MSiA 400
Build Change Project Report

Team Grand Teton
Bhavya Kaushik
Jing Ren
Jue Wang
Zach Zhu

Northwestern

Background

After natural disasters, retrofitting becomes a cost-effective and rapid solution for preserving damaged houses and making them structurally sound; however, due limited availability of professionals, sending engineers on-site for structural evaluation is costly and inefficient.

In order to fix this problem, Build Change is considering building an app that will implement machine learning model, coupled with Computer Vision techniques, to allow users in a country to assess whether their house is a good candidate for retrofitting via photo recognition.

Dataset

Example of generated images

❑ Generated images:
TestGo (3370 images)
TestNoGo (5001 images)

Example of real images: Variations of Mud and Stone houses in Nepal

❑ Real images:
Real_Images (192 images)

Tasks

- ❑ Create a feature vector based on syntactic images by using modules including OpenCV
- ❑ Create a classification model for syntactic images
- ❑ Evaluate the model on real world images
- ❑ Improve feature engineering to hopefully improve the performance on real world images

Our Approach

- Build classification model using logistic regression and XGBoost based on training images.

- Use package OpenCV in python to extract feature vectors (# openings, # levels, house sizes, etc.) from the real-world images, and feed these feature vectors into the classification model to get final result.

- Without the labelling of these real-life images, i.e., suited for retrofitting or not, manually assess a sample of these images based on the rules provided in the excel sheet,

  Retrofit_Module_Inputs_Table_Questionnaire.xlsx

  then evaluate the model based on several metrics including F1-score and Recall.

Image Pre-processing

Turn images into grayscale to avoid dealing with different colors within images.

- ❏ Syntactic images: Binary thresholding

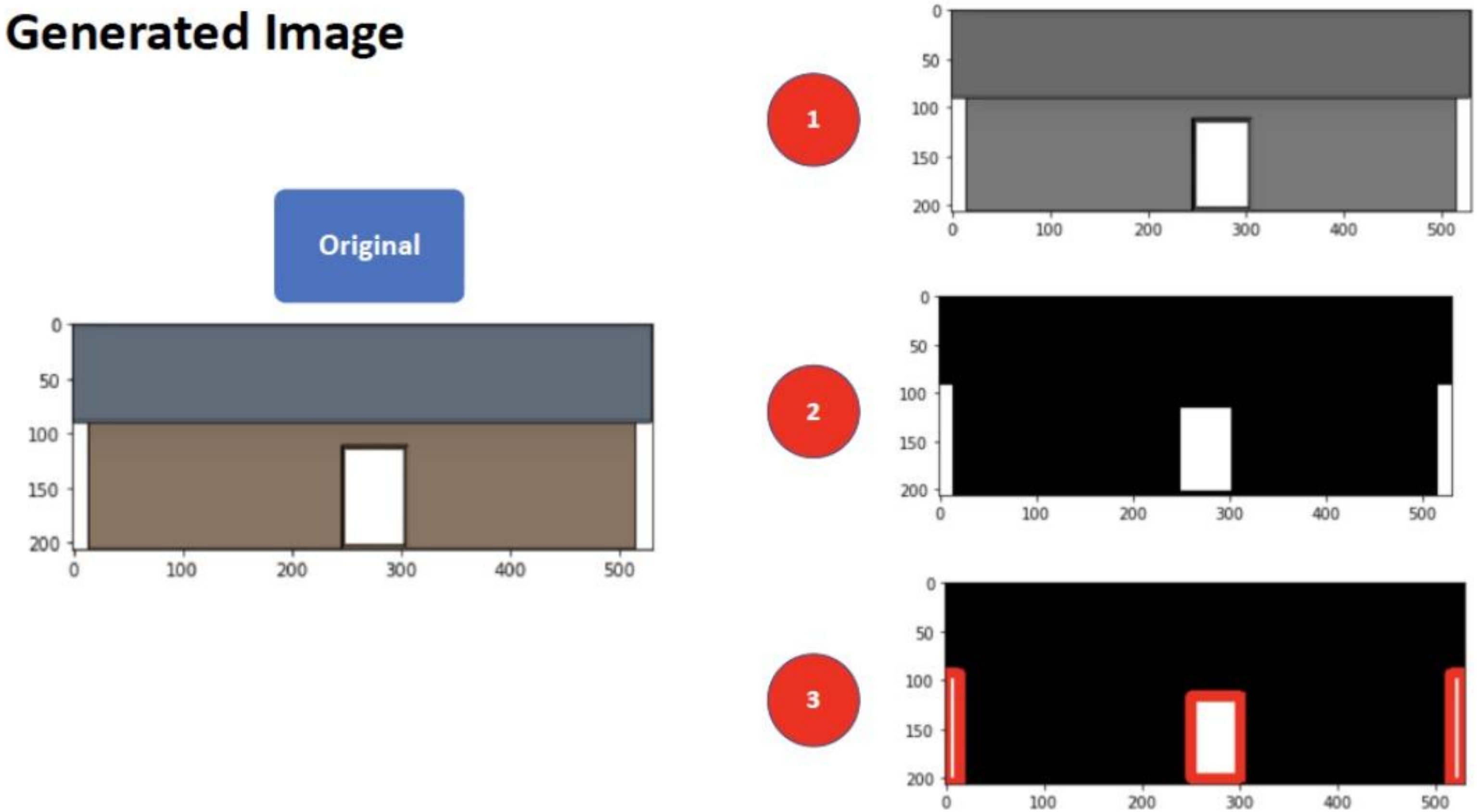
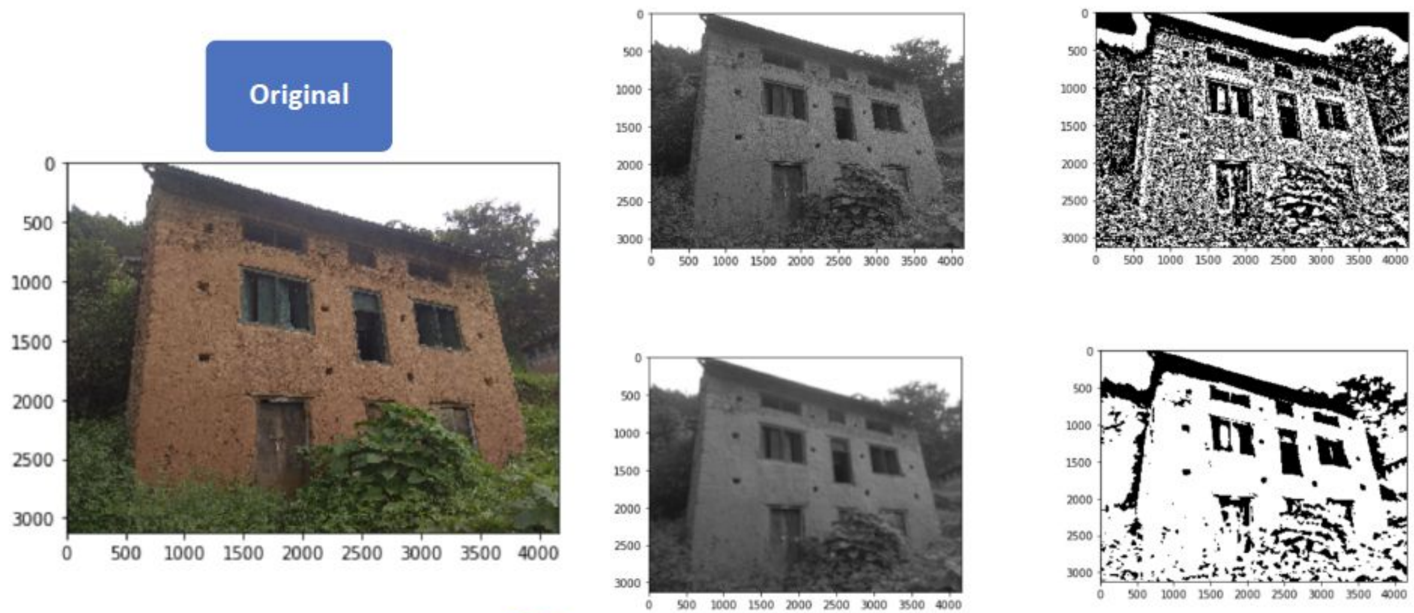


Image Pre-processing

Turn images into grayscale to avoid dealing with different colors within images.

- ❏ Real images: 1. Gaussian adaptive thresholding;
  2. Bilateral filtering for smoothing and reducing noise.

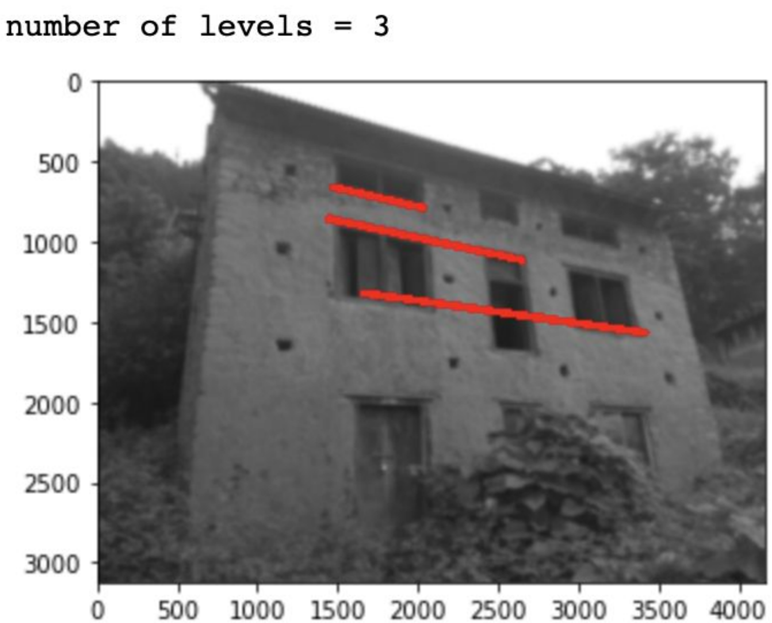
Real Image

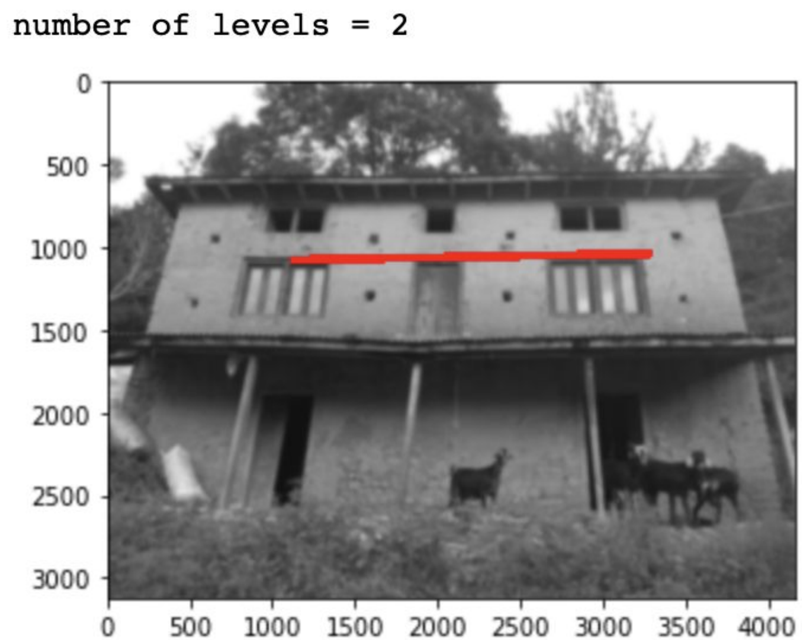


1 **Bilateral Filtering & Adaptive Thresholding**

Level Detection: *Number of Floors*

- Syntactic images:
  - OpenCV's canny edge detection
  - Hough Line Transform to detect the horizontal straight lines that exceeds 80% of the total width of the images.
  - Delete repeated lines and line detected from the roof (only need floor lines to count the number of floors).
- Real images: Detect lines based on contours.



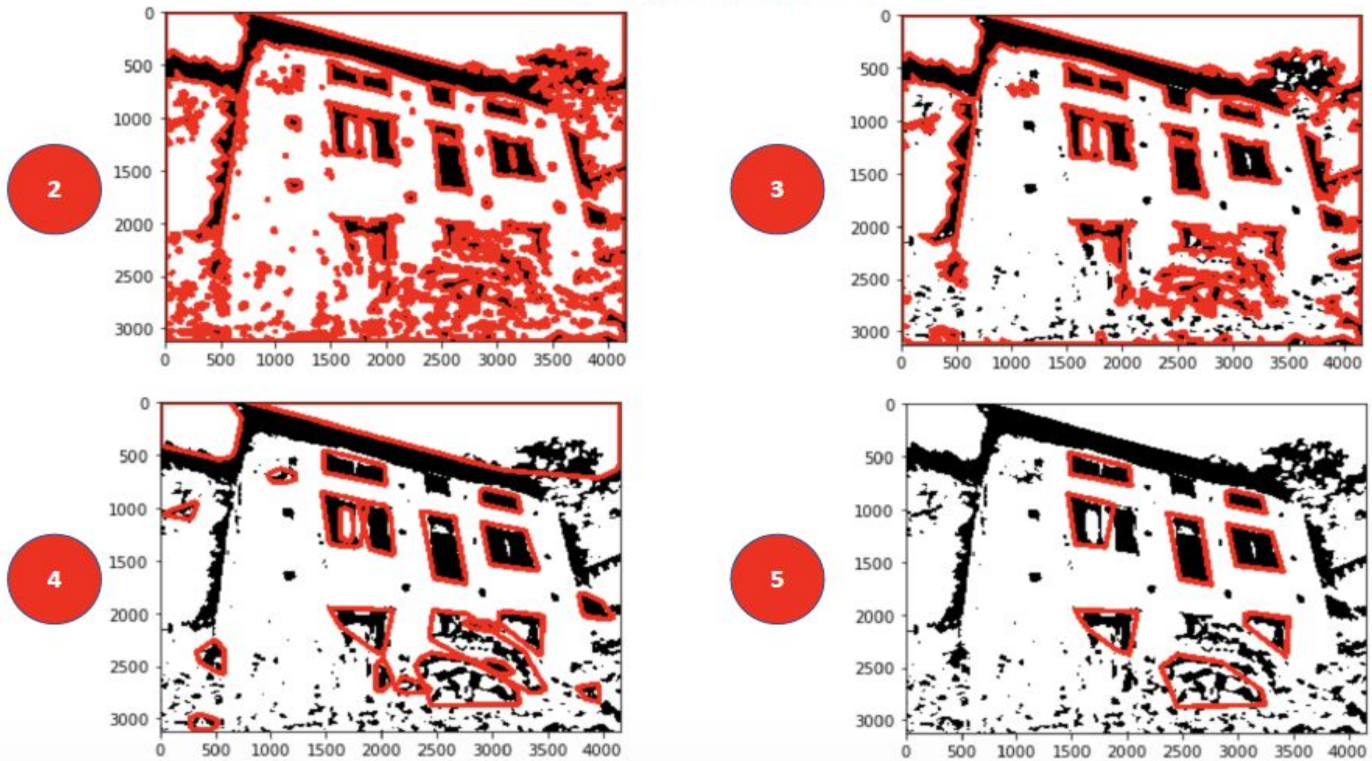


Opening Detection:

Doors, Windows, Openings Caused by Natural Disasters

- Extract the contours in the image with OpenCV method findContours
- Classify the contours based on the number of their sides with UseapproxPolyDP
- Return the number of all quadrilateral contours

Filtering out bad contours

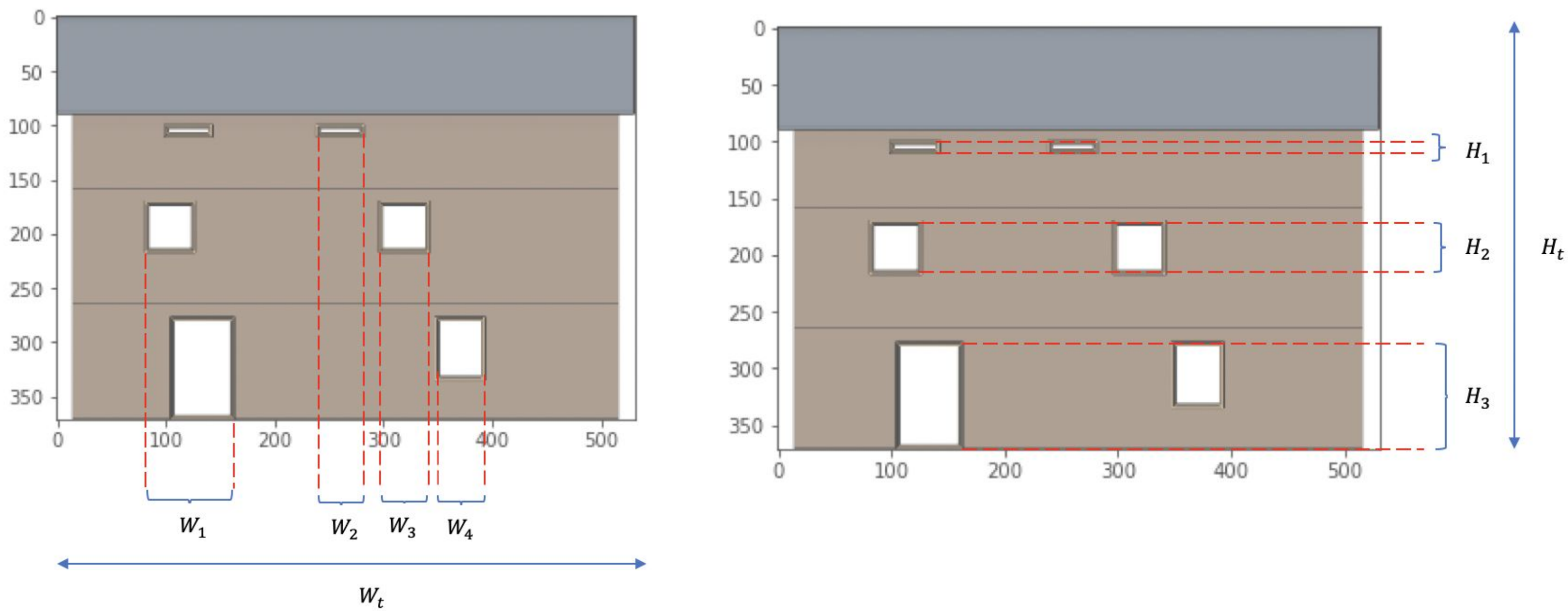


Northwestern

Other Derived Attributes:

Fractional Width and Height of the Openings

- **fraction_width:** fractional width of the openings compared to the width of the house.
- **fraction_height**: fractional height of the openings compared to the height of the house.
- **avg_fraction_width & aggregate_fraction_height:** sum of all windows widths (heights) (divided by the number of floors), and derive the ratio of that sum to the overall width (height) of the building.
- fraction of the total area of the openings of the house to the entire area of the image.

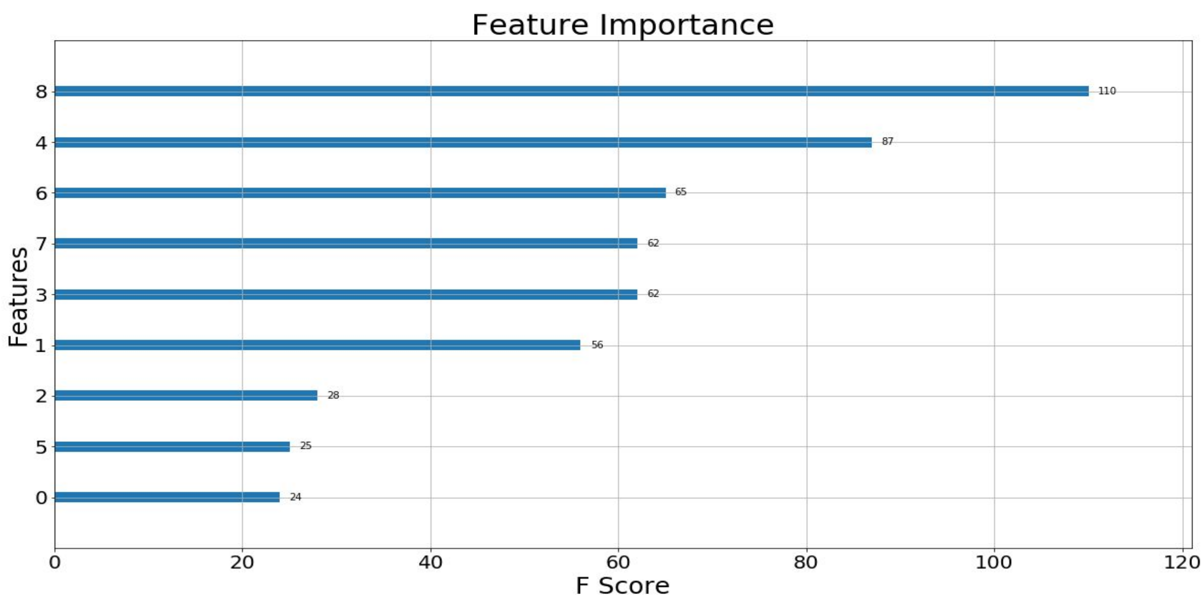


Model Fitting

Evaluation of two models show XGBoost is better than logistic.

Logistic			XGBoost		
Precision	Recall	F1 Score	Precision	Recall	F1 Score
71.40%	78.00%	74.55%	82.43%	79.75%	81.07%

Determine which features are most useful in constructing the boosted decision trees, by taking a look at the feature importance plot from XGBoost model.

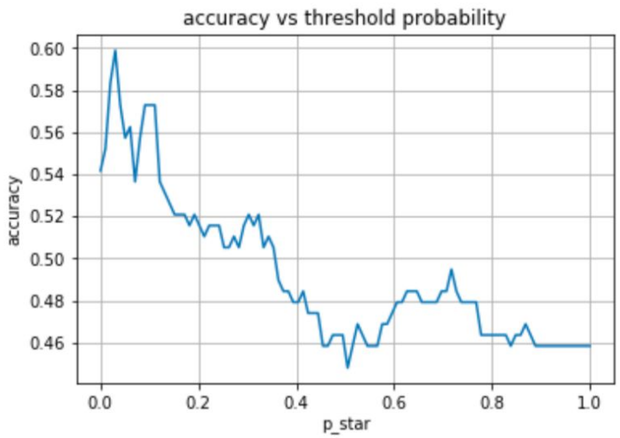


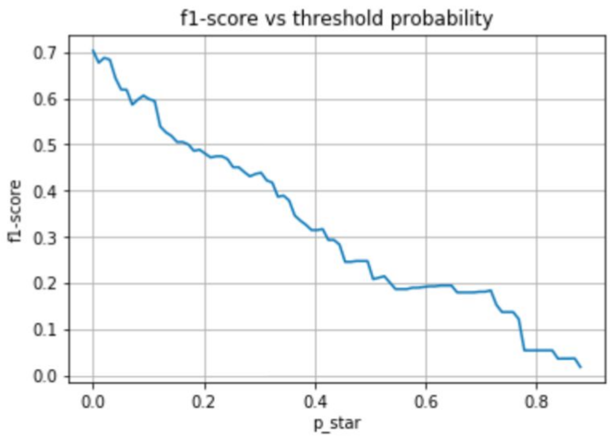
Real Images Classification

- Comparing the manually set labels (based on the rules set out in the excel sheet) and the predicted labels given by XGBoost model:
  accuracy = 0.45 precision = 0.5, recall = 0.15, F1-score = 0.24

- More concerned with precision over recall, since the cost of False Negatives is of the primal concern (falsely classifying a building to not suitable for retrofitting).

- Need more consultation from Build Change to quantify the cost of False Positives and False Negatives





Conclusions

- ❑ Our XGBoost model (with 9 features) yielded a great performance on generated image dataset (82.43% precision), and also worked fine on the real image (50% precision)
- ❑ With more granular search and identification of the features of the houses we might be able to capture more information and provide a more predictive model.
- ❑ In the future, we will also try Neural Style Transfer to transform the styles of the training images (colors, lines, etc.) onto the real images, so that the feature extraction methods we implement on the training images can be more conveniently reused on the real images.