Mannequin Image Detection

Chase Weber

Question

How can we save time labelling images?

- WindowsWear is the world's largest creative database for brands and retailers. Each month, the company captures images of retail window displays around the world, manually analyzes to label features of the images, and uploads the data onto their website.
- A process to automate labelling can cut days of work.

Data Wrangling

- Our data is obtained directly from WindowsWear
- Over 5,000 window display images
  - Monthly images from the year 2022
  - All images from fashion districts in Hong Kong, London, Paris, New York, and Milan
- All images were manually sorted into two folders — those with mannequins and those without
- Per request of WindowsWear, images were compressed to lower resolution using Adobe Bridge
- The sorted and compressed data can be found on my github

Data Cleaning

- **Grayscale:** convert all images to grayscale
- **Scale Images:** set values of images between 0 and 1
- **Adjust Contrast**: adjust the contrast of each image by .7
- **Crop**: crop the photo to include 75% of the original width and 85% the original height
- **Random Flip:** flip images vertically at random

Imbalance Check

- Note, of the 5,000+ images, only 4,912 could be imported correctly
- An imbalance check shows that the data is reasonably distributed

  2,499 contain mannequins

  2,413 do not contain

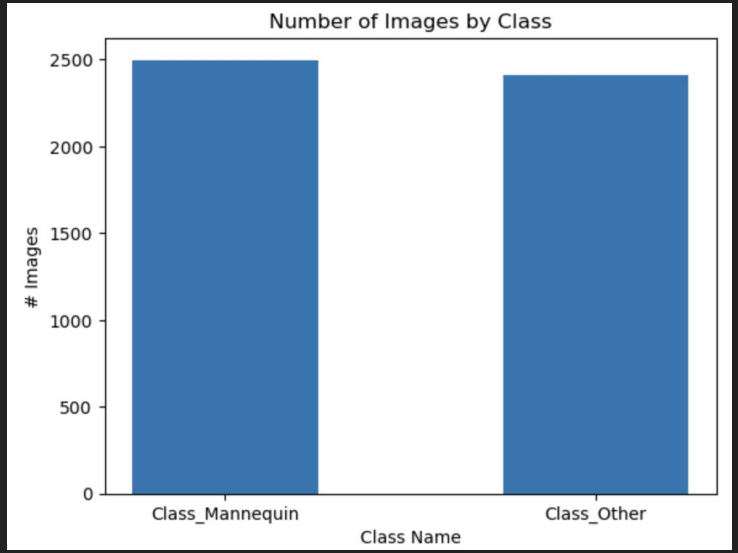


Image Check - raw data

Image Check - Raw Import vs Cleaned

- Images are imported in grayscale
- Images are then contrast adjusted, cropped, and scaled

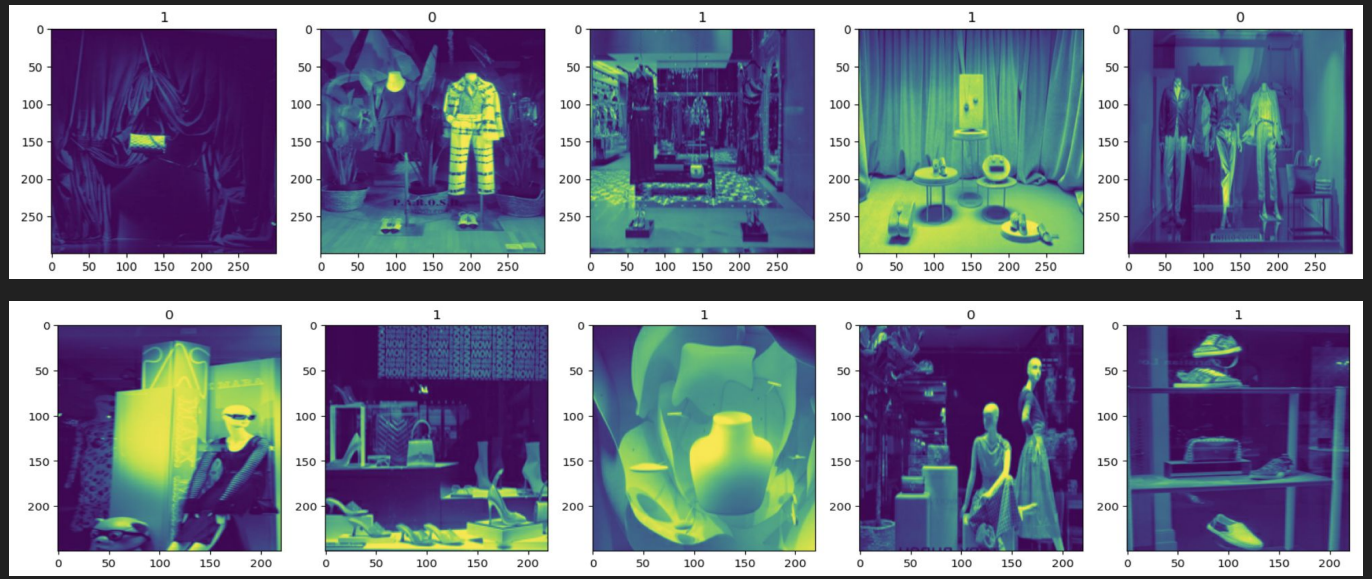
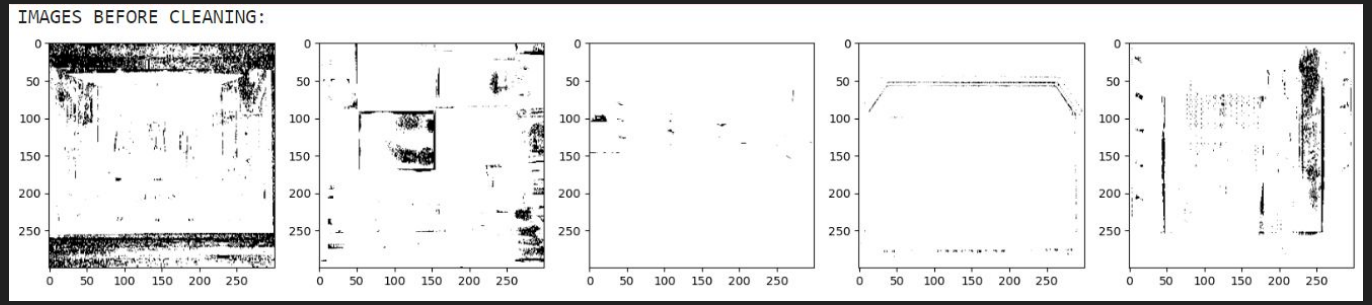
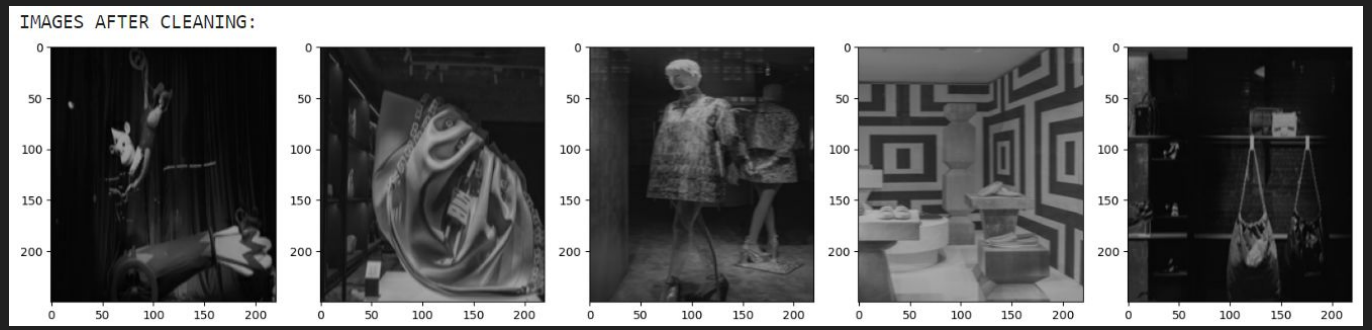


Image Check - Raw Import vs Cleaned

- Same as previous slide, converted to visual black and white



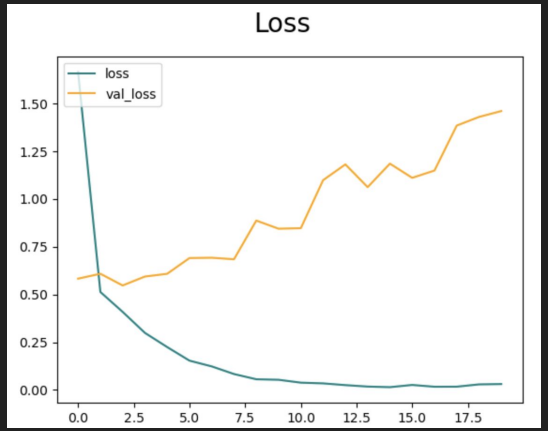


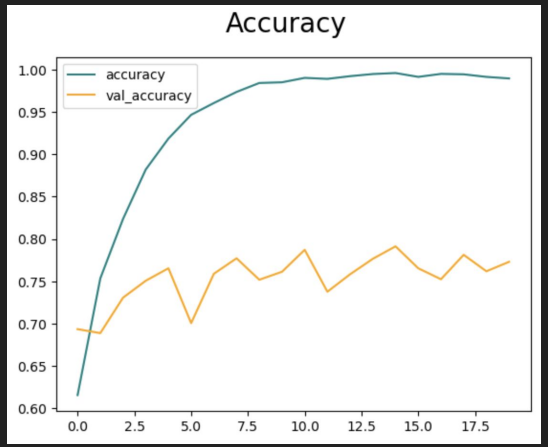
Data Split

- Our data has 4,912 images split into 2 classes
- Images are imported in 492 batches with about 10 images per batch
- Train-Test-Validation Split
    - 70% Train (344 batches)
    - 20% Test (98 batches)
    - 10% Validation (49 batches)

Initial Model Construction

- Convolutional Neural Network
- Initial Construction:
  - 3 hidden layers
  - Maxpooling
  - “relu” activation
  - “adam” optimizer
- Test Scores:
  - Precision : 0.79
  - Recall : 0.61
  - Accuracy : 0.75



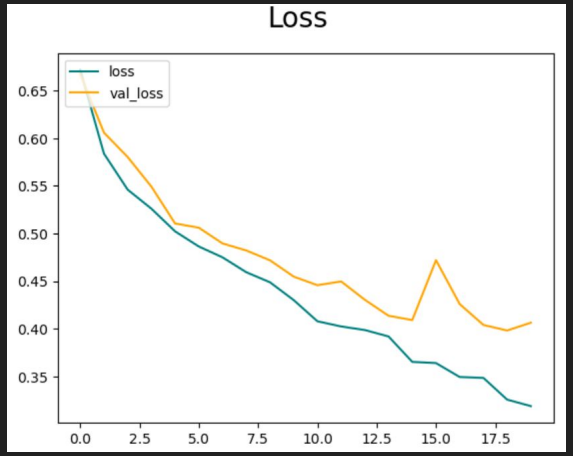


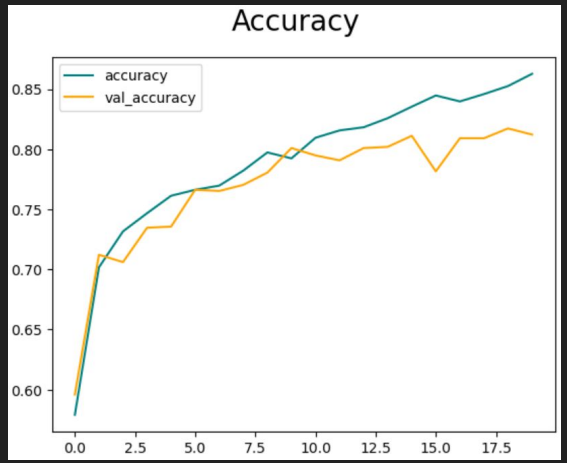
Hyper-parameter Tuning

- Several versions of hyper parameters were tested and tuned
- The following resulted as optimal:
  - Grayscale vs Color - grayscale
  - Cropping Dimensions - 250 x 220
  - Contrast - .7
  - Hidden layers - 5
  - Nodes - 16 - 32
  - Dropout - .25
  - Activation - ReLu

Final Model Construction

- Convolutional Neural Network
- Initial Construction:
  - 5 hidden layers
  - Maxpooling
  - Dropout (.25)
  - 16-32 nodes
  - “relu” activation
  - “adam” optimizer
- Test Scores:
  - Precision : 0.80
  - Recall : 0.80
  - Accuracy : 0.82





Results

- Our regularized model provides the best results with an accuracy of .82 on our test dataset
- This algorithm can be utilized by WindowsWear to accurately predict approximately 82% of their photos correctly
- Additional Scores
  - Precision - .80
  - Recall - .80

Future Improvements

- Reproduce the model using high resolution images
- Include additional images
  - Images from other time-periods, cities, sources, etc
- Analyze images for multi-label feature extraction
- Expand scope of problem to extract from other images
  - Store interior, marketing campaigns, packaging, etc

Appendix

- All work can be found on my GitHub:
  - *https://github.com/chase-weber/Mannequin-Image-Detection*