Scene Parsing and Room Classification with Deep Learning

Ahmet Tarik KAYA
21527156
ahmet.tarik.kaya@gmail.com
Hacettepe University
Computer Engineering

Ayca Meric CELIK
21627103
aycameric.celik@gmail.com
Hacettepe University
Computer Engineering

Kaan MERSIN
21527214
kaanmersin07@gmail.com
Hacettepe University
Computer Engineering

Abstract

Machine learning displaces human minds on analyzing and evaluation day by day. We believe that labeling the room in the image is not a task that the human mind should perform. In today's world, machines can examine an image. Machines can detect different objects in the image and machines can even tell what objects are these and show the exact location with outliers. This can be achieved with scene parsing based on semantic segmentation. It is not just that, machines can also analyze data in a way which is much more efficient than a human mind. Machines can learn from those analyses and use that information on decision making. So, why bother with classifying the room by ourselves when machines are capable of doing those operations, especially when they can be more efficient on doing so?

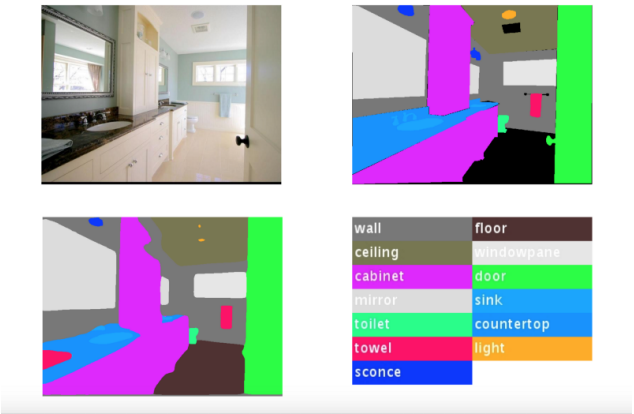

Figure 1. Parsing a scene of a bathroom. Every object in the bathroom is located and labeled.

1. Introduction

In today's world, many technologies are getting more and more effective with the aid of more information every single day. Computers are better than ever at making inquiries with machine learning, and considerably better and more efficient than humans. A project which classifies the room by images could provide useful information for some other applications. Humans would make more accurate predictions about what is the room's type based on an image. However, it would not be nearly as fast as computers. Thus, answering the question "Which room is this image from?" can be a useful task for machine learning. The models can be used on real estate business or managing digital galleries.

We will use the object information of images to predict their room type. In order to achieve that information, we need to detect the objects in the scene. This is one of the problems of scene parsing, which is a fundamental topic in computer vision

We need to know different object types in an image to predict room types, so our operations will be based on semantic segmentation. The goal of semantic segmentation is to assign each pixel in the image a category label. At the end of our scene parsing process, we will have predictions for the label, location and shape for each element in the scene.

Scene parsing frameworks are usually based on deep convolutional neural networks (CNN) and fully convolutional networks (FCN). The deep convolutional neural networks based methods are better at dynamic object understanding but issues like diverse scenes and unrestricted vocabulary results in the fact that most advanced scene parsing frameworks are usually based on fully convolutional networks. That's why we are going to implement our model based on FCN. Since creating a scene parsing framework is a very difficult task, we will shape our model based on other FCN models created before.

2. Related Work

There are two kinds of image datasets for scene parsing problem: fully-labeled and partially labeled. Different problems had arisen, so different approaches have been developed by researches for partially labeled images. We mainly interested in parsing fully labeled images. There are various datasets which are made of them. We are using ADE20K dataset[9] for our research. It contains 20,000 indoor and outdoor images labeled with 150 objects. There are several open-source state-of-the-art models built upon this dataset in PyTorch. That models are Multi-Scale Context Aggregation model, PSPNet, UPERNet, AlexNet, ResNet and MPF(Multi-Path Feedback model).

Multi-Scale Context Aggregation is an open-source state-of-the-art model that uses the dilated convolution network of Yu and Koltun.[7]The goal of this model is to compute discrete or continuous label for every pixel in the image. Dilated convolution is a convolution which is applied to input with defined gaps. The biggest advantage of dilated convolution in this model is, dilated convolution can exponentially expand a chosen field without losing resolution.

PSPNet[8] is an open-source state-of-the-art model that produces good quality results in scene parsing. In PSPNet, the first step is a CNN to get the features map. Then, to get the sub-region views of the features maps, a pyramid parsing module is applied. Finally, to get the final per-pixel predictions, the model uses a convolution layer. This model is the first model, that uses Pyramid Pooling Module(PPM) to get the multi-scale contextual information for a scene.

UPerNet is an open-source state-of-the-art model for scene parsing of images in ADE20K dataset.[10] UPerNet developed for adopting architectures Feature Pyramid Network (FPN) to incorporate multi-scale context more efficiently. The most significant work in UPerNets development was on batch normalization. The authors discovered that if a network is trained with batch normalization, only by a sufficiently large batch size of batch normalization can the network achieves state-of-the-art performance.

AlexNet is an open-source state-of-the-art-model for classifying 1.2 million high-resolution images in ImageNet LSVRC-2010 into the 1000 different classes[5]. In AlexNet, the neural network that was used has five convolutional layers, 60 million parameters and 650,000 neurons. Five convolutional layers contain three fully-connected layers, one max-pooling layer and a final layer that has 1000-way softmax in it. This model was a huge success because, with AlexNet, authors achieved considerably better results than the previous state-of-the-art models.

ResNet is an open-source state-of-the-art-model for a residual learning framework to ease the training of networks which are substantially deeper than those used previously.[6]ResNet uses ImageNet dataset. Before using ResNet model, authors were trying to make scene parsing and classification in ImageNet dataset and different than other VGG nets, they worked with a depth of up to 152 layers(which is 8 times deeper than other VGG nets) but the results had very low complexities. An ensemble of these residual nets achieves 3.57 percentage error on the ImageNet dataset. Which is the lowest error on ImageNet dataset. So, the residual logic in ResNet is very successful.

3. The Approach

Since our aim is to get object information from the segmented image, we selected ADE20K as our dataset. We searched for models which are trained on this dataset, and the one which impressed us the most was PSPNet. Thus, we decided to use it as our base model.

3.1. Our Dataset: ADE20K

ADE20K is one of the most popular image datasets, which is used for object detection and scene parsing projects. It contains RGB images of various indoor and outdoor scenes, their segmentation masks for both objects and parts, and object annotations text file. It is a wide dataset, which contains 20210 scenes for training and 2000 scenes for validation. Scenes were annotated using the LabelMe interface by a single person. There are 150 stuff/object category labels and 1,038 image level scene descriptors in the dataset.

3.2. A Brief Look At Scene Parsing PSPNet

Nowadays, fully convolutional network (FCN) is the base of most of the state-of-the-art scene parsing frameworks. Convolutional neural network (CNN) based methods give better results in object understanding, but the confusions still exist on diverse scenes. For example, the model might mistake a lighthouse with a tower due to similar appearance. However, if the model knows it is a seashore scene, it can differentiate objects more correctly. Its prediction will more likely to be a lighthouse.

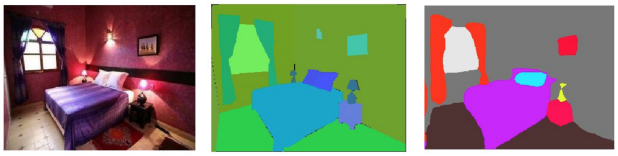

Figure 2. From left to right, original image from ADE20K, masked image from ADE20K, masked image created by PSPNet

Researchers concluded that fully convolutional networks need to have strategies to detect the global scene context to solve this problem. As a famous example, spatial pooling module is widely used to get the global image-level feature. Another approach is the one we employed for our project, pyramid scene parsing network (PSPNet). The idea behind is to fuse local and global clues in one pot and make predictions with it. PSPNet has a special design, which is called pyramid pooling module, to extend pixel-level features which are resulted from dilated FCN.

Common Issues on Complex-Scene Parsing

- Mismatched Relationship: There is a high chance of misclassification when there is a lack of contextual information. The model might mistake a pigeon in the air with a duck only based on their similar appearance, but we know that ducks cannot fly like pigeons.
- Confusion Categories: There are many class labels with a similar appearance, such as mountain and hill. It might cause multiple predictions on a single object, such as a house is labeled with both ”house” and ”building”. Thus, the relationships/hierarchy between the labels is very important to solve this problem.
- Inconspicuous Classes: Scene contains various objects of different size. Smallest objects could be the greatest indicators but easily misclassified if there is not enough information about the global scene category.

In summary, all these problems are linked with the global information and the contextual relationship. PSPNet's pyramid pooling module provides a great solution to these issues.

Pyramid Pooling Module makes PSPNet special. It helps with problems mentioned above because it provides much better sub-region observations. Using its 4 level pyramid, Pyramid Pooling Module extracts feature maps for the whole, half, 1/4 and 1/8 of the image with the usage of pretrained ResNet model. For each pyramid level, a 1x1 convolution layer is used in order to prevent variances in the weight of global feature. After those operations, pyramid levels result in global prior. Ultimately, global prior is concatenated with final feature map from pyramid pooling module and with a convolution layer, it results in the final prediction map.

3.3. Our Implementations

We had issues with training our PSPNet module by ourselves due to inabilities on hardware and software technologies. As a solution for it, we used pre-trained data of PSPNet[1] on ADE20K in order to make evaluations on the test images.

As for the object detection part, even though we were able to get the segmented image by running PSPNet, we were not able to get information on which objects the image have explicitly, such as console or file output. In order to achieve that, we made some adjustments in the source code of PSPNet. Since PSPNet provides a framework based on pixel-level prediction, it generates segmentation masks by using pixel labels. We extracted the data we need from the part where the segmented image was not even created.

PSPNet predicts the label of pixels, decodes those labels to get the corresponding color of the label, and creates a segmented image with all these colors. We filtered unique labels from all pixels in decoding function, and get values(color codes) of those labels. We have already created a colors-annotations map of ADE20K beforehand, so we got the corresponding object names by their color values. This way, we achieved information on which objects are detected in the image and wrote it on a text file.

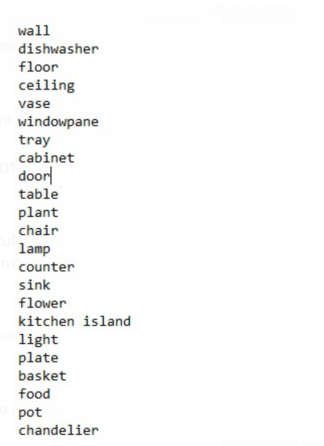

Figure 3. Example .txt file of an image from our PSPNet Implementation

After we got the object information from scenes, we tried three different machine learning models to predict room label: Random Forest Classifier, K-Nearest Neighbors Classifier, and Multinominal Naive Bayes Classifier.

Random Forest[2] is one of the most used machine learning algorithm, which is used for classification and regression. Even though Random Forest is working without hyper-parameter tuning, the results of it are spectacular in many different cases.

The k-nearest neighbors algorithm[3] is a non-parametric method used for classification and regression. k in the kNN stands for the number of nearest neighbors that the method it will look while the classification or regression

process. From those neighbors that will be looked, the most frequent class will be assigned to the data which the method is testing.

Naive Bayes[4] is a classifier based on applying Bayes theorem with strong independence assumptions between the features. In other words, Nave Bayes algorithm is making predictions of conditional independence between the training dataset.

4. Experimental Results

Our experiment and observation part of the project mainly contains two steps. In the first step, we used the human-written object annotation files of scenes of ADE20K for training and prediction. The reason for that was to get baseline accuracies. We tried different machine learning models and tuned the hyperparameters to increase accuracy. In the second step, specialized our PSPNet model and got the object annotations, we trained different models with these outputs and tuned hyperparameters. That was our desired final model.

Our dataset contains 1389 bedroom, 671 bathroom, 652 kitchen, 697 living room, and 412 dining room scenes. We started with creating a dataframe, where each row entry stands for an image and each column for an object name. Also, we have an additional column for the room label as target values for our model. Then, we split the dataframe into train and test frames and fed the models with them. The classifiers which we selected for this mission are Naive Bayes, kNN and Random Forest classifers.

	wall	floor	chair	door	screen	seats	cabinet	curtain	table lamp	pendant lamp	...	onions	onion	pinecones	sliding	generator	controls	ceiling decor	tomato
1	3.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	3.0	1.0	7.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	2.0	1.0	5.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	3.0	1.0	8.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	3.0	1.0	4.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 1022 columns

Figure 4. First Five Rows of our Dataframe

```
001 # 0 # 0 # wall # wall # ""
002 # 0 # 0 # wall # wall # ""
003 # 0 # 0 # wall # wall # ""
004 # 0 # 0 # floor, flooring # floor # ""
005 # 0 # 0 # floor, flooring # floor # ""
006 # 0 # 0 # wall socket, wall plug, electric outlet, electrical outlet, outlet, electric receptacle # outlet # ""
007 # 0 # 0 # toilet tissue, toilet paper, bathroom tissue # toilet paper # ""
008 # 0 # 0 # towel # towel # ""
009 # 0 # 0 # ashcan, trash can, garbage can, wastebin, ash bin, ash-bin, ashbin, dustbin, trash barrel, trash bin # trash can # ""
010 # 0 # 1 # toilet, can, commode, crapper, pot, potty, stool, throne # toilet # ""
011 # 0 # 0 # hair dryer # hair dryer # ""
012 # 0 # 0 # stool # stool # ""
013 # 0 # 0 # countertop # countertop # ""
014 # 0 # 0 # sink # sink # ""
015 # 0 # 0 # toilet paper holder # toilet paper holder # ""
016 # 0 # 0 # mirror # mirror # ""
017 # 0 # 0 # mirror # mirror # ""
018 # 0 # 0 # bidet # bidet # ""
001 # 1 # 0 # bowl # bowl # ""
002 # 1 # 0 # faucet # faucet # ""
003 # 1 # 0 # faucet # faucet # ""
```

Figure 5. Example .txt file of an image

First machine learning method which we tried is kNN(k-Nearest Neighbors). There are 5 hyperparameters we tuned.

- ”k” parameter stands for the number of neighbors the model evaluates in prediction. We tried different values of k from 2 to 10 to tune every other hyperparameter as well. 7 was the best k value among others.
- ”Weight” parameter is for determining the weight calculation for predictions. We observed that choice of ”distance” gives better accuracies, which is also the default choice.
- ”Parameter” argument, which equals ”auto” in default, is for computing the nearest neighbors. We observed that kdtree choice gives the highest accuracies, and we selected it.
- ”Metric” parameter, which equals minkowski in default, is for determining the tree structure which will be used in kNN. We observed that the best choice of it is manhattan, and we selected it.
- ”Leaf size” parameter, which equals 30 in default, is for setting the memory required to store the tree. We decided on to leave it as default after the accuracies did not change too much.

As a result, we got the maximum test accuracy as 92.67% all the parameters we tuned.

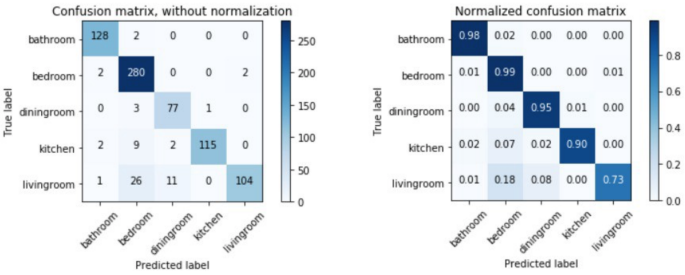

Figure 6. Confusion matrix with and without normalization of kNN

Second machine learning method we tried is the Multinominal Naive Bayes Classifier. There are 2 hyperparameters we tuned.

- ”alpha” parameter decides whether Naive Bayes makes smoothing or not.
- ”fitprior” parameter decides whether Naive Bayes learns class prior probabilities or not.

After our observations, we decided to leave this parameter as default ones. We got the maximum test accuracy as 96.7%.

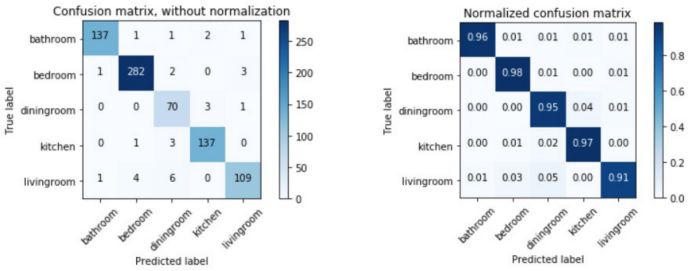

Figure 7. Confusion matrix with and without normalization of Naive Bayes

The last machine learning method we tried is Random Forest Classifier. After our observations, we decided to leave this parameter as default ones since they do not make significant changes. We got the maximum test accuracy as 96.8%.

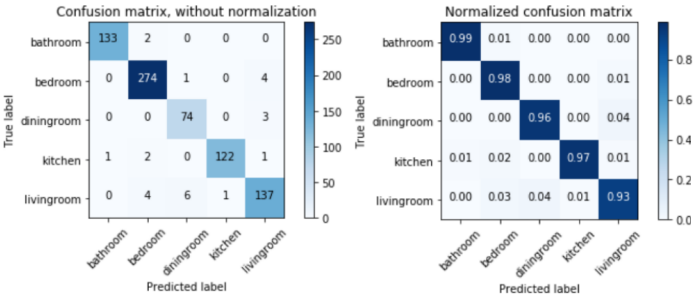

Figure 8. Confusion matrix with and without normalization of Random Forest

After working on each classifier, the common cause we observed which decreases the accuracy is living room scenes. We concluded the reason is living rooms do not contain so many significant objects. Objects in a living room can be easily found in any other room. As we see in confusion matrices of all models, the room label with the lowest accuracy is always living room.

When we train and test these three models with object information obtained from ADE20K dataset, the results we obtained were considerably higher than the object information obtained from our PSPNet model. There are two main reasons for this and both causes the loss of information:

- Less Object Labels: PSPNet works with only 150 different object types. ADE20K object annotation files we used on our models contained almost 9 times more than that, 1022 different object labels.
- Existence vs. Frequency: ADE20K object annotation files allowed us to calculate frequencies of objects in that scene. Our implementation with PSPNet only allowed us to obtain information about the existence of the objects. Thus, one of them was a frequency data while the other one was the binary data.

The highest accuracy we obtained from kNN method by setting the hyperparameters as we tuned is 90.28%.

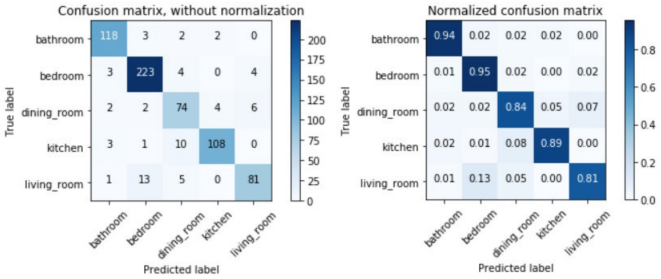

Figure 9. Confusion matrix with and without normalization of kNN with our PSPNet Implementation

The highest accuracy we obtained from Naive Bayes method is 91.03%.

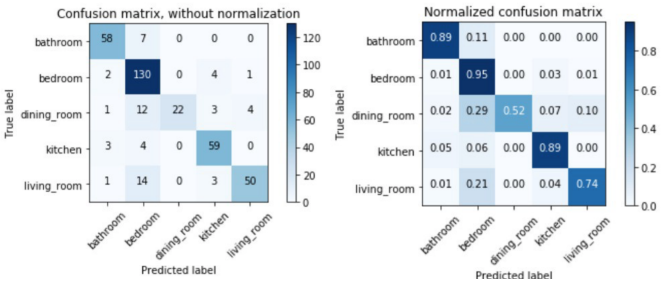

Figure 10. Confusion matrix with and without normalization of Naive Bayes with our PSPNet Implementation

The highest accuracy we obtained from Random Forest is 89.08 percentage.

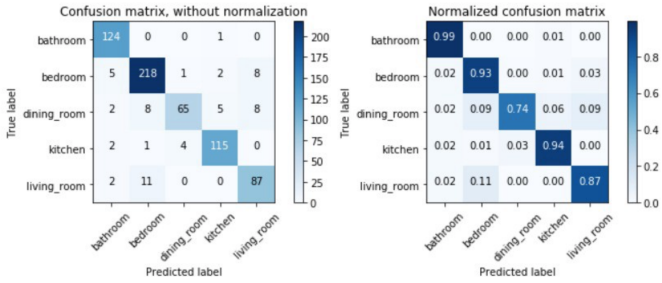

Figure 11. Confusion matrix with and without normalization of Random Forest with our PSPNet Implementation

5. Conclusions

We have completed an effective object detection and room classification model. Our project has two parts. The first part is the semantic segmentation of scenes and the second part is the room classification by objects. In the first part, we segmented the scenes and collected object information with our specialized PSPNet implementation. By looking at the results, our PSPNet is not able to detect and label items as successful as humans. However, we concluded that classifying the rooms by their items in it is a great method to follow. The accuracies we got from both preprepared and segmented object data are satisfying. There can be various methods, which are able to predict room types by images, and gives far better results. Yet, we are happy to come up with an inspiring alternative.

References

[1] https://github.com/hellochick/PSPNet-tensorflow/.

[2] https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd.

[3] https://towardsdatascience.com/k-nearest-neighbours-introduction-to-machine-learning-algorithms-18e7ce3d802a.

[4] https://towardsdatascience.com/naive-bayes-in-machine-learning-f49cc8f831b4.

[5] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, M. Hasan, B. C. V. Esesn, A. A. S. Awwal, and V. K. Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *CoRR*, abs/1803.01164, 2018.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[7] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[8] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017.

[9] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[10] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, pages 1–20, 2016.