Defining Public Scenes from Private Using MIT Indoor67

1st Tianrui Wang
CPE 695
BIA
Hoboken, US
twang55@stevens.edu

2nd Yating Liu
CPE 695
BIA
Hoboken, US
yliu260@stevens.edu

3rd Xing Fang
CPE 695
BIA
Hoboken, US
xfang5@stevens.edu

***Abstract*—In this paper, we mixed and matched several feature extraction methods and classifiers to separate public images from private ones. We compared the performance of image feature extraction tools, including Convolutional Neural Networks (CNNs), Scale-Invariant Feature Transform (SIFT), and VGG16. We also studied through several state-of-the-art classifiers, including Multilayer Perceptron (MLP), Support Vector Machine (SVM) and K-Means, Linear Discriminant Analysis (LDA), Random Forest (RF) to see which classifier works best with which feature extraction tool.**

***Index Terms*—Indoor Scene Classification, Transfer Learning, SIFT, CNN, VGG16, SVM**

I. Introduction

Images today are increasingly shared online on social networking sites, such as instagram, facebook, twitter; or stored online on cloud drives such as google photos and icloud [1]. For the users, it would be a cumbersome task to manage the privacy settings over thousands, or even millions of photos they stored or shared online.As for the web service providers, it is critical for them to maintain the data integrity and protect their users' privacy. Naturally, automatically identifying private scenes from public ones become an on-demand task.

In this paper, we compared three approaches to extract features from the images selected from MIT Indoor67 dataset, and then conducted binary classification using different classifiers including Multilayer Perceptron (MLP), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN).

We created a benchmark CNN model through trial and error of several different structures on our data set to extract features and did binary classification with MLP. Next we tried traditional scene classification methods, such as Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF), and Oriented FAST and Rotated BRIEF (ORB) to detect and describe the local features in images to derive extracted features, and used KNN and SVM as classifiers. At last, we used VGG16,a well-known CNN model proposed by [2] and pre-trained on ImageNet to extract features from our data set, and then classified using MLP and SVM.

Our work shown that the combined result of a pre-trained VGG16 for feature extraction, and a SVM classifier ranked first among the others for a test accuracy of 74.2%. The benchmark CNN model could reach an accuracy of up to 63.08%. And extracting features using SIFT would give use a test accuracy of up to 71%. Among all classifiers, SVM seems work well with both features extracted from CNN models, feature extracted from SIFT.

II. Related Work

Scene cognition has been an active research in the computer vision field. A great deal of researches has been done on indoor scenen classification. As mentioned by [3], "... The main difficulty is that while some indoor scenes (e.g. corridors) can be well characterized by global spatial properties, others (e.g, bookstores) are better characterized by the objects they contain." We found that public spaces such as airports, movie theaters, usually contain global spatial properties, as they were designed for specific purposes. On the other hand, small spaces such as stores, bedrooms, kitchens, etc, are not uniquely designed but are defined by the objects they contain. Thus, identifying public scenes from private ones can be fundamentally meaningful for indoor scene classification as it addresses the main difficulty of the problem.

Traditional scene classification methods like SIFT, SURF, ORB were based on manual vision features. Reference [4] utilized an improved SIFT feature named RootSIFT to build a BoW (Bag of Word) model and combined selective attention for scene classification. Reference [5] extracted fixed number of SIFT features, and a feature matrix consisting of the extracted SIFT feature vectors was used as input to CNN. In [6], SPM(spatial pyramid matching) was proposed on BoW to classify scenes.

Recent years, deep learning in image classification made great progress especially in CNN. In [7], the report extracted features by neural network and showed that neural network could learn patterns. In [8], accuracy of scene classification was improved by transferring learning. A pre-trained model like VGG16 has been well trained on large scale data set. Based on pre-trained model, adding a few more layers can obtain better accuracy than models trained from scratch.

III. Solutions

A. Dataset Description

MIT67 is a refined dataset for indoor scenes and has been used widely in previous research and literature in indoor scenes recognition. On top of the MIT Indoor67 dataset,we

manually picked the 45 categories and split them into public private labels based on the image characteristics, this results in over 10,000 images left for us to train and test. During the modelling phase, we used 6,000 images for training, 2,000 for validation and another 2,000 for testing. Fig. 1 is a clear illustration of all the sub-categories under public and private labels. Fig. 2 shown the relative proportion of the sub-categories inside each labels. 18 sub-categories are included in private scenes and 27 are included in public scenes.

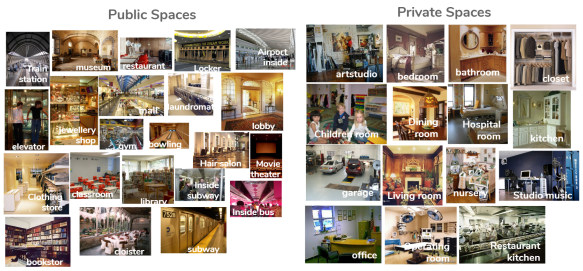

Fig. 1. Public Spaces & Private Spaces

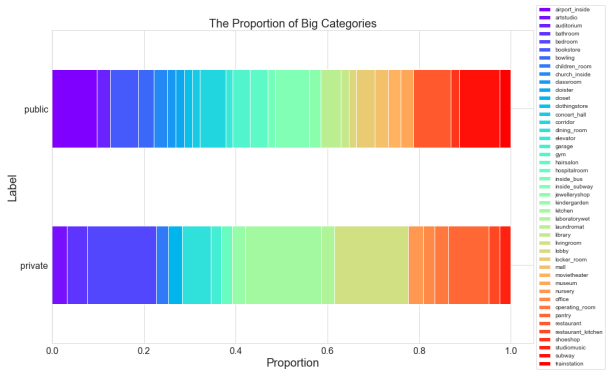

Fig. 2. Proportion of two big labels

B. Implemented Algorithms

1) Benchmark CNN Model - Tianrui:

a) Feature Extraction through different Convnet Architectures: We arbitrarily tested several simple structures of convnet models to see which structure could efficiently extract the features from our public vs private images. Initially, to compare the feature extraction power of the convnet, we controlled the effect of classifier by using a fully connected `Dense` layer with a size of 512 after the `Flattern` layer, end by a single unit and a `sigmoid` activation.

The first convnet was a stack of three `Conv2D` (with `relu` activation) and one `MaxPooling2D` layer [Fig. 3]. The second convnet was a stack of four alternated `Conv2D` and `MaxPooling2D` layers [Fig. 4]. After 30 epoches of training, we could see clearly that the later model results in a X% higher test accuracy and a much more efficient training process [Fig. 6].

It is also interesting to us that the training accuracy did not go up greatly over the 30 epochs for the first model [Fig. 5]. This implies that the model did not work for some reason. In order to improve the performance of this model, we had to adjust the classifier by adding a `Dropout` layer and adjusting the structure of the MLP classifier [Fig. 7]. After adjusted, we could see that the training is able to converge around 100% [Fig. 8]. Although this way we cannot control the effect of the classifier, it is worth to think about the effect of a `Dropout` layer and a two fully connected hidden layers versus one hidden layers. We infer that it is probably because one hidden layer is not enough to handle and adjust the weights of the extracted features from the convnet.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 148, 148, 32)	896
conv2d_2 (Conv2D)	(None, 146, 146, 32)	9248
conv2d_3 (Conv2D)	(None, 144, 144, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 36, 36, 32)	0
flatten_1 (Flatten)	(None, 41472)	0
dense_1 (Dense)	(None, 512)	21234176
dense_2 (Dense)	(None, 1)	513

Total params: 21,254,081
Trainable params: 21,254,081
Non-trainable params: 0

Fig. 3. First CNN Model

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 148, 148, 32)	896
max_pooling2d_1 (MaxPooling2	(None, 74, 74, 32)	0
conv2d_2 (Conv2D)	(None, 72, 72, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 36, 36, 64)	0
conv2d_3 (Conv2D)	(None, 34, 34, 128)	73856
max_pooling2d_3 (MaxPooling2	(None, 17, 17, 128)	0
conv2d_4 (Conv2D)	(None, 15, 15, 128)	147584
max_pooling2d_4 (MaxPooling2	(None, 7, 7, 128)	0
flatten_1 (Flatten)	(None, 6272)	0
dense_1 (Dense)	(None, 512)	3211776
dense_2 (Dense)	(None, 1)	513

Total params: 3,453,121
Trainable params: 3,453,121
Non-trainable params: 0

Fig. 4. Second CNN Model

b) Addressing Over-fitting through Image augmentation: After choosing to continue with the second model, we realized that there is severe over-fitting issue as the training accuracy could converge to approximately 100% after 30 epochs, but the validation accuracy is around 70% to 75% [Fig. 6]. With only 6,000 training data and 2,000 validation data, it is possible that the over-fitting is caused by having too few samples to learn from, resulting in the model unable to adjust to new data. To address this problem, we used data augmentation techniques by augmenting the samples via several random transformations on the original images. To further improve the model result and boost the efficiency, we added a `Dropout` layer before the MLP classifier [Fig. 9].

The result of image augmentation is convincing. Fig. 10 shows that, although the validation accuracy is very fluctuated, it could increase with the training accuracy as the level of

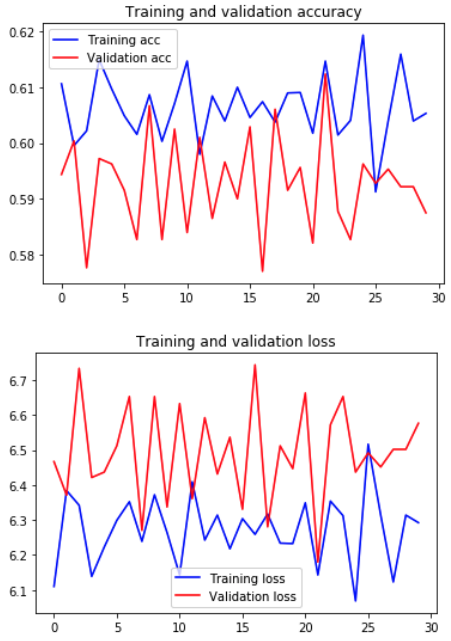

Fig. 5. First CNN Model: Accuracy & Loss Plot

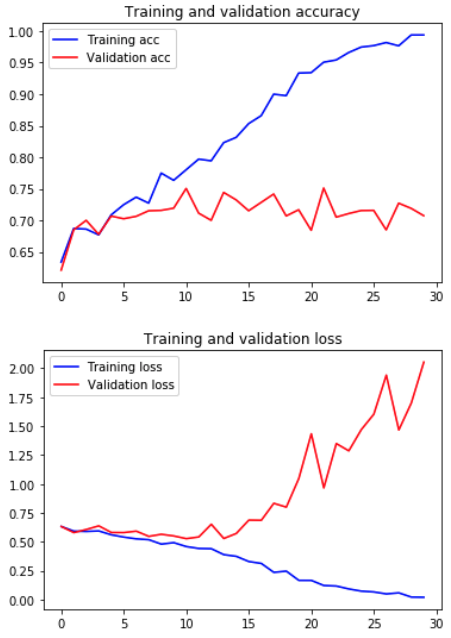

Fig. 6. Second CNN Model: Accuracy & Loss Plot

Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 148, 148, 32)	896
conv2d_8 (Conv2D)	(None, 146, 146, 32)	9248
conv2d_9 (Conv2D)	(None, 144, 144, 32)	9248
max_pooling2d_3 (MaxPooling2	(None, 36, 36, 32)	0
flatten_3 (Flatten)	(None, 41472)	0
dense_6 (Dense)	(None, 256)	10617088
dense_7 (Dense)	(None, 128)	32896
dropout_2 (Dropout)	(None, 128)	0
dense_8 (Dense)	(None, 1)	129

Total params: 10,669,505
Trainable params: 10,669,505
Non-trainable params: 0

Fig. 7. First CNN Model(adjusted)

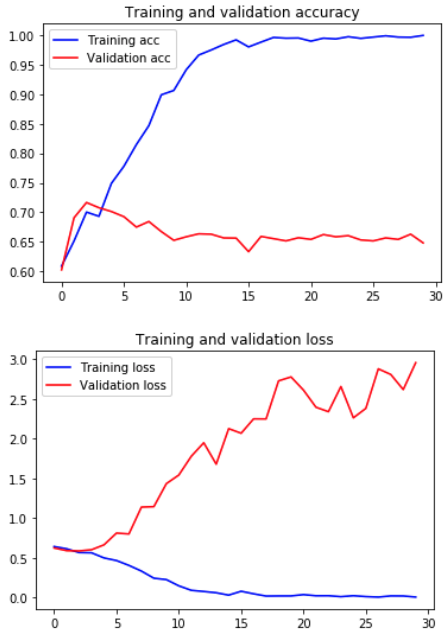

Fig. 8. First CNN Model(adjusted): Accuracy & Loss Plot

epochs goes further. In the end, we were able to achieve a same level of overall test accuracy of 65.55% for both before and after augmentation. This means that the over-fitting can be fixed greatly through image augmentation, and surprisingly, it does not sacrifice much of the accuracy as shown in Fig. 11 & Fig. 12.

2) SIFT - Xing:

a) Extract Features: SIFT And K-means(Bag of Visual Words)

At the very first, we tried three traditional different ways rather than CNN to extract features of images: SIFT, SURF, ORB. We took one picture to see key-points locations. By comparing of three key-points on same image as shown in Fig. 13, Fig. 14 and Fig. 15 and academic research, we finally chose SIFT to extract key-points.

After locating key points, direction and magnitude of picture gradient are calculated using key-point neighboring pixels. SIFT transform each image into 2 dimensions, like [x, 128].

Layer (type)	Output Shape	Param #
conv2d_25 (Conv2D)	(None, 148, 148, 32)	896
max_pooling2d_27 (MaxPooling	(None, 74, 74, 32)	0
conv2d_26 (Conv2D)	(None, 72, 72, 64)	18496
max_pooling2d_28 (MaxPooling	(None, 36, 36, 64)	0
conv2d_27 (Conv2D)	(None, 34, 34, 128)	73856
max_pooling2d_29 (MaxPooling	(None, 17, 17, 128)	0
conv2d_28 (Conv2D)	(None, 15, 15, 128)	147584
max_pooling2d_30 (MaxPooling	(None, 7, 7, 128)	0
flatten_7 (Flatten)	(None, 6272)	0
dropout_6 (Dropout)	(None, 6272)	0
dense_21 (Dense)	(None, 512)	3211776
dense_22 (Dense)	(None, 1)	513

Total params: 3,453,121
Trainable params: 3,453,121
Non-trainable params: 0

Fig. 9. Second CNN Model: after Image Augmentation & Dropout

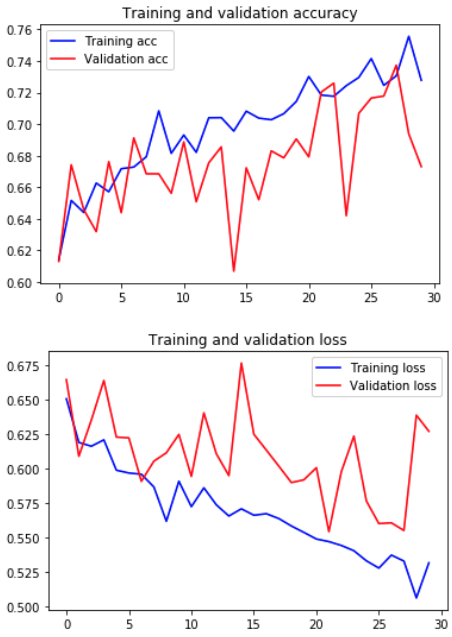

Fig. 10. Second CNN Model Augmented: Accuracy & Loss Plot

The accruacy score is 0.655500

	precision	recall	f1-score	support
0	0.68	0.57	0.62	989
1	0.64	0.74	0.68	1011
accuracy			0.66	2000
macro avg	0.66	0.65	0.65	2000
weighted avg	0.66	0.66	0.65	2000

Fig. 11. Second CNN Model: Test Accuracy

The accruacy score is 0.655500

	precision	recall	f1-score	support
0	0.76	0.17	0.28	989
1	0.54	0.95	0.69	1011
accuracy			0.56	2000
macro avg	0.65	0.56	0.49	2000
weighted avg	0.65	0.56	0.49	2000

Fig. 12. Second CNN Model Augmented: Test Accuracy

Fig. 13. SIFT

Fig. 14. SURT

Each image has different data size but all with fixed 128 dimensions. Then we used `pad_sequence` to set our data as [1000, 128].

In order to use the key-point descriptors in classification, a vector of fixed-size is needed. For this reason, K-Means is used to group all descriptors into a set of clusters which is bag of visual words. In this step, because of machine limitation, it took us about 5 hours, so we just selected 50 centers, no other trials. Then we used Random Forest to visualize the importance of 50 centers like Fig. 16.

After that, we calculated the distance between each descriptors and 50 centers. By doing this step, we transformed data into one dimension [50] which would fit into classifi-

Fig. 15. ORB

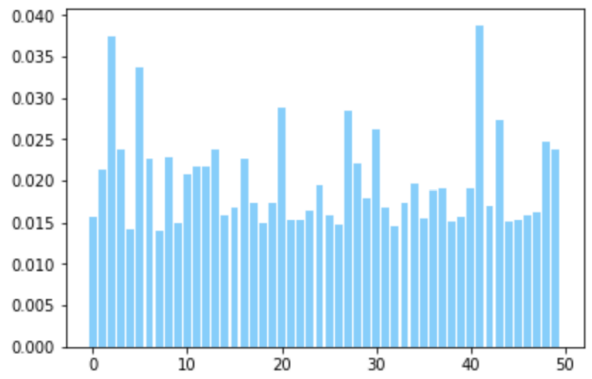

Fig. 16. Importance of 50 Centers

cation models. Finally to change values to a common scale without distorting differences in the range of values, we used `StandardScaler` to normalize all the data.

b) Classification Models: SVM & PCA + SVM

Support Vector Machine is supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. In our project, we chose SVM and used `GridSearchCV` to tune parameter of SVM model. We set C:[0.1,1,10,100,1000], GAMMA:[1,0.1,0.01,0.001,0.0001]. After training the model on training dataset, we put test data into trained model. Accuracy of SVM has arrived 71%.

Then we tried PCA to extract important features, as shown in Fig. 17. After number of 10, the line becomes stable, so we finally chose 20. Then we used PCA to transform all data, and put train data into SVM again with same `GridSearchCV` function. Accuracy of PCA + SVM reaches 68%.

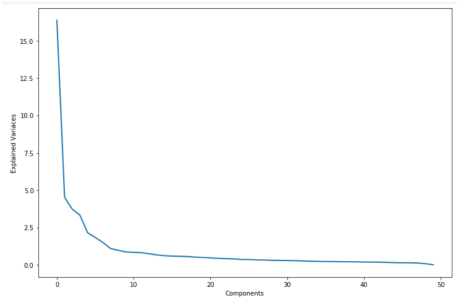

Fig. 17. PCA

c) Classification Models: LDA & PCA + LDA

LDA works fine in both dimensions reduction and in classifier. LDA basically separate example of classes linearly, but for non-linearly characteristic of data, LDA may not have good performance. In this part, we use LDA to see if linear classification model can also have better performance and also if images processed by SIFT and `KMeans` are kind of linear which may have deep meaning in the future work. Accuracy of LDA is 68%. Put PCA data into LDA model, accuracy goes down to 65%. We think that LDA itself has the function of dimensions reduction, so if we put PCA again which also plays the role of dimensions reduction, it will have negative effect on model accuracy.

d) Classification Models: Random Forest & PCA + Random Forest

Random forests algorithm is increasingly being used for image classification. Random forest is an ensemble model which means it calculate the results from many models which will have a better performance than individual models([9]). Especially in the case of random forests, several decision trees are made and the response is produced based on all decision trees.

In our project, we also used `GridSearchCV` to tune the parameters which including max depth, max features, min sample leaf, min sample split and estimators. Accuracy is 68%. Then we use PCA data to tune RandomForestClassifier again, then we got the accuracy of 67%.

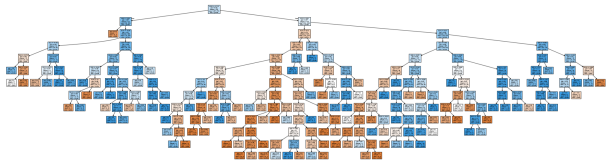

Fig. 18. Random Forest

e) Classification Models: Comparison

By comparing 3 models and 3 models with PCA, we concluded that SVM has the best performance. PCA can't help improve image classification accuracy. For SVM & LDA, they both have the function of dimension reduction, so applying PCA in the two models will decrease the accuracy. For another reason, we firstly chose 50 centers may be too small. In the future, we can use more centers, then we will see if it will change in accuracy.

TABLE I
ACCURACY COMPARISON

	SVM	*LDA*	*Random Forest*
Without PCA	***71%***	***68%***	***67%***
With PCA	***68%***	***65%***	***67%***

3) VGG16 - Yating:

a) Feature Extraction through VGG16 & Binary Classification SVM: Feature Extraction We tried two ways to import the image data and extract features. The first way is extracting the features through `VGG16`, then train the CNN model using the features. Using a pre-trained model makes deep learning very effective for small-data problems.

`VGG16` [Fig. 19] is a convolutional network model for classification and detection proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was one of the famous model

submitted to ILSVRC-2014. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another.

Model: "vgg16"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 150, 150, 3)	0
block1_conv1 (Conv2D)	(None, 150, 150, 64)	1792
block1_conv2 (Conv2D)	(None, 150, 150, 64)	36928
block1_pool (MaxPooling2D)	(None, 75, 75, 64)	0
block2_conv1 (Conv2D)	(None, 75, 75, 128)	73856
block2_conv2 (Conv2D)	(None, 75, 75, 128)	147584
block2_pool (MaxPooling2D)	(None, 37, 37, 128)	0
block3_conv1 (Conv2D)	(None, 37, 37, 256)	295168
block3_conv2 (Conv2D)	(None, 37, 37, 256)	590080
block3_conv3 (Conv2D)	(None, 37, 37, 256)	590080
block3_pool (MaxPooling2D)	(None, 18, 18, 256)	0
block4_conv1 (Conv2D)	(None, 18, 18, 512)	1180160
block4_conv2 (Conv2D)	(None, 18, 18, 512)	2359808
block4_conv3 (Conv2D)	(None, 18, 18, 512)	2359808
block4_pool (MaxPooling2D)	(None, 9, 9, 512)	0
block5_conv1 (Conv2D)	(None, 9, 9, 512)	2359808
block5_conv2 (Conv2D)	(None, 9, 9, 512)	2359808
block5_conv3 (Conv2D)	(None, 9, 9, 512)	2359808
block5_pool (MaxPooling2D)	(None, 4, 4, 512)	0

Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0

Fig. 19. VGG16 Model

Initially, we define a simple network based on `VGG16` as shown in Fig. 20. Then we test the model and get accuracy score as 0.71.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
flatten_1 (Flatten)	(None, 8192)	0
dense_1 (Dense)	(None, 256)	2097408
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 1)	257

Total params: 2,097,665
Trainable params: 2,097,665
Non-trainable params: 0

Fig. 20. Model 11

But from figures of history [Fig. 21], you can obviously notice that there is a far distance between training curve and validation curve, which means it is over-fitting seriously. Our training accuracy increases linearly over time, until it reaches nearly 100%, while our validation accuracy stalls at 70-72%. Our validation loss reaches its minimum after about fifteen epochs then stalls, while the training loss keeps decreasing linearly until it reaches nearly 0.

To reduce overfitting and try more layers, we add a `global average pooling` layer. `Global` pooling is mainly used to solve the problem of full connection, which is mainly to pool the feature graph of the last layer to form a feature point. Finally we get an accuracy which is increased to 0.73, and the overfitting has been relatively reduced.

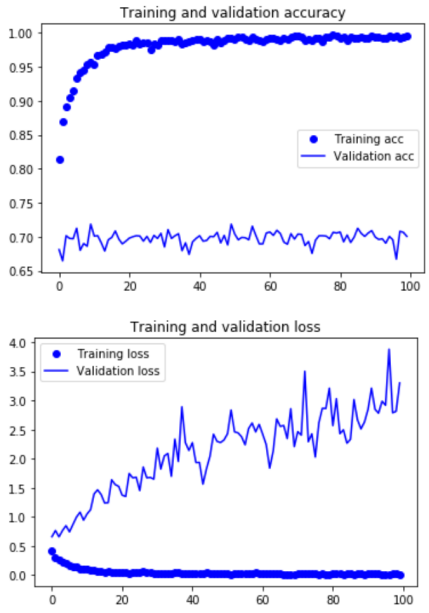

Fig. 21. Accuracy of Model 1

Model: "sequential_2"

Layer (type)	Output Shape	Param #
global_average_pooling2d_1 (	(None, 512)	0
dense_3 (Dense)	(None, 1)	513

Total params: 513
Trainable params: 513
Non-trainable params: 0

Fig. 22. Model 2

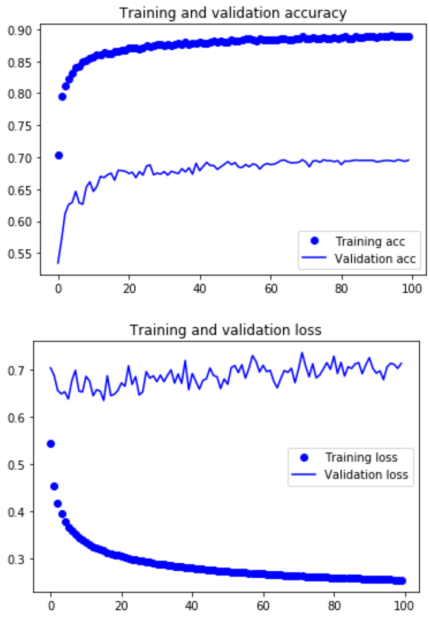

Fig. 23. Accuracy of Model 2

Binary Classification SVM At last, we apply SVM to classify by using features we extractedand tune parameters by GridSearchCV, then we get the highest accuracy score as 0.742. So in the future, we will try to combine the methods we used, continue to tune the parameters and find more ways to decrease overfitting.

```
GridSearchCV(cv=10, error_score='raise-deprecating',
       estimator=LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
     intercept_scaling=1, loss='squared_hinge', max_iter=1000,
     multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
     verbose=0),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid=[{'C': [0.01, 0.1, 1, 10, 100]}],
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
```

Fig. 24. GridSearchCV of SVM Model

b) Binary Classification MLP: The second way of loading image data is using ImageDataGenerator, it can preprocess the data and generate tensor image data batches. So using the generators, we trained the CNN model based on VGG16 and the accuracy score is 0.703 this time. This result is not as good as we thought, it may because that ImageNet categorized the image into precise categorizes, but our dataset contains only selected indoor scenes and we have a wider category for public and private data, which introduces noise. From Tianrui's study, we can clearly learn effective methods to reduce overfitting, which is image augmentation. So there is no need to apply it again.

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 4, 4, 512)         14714688
_________________________________________________________________
flatten_3 (Flatten)          (None, 8192)              0
_________________________________________________________________
dense_6 (Dense)              (None, 256)               2097408
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 257
=================================================================
Total params: 16,812,353
Trainable params: 2,097,665
Non-trainable params: 14,714,688
```

Fig. 25. MLP

IV. Comparison

As shown in Table II, we computed accuracy of all models. In function of extracting features, CNN has better performance than SIFT, especially when using pre-trained VGG16 model to extract features from the images. However, there exists a serious overfitting which can be effectively reduced by Image Augmentation. On the other hand in function of classification models, SVM achieved the best performance.

TABLE II
Comparison Accuracy of Models

Extract Features	*Models*	*Accuracy*
CNN	***MLP***	***61.61%***
SIFT	***SVM***	***71%***
SIFT	***LDA***	***65%***
SIFT	***RF***	***67%***
VGG16	***SVM***	***74.2%***
VGG16	***MLP***	***70.3%***

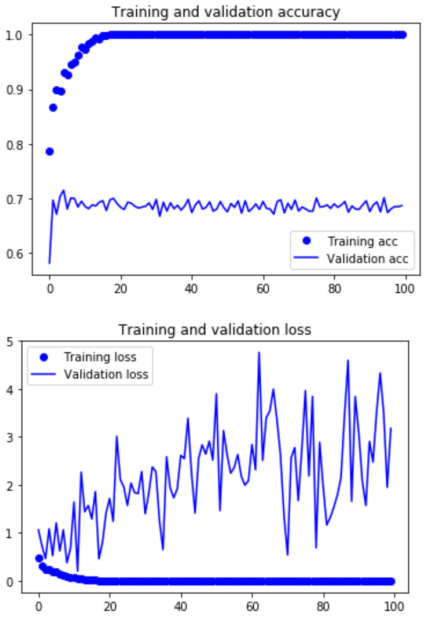

Fig. 26. Accuracy of MLP

V. Conclusions

In our project, we found that the pre-trained model VGG16 has the best performance in terms of features extraction. Although SIFT is an old fashion tool for image processing, it still works well beyond our expectation. It could be more effective to combine SIFT with CNN, which leads to possibilities of future development for this project. During the process of using CNN to train the models, severe over-fitting issue can be solved by applying image augmentation techniques in current situation where we have limited image data.

Currently, VGG16 cannot achieve higher accuracy when comparing to its powerful results on ImageNet. We think possible reasons include the differences in the structure of our dataset comparing to the ImageNet dataset. VGG16 is pre-trained on the ImageNet with thousands of precise categories, while our dataset separates a few precise scene categories that is defined as 'Public' from others that is defined as 'Private', which introduces noise for both classes.

We also found that SVM could improve classification accuracy when comparing to MLP, which is traditionally used after Convnet. It also works well with the features extracted from SIFT, resulted in the highest test accuracy of 71%, comparing to LDA and Random Forest. It is quite amazing to see that a simple SVM classifier could out-perform other classifiers such as Random Forest and MLP which requires more computing power. However, it is also reasonable since SVM is more efficient in high dimensional spaces and when number of dimensions is greater than the number of samples. We think the results implied that there is clear margin of separation between the public and the private classes.

VI. Future Work

- There are several approaches to improve the benchmark CNN model: 1. By fine-tuning the hyper-parameters of the convolutional layer; 2. By adding additional `Conv2D` + `Maxpooling2D` pair layers to the model; 3. By fine-tuning the hyper-parameters of MLP classifier for binary classification.
- As for the SIFT method, we could try hybrid model to concatenate SIFT with CNN using same input. In this model, put SIFT and CNN in one deep learning model. And we will see if this model can have better accuracy than just CNN individually. As the result of our project, SVM has the best performance in image classification, so we can also use this hybrid model to extract features, and put the output into SVM to see if it will have positive effect on classification.
- On top of the VGG16 model, we can try to utilize the power of `ImageDataGenerator` and Image Augmentation. We should definitely try to fine-tune the existing model, by frozening the topper blocks and fine-tuning the last block to make the model fit to our classification purpose.
- We can also find ways to decrease over-fitting and classify images more precisely.

References

[1] A. Tonge and C. Caragea, "Image privacy prediction using deep neural networks," *arXiv preprint arXiv:1903.03695*, 2019.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 413–420.

[4] J. Niu, X. Bu, K. Qian, and Z. Li, "An indoor scene recognition method combining global and saliency region features," *Robot*, vol. 37, no. 1, pp. 122–128, 2015.

[5] T. Zhang, W. Zheng, Z. Cui, Y. Zong, J. Yan, and K. Yan, "A deep neural network-driven feature learning method for multi-view facial expression recognition," *IEEE Transactions on Multimedia*, vol. 18, no. 12, pp. 2528–2536, 2016.

[6] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 2169–2178.

[7] P. Khorrami, T. Paine, and T. Huang, "Do deep neural networks learn facial action units when doing expression recognition?" in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 19–27.

[8] T. Akilan, Q. J. Wu, A. Safaei, and W. Jiang, "A late fusion approach for harnessing multi-cnn model high-level features," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017, pp. 566–571.

[9] C. Dahinden and M. ETHZ, "An improved random forests approach with application to the performance prediction challenge datasets," *Hands-on Pattern Recognition, Challenges in Machine Learning*, vol. 1, pp. 223–230, 2011.