COMP6721 Applied Artificial Intelligence Project Proposal

Chaima jaziri : 40305743 **Yajing Liu 40268996** **Taranjeet Kaur 40263787**

A. Abstract

Image classification is essential in fields such as image search. This project aims to categorize venue images into five categories using supervised decision trees, semi-supervised decision trees, and supervised convolutional neural networks (CNNs). Three main obstacles are the absence of labeled data, data imbalance, and the black-box nature of models. Data augmentation addresses data imbalance by generating images through techniques like flipping and rotating although it may lead to overfitting. High-confidence pseudo-labels are used by semi-supervised models to manage limited labeled data. Although grid search is computationally costly, it optimizes hyperparameters. Our methodology involves image analysis, data processing, and model training. Evaluation metrics include accuracy, precision, recall, F1-score, and confusion matrix analysis. The supervised decision tree obtained 65% accuracy, the semi-supervised decision tree reached 57% accuracy, and the supervised CNN achieved 85% accuracy.

B. Introduction

B.1. Problem Statement and Associated Challenges

Image classification is crucial in various fields, including medical diagnosis, automated driving, and image search. For example, image categorization is necessary for image search to get relevant photos based on visual information from enormous databases, improving search results' efficiency and accuracy. This project aims to enhance venue image search by classifying venue images into five categories using a combination of machine learning techniques: a supervised decision tree, a semi-supervised decision tree, and a supervised convolutional neural network (CNN). By using these technologies, we seek to address the challenges of venue classification and improve accuracy.

The primary challenge of venue image classification includes:

- **Data Imbalance**: Data imbalance refers to a situation where some classes have much fewer images than other classes. Data imbalance can affect the model and reduce prediction accuracy.
- **Limited Labeled Data**: Most of the data is unclassified and labeling the data is typically highly costly. A small amount of labeled data can hinder the performance of supervised models.
- **Optimization**: Image classification models often function as black boxes, making it challenging to understand their internal workings. This opacity can create difficulties when attempting to improve model accuracy.

There are some existing solutions to address these challenges:

- **Data Augmentation**: Data augmentation is used to address the data imbalance problem. Data augmentation is to generate images automatically based on existing images by using technologies such as flipping, rotating, changing brightness, and Gaussian blur. Data augmentation can increase image size effectively but it can cause potential for overfitting.
- **Semi-supervised Learning**: Semi-supervised models are used to address the limited labeled data. Semi-supervised models use high-confidence pseudo-labels to handle the unlabeled data. Semi-supervised models can effectively use large amounts of unlabeled data but poor-quality pseudo-labels can lead to suboptimal model performance.
- **Grid Search**: Grid search is used to find the optimal model hyperparameters to improve the accuracy. Grid search uses each combination of hyperparameters and chooses the set of hyperparameters getting the best performance as a result. Grid search can search comprehensively through all combinations of specified hyperparameter values but it is expensive to get the results.

Our report aims to solve the classification problem by employing a combination of supervised decision trees, semi-supervised decision trees, and supervised CNNs. Our methodology involves:

- **Image Analysis and Data Processing**: We analyze data with descriptive image analysis (EDA) and we process data with data cleaning, data resizing, and data augmentation.
- **Model Training**: Training supervised decision trees and CNNs on labeled data, and semi-supervised decision trees using both labeled and unlabeled data.
- **Evaluation**: We use accuracy, precision, recall, and F1-score of the whole dataset and every class to evaluate our model. We also generate the confusion matrix.

We employed a supervised decision tree, a semi-supervised decision tree, and a supervised CNN to classify images into five categories. The achieved accuracy of the semi-supervised decision tree is 57%, whereas the supervised decision tree achieved 65% accuracy. The supervised CNN achieved 85% accuracy.

B.2. Related Work

B.2.1 Data Augmentation

Data Augmentation is a method used to enhance the performance of deep convolutional neural networks in fields with limited dataset availability. Data augmentation involves geometric transformations, color space augmentations, kernel filters, and other technologies [5].

B.3. Semi-supervised Learning

The labeling process's expense might make obtaining a completely labeled training set impractical. However, Semi-supervised learning may be quite helpful in these kinds of circumstances. Semi-supervised learning uses a small amount of labeled data and a large amount of unlabeled data. Semi-supervised learning generates pseudo-labels for unlabeled data [4].

B.4. Grid Search

Grid search is an exhaustive search to try every combination of hyperparameter values to find a combination performing best in some performance metric [1].

C. Methodology

C.1. Image Analysis and Data Processing

C.1.1 Data Description

We use 2,960 colorful images categorized into five datasets: bar (604 images), beach (622 images), Bookstore(746 images), restaurant (449 images), and subway (539 images). We eliminated parts of the datasets for having specific classes and combined datasets from different sources: image.cv [2] and mit.edu [3].

C.1.2 Data Exploration

To identify the class imbalance, we plot the number of images in each class in a bar chart, shown in figure 1.

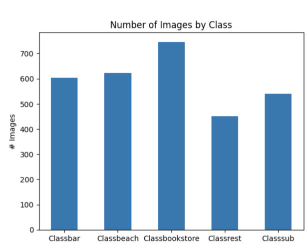

Figure 1. Identifying Class Imbalance

C.1.3 Data Cleaning

The acquired dataset has good quality. No duplicated images are present; results were shown in the notebook.

C.1.4 Data resizing

In this project, we did two data resizing processes. First, it was to the bar images to make its dimensions 256*256 pixels like the rest of the classes. Second, we found that each took over 10 minutes to load. Therefore, we resized them to 64x64 pixels.

C.1.5 Data Augmentation

To cater to class imbalance, around 200 new images are added to the restaurant class by augmentation. The augmentations include horizontal and vertical flips, randomly rotates images, brightness adjustment and Gaussian blur. In the figure 2, we share the new distribution of our dataset.

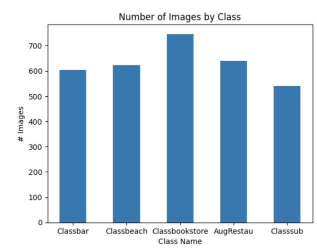

Figure 2. Augmented data

C.2. Model Training

C.2.1 Supervised Decision Tree

The supervised decision tree is mainly implemented through the scikit-learn library. Our solution contains two major steps: data preparation and model training.

- **Data Preparation**: Images are loaded via PIL and transformed with torchvision, resized to 64x64 pixels, and converted into tensors. Pixel values are normalized to a range of 0 to 1. The resulting 3D matrix is flattened into a 1D array to meet scikit-learn's input requirements.

- **Model Training**: In a first approach, the original dataset is then split into 80% training and 20% testing sets and later we worked on the augmented dataset. The Decision Tree Classifier is trained first with specified parameters: entropy criterion, maximum depth, minimum samples split, and minimum samples leaf and later it was choosen with a Hyperparameter tuning using Grid Search.

C.2.2 Semi-supervised Decision Tree

The semi-supervised decision tree has similar data preparation processes to the supervised decision tree. But model training shows differently.

- **Model Training**: The dataset is split into 80% training and 20% testing sets. The labels are removed from 80% of the training data. The remaining labeled data is used to train the decision tree model. The model is trained in several iterations where in each iteration the top 10% of the high-confidence predictions from the unlabeled data are added to the labeled set. The training is done until all the data is labeled. The hyper-parameters used for classification are entropy criterion, maximum depth, minimum samples split, and minimum samples leaf.

C.2.3 Supervised Convolutional Neural Network

The supervised convolutional neural network is implemented using Pytorch library. Our solution contains three major steps: data preparation, CNN model design, and model training.

- **Data Preparation**: An $ImageDataset$ class expands the $Dataset$ class in Pytorch library. In this $ImageDataset$ class, we define methods len and $getitem$ to cooperate $DataLoader$. We randomly split the datasets into 80% training datasets and 20% testing datasets. Then out of the training dataset, we take 20% of the data as validation dataset.$DataLoader$ class is used to load training, validation and testing datasets. We normalize the dataset before loading to help the CNN perform better as it helps get data within a range and reduces the skewness since it's centered around 0. This helps it learn faster and better.

- **CNN Model Design**: A $CNNModule$ class expands the $Module$ class. In this $CNNModule$ class, we define the CNN model structure. Our CNN model contains two convolutional layers and one fully connected layer.

  The first convolutional layer includes $Conv2d$ (32 filters, 3x3 kernel, stride 1, padding 1), $BatchNorm2d$, $LeakyReLU$, $Conv2d$ (32 filters, 3x3 kernel, stride 1, padding 1), $BatchNorm2d$, $LeakyReLU$, $Max-Pooling$ (2x2 window, stride 2).

  The second convolutional layer includes $Conv2d$ (64 filters, 3x3 kernel, stride 1, padding 1), $BatchNorm2d$, $LeakyReLU$, $Conv2d$ (64 filters, 3x3 kernel, stride 1, padding 1), $BatchNorm2d$, $LeakyReLU$, $Max-Pooling$ (2x2 window, stride 2).

  The fully connected layer includes $Dropout$ (p=0.1), $Linear$ (64*16*16 to 1000), $ReLU$, $Linear$ (1000 to 512), $ReLU$, $Dropout$ (p=0.1), $Linear$ (512 to number of classes).

  This model structure is suitable for this practice for several reasons. First of all, convolutional layers can help to capture features from low-level edges to complex patterns. Additionally, batch normalization can also speed up and stabilize training. Moreover, LeakyReLU also enhances learning by preventing vanishing gradients. In the end, MaxPooling reduces spatial dimensions, retains key features, and mitigates overfitting.

- **Model Training**: We train the model through multiple epochs. For each epoch, we first perform a forward pass to generate predictions. Subsequently, we calculate the loss between predictions and true labels. Finally, we perform a backward pass with Adam optimizer to calculate gradients and upload model parameters based on these gradients.

- **Plotting Loss and accuracy over epochs**: We use the validation dataset to get accuracy and mean loss at the end of each epoch. The accuracy and Loss is observed over the training and validation data in each epoch.

C.3. Evaluation

Evaluation is performed on the testing set, computing accuracy, precision, recall, and F1 score metrics. Additionally, a confusion matrix offers detailed insight into classification performance across various classes.

D. Results and discussion

D.1. Supervised Decision Tree

Originally, we used the following value of hyper-parameters: $criterion(entropy)$, $max\_depth(35)$, $min\_samples\_split(20)$, $min\_samples\_leaf(5)$. We got the improved results: accuracy (46.79%), precision (45.67%), recall (45.93%), and F1-score (45.70%) (Figure 3). The evaluation of each class is shown in the following table (Figure 4).

We kept the hyperparameters unchanged and trained the model with our new augmented datasets obtained through data augmentation techniques. We got the improved results: accuracy(64.18%), precision(64.27%), recall(64.65%), and F1-score(64.09%) (Figure 5). The evaluation of each class is shown in the following table (Figure 6). Significant performance improvement is shown in the restaurant class.

We chose the better value of hyperparameters with GridSearch and trained on the augmented dataset. The best hyperparameters were the following value: $criterion(entropy)$, $max\_depth(48)$,

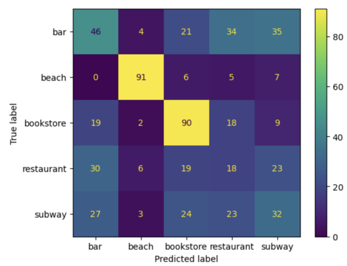

Figure 3. Supervised learning original data

	Accuracy	Precision	Recall	F1-score
Bar	0.3286	0.3770	0.3286	0.3511
Beach	0.8349	0.8585	0.8349	0.8465
Bookstore	0.6522	0.5625	0.6522	0.6040
Restaurant	0.1875	0.1837	0.1875	0.1856
Subway	0.2936	0.3019	0.2936	0.2977

Figure 4. Supervised learning original data evaluation table

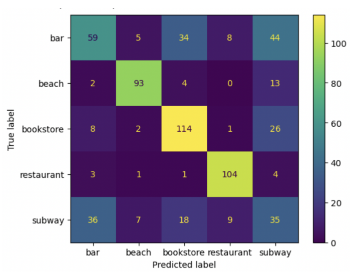

Figure 5. Supervised learning augmented data

	Accuracy	Precision	Recall	F1-score
Bar	0.3933	0.5463	0.3933	0.4574
Beach	0.8304	0.8611	0.8304	0.8455
Bookstore	0.7550	0.6667	0.7550	0.7081
Restaurant	0.9204	0.8525	0.9204	0.8851
Subway	0.3333	0.2809	0.3333	0.3084

Figure 6. Supervised learning augmented data evaluation table

$min\_samples\_split(2)$, $min\_samples\_leaf(1)$. We got the improved results: accuracy(64.81%), precision(63.70%), recall(65.11%), and F1-score(63.91%) (Figure 7). The evaluation of each class is shown in the following table (Figure 8).

D.2. Semi-supervised Decision Tree

We experimented with different hyper-parameters manually and chose the following value: $criterion(entropy)$, $max\_depth(35)$, $min\_samples\_split(20)$, $min\_samples\_leaf(5)$. We got the results on the original datasets: accuracy (41.39%), precision (40.41%), recall (41.12%), and F1-score (40.65%) (Figure 9). The

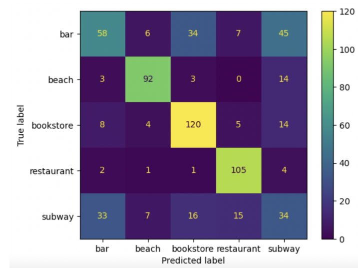

Figure 7. Supervised learning with grid search

	Accuracy	Precision	Recall	F1-score
Bar	0.3867	0.5577	0.3867	0.4567
Beach	0.8214	0.8364	0.8214	0.8288
Bookstore	0.7947	0.6897	0.7947	0.7385
Restaurant	0.9292	0.7955	0.9292	0.8571
Subway	0.3238	0.3063	0.3238	0.3148

Figure 8. Supervised learning with grid search evaluation table

evaluation of each class is shown in the following table (Figure 10). Then, we run the model on the augmented dataset: accuracy (57.21%), precision (57.45%), recall (58.67%), and F1-score (57.89%) (Figure 11). The evaluation of each class is shown in the following table (Figure 12).

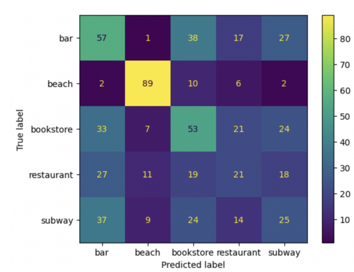

Figure 9. Semi-Supervised learning original data

D.3. Supervised Convolutional Neural Networks

We used the following hyperparameters $no_{e}poch(10)$, $batch_{s}ize(32)$, and $learning_{r}ate(0.001)$. We got the results on the original datasets: accuracy (60.30%), precision

	Accuracy	Precision	Recall	F1-score
Bar	0.4071	0.3654	0.4071	0.3851
Beach	0.8165	0.7607	0.8165	0.7876
Bookstore	0.3841	0.3681	0.3841	0.3759
Restaurant	0.2188	0.2658	0.2188	0.2400
Subway	0.2294	0.2604	0.2294	0.2439

Figure 10. Semi-Supervised learning original data evaluation table

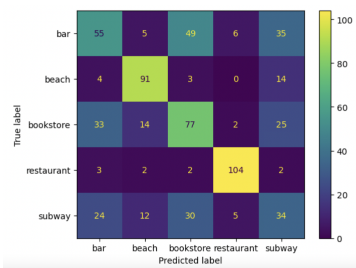

Figure 11. Semi-Supervised learning augmented data

	Accuracy	Precision	Recall	F1-score
Bar	0.3667	0.4622	0.3667	0.4089
Beach	0.8125	0.7339	0.8125	0.7712
Bookstore	0.5099	0.4783	0.5099	0.4936
Restaurant	0.9204	0.8889	0.9204	0.9043
Subway	0.3238	0.3091	0.3238	0.3163

Figure 12. Semi-Supervised learning augmented data evaluation table

(65.85%), recall (60.97%), and F1-score (59.77%) (Figure 13). The evaluation of each class is shown in the following table (Figure 14).

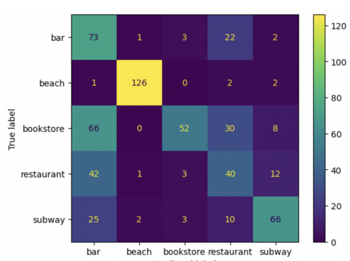

Figure 13. Supervised CNN original data (1)

We changed hyperparameters to $no_epoch(20)$, $batch_size(128)$, and $learning_rate(0.001)$. We got the results: accuracy (67.40%), precision (66.61%), recall (66.77%), and F1-score (66.20%) (Figure 15). The evaluation of each class is shown in the following table (Figure 16).

We kept the hyperparameters unchanged and ran the model on augmented datasets. We got the results: accuracy (84.76%), precision (83.92%), recall (83.55%), and F1-score (83.71%) (Figure 17). The evaluation of each class is shown in the following table (Figure 18).

	Accuracy	Precision	Recall	F1-score
Bar	0.7228	0.3527	0.7228	0.4740
Beach	0.9618	0.9692	0.9618	0.9655
Bookstore	0.3333	0.8525	0.3333	0.4793
Restaurant	0.4082	0.3846	0.4082	0.3960
Subway	0.6226	0.7333	0.6226	0.6735

Figure 14. Supervised CNN original data (2)

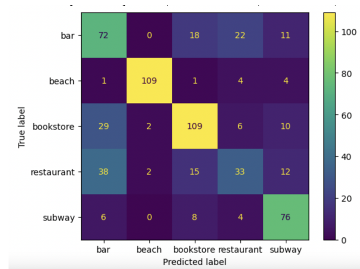

Figure 15. Supervised CNN original data (2)

	Accuracy	Precision	Recall	F1-score
Bar	0.5854	0.4932	0.5854	0.5353
Beach	0.9160	0.9646	0.9160	0.9397
Bookstore	0.6987	0.7219	0.6987	0.7101
Restaurant	0.3300	0.4783	0.3300	0.3905
Subway	0.8085	0.6726	0.8085	0.7343

Figure 16. Supervised CNN original data (2)

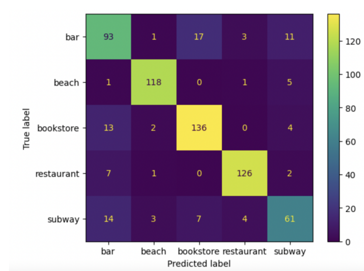

Figure 17. Supervised CNN augmented data

In the validation loop, we accumulated the losses from each batch of validation dataset, and calculated an average validation loss to get an idea about how well the model generalizes to new unseen data. It helped in tuning the hyperparameters such as number of epochs.Figure 19 depicts the plot of training, validation losses and accuracy of the CNN model over each epoch on the augmented dataset.

In summary, the changes in accuracy in these three

	Accuracy	Precision	Recall	F1-score
Bar	0.7440	0.7266	0.7440	0.7352
Beach	0.9440	0.9440	0.9440	0.9440
Bookstore	0.8774	0.8500	0.8774	0.8635
Restaurant	0.9265	0.9403	0.9265	0.9333
Subway	0.6854	0.7349	0.6854	0.7093

Figure 18. Supervised CNN augmented data

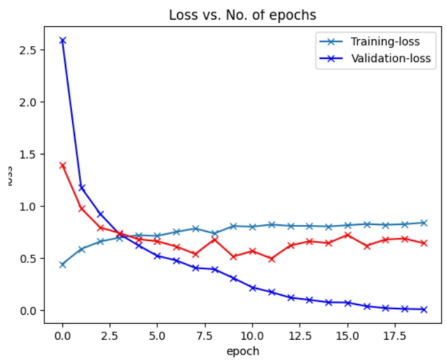

Figure 19. Training and validation loss over each epoch

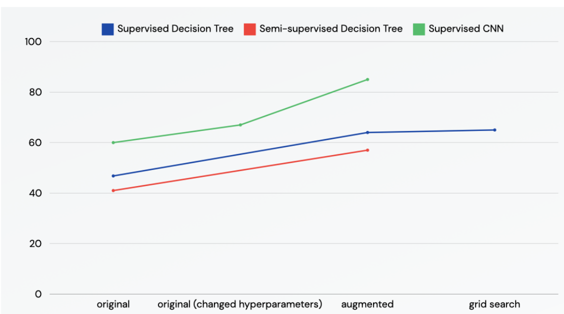

Figure 20. Summary chart

methodologies are shown in Figure 20.

E. Conclusion

In conclusion, we employed a supervised decision tree, a semi-supervised decision tree, and a supervised CNN to classify 2,960 images into five categories. We achieved 64.81% accuracy for the supervised decision tree, 57.21% accuracy for the semi-supervised decision tree, and 84.76% for supervised CNN. To improve accuracy in the future, we will use cross-validation in our three models.

References

[1] grid-search. https://en.wikipedia.org/wiki/Hyperparameter_optimization.

[2] images.cv. https://images.cv/.

[3] MIT. https://web.mit.edu/torralba/www/indoor.html.

[4] semi-supervised. https://debategraph.org/Details.aspx?nid=305594.

[5] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, July 2019.