_j), fix α_j, minimize the function by finding α_i.
2. Fix α_i, minimize the function by finding α_j.
3. Repeat 1 and 2 until convergence.

3) Testing phase: We apply the trained model to the test set to derive the predicted label $\widehat{Y} = \{\widehat{y_1}, ..., \widehat{y_n}\}$. Then evaluate the accuracy.

4) Experiments: We test two hyper parameters: kernel mode and C, which is the punishment coefficient. The following is the test result:

SVM-kernel-C	SVM-poly-10	SVM-poly-1	SVM-poly-0.1
training time	15.91	14.95	14.59
accuracy	0.7	0.7	0.7

SVM-kernel-C	SVM-rbf-10	SVM-rbf-1	SVM-rbf-0.1
training time	14.75	14.94	14.97
accuracy	0.5	0.5	0.5

SVM-kernel-C	SVM-linear-10	SVM-linear-1	SVM-linear-0.1
training time	14.22	14.52	19.71
accuracy	0.6	0.6	0.6

Among all settings, the poly kernel performs the best. Obviously, changing C does not affect the final result, only resulting in various training time.

B. CNN

1) Method Overview: Convolutional Neural Network(CNN) is a commonly used method to deal with image classification problem. In our task, we need to let our model output a messy probability to represent the information encoded in the dataset. To achieve this goal, we design the neural network based on AlexNet structure. The input 3-channel matrix is firstly fed to a batch normalization layer for regularization. Then the data goes through two convolutional layers, each associated with a max-pooling layer. The output is linked to three fully connected layers to be mapped to the dimension of two. This final output is activated by softmax function to represent the probability of messy or clean:

$$\sigma(z_i) = \frac{exp(z_i)}{\sum_{i=1}^{n} exp(z_i)}$$

2) Training phase: To train the network we designed, a few techniques are introduced. The loss function is defined by Cross-Entropy Loss:

$$L = -(y \log p + (1-y) \log(1-p))$$

We update our model by Back Propagation. Its efficiency makes it feasible to use gradient methods for training multilayer networks, updating weights to minimize loss; In our settings, stochastic gradient descent is used.

3) Testing phase: After training our model and tuning the hyper parameters, we directly apply the model to the input image from the test set. Our model will generate a messy probability and use the max function to determine whether the room is tidy or not.

4) Experiments: We set the hyper parameters as following:

batch_size	10
repeat	1
shuffle	True
num_epochs	10
learning_rate	0.001

The result is shown as following:

No.	clean prob	messy prob	label
0	1.3393e-03	9.9866e-01	1
1	9.9437e-01	5.6297e-03	0
2	5.2131e-03	9.9479e-01	1
3	1.9273e-01	8.0727e-01	1
4	3.8652e-02	9.6135e-01	1
5	7.5680e-02	9.2432e-01	1
6	9.9999e-01	1.2187e-05	0
7	7.6671e-01	2.3329e-01	0
8	9.6782e-01	3.2175e-02	0
9	9.9996e-01	3.5226e-05	0

In the end, our model has only the accuracy of 0.6. This is actually worse than a finely tuned SVM. This shows that a shallow network is not enough for dealing with image-like information with tons of features. Thus we need to import more complicated network in order to perform better.

C. ResNet

1) Method Overview: Residual neural network(ResNet) is an artificial neural network (ANN) of a kind that builds on designs adapted from pyramidal cells in the cerebral cortex. Residual neural networks implement this by utilizing skip connections, or shortcuts to jump over some layers. Typical ResNet models are implemented with triple- layer skips that contain nonlinearities (ReLU) and batch normalization in between. In our implementation, we load a ResNet18 pretrained on the ImageNet dataset. The output is then mapped to the dimension of two through three fully connected layers to represent the two probabilities.

2) Training phase: Like CNN, we also use Cross-Entorpy Loss as loss function and Back propagation to update weights.

3) Testing phase: The parameters settings are as below:

batch_size	10
repeat	1
shuffle	True
num_epochs	10
learning_rate	0.001

4) Experiments: The result is shown as following:

No.	clean prob	messy prob	label
0	8.7576e-01	1.2424e-01	0
1	8.9349e-01	1.0651e-01	0
2	1.2975e-01	8.7025e-01	1
3	8.5477e-01	1.4523e-01	0
4	4.7249e-16	1.0000e+00	1
5	3.8632e-10	1.0000e+00	1
6	8.9842e-01	1.0158e-01	0
7	1.3818e-02	9.8618e-01	1
8	8.4371e-10	1.0000e+00	1
9	8.8990e-01	1.1010e-01	0

The generated label is the same as the ground truth. It demonstrates that ResNet fully capture the features within the dataset and thus can make inference based on the hidden information.

III. Comparison and Discussion

The following table shows the training time and accuracy among all models:

model	SVM-poly-10	SVM-rbf-10	SVM-linear-10
training time	15.91	14.75	14.22
accuracy	0.7	0.5	0.6

model	CNN	ResNet
training time	4.28	14.39
accuracy	0.6	1.0

Note that we train neural network on GPU such that it is accelrated. In reality, the computation cost for neural network is significantly larger than traditional machine learning methods like SVM. From these results, we can interpret that ResNet is the best model of all. It shows that deep neural network can clearly capture the hidden information within the dataset and can make good inference based on that. Besides, SVM also has the power of feature extraction and its result is equivalent to shallow network models with one or two convolutional layers.

IV. Conclusion

In our project, we implement three models to detect room cleanliness. Among all three models, ResNet is the most promising one, correctly classifying all test data points. All models require fine tuning parameters. Though neural network has a higher computation cost, it clearly captures all features provided in the training set.

References