COSMOP: Synthesize quasi-realistic immigration dataset using GAN

Subin Varghese
Cullen College of Engineering
University of Houston
Houston, Texas
srvargh2@cougarnet.uh.edu

Xiaoqing Liu
College of Natural Sciences and Mathematics
University of Houston
Houston, Texas
xliu81@cougarnet.uh.edu

Sunita Villarreal
College of Natural Sciences and Mathematics
University of Houston
Houston, Texas
svillar8@cougarnet.uh.edu

***Abstract* - We will investigate the effect of image augmentation on the predictive performance of a one-class classifier (SVM) modeled to learn the behavior of immigrant movements. We start by collecting related images for our dataset; preprocessing them; generating the synthetic images; classifying the augmented dataset; and then evaluating the outcome.**

Keywords - GAN, CNN, classification, data augmentation, synthetic images, deep learning

I. Introduction

Large immigration datasets are not readily available due to a lack of research studies conducted in this area. Constructing one is also extremely tedious due to the presence of many feature variations in images of immigrants. Our aim is to investigate the effects of image augmentation on the predictive performance of different classification models, such as one-class classifier (SVM), to learn the behavior of immigrant movements.

A. Problem Statement

Synthesizing realistic immigration images could generate large volumes of augmented image data, which can facilitate advanced computational methods, such as deep learning algorithms.

B. Project Objectives

We will attempt to generate a larger immigration dataset comprising various quasi-realistic immigration images from a smaller dataset of real images collected from various online sources.

II. Related work

A. Generative Adversarial Networks

A Generative Adversarial Network (GAN) [1, 2] consists of two models: a generator (G) and a discriminator (D). G attempts to learn the distribution of a dataset (X). Given random noise (z) G will output an approximated sample from X. D attempts to distinguish generated data from G and the real dataset X. Both D and G are functions that are approximated using a neural network and trained using gradient descent. The loss function used in this framework essentially scores the ability of D to distinguish between X and G(z). Gradient descent can be used on this function to update the parameters in G, while gradient ascent can be used to update the parameters in D. In the case of a dataset consisting of immigration images, G(z) will output a synthetic image of a immigration image.

B. Fréchet Inception Distance

A metric is needed to measure the quality of the synthetic images produced by a GAN. One such metric is the Fréchet Inception Distance (FID) [3], which utilizes the Inception-v3 [4] network (ICN) as a feature extractor. It is assumed features extracted from images using the ICN are from a Gaussian distribution. By extracting features from all of the images in the training set and from the generated images, two distributions can be made - one for the real images in the training set and one distribution from the generated images. By calculating the Fréchet distance between these distributions, the FID can be calculated. The lower this measure the closer the training set distribution is to the generated image distribution.

III. Methodology

A. Real immigration dataset collection

We had to collect images for our dataset since we did not have one to work with. We resorted to webscraping images online using two different methods: Selenium Webdriver and Twitter API.

Since they are online searches that use keywords, we also had to define a set of criteria for how we determined which images were relevant or not. Our criteria required images to have at least one of the following: people carrying bags, people with children, people crowded around an immigration border, and/or border control people. Based on the criteria, our search keywords were "[country] + immigration border" (the country here refers to only those countries with land borders between neighboring countries [5]); "ukraine refugees", "ukraine"; "russia"; "refugees"; "ukrainian refugees"; and "ukraine" war (Athina Bikaki, personal communication, February, 8, 2022).

We were able to collect a total of 22, 000 images using the methods described below. However, it turned out that a majority of them were irrelevant - images that were either duplicates or did not even contain humans (described below). We had to filter out the irrelevant images; after which we were left with only 5, 094 images that were relevant (2, 272 "immigration" images, and 2, 822 "non-immigration" images.

1) Selenium Webdriver

Selenium WebDriver is a web framework that permits you to execute cross-browser tests. It is used for automating web-based application testing to verify that it performs expectedly. It lets one's program do what a human user would normally do on the browser, like searching for something and such can all be possible by using selenium to simulate a user's actions.

For our purposes, we paired up Selenium with Google Chrome and automated it to fetch the images we need for our dataset [6]. To be able to do that we also needed the Chrome driver. Once the Chrome driver was set up, we searched for queries using our set of keywords listed above, then moved into the images section, and got the image links. Then the images were downloaded and saved to a folder of our choice.

2) Twitter API and Tweepy

To access the API, a combination of API access keys are provided after creating an app [7]. We then conducted a search for tweets containing the keywords of interest using the Twitter API. Tweets were then fetched and then filtered for only those that contained media. The image links were obtained, and the images were then downloaded to a folder of choice.

B. Detect and Remove Duplicate Images(xiaoqing)

The image dataset we got after downloading images using web scraping contained many duplicate images. These duplicate images can introduce bias to the dataset, causing the model prone to learn specific of those duplicate features, this'll weaken the model ability to learn the general features. Thus, we must detect and remove those duplicate images.

The strategy we used to implement this goal is described in Figure 1 below:

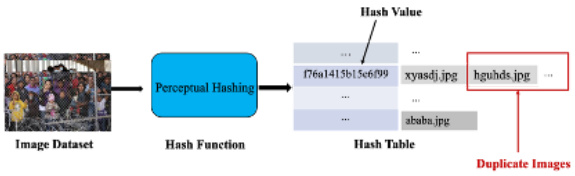

Figure 1. The process to detect and remove duplicate images

The hash function we used in this project is perceptual hash [8]. This method is easy, robust, and quick, if two or more images are the same, this function can yield the same hash value. We can also use pHash to get the similarity of the two images through Hamming distance though we don't need this in this project. The approach for pHash is described in Figure 2.

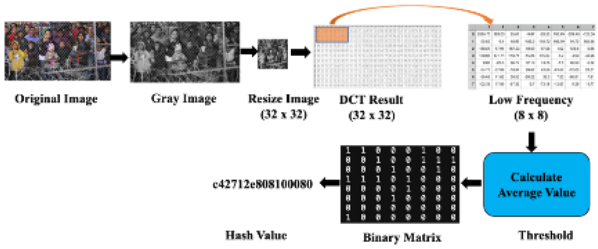

Figure 2. The process of the cnn binary classifier

First, we convert the original image to gray image, resize it to 32x32, then we apply the DCT transform on the 32x32 image, analysis the low frequency 8x8 matrix, using the average value as the threshold to get binary matrix, finally convert the 64 bits binary matrix into 64-bit integer.

C. Human Detection

We train a binary classifier to detect if humans are present in an image. To do so we combine the Pictures without humans for Self-Supervision dataset [9] with the CrowdHuman dataset [10]. We use the ResNext 101 32x8D [11] architecture and train it on the combined dataset. We were able to achieve 90% accuracy. This model was then used to filter images for humans in our immigration dataset.

D. Manual Annotation

After the removal of irrelevant images (duplicate and/or those not containing humans), we had to annotate our images based on our afore-mentioned criteria.

Initially, we attempted to use a classifier to annotate our images. However, despite a seemingly good test accuracy score of 80%, upon inspection of our dataset, we deemed it unsatisfactory, and we proceeded with manually annotating all 5094 images.

E. Generate synthetic images using GAN

Based on our literature search of GAN architectures we have selected StyleGAN2 [12] for its performance on the Flicker-Faces-HQ dataset [13], in which it was able to generate realistic faces. This was a major consideration as realistic facial features are a desired property for the immigration images we would like to generate. We trained StyleGAN2 for a maximum of 18,000k iterations and found the FID was minimum at 10,000k iterations. We present a sample of images generated using the StyleGAN2 architecture trained on our immigration dataset at 10,000k iterations in Figure 3. Additionally Table 1 shows the FID as the number of iterations increases from 0 to 15,000k. We can see the dip at 10,000k. The increase in FID after 10,000k is due to mode collapse.

Figure 3. Sample generated images produced using StyleGAN2

TABLE I: ITERATIONS VS FID FOR STYLEGAN2	
Iterations	FID
0	207.25
10,000k	46.06
15,000k	127.72

F. *Classification*

In this project, we have two purposes to build the classifiers on the real images: one is to validate the generated images from GAN model, to evaluate these synthetic images is relevant to immigration or not; another one is to help us to automatically annotate the new images since annotation manually is too expensive.

We implemented two types of classifiers: CNN binary classifier with transfer learning and one-class classifier with feature extraction.

1) *CNN Binary Classifier with Transfer Learning*

First, we annotate the dataset after pre-processing manually into immigration and non-immigration. Due to the dataset being small, we imply image data augmentation, like rotation, width, height shift, zoom, shear, horizontal flip. Then we feed these data into the pre-trained ResNet50 [14] and fine tune it to get the final model.

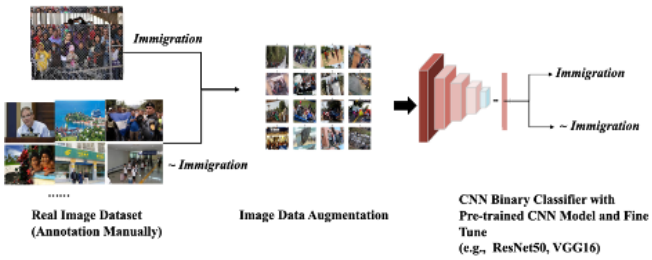

Figure 4. Process of the CNN image classification.

2) *One-Class Classifier with Feature Extraction*

Different from the binary classifier, we can take our problem as a novelty detection problem, the training data consist only of the immigration data, we trained our model to learn the features of these data, and then we can use this model to predict if the new image is an outlier or not. In our project, immigration images may have some common features, however the non-immigration images are quite different.

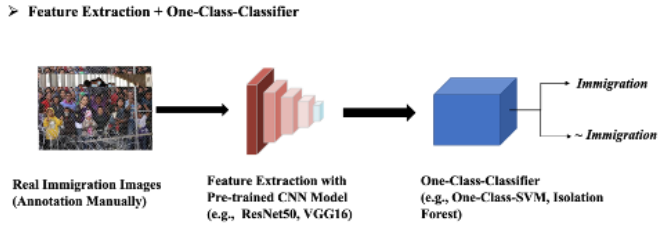

Figure 5. The process of the one-class classifier

In this project, we use one-class SVM as our classifier and pre-trained ResNet50 as the feature extractor. We feed the pre-trained ResNet50 with only the real immigration images to let the model extract the features of immigration. After ResNet50 we get 7x7x2048 dimension features, this is too large to feed to SVM directly, so in our implementation we apply standard scaler and PCA dimensional reduction to those features. We remain 93% explained variance of the original features with the dimension only 1024 features. Finally, we feed these features to one-class SVM to train the classifier. Then we can use this classifier to predict whether the new image is immigration or not.

IV. RESULTS

A. *Classification score for the dataset of real images*

We split the manually annotated real image dataset into training and test.

TABLE II: TRAIN AND TEST DATASET SIZES		
Subset	Immigration	Non-immigration
Train	1357	2100
Test	340	526

We trained the two classifiers on training image dataset and evaluated the performance of CNN binary classifier and one-class SVM on the test dataset with three measures: accuracy, precision, and recall.

Performance Measure on Real Image Dataset

Method	Precision	Recall	Accuracy
Binary CNN Classifier	**0.79**	0.71	**0.81**
One-Class-SVM	0.33	**0.75**	0.31

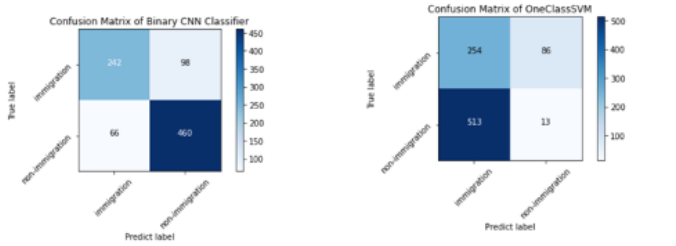

Figure 6. Performance measure on the real image dataset

From Figure 6, the recall of one-class SVM is 4% higher than the CNN binary classifier, which means one-class SVM is better at predicting the positive in the positive dataset. However, one-class SVM is 46% lower than CNN binary classifier due to it being bad at negative dataset, we can see that by comparing the two-confusion matrix, one-class SVM miss classified most of the non-immigration as immigration.

From the misclassified images we can get that: the fence, crowd gathered people are the general features the classifier learned from the training dataset. So those kinds of images are prone to be misclassified.

Figure 7. False positive sample image set and false negative sample image set

*B. Classification score for the dataset of augmented d*ataset

We also do the evaluation on the 2276 synthetic images using CNN binary classifier.

$$\frac{\text{Classified as Immigration Images Number}}{\text{Total Number of Synthetic Images}} = \frac{1605}{2276} = 0.70$$

Figure 8. Classification results of the CNN binary classifier on the synthetic images

From the results, we can see that: the features that GAN model learned through the real images are much more than the classifier, though the synthetic images looked distorted.

V. Conclusion

By scraping images from Google images and Twitter, we were able to make a small dataset of images related to immigration. Using this dataset we were able to successfully train a GAN to generate synthetic immigration images. Though the synthetic images failed to produce realistic faces, quantitatively, they have been shown to align well with the real immigration images. 70% of the synthetic images were classified as real, using a classifier trained on real immigraion images. Additionally, we were able to achieve a minimum FID of 46. Further work is required in experimenting with more GAN architectures as well as further expanding the immigration dataset size. However, we believe this initial dataset can assist in further expanding a future dataset for immigration related images.

VI. Roles

Student	Time Spent per week	Time Spent Total
Subin Varghese	5 hours each week	60
Xiaoqing Liu	5 hours each week	60
Sunita Villarreal	5 hrs each week	60

Student	Tasks
Subin Varghese	Tested web scraping with beautiful soup, GAN literature search, method for filtering for humans in web scrapped images, found datasets to create classifiers for human detection,wrote code and trained human classifier model, wrote code to train StyleGANv2 on Carya cluster, wrote documentation on using written code, wrote all GAN and FID related text in report.
Xiaoqing Liu	Data collection: implement web scraping with Selenium; Dataset pre-processing: do some research and implement detect and remove duplicate images; GAN: read papers and learn the original architecture of GAN; Evaluation: do some research and implement two types of classifiers CNN binary classifier and one-class-svm classifier.
Sunita Villarreal	Handled the collection image dataset using Selenium and Twitter API, investigated and researched

	the use of a classifier for classifying our dataset, manually annotated all the images based on the predetermined criteria, and learned more about StyleGAN

Acknowledgment

We would like to thank Athina Bikaki for her helpful guidance throughout the semester.This work was completed in part with resources provided by the Research Computing Data Core at the University of Houston.

References

[1] I. J. Goodfellow et al., “Generative Adversarial Networks,” Jun. 2014.

[2] I. Goodfellow et al., “Generative adversarial networks,” Commun ACM, vol. 63, no. 11, pp. 139–144, Oct. 2020.

[3] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium,” Advances in Neural Information Processing Systems, vol. 2017-December, pp. 6627–6638, Jun. 2017.

[4] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[5] “List of countries and territories by Land Borders,” Wikipedia, 19-Apr-2022. [Online]. Available: https://en.wikipedia.org/wiki/List_of_countries_and_territories_by_land_borders. [Accessed: 20-Jan-2022].

[6] A. Suresh, “Web scraping images from google,” Medium, 19-Mar-2020. [Online]. Available: https://medium.com/@wwwanandsuresh/web-scraping-images-from-google-9084545808a2. [Accessed: 15-Mar-2022].

[7] T. Nguyen, “How to automatically download pictures from any Twitter feed,” *Medium*, 01-Nov-2018. [Online]. Available: https://medium.com/@nguy3409/how-to-automatically-download-pictures-from-any-twitter-feed-e0f97ae4d193. [Accessed: 08-May-2022].

[8] “Looks like it,” *Looks Like It - The Hacker Factor Blog*. [Online]. Available: https://www.hackerfactor.com/blog/index.php?%2Farchives%2F432-Looks-Like-It.html. [Accessed: 08-May-2022].

[9] Asano, Yuki M., et al. "PASS: An ImageNet replacement for self-supervised pretraining without humans." arXiv preprint arXiv:2109.13228 (2021).

[10] Shao, Shuai, et al. "Crowdhuman: A benchmark for detecting human in a crowd." arXiv preprint arXiv:1805.00123 (2018).

[11] Xie, Saining, et al. "Aggregated residual transformations for deep neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[12] Karras, Tero, et al. "Training generative adversarial networks with limited data." Advances in Neural Information Processing Systems 33 (2020): 12104-12114.

[13] Karras, Tero, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.

[14] *He*, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.