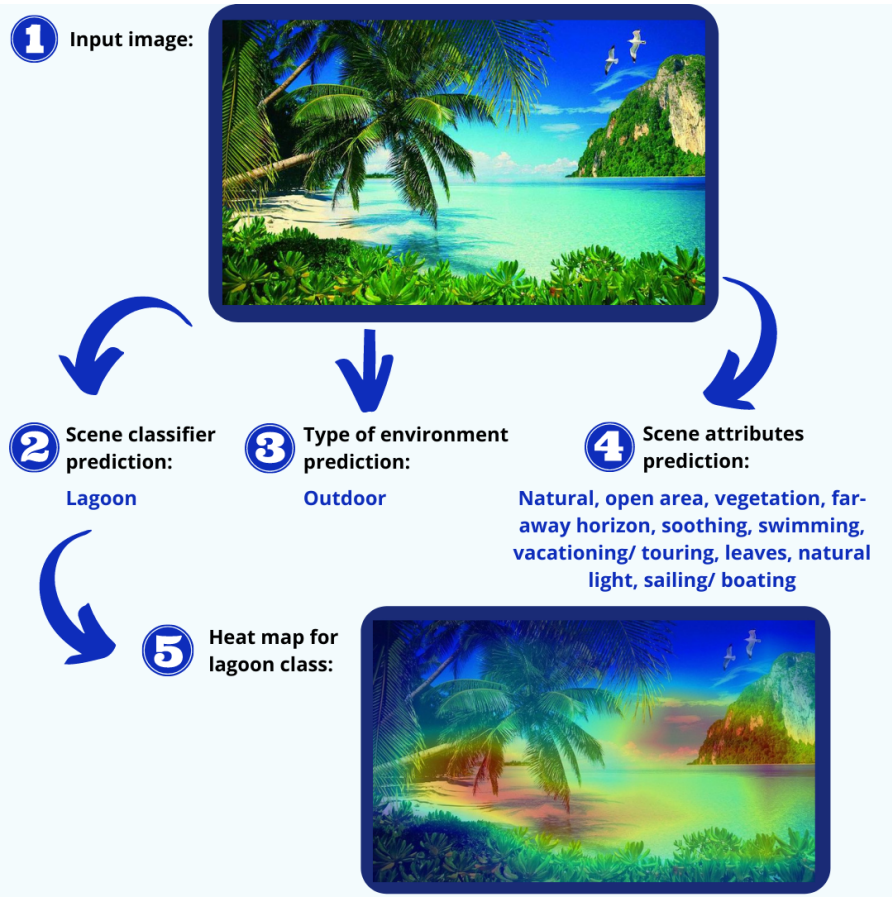
Scene recognition models and their deployment online

Team: Sandip Sonawane (sandip2) and Nursultan Baitlessov (nb17)

Executive summary: The main idea of this project is to supplement object detection models by predicting the scene or context of an image in a simple and concise way. Even though object detection models can provide a description of spatial relationship between objects, the descriptions are usually long and complex. For this problem, we have implemented the classification models that predict the scene of the image and the scene attributes present in the image. We also provide the heat map that shows the most important regions for the prediction of the scene classifier. The most surprising finding of this project was that our multi-label classifier has reached a high accuracy, which shows that modern CNN models are very powerful in classification task. Our DenseNet model pre-trained on ImageNet has reached a 95% mAP score for predicting the presence of attributes in the image. Finally, we have built and deployed the web application so that the models can be tested in real time. You can find the diagram below that shows what are the outputs of our models for the sample image.



1. Introduction

In recent years, many advanced methods to detect objects in an image have come into existence, but object detection models still miss to predict an important detail of an image - the context or a place in which the objects are shown. It is possible to provide a description of spatial relationships of the objects in an image, but predicting a scene category for an image can help to avoid producing such long and complex descriptions. We believe that predicting the scene or a context of an image should be very useful for an intelligent system to extract the same amount of information from an image as a human can do. For this reason, we found this problem particularly interesting to explore and in this project we have used the ideas, algorithms and databases provided by Bolei Zhou et al. in “Places: A 10 million Image Database for Scene Recognition” research paper.

In this paper by Bolei Zhou et al., the authors described how they have constructed the Places database and built and tested the latest CNN models for a scene classifier. They have found that the ResNet model has performed best on the Places-365 database, so we have decided to use their pre-trained model for the scene classification task. Even though it is not mentioned in the paper, their codebase has also used another classifier trained on the SUN Attribute database for the task of getting top-10 scene attributes of an image. For this multi-label scene attribute classifier, they have used the pre-trained weights but did not provide any code for implementation of this model. Therefore, we built and trained our own model to predict the top-10 highest-ranked scene attributes of an image. For this task, we have trained the pre-trained DenseNet with 201 layers on SUN Attribute Database and achieved about 95% mean Average Precision (mAP) score on the testing test. Apart from these two models, this paper’s codebase contained code for the heat map generator, so we have also implemented it to show what parts of the image were the most important for predicting the scene of an image. Afterwards, we have built a web application that incorporates the scene classifier, multi-label scene attributes classifier and a heat map generator for making scene predictions for an uploaded image. We have also deployed the web application online and it is possible to test it by uploading your own images.

2. Details of the approach

2.1 Data transformation and data loader for the SUN Attribute database

We have started from implementing the multi-label classifier for predicting the scene attributes present in the image. This classifier was trained on the SUN Attribute database, which contains information about presence of 102 scene attributes (e.g. dirty, natural light, ocean, etc.) in each of the 14340 images. One of the tables that we had in the database contained 14340 rows corresponding to images and 102 columns corresponding to scene attributes and this table had continuous values between 0 and 1, which correspond to how often a given attribute was voted as being present in a given image by Mechanical Turk Workers. These continuous values are calculated from 3 votes given by the AMT workers for each image. We decided that at least 2 votes for a particular attribute are enough to claim that the image contains this attribute and it also helped to avoid the sparse attribute vectors for images, which was the case when only 3 votes for an attribute were selected to claim its presence in the image. Thereby, we have one-hot encoded this table with the following condition: if the value was greater than 1/3 or if the attribute had at least 2 votes it became 1 and otherwise it became 0.

We have shuffled data from the database and divided it into training (80% of data), validation (10% of data) and testing (10% of data) sets. Afterwards, we have implemented our

own DataLoader class to iterate through the images and their attribute labels. The images were transformed to have a width and height of 227 pixels and the RGB values of an image were normalized using the means and standard deviations of the RGB values from ImageNet.

2.2 Training and fine-tuning the multi-label scene attribute classifier

For training the multi-label scene attribute classifier, we have decided to use the pre-trained DenseNet with 201 layers. DenseNet has many compelling advantages over other Convolutional Neural Network (CNN) models, such as reduced risk of having a vanishing gradient problem, reduced number of parameters, strengthened feature propagation and it encourages feature reuse. Afterwards, we have loaded the pre-trained DenseNet model and have changed the last linear layer of the model to output 102 predictions corresponding to the number of the scene attributes. We have chosen multi-label one-versus-all loss based on max-entropy for the computation of the losses on the predicted and target values. The formula for calculating the loss function is given in **Formula 1**, where x is a predicted value and y is a target value of size (N, C).

$$loss(x, y) = -\frac{1}{C} * \sum_i y[i] * log((1 + exp(-x[i]))^{-1}) + (1 - y[i]) * log(\frac{exp(-x[i])}{1+exp(-x[i])}),$$

$$where\ i \in \{0, \ldots, x.nElement() - 1\}\ and\ y[i] \in \{0, 1\}$$

Formula 1. Multi-label one-versus-all loss function

For minimizing our loss function, we have used a Stochastic Gradient Descent (SGD) with momentum optimizer. We have chosen SGD over Adam optimizer, since SGD generalizes better than Adam for a greater number of epochs. To measure the overall accuracy of our model, we have used the mean Average Precision (mAP) metric. After setting the model, loss function and optimizer, we have performed hyperparameter fine-tuning to get the hyperparameters that give the best accuracy or mAP score. Following this, we have saved our best performing trained model's weights so that the predictions of image attributes could be made in real-time for our web application.

2.3 Implementing the pre-trained scene classifier

In "Places: A 10 million Image Database for Scene Recognition" paper by Bolei Zhou et al., the authors have trained the scene classifier on Places-365 database (the database has 305 scene categories) using AlexNet, GoogleNet, VGG and ResNet and they have reached the highest accuracy for ResNet model predictions (85.08% in top-5 accuracy, the percentage of testing images where the ground-truth label is among the top ranked 5 predicted labels given by an algorithm). Therefore, we have decided to use their pre-trained ResNet model, provided in their codebase, for a scene classification task. We did not train our own ResNet classifier due to the limited computation power that we own, since the Places-365 database has 1.8 million images with a total size of 125 GB.

2.4 Predicting the indoor or outdoor type of environment for an image

Along with predicting the scene category for an image, we also predict whether the type of environment of an image is indoor or outdoor. We do not use any classification model for this task. Each of the 365 scene categories that are provided by Places-365 database have an indoor or outdoor macro-classes. We have calculated the indoor/outdoor score for an image based on the macro-classes of the top 10 scene category predictions. To make this calculation possible, we have encoded scene categories with indoor class as 0 and with outdoor class as 1.

By averaging these values for the top-10 predicted categories, we predicted that an image has an indoor environment if the score was less than 0.5 and the image has an outdoor environment otherwise.

2.5 Implementing the heat map generator

We have also implemented the heat map generator from the codebase provided by Bolei Zhou et al. The heat map is generated using the class activation maps technique, where class activation maps are generated using the global average pooling (GAP) in CNNs. You can find the diagram for the procedure of generating the class activation maps in **Figure 1**, which was also provided by Bolei Zhou et al. in the "Learning Deep Features for Discriminative Localization" paper. A class activation map for a separate scene category shows the important regions of the image for identifying this category. Thereby, the heat map will show which regions of the image were the most important for a scene category prediction with the highest score.

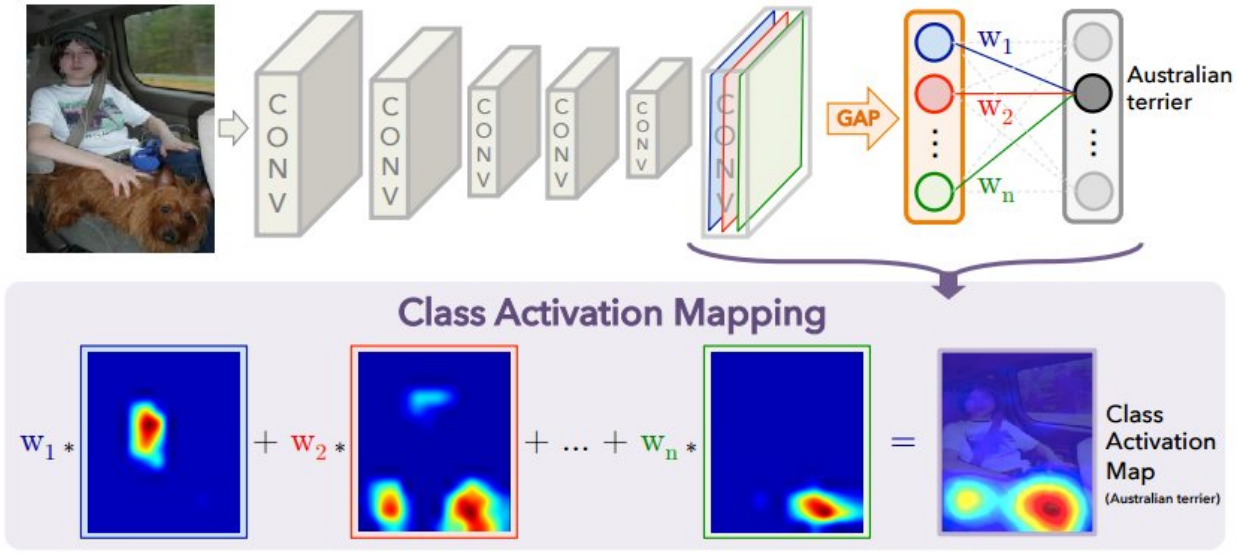


Figure 1. Procedure for generating the class activation maps

2.6 Building the web application

After implementing all of the classification models, we have built a web application to incorporate these models and deployed it online by using the following technologies: i) Python, ii) Django Framework, iii) Django Dash, iv) CSS, v) HTML, vi) JavaScript, vii) AWS. We started with an initial sketch of the user interface. You can find an initial sketch for our web application in **Figure 2,** which shows the inputs and outputs for our model. At the top left of the page, the user can upload an image to run our classification models on it and see the results on the right. The user has an option to upload an image from the local directory or use the link for a publicly available image. If the user does not have an image on the local machine, they can test our models on one of our sample images on the bottom left part of the page.

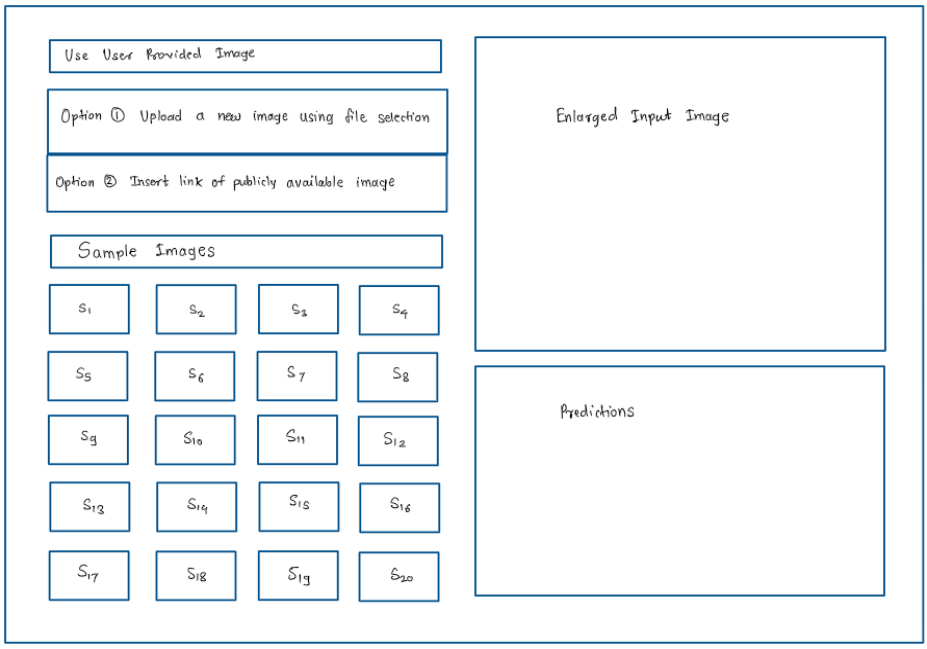


Figure 2. Initial sketch of the user interface

3. Results

3.1 Results of fine-tuning and training the multi-label attribute classifier

After setting the pre-trained DenseNet model, multi-label one-versus-all loss function and SGD optimizer, we have started training our model and performed the hyperparameter fine-tuning. You can find the hyperparameters that gave us the highest mAP score in **Table 1**. After training for 40 epochs with these hyperparameters, the mAP score has converged and we could finally achieve a 95% mAP score on the testing set, which we believe is sufficient for our task.

Hyperparameter	**Value**
Batch size	50
SGD momentum	0.9
Initial learning rate	0.1
Exponential learning rate decay for every epoch	0.97

Table 1. Hyperparameter values that gave 95% mAP score for the multi-label attributes classifier

We believe that we do not have an issue of overfitting our training data, because during training the gap between the loss and mAP for training and validation sets was not significant. In **Figure 3**, we can clearly see that the loss and mAP scores start to converge from epoch 30 to 40. At epoch 40, the validation set reached an mAP of almost 94%. As we have already mentioned, the testing set gave us an mAP of 95%, which also proves that there is no problem of overfitting the training data.

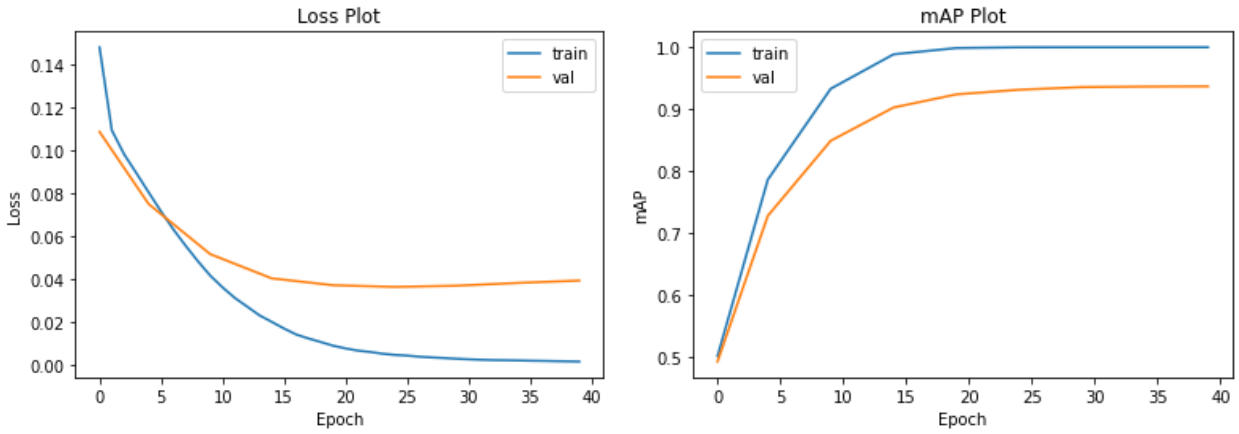


Figure 3. Loss and mAP plots for training and validation sets during training

The model has performed the worst on the following scene attributes: fencing (77.58% mAP), concrete (79.1% mAP), vinyl/ linoleum (80.94% mAP), conducting business (81.89% mAP) and rusty (85.97% mAP). We believe that considering the variety of image attributes, the predictions for these attributes are still acceptable. The model performed worse on these attributes, since they appear in multiple forms adding the variance in their distribution, which makes the task of predicting the presence of these attributes in an image harder. For example, you can find the original sample images that contain fencing attribute in **Figure 4**. The images from these figure prove that there is a high variety of fencing types in different images and the task of learning them is not trivial considering the small number of images that contain this attribute (see **Figure 5**).

Figure 4. Different types of fencing present in images from SUN database

On the other hand, the model has 100% mAP score for the following attributes: smoke, railroad, praying and cleaning. We believe that predicting the presence of these attributes in an image is an easier task, since these attributes appeared in less forms and from **Figure 5** it can be seen that the SUN Attribute database has the smallest number of images that have these attributes which made it easier for the classifier to learn them.

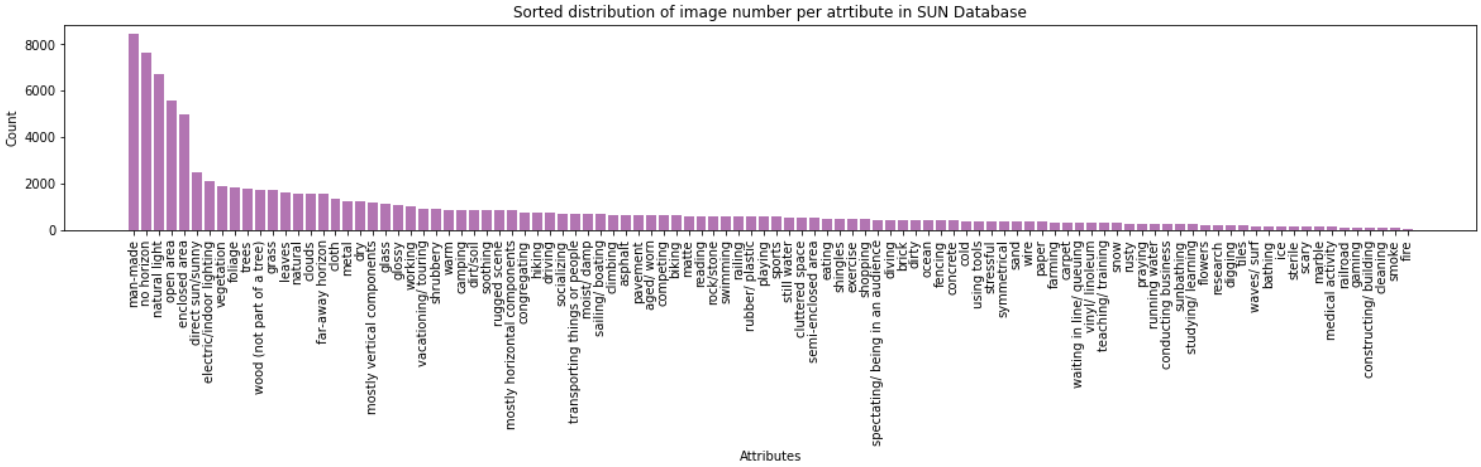


Figure 5. Sorted distribution of image number per attribute in SUN Database

The final output of our multi-label scene attribute classifier is the scores for each of the 102 attributes that show confidence on whether a specific attribute is present in the image. We believe that listing the top 10 predicted attributes is sufficient for the task of scene recognition. In **Figure 6**, you can find a sample image for which the model has predicted the following top 10 attributes: man-made, natural light, driving, transporting things or people, glossy, direct sun/sunny, open area, clouds, mostly vertical components, glass. We believe that the results of this prediction is very accurate, since we can find all of these attributes in the given image.

Figure 6. Sample image for multi-label classifier

3.2 Results of the scene classifier, indoor/outdoor predictor and heat map generator

As we have already mentioned, we have implemented the scene classifier, indoor/outdoor type of environment predictor and a heat map generator from the codebase provided by Bolei Zhou et al. We have used the pre-trained weights of ResNet model with 18 layers to generate scene category predictions for an image. In this classifier, we output the top 5 predicted categories of the image that have the highest confidence scores, since scene categories in top 5 predictions are usually very similar and the image can contain regions for each of these categories. In **Table 2**, we can see the classifier prediction on the image from **Figure 6** and the street and crosswalk are the two predictions with the highest confidence

scores. As we can see this prediction is reasonable as this could be both the street and the crosswalk. The predicted type of environment is outdoor, which is also true about the image.

Predicted scene category	**Confidence score (out of 1)**
street	0.414
crosswalk	0.358
downtown	0.063
parking_garage/outdoor	0.023
bazaar/outdoor	0.021

Table 2. Predicted scene categories and confidence scores for the sample image in **Figure 6**

In **Figure 7,** we can see that the bottom part of the image and the car on the right were the most important for a scene classifier to make a prediction that this is a street. We believe that it is rational to say that the bottom part of the image signals most that it is the street.

Figure 7. Heat map for the predicted street category

3.3 Final version of the web application

We have incorporated each model in our web application and deployed it online. The application was deployed online using aws ec2 instance t2.small with TCP protocol using Gunicorn at the following link Scene Recognition model. Alternate link.

Below is the snapshot of the application. The app layout is formed such that users can give input image based on their preferred method and the output will be generated immediately after the image input is given. Users can give input from below methods:

1. By uploading an image from their local device
2. Inserting a link from a publicly available and with creative commons licence
3. Select an image from samples provided

You can find the demo video of the app here.

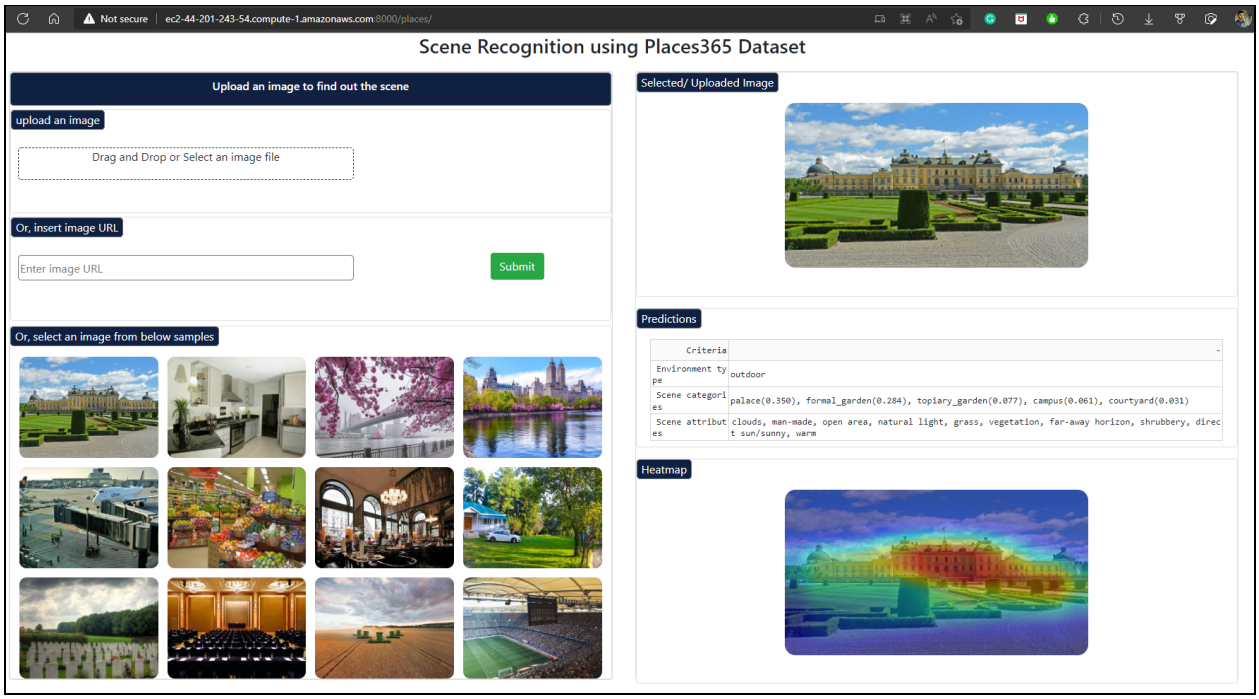


Figure 8. Web application snapshot

4. Discussion and conclusions

In this project we have built and trained the multi-label attribute classifier from scratch and implemented the scene classifier using the pre-trained ResNet model weights and heat map generator from the codebase provided by Bolei Zhou et al. After we have trained the multi-label classifier, it has reached a 95% mAP score on the testing set and we believe that it does a decent job in predicting the presence of attributes in the image. However, there were some attributes which had the prediction accuracy close to 80% mAP. This has happened because these attributes had a lot of different types and a small number of images that include these attributes in the SUN database were feeded to the model, which means that the classifier was not able to learn all of the properties of these attributes. We believe that the potential solution to increase the prediction accuracy for these attributes is to use more data augmentation steps and/or greater number of images that include these attributes feeded to the model in the training phase.

5. Statement of individual contribution

Both Nursultan and Sandip were working on implementation of algorithms given in our paper by Boleu Zhou et al. Apart from this, Nursultan has mainly focused on implementing multi-label scene attribute classifier and Sandip was working on building the web application and deploying our models online.

6. References

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2016.319

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2018). Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(6), 1452–1464. https://doi.org/10.1109/tpami.2017.2723009

Places365 database: http://places2.csail.mit.edu/download.html

SUN Attribute database: http://cs.brown.edu/~gmpatter/sunattributes.html

Github Repository: https://github.com/CSAILVision/places365