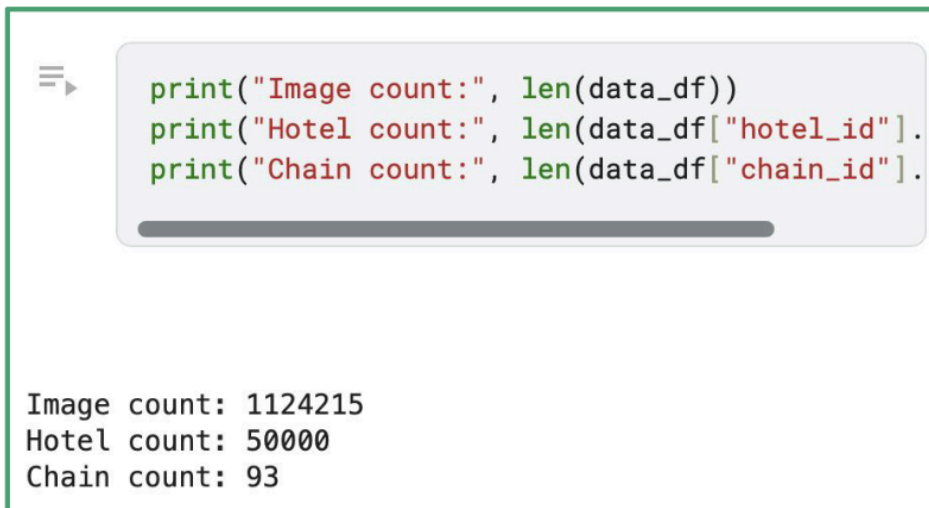


Data Report Human Trafficking – PD 3.2

1. Which data cleaning and processing techniques, if any, were used to make the data usable?

We started the data preparation phase by first cleaning the dataset by removing missing image or hotel identifiers to make sure only valid entries are used in the training set for the model. Then, to standardize the entire dataset of over 1.1 million hotel room images, where there exist 50,000 unique hotels across 93 chains, all the images were resized to 384x384 resolution to establish consistency by using padding and normalization. Pixel values were normalized (scaled to a common range such as 0-1) so that the data distribution remains consistent across training batches. Having this uniformity is essential for the convolutional neural networks (CNNs) we use in this project, as CNNs require consistent input dimensions to learn effectively.

Standardizing methods were applied to categorical data like hotel names and chain names to ensure consistent formatting. Columns were also transformed to appropriate data types (integers for counts, dates for timestamp fields, and floats for normalized pixel values). Ensuring correct data types across the dataset facilitated more efficient computations and compatibility with downstream libraries where a more uniform data is used for training.



```
print("Image count:", len(data_df))
print("Hotel count:", len(data_df["hotel_id"]).
print("Chain count:", len(data_df["chain_id"]).
```



```
Image count: 1124215
Hotel count: 50000
Chain count: 93
```

The code output indicating the overall size of the data set, specifically that it contains 1,124,215 images, 50,000 hotels, and 93 hotel chains.

This confirms the large-scale nature of the project's data and underscores the importance of robust data management and model strategies. Statistical analysis used to display a computational summary of the data statistics like mean, median, quantiles, and standard deviation for continuous variables to identify central tendencies and dispersions. Knowing the competition provided 15 GB of 1.1 million images, initial data acquisition faced challenges such as local resource limitations, including slow processing due to CPU constraints and limited disk space. Ultimately, the team switched to Kaggle for its ease of use and ability to directly submit to the competition to obtain the MAP@5 score for performance analysis and model comparison. These platforms also offered enhanced GPU availability and memory, ultimately streamlining our preprocessing and model development. This image pipeline was compatible with augmentation libraries like Albumentations, which is used to enhance robustness during training.

These techniques collectively set a solid foundation for a subsequent modeling, ensuring that both CNN-based feature extraction and augmentation-based robustness are well grounded in the data's underlying structure. The rigorous process of handling missing values, standardizing text data, and normalizing images ensures that only valid trained data is used for training.

2. Which feature extraction techniques, if any, were used to generate new features to the dataset?

With the data cleaned and standardized, the main feature extraction techniques were all CNN-based models. Specifically, the ResNet-34 and EfficientNet architectures are the backbones of our project models. With these models pre-trained on the ImageNet dataset, the networks were fine-tuned in detecting any patterns and visual cues that are unique to each hotel's interior. To complement this training and improve accuracy, we added a test-time augmentation (TTA), which applies transformations to test images to improve model generalization. We want to add this as it averages predictions from multiple augmented versions of the same image to reduce uncertainty. Specifically, it returns the top 5 most confident predictions per image and then generates 3 augmented versions of each image to improve robustness. Thus, multiple augmented versions of a test image are generated and passed through the model. Where there are predictions for each augmented version. Lastly, the final prediction is obtained by averaging or voting among these predictions to increase stability and reduce noise.

For the EfficientNet, feature extraction was done by generating embeddings of the images using the "efficientnet_b0" as the backbone. These embeddings were vectors that included float-32 objects. Once the images were turned into embeddings, they were

put through the model, and that output was embedded as well. These vectors were then analyzed against the target image, and a cosine similarity was found. This cosine similarity is the cosine of the angle between the two vectors; the more similar the vectors, the closer to one the cosine similarity will be. The features of the images are the embeddings, and those embeddings are compared to each other to determine how similar they are to each other.

We used the albumentations library for data augmentation. Albumentations contains multiple methods for image augmentation which can be applied to the original dataset randomly while updating the model. This effectively means that every time the model is updating its weights it is being tested on a different dataset, without having to permanently increase the size of the original dataset. We are using the following augmentations (p is the chance that the augmentation will be applied to any given image during each iteration):

ResNet-34:

```
A.RandomCrop(width=64, height=64),  
  
    A.HorizontalFlip(p=0.75),  
  
    A.ShiftScaleRotate(p=0.5, shift_limit=0.0625, scale_limit=0.1, rotate_limit=10,  
interpolation=cv2.INTER_NEAREST, border_mode=cv2.BORDER_CONSTANT),  
  
    A.OpticalDistortion(p=0.25, distort_limit=0.05, shift_limit=0.01),  
  
    A.Perspective(p=0.25, scale=(0.05, 0.1)),  
  
    A.ColorJitter(p=0.75, brightness=0.2, contrast=0.2, saturation=0.1, hue=0.05),  
  
    A.CoarseDropout(p=0.5, min_holes=1, max_holes=5,  
  
                    min_height=IMG_SIZE//16, max_height=IMG_SIZE//8,  
  
                    min_width=IMG_SIZE//16, max_width=IMG_SIZE//8), # normal coarse  
dropout  
  
    A.CoarseDropout(p=0.75, max_holes=1,
```

```

min_height=IMG_SIZE//4, max_height=IMG_SIZE//2,

min_width=IMG_SIZE//4, max_width=IMG_SIZE//2,

fill_value=(255,0,0)),# simulating occlusions in test data

```

Albumentation Augmentations in Res Net Model

EfficientNet:

```

import albumentations as A
import albumentations.pytorch as APT
import cv2

# used for training dataset - augmentations and occlusions
train_transform = A.Compose([
    #A.RandomCrop(width=256, height=256),
    #A.HorizontalFlip(p=0.75),
    A.VerticalFlip(p=0.5),
    A.ShiftScaleRotate(p=0.5, shift_limit=0.0625, scale_limit=0.1, rotate_limit=10, interpolation=cv2.INTER_NEAREST),
    A.OpticalDistortion(p=0.25, distort_limit=0.05, shift_limit=0.01),
    A.Perspective(p=0.25, scale=(0.05, 0.1)),
    A.ColorJitter(p=0.75, brightness=0.25, contrast=0.25, saturation=0.15, hue=0.065),
    A.CoarseDropout(p=0.5, min_holes=1, max_holes=5,
                    min_height=IMG_SIZE//16, max_height=IMG_SIZE//8,
                    min_width=IMG_SIZE//16, max_width=IMG_SIZE//8), # normal coarse dropout

    A.CoarseDropout(p=0.75, max_holes=1,
                    min_height=IMG_SIZE//4, max_height=IMG_SIZE//2,
                    min_width=IMG_SIZE//4, max_width=IMG_SIZE//2,
                    fill_value=(255,0,0)),# simulating occlusions in test data
    #A.RandomBrightnessContrast(p=0.75),
    A.ToFloat(),
    APT.transforms.ToTensorV2(),
])

# used for validation dataset - only occlusions
val_transform = A.Compose([
    A.CoarseDropout(p=0.75, max_holes=1,
                    min_height=IMG_SIZE//4, max_height=IMG_SIZE//2,
                    min_width=IMG_SIZE//4, max_width=IMG_SIZE//2,
                    fill_value=(255,0,0)),# simulating occlusions

    A.ToFloat(),
    APT.transforms.ToTensorV2(),
])

# no augmentations
base_transform = A.Compose([
    A.ToFloat(),
    APT.transforms.ToTensorV2(),
])

```

Albumentation Augmentations in EfficientNet Model

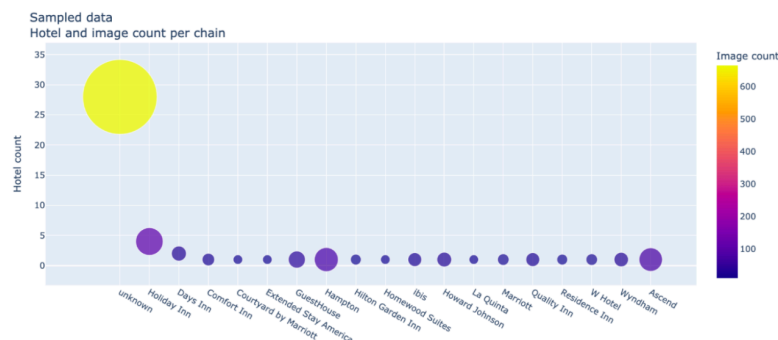
Upon generating these high dimensional aggregated embeddings using ResNet/EfficientNet, we use two key techniques to verify that these embeddings capture meaningful visual patterns. These feature extraction techniques include Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE). Using the TTA pipeline compliments the earlier standardization/normalization to ensure that each test image's embedding is robust to variations. The aggregated

embeddings when visualized using PCA or t-SNE provide a more concrete confirmation that meaningful visual patterns are being captured consistently contributing to a more stable and improved model prediction.

3. What are the trends in notable features in the data?

EDA shows key trends in our vast dataset that shape our approach to modeling and interpretation. Specifically, the variety in lighting conditions, camera angles, and visual quality is consistent across all images. Such differences in the data are valid due to the natural inconsistencies in images submitted by users or surveillance photos that law enforcement might rely on. Thus, there exists a certain class imbalance, which results in certain hotel chains contributing more images than other chains and an imbalance in images per hotel. This results in imbalances that skew the model's predictions toward well-represented chains and ultimately limit the model's ability to generalize to less common hotels.

To further understand these imbalances, we quantified them into a visualization that shows the distribution of images across hotel chains and included a distribution of images per hotel. Here, the X axis is the hotel chain, and the Y axis is the Hotel count. As shown in the legend, the color indicates the image count, where yellow bubbles are images over 600, purple are chains with minimal representation, and the color gradient is proportional to the size of each bubble. As shown in the bubble chart below, some chains — such as the one labeled “unknown”, “Holiday Inn”, “Hampton” — are a disproportionately high representation compared to the other chains that have relatively lower image counts.



Bubble Chart Showing Image Counts of Several Hotel Chains

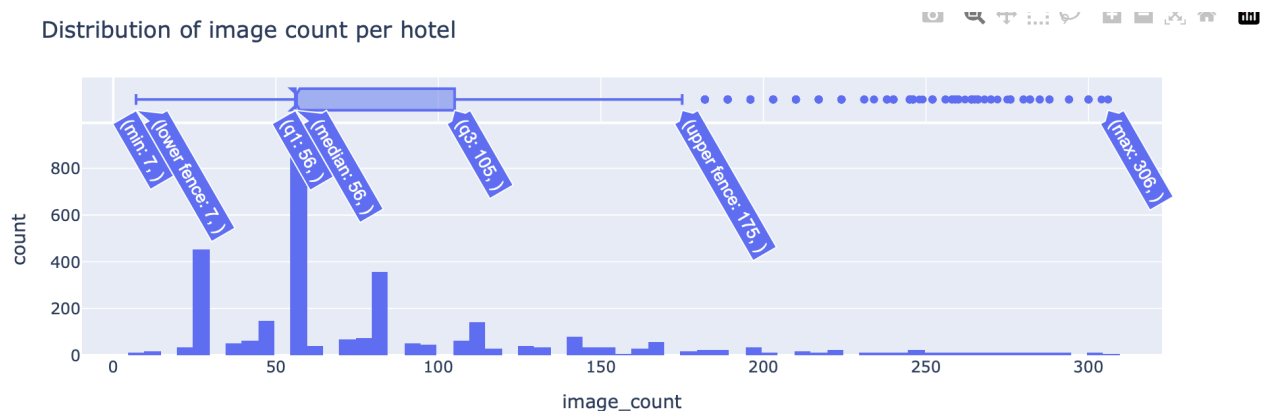
This chart shows a clear class imbalance in the dataset. Most notably, the “unknown” hotel chain stands out as a significant outlier, represented by a large yellow bubble. This indicates that it contains over 30 hotels and more than 600 images, which exceeds any other chain in both hotel count and image volume. Such a disproportionate representation could skew the model's learning process, as it may become biased

toward this majority class. Additionally, this category may include mislabeled or aggregated entries, which pose a risk to label quality and model interpretability.

In contrast, the majority of named hotel chains, such as Holiday Inn, Comfort Inn, and Courtyard by Marriott, have fewer than five hotels and are shown in dark purple, indicating fewer than 100 images. These underrepresented classes are also at risk of being underfit, as the model has limited examples from which to learn distinguishing visual features. A few chains like Ascend, Hampton, and Wyndham fall into a middle range, with moderate representation in both hotel and image count. While these provide more reliable training data, they are still overshadowed by the dominance of the “unknown” label.

Overall, this visualization portrays the need for careful handling of class imbalance during training. Without strategies like data augmentation, re-weighting, or sampling techniques, the model is likely to perform well only on well-represented chains and struggle to generalize to less common hotels—a major concern given the real-world objective of accurately identifying a wide range of hotels based on image input.

Regarding the distribution of image count per hotel, the majority of outliers are present on the right-hand side, which introduces class imbalance at the individual hotel level. This supports the need for having robust feature learning and augmentation to prevent overfitting to these overrepresented hotels.



Histogram of Hotels by Number of Images

This histogram shows a strong imbalance in how many images are associated with each individual hotel in the dataset. Most hotels have between 50 and 105 images, with the median image count at 56, meaning that half of the hotels have fewer than 56 images available. This suggests that while a decent portion of hotels have a moderate amount of image data, a large number still hover just above the minimum threshold for

model training. The minimum and lower fence values are both 7, indicating that some hotels contribute very few images (potentially too few for effective learning without augmentation.)

A key concern is the presence of outliers, as shown by the extended tail on the right of the distribution. The upper fence is 175, but many hotels exceed this threshold, with the maximum number of images reaching 306 for a single hotel. These outliers represent hotels that are highly overrepresented in the dataset. While this abundance of data might improve the model's performance for those particular hotels, it also introduces a risk of overfitting, where the model becomes too specialized and fails to generalize to hotels with less data.

This skewed distribution emphasizes the need for careful model training strategies. To address these disparities, techniques such as data augmentation for underrepresented hotels, weighted loss functions, and sampling adjustments should be reconsidered. These methods help ensure that the model does not disproportionately favor hotels with abundant images while neglecting those with fewer examples. Ultimately, recognizing and addressing these imbalances is essential to building a model that performs well across all hotels, especially since our use case involves matching real-world, sometimes limited, images to hotels.

4. How do the observed trends relate to your project problem statement?

The trends we uncovered during our EDA have a direct impact on our project's overall goal: building a hotel image classification model that can accurately identify hotels from room images to support law enforcement efforts in human trafficking investigations. One of the biggest takeaways is how much variation exists in the image data: everything from lighting and camera angles to overall quality differs from photo to photo. This reflects the reality of how these images are often captured, whether through user submissions or surveillance footage, and highlights the need for a model that can handle these inconsistencies and still perform reliably.

We also noticed a significant class imbalance in the dataset. Some hotel chains and individual hotels have hundreds of images, while others have very few. This imbalance could lead the model to favor overrepresented classes and ignore those with limited data, ultimately affecting its ability to generalize and make accurate predictions on less common hotels. To deal with this, we've applied data augmentation techniques (rotating, flipping, and cropping images) to help the model see more variations of the

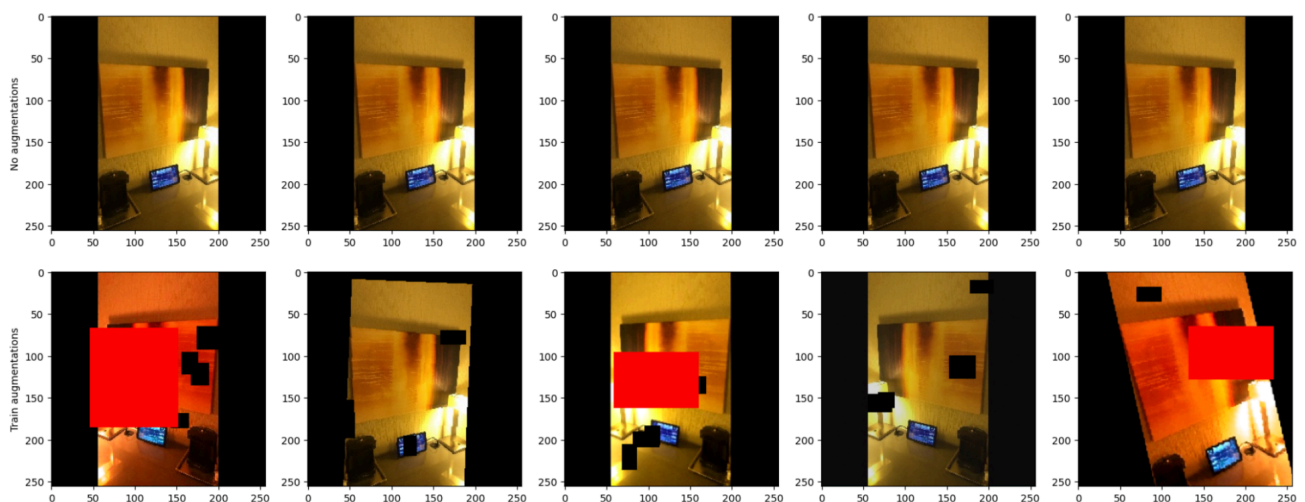
underrepresented classes. Tools like Albumentations have made this process efficient and flexible.

In addition to augmentation, we're using weighted loss functions to make the model more sensitive to underrepresented hotels by penalizing incorrect predictions more heavily in those cases. We also apply test-time augmentation (TTA), which involves generating multiple transformed versions of a test image and averaging the predictions. This helps the model make more stable and confident predictions, even when the image conditions are challenging. Techniques like data augmentation and TTA are essential to simulate these real world challenges during both training and inference.

TTA generates several transformed versions of each test image and averages their predictions. This process helps stabilize the model's output reducing the impact of any single transformation that might otherwise lead to inconsistent predictions. By generating more confident and averaged predictions, TTA directly contributes to a higher MAP@5 score, enhancing the model's reliability and effectiveness in real-world challenging conditions.

5. Does the data foretell any issues that may arise in later stages of the project lifecycle?

Consistent inputs ensure the model's foundation. The data is first standardized, and then grid augmentation is applied where additional transformations are layered on top of the cleaned data. This implies that while the core image quality is maintained, the augmentations introduce variability. While the data preparation and feature extraction pipelines have been carefully designed, the variability in image data displays potential obstacles.



Examples of Hotel Images with Red Masks Applied

This image grid layout provides a visual way to compare the effects of different augmentation techniques on a single image. Grid augmentation is applied to cleaned images, which demonstrates how layered transformations (such as cropping, slipping, color adjustments, and synthetic occlusions) preserve core image quality while introducing variability. This variability, while boosting robustness, also illustrates potential challenges.

The large-scale nature of the dataset implies ongoing challenges with computational efficiency. We expect unpredictable performance drops from alterations that affect augmentation or learning settings. Inconsistencies in feature extraction are also expected due to high image variability. These issues require ongoing monitoring and rigorous fine-tuning to optimize both ResNet and EfficientNet models, which each have unique sensitivities to tuning hyperparameters. We expect continuous fluctuations in results due to various alterations and high variability in image data, especially after aggressive augmentations that may lead to inconsistencies in feature extraction. Any disruption caused by over-augmentation can hinder the learning network from capturing the necessary features reliably. The goal is to optimize both the ResNet and EfficientNet models concurrently so they collectively achieve a high MAP@5 scorer even with the data variability to account for.