Using Deep Learning Neural Networks for Image Processing

Uddom Lee
ulee001@odu.edu

Peng Jiang
pjiang@odu.edu

Hongyi Wu
h1wu@odu.edu

Chunsheng Xin
cxin@odu.edu

Old Dominion University
School of Cybersecurity

***Abstract*— The Internet has allowed people to visit places virtually at the click of a button. Many different softwares and applications allow users to explore different buildings, events, or historical sites. Google Street View is a software used to see what an area is going to look like before travelling to a specific location. Google Street View also shows what the surrounding area and landscape is going to look like as well. However, Google Street View provides images that are outdated and do not show what the location would look like under different forms of weather or lighting. To try and view different locations at various weather and lighting conditions a database of different scenes was needed. This database needed to have many images and have a trained AI model that could identify the different features of the images. Then another AI model was needed to combine the features of Google Street View and the features of the image database to produce many Google Street View images of the same location with different weather and lighting conditions. This paper explores the steps to convert the Google Street View images into images that can represent a variety of scenarios based on different weather and lighting conditions.**

Keywords — Google Street View, Image processing, deep learning, neural networks, machine learning, security, autoencoder, image feature analysis, image recognition

I. Introduction

Visiting places can easily be done by going online and buying ticket to a specific location. However, that cost time and money. People have created different softwares and applications that allow users to visit a future vacation area, famous buildings, historical sites, or a local area that is 30 minutes away. Some of these applications are free and can even be used on mobile devices.

Google Street View is a software developed by Google and is integrated into Google Maps, a GPS application. Google Street View shows the user a set of images of what a specific location looks like. The user is also able to interact with the images by clicking on certain areas and allowing them to travel as if they were physically at the location. They can also look around and get a visual on what their surroundings would look like if they were at that exact spot. Some applications online use this technology to create a game where the user must guess where they are based on the images shown.

These photos are taken globally and uploaded to the Google Street View database and updated at an infrequent time. These images can help the user locate certain landmarks or look for specific buildings. However, Google Street View provides outdated images to the user depending on the location the user is trying to view.

Many images that are used in Google Street View can be outdated from a day, to months, or years depending on the location. Many cities in certain areas are being developed or constantly changing. This means that an image on Google Street View could show a building that is not there anymore, or not show a new shop that recently opened. Outdated images on Google Street View could confuse the user if they travel to that location.

This paper seeks to explore the different steps needed to convert the images provided by Google Street View to a variety of scenarios that accurately represent the physical location shown.

To convert the images from Google Street View into a new image other images from different locations are needed. The images need to have different times of day, different seasons, and different forms of lighting. These different attributes are going to help show how the Google Street View image can look with these different attributes. To get a variety of images with these different attributes the Flickr database will be used.

The images from Flickr and Google Street View will then need to be combined. This will be done by using an autoencoder which will produce an entirely new image with similar attributes.

To ensure that the old attributes are being transferred over to the new image a scene recognition model will be used to scan the previous images and the new image. The images will then be compared to see if they have any related attributes.

This paper seeks to offer the following:

- A design model on how to obtain images, curate images, and combine them into new images.
- An algorithm on how to calculate the accuracy of an autoencoder model.
- Results of the testing and how to analyze them.

- A discussion on the next steps of improving the current tests.

The paper then goes into section 2 about the related works. Section 3 the design model and the functions that make up that model. Section 4 how the images are collected and processed. Section 5 the evaluations of the data. Section 6 the challenges that were faces when designing and using the model. Lastly, section 7 the conclusion and discussion on what we will seek to do next.

II. Related Work

Karras, Aittala, Janne et al wanted to use StyleGAN2 to create more images for training specific applications. These researchers observed that many specific applications required more images and the more niche the subject was the less images there were [6]. Moreover, some of these specific applications required newer images. Without a variety of images for training the application would produce results that had similar patterns and not a variety of sets. These researchers aimed to use StyleGAN2 to produce new training images that prevented a pattern of results from forming [6].

Brock, Lim, Ritchie, and Weston saw an issue with the original Generative Adversarial Network (GAN) and the Variational Autoencoder (VAE). They saw that GAN had unstable training dynamics but produced realistic photo [2]. VAE also had issues of discarding high-frequency details but had stable training dynamics [2]. To fix the issues of both models the researchers developed their own model called the Introspective Adversarial Network (IAN) which was a combination of the GAN and VAE models. The IAN model used and interpolating mask, with multiscale dilated convolution blocks, and orthogonal regularization to produce small quality changes on pre-existing images [2].

CNN models trained in object recognition were found to also be good extracting high-level image content, textures, and artistic style [4]. Gatys, Ecker, and Bethge created a "Neural Algorithm of Artistic Style" [4] which allowed a CNN model to accurately transfer the style of an artist to another pre-existing image. It was found that a CNN model can separate an image between the contents of the image and the style of an image [4]. With the proper algorithm the CNN model could freely manipulate the content and or the style of an image.

Zhou, Lapedriza, Khosla et al created the Places365 database inferred that a "place and context is important for an intelligent system" for scene recognition [1]. Places consist of various parts and features that can be broken down to identify what the place is representing. They created a database of different scenes that many people would similarly see every day [1]. They then decided to train a convolution neural network using the Places365 database to compare how well the CNN model fare in "place recognition [1]." Then compared how well the CNN model would perform against other "visual recognition" tasks and wanted to show how the Places365 database could be used to train other models [1].

Park, Zhu, Wang et al wanted to create an "image editing process" that allowed users to combine images together, replace objects, and change the time of day the image was taken. They saw that current photo editing tools were limited based on the edited image. [8]. This meant that the photo editing tool would not be able to easily change the features of a photo like time of day. They wanted to use machine learning with an autoencoder to create a software capable of these new photo editing tasks [8].

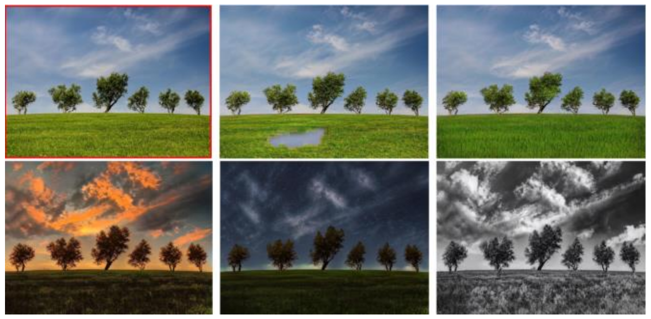


Figure 1: Example of the attribute changing between the images from another research paper [8].

III. The Design Model

A. Flickr

Flickr is an image database where users join to upload their images to the database. The database is updated frequently so the images within the database are often changing or being archived with newer images [3]. This database will be used obtain images to use in the model. Flickr ensures that the model has a variety of images with different lighting, weather, and file sizes. The Flickr images will be used as one of the base images to be combined with the images from Google Street View.

B. Places365

Places365 is a deep learning neural network and scenery database. [1]. The Places365 model is used for scanning images and showing what features the image has. This is used in two ways: one to curate images from Flickr looking for a specific attribute and two to show what attributes the image from Flickr has and the attributes the new combined image will have. To get a general group of attributes a base keyword is used keywords like forest, cloudy, snow, etc.

C. Google Street View

Google Street View is also another image database where each image has a GPS coordinate tied to the image's location. The images are also connected to one another which allows any user using Google Street View to travel online using the images provided. For example, a user can view and travel the streets of New York City from a computer in California. The images from Google Street View will also be used as one of the base images to be combined with the images from Flickr.

D. Swapping Autoencoder

The two main parts of the AI model contain an encoder and a decoder. The encoder will read the image and the decoder will create the same image again. By having the decoder create the same image the autoencoder can then manipulate the image and change specific features like lighting, weather, or the texture of a building [8]. The autoencoder was used to combine two images to produce a new image with similar attributes or attributes that are carried over.

$$Places365(Flickr) = P_F \,\&\, A_F$$

The Places365 model can be seen as a function to curate the images from the Flickr database and produce attributes associated with the curated Flickr images.

$$AE(P_F, GSV) = P_{AE}$$

The Autoencoder model can also be seen as a function combining P_F, the images from Flickr curated by Places365, and the Google Street View images to produce P_{AE}, the autoencoder's output.

$$Places365(AO) = A_{AE}$$

A_{AE} is the attributes from P_{AE} produce by the Places365 model.

$$Places365(GSV) = A_{GSV}$$

A_{GSV} is the attributes from the Google Street View images produced by the Places365 model.

Flickr	Image database
AE	Autoencoder model
Places365	Model for finding attributes of images
Keywords	Words to curate for specific images
P_F	Flickr image curated by Places365
P_{GSV}	Google Street View images
P_{AE}	Autoencoder output
A_F	Flickr image's attributes
A_{GSV}	Google Street View attributes
A_{AE}	Autoencoder output's attributes

Table 1: Legend for the flowchart

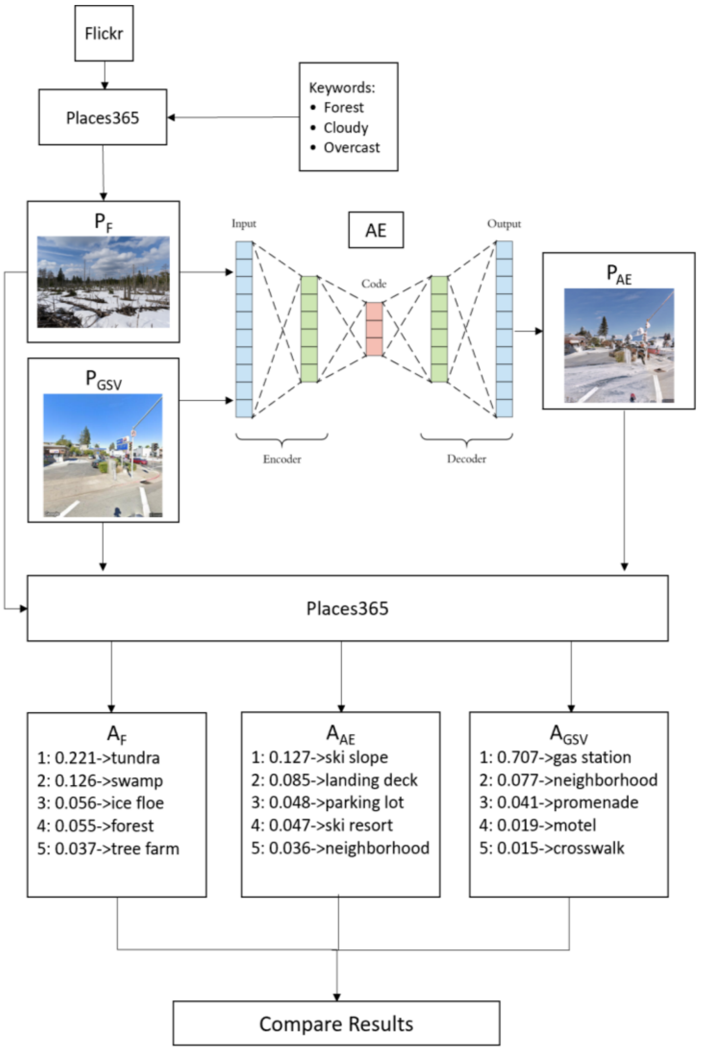


Figure 2: Flowchart on how all the functions work together (AE image from [7].)

IV. Data Collection

Images were gathered from the Flickr database [3]. Then processed using the CNN model within Places365 to identify the different types of images given to the neural network [1].

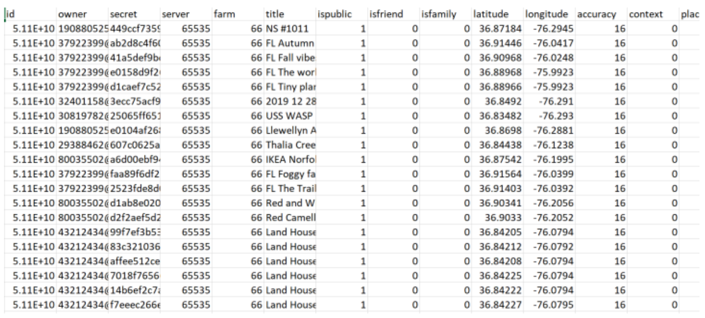
A. *Indefinite Source*

Originally, to associate images and coordinates with GPS the images from Google Street View were obtained [5]. However, the images were not up to date. Google Street View is categorized as an indefinite source because the images were taken at a broad timeframe and were updated at an unknown frequency, this led to the use of Flickr.

Figure 3: Image from Google Street View

B. Time-Sensitive Source

Flickr as mentioned before is an image database [3]. The database allows users to upload images taken on a specific date or time and categorize them [3]. This means the images can be organized based on the time of day, date, or season. Flickr is a time-sensitive source because each image that is taken can be updated, and the database is updated frequently.

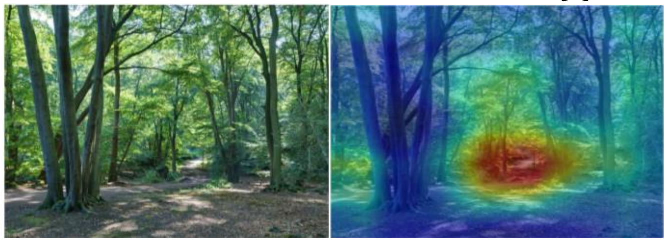


id	owner	secret	server	farm	title	ispublic	isfriend	isfamily	latitude	longitude	accuracy	context	plac
5.11E+10	190880525	449ccf735	65535	66	NS #1011	1	0	0	36.87184	-76.2945	16	0	
5.11E+10	37922399@	ab2d8c4f6	65535	66	FL Autumn	1	0	0	36.91446	-76.0417	16	0	
5.11E+10	37922399@	41a5def9b	65535	66	FL Fall vibe	1	0	0	36.90968	-76.0248	16	0	
5.11E+10	37922399@	e0158d9f2	65535	66	FL The worl	1	0	0	36.88968	-75.9923	16	0	
5.11E+10	37922399@	d1caef7c52	65535	66	FL Tiny plar	1	0	0	36.88966	-75.9923	16	0	
5.11E+10	32401158@	3ecc75acf9	65535	66	2019 12 28	1	0	0	36.8492	-76.291	16	0	
5.11E+10	30819782@	25065ff651	65535	66	USS WASP	1	0	0	36.83482	-76.293	16	0	
5.11E+10	190880525	e0104af26	65535	66	Llewellyn A	1	0	0	36.8698	-76.2881	16	0	
5.11E+10	29388462@	607c0625a	65535	66	Thalia Cree	1	0	0	36.84438	-76.1238	16	0	
5.11E+10	80035502@	a6d00ebf9	65535	66	IKEA Norfol	1	0	0	36.87542	-76.1995	16	0	
5.11E+10	37922399@	faa89f6df2	65535	66	FL Foggy fa	1	0	0	36.91564	-76.0399	16	0	
5.11E+10	37922399@	2523fde8d	65535	66	FL The Trail	1	0	0	36.91403	-76.0392	16	0	
5.11E+10	80035502@	d1ab8e020	65535	66	Red and W	1	0	0	36.90341	-76.2056	16	0	
5.11E+10	80035502@	d2f2aef5d2	65535	66	Red Camell	1	0	0	36.9033	-76.2052	16	0	
5.11E+10	43212434@	99f7ef3b53	65535	66	Land House	1	0	0	36.84205	-76.0794	16	0	
5.11E+10	43212434@	83c321036	65535	66	Land House	1	0	0	36.84212	-76.0792	16	0	
5.11E+10	43212434@	affee512ce	65535	66	Land House	1	0	0	36.84208	-76.0794	16	0	
5.11E+10	43212434@	7018f7656	65535	66	Land House	1	0	0	36.84225	-76.0794	16	0	
5.11E+10	43212434@	14b6ef2c7	65535	66	Land House	1	0	0	36.84222	-76.0794	16	0	
5.11E+10	43212434@	f7eeec266	65535	66	Land House	1	0	0	36.84227	-76.0795	16	0	

Figure 4: Flickr excel spreadsheet query

C. Places365

The Places365 model has a database of different sceneries. The database was developed by looking at millions of different sceneries and identifying the different attributes within the sceneries [1]. This database was used to train the model to help predict what the image that is being shown to the model is. The Places365 model shows the type of environment, scene categories, scene attributes, and another image of what the model looked at to determine the different attributes [1].



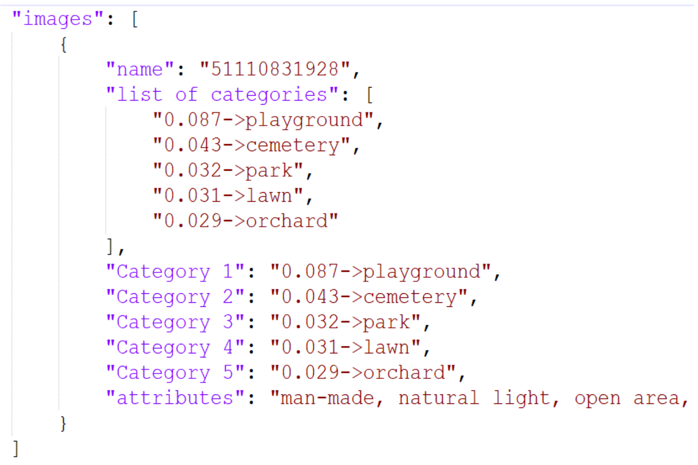
Predictions:

- **Type of environment:** outdoor
- **Scene categories:** forest path (0.326), forest/broadleaf (0.165)
- **Scene attributes:** trees, natural light, foliage, leaves, vegetation, open area, natural, no horizon, camping

Figure 5: Places365 Model predicting the different attributes of an image [1]. Also, producing a heatmap of what the model was looking at.

D. Data Processing

All images used to obtain specific scenic features were from Flickr. The Flickr API was used to gather images for the Places365 model to predict the environment, scene category, scene attribute, and heatmap image. The image information was then taken into a Python function to sort the different environments into two categories, indoor and outdoor. The images would be placed in the respective folders based on their environment and all the image attributes was saved in a JSON file to allow for further analysis.



```
"images": [
    {
        "name": "51110831928",
        "list of categories": [
            "0.087->playground",
            "0.043->cemetery",
            "0.032->park",
            "0.031->lawn",
            "0.029->orchard"
        ],
        "Category 1": "0.087->playground",
        "Category 2": "0.043->cemetery",
        "Category 3": "0.032->park",
        "Category 4": "0.031->lawn",
        "Category 5": "0.029->orchard",
        "attributes": "man-made, natural light, open area,
    }
]
```

Figure 6: Example of JSON file for images

In addition to the sorting of images, there is also a search function. After the Python script is ran another python script will look for a category to search for. This is done by executing the python script with the "-s" parameter. If the user adds a category to search for them an output will be shown to the screen of all the listed files with the specified category.

The python search script is done by looping through the entire folder of JSON files. Then reading each JSON file and checking if the parameter the script is looking for is in the JSON file. If the JSON file does not have what the python script is looking for then it will continue with the loop. If the JSON file has the parameter, the python script is looking for then the python script will save the file name of the JSON file to a dictionary with the key being the JSON file and the value being the parameter the python script was looking for. After the python script has finished running the script will output the dictionary to the screen with a list of all the files with the parameter the python script was looking for.

Once the Flickr images were collected, they were placed in in the style folder for the swapping autoencoder. The swapping autoencoder has two main folders for its models: the structure, and the style folder [8]. The images that were obtained from Google Street View were placed in the structure folder. While the images collected from Flickr were placed in the style folder. Park, Zhu, Wang, et al. refer to the different styles as textures [8]. Having the Google Street View images in the structure's folder means the new images will be based on the Google Street View structure.

Below are images that were chosen from Flickr to test the ability of the different models from the swapper autoencoder. There are faces, 512 and 1024, mountain, church, and bedroom models. The mountain, church, and bedroom models were tested since the models were trained to swap environments and not facial features. Figures 11, 12, and 13 show the different models swapping the textures from images with the forest tag.

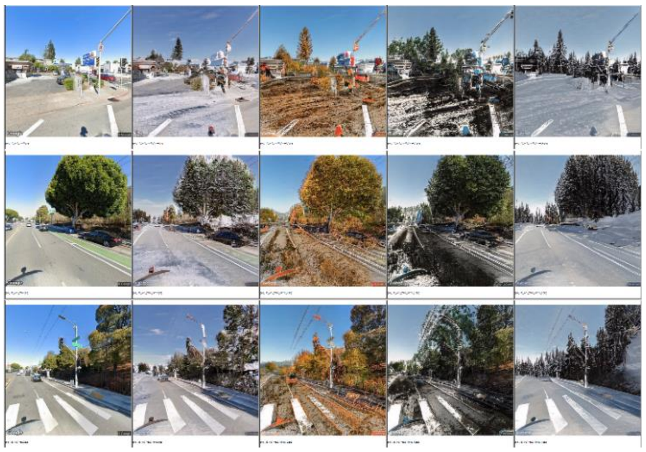


Figure 7: The mountain model using forest images to convey different weather and lighting conditions.

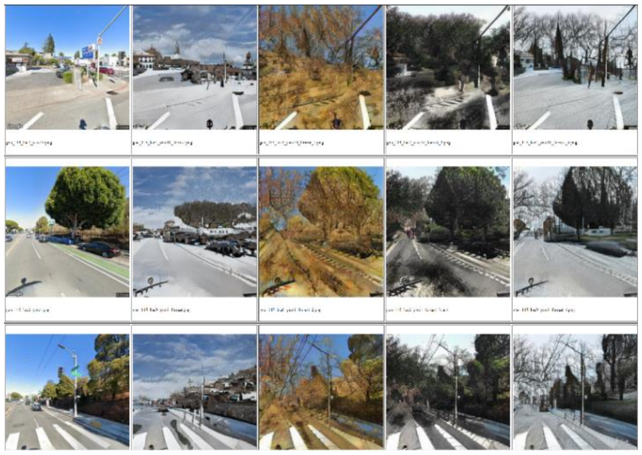


Figure 8: The church model using forest images *to* convey different weather and lighting conditions.

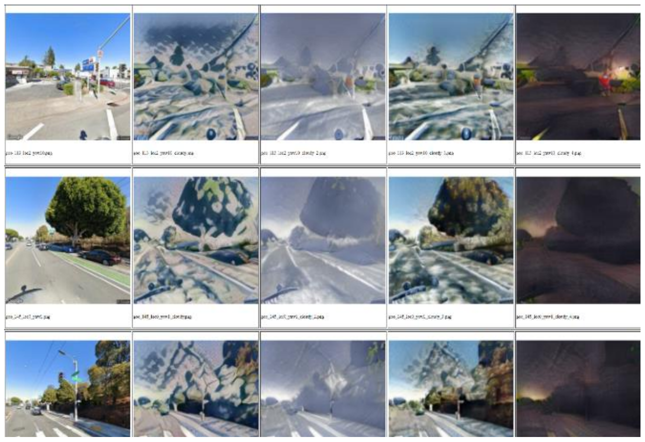


Figure 9: The bedroom model using forest images to convey different weather and lighting conditions.

The first column in figures 7, 8, and 9 show the original Google Street View images. The subsequent columns showed the new images with the textures from the images in the style folder, the Flickr images, combined with the Google Street View images in the structure folder. Using four different image sets: forest, cloudy, overcast, and roads to create a matrix to compare the models on how well the model kept the structure of the original Google Street View images, subjectively. The "X" shows that the model did not retain the original structure. The "O" shows that the model kept the original structure of the Google Street View images.

Bedroom	X	X	X	X
Church	X	O	X	X
Mountain	X	O	X	O
	Cloudy	Forest	Overcast	Roads

Table 2: Subjective table on how well the images look based on the model

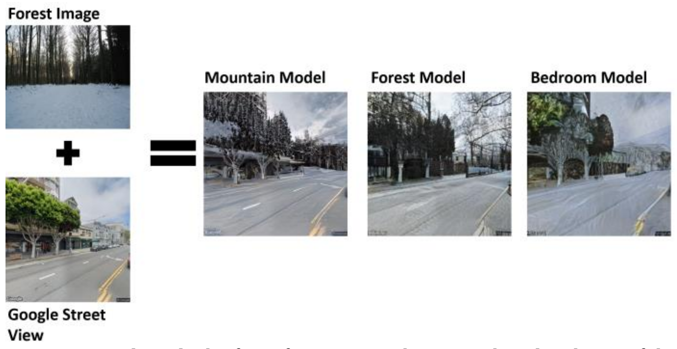


Figure 10: A closer look of one forest images being combined with one of the Google Street View images. Each model produces a different image.

Here as Figure 10 states a closer look at how the Google Street View and Flickr images are being combined to create three new images. The mountain, forest, and bedroom model create somewhat similar images. The three models keep the road and the trees within the images. These three models also have trouble keeping the buildings behind the trees in the image. Upon a closer inspection of the three model images, it shows that the texture of the road and sky is different from the original Google Street View image. Also, the forest model image adds extra branches in the top right corner. While the images are not perfect it can be inferred that the original Google Street View image was changed to include snow on a cloudy day.

V. Evaluations

To determine if the new images created by AE were accurate images they would be used as another input in the Places365 model to obtain their attributes. A_F, A_{AE}, and A_{GSV} attributes were compared with a recall rate of five. The Places365 model would output the top five attributes, this meant the attributes with the highest percentages determined by Places365, from the new images. AE has three different models as mentioned before mountain, church, and bedroom. Each model had four sets of images: cloudy, forest, overcast, and roads. From these image sets twelve images were taken each with five attributes for a total of sixty attributes across the twelve images. If a previous attribute from A_F or A_{GSV} was seen in A_{AE} that attribute was added to the total amount of correct attributes carried over, this will be the rules for a strict set of matching. Only attributes that were on the previous images A_F or A_{GSV} will be tallied. Then they will be divided by the total amount of attributes to obtain the average accuracy of the model.

For a subjective matching, if any attributes related to the respective image groups were seen, then those attributes were added to the total amount of matched attributes carried over. For example, if the A_{AE} in the forest image set had any attributes related to forests, flora, or fauna those attributes would be added to the total correct attributes. Once the total matched attributes carried over were tallied, they were then divided by the total amount of attributes in the image set to produce the average that respective model would have in carrying an attribute over to A_{AE}.

$$Accuracy = \frac{\sum_{n=1}^{N}\sum_{k=1}^{K}(A_{nk})}{N \times K}$$

This is the algorithm used to find the average accuracy percentage in each table. $\sum_{n=1}^{N}\sum_{k=1}^{K}(A_{nk})$ represents the summation of all the matched attributes from each image in the category. N represents the total number of images. K represents the total number of attributes. A_{nk} is a binary value of either 0 or 1. $A_{GSV}(n)$ is the attribute list of the nth GSV image. $A_F(n)$ is the attribute list of the nth Flickr image. $A_{AE}(n)$ is the attribute list of the nth AE image. A_{nk} is equal to 1 if the Kth attribute of the nth image belongs to the attribute list of the nth GSV image or the nth Flickr image.

$$O_{GSV}(n) = \sum_{k=1}^{K} G_{nk}$$

G_{nk} is equal to 1 if $A_{GSV}(n)$ Kth belongs to $A_{AE}(n)$. G_{nk} is equal to 0 if $A_{GSV}(n)$ Kth does not belongs to $A_{AE}(n)$.

$$O_F(n) = \sum_{k=1}^{K} F_{nk}$$

F_{nk} is equal to 1 if $A_F(n)$ Kth belongs to $A_{AE}(n)$. F_{nk} is equal to 0 if $A_F(n)$ Kth does not belongs to $A_{AE}(n)$.

$$R(n) = Jain\big(O_{GSV}(n), O_F(n)\big)$$

$$= \frac{\big(O_{GSV}(n) + O_F(n)\big)^2}{2(O_{GSV}(n)^2 + O_F(n)^2)^2}$$

This equation will judge the how many attributes from A_{GSV} and A_F carried over to A_{AE}. This equation is known as the Jain index which will judge fairness. Fairness meaning A_{AE} attributes' come from A_{GSV} and A_F equally. For example, if the total attributes of A_{AE} are 4 then 2 attributes should come from A_{GSV} and 2 attributes should come from A_F.

	Cloudy		Forest		Overcast		Roads	
	Sub.	Str.	Sub.	Str.	Sub.	Str.	Sub.	Str.
Bedroom	0%	0%	0%	0%	0%	0%	0%	0%
Church	15%	15%	5%	0%	10%	10%	8.33%	8.33%
Mountain	10%	5%	5%	3.33%	8.33%	1.67%	0%	0%

Table 3: Average accuracy at a recall rate of 1. The "Sub." means subjective ruling and the "Str." means strict ruling.

	Cloudy		Forest		Overcast		Roads	
	Sub.	Str.	Sub.	Str.	Sub.	Str.	Sub.	Str.
Bedroom	5%	5%	28.33%	11.67%	1.67%	1.67%	18.33%	10%
Church	33.33%	31.67%	50%	28.33%	20%	20%	43.33%	33.33%
Mountain	41.67%	41.67%	40%	40%	46.67%	46.67%	6.67%	6.67%

Table 4: Average accuracy at a recall rate of 5.

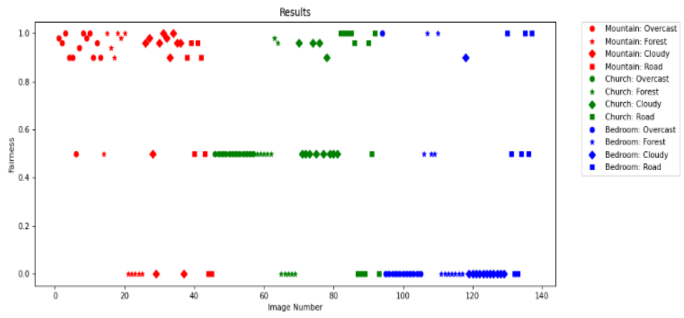


Figure 11: Jain Index Results of the mountain, church, and bedroom model

In Figure 11 the data points are broken into three different colors and four different shapes. The data points in red come from the mountain model, the data points in green come from the church model, and the data points in blue come from the bedroom model. Then the data points with the circle symbol are from the overcast image group, the data points with the star symbol are from the forest image group, the data points with the diamond symbol are from the cloudy image group, and the data points with the square symbol are from the road image group.

Figure 11 shows the Jain index results from all the images that were tested. The results that were the closest to 1.0 showed the image had an even distribution of attributes from both Flickr and Google Street View. Results that have a Jain index result of 0.5 or less shows an uneven distribution of attributes from both Flickr and Google Street View. Lastly, results with a Jain index of 0 show that the image did not carry any attributes from either the Flickr or Google Street View image.

Based on the results in Figure 11 the mountain model created more images with a Jain index closer to 1.0 than the church and bedroom model. While the bedroom model performed the worst out of the three models, by having the most images with a Jain index result of 0. The church model had the most images with a Jain index of 0.5, if the model was trained more the church model has an opportunity to perform just as well as the mountain model.

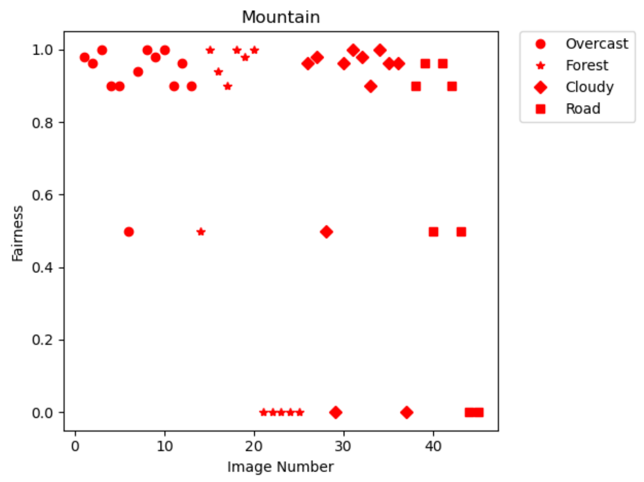


Figure 12: Jain Index of the Mountain model

Based on figure Figure *12*, the mountain model had the most trouble with the forest image group. The mountain model also performed very well with the overcast and cloudy image group. Overall, the mountain model with more training could achieve higher Jain index results. However, since the model was mostly trained on images consisting of different mountains the model would struggle to adapt to images with many buildings.

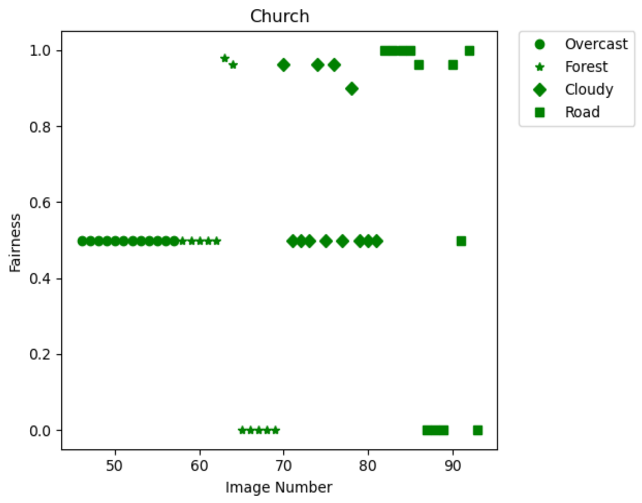


Figure 13: Jain Index of the Church model

Based on Figure 13, the church model had trouble with the overcast image group. All the overcast images only have a Jain index of 0.5. With more the training the church could achieve results similarly to Figure *12*, the mountain model. However, the church model was primarily trained on buildings so any images that do not have many buildings would pose a problem for the church model.

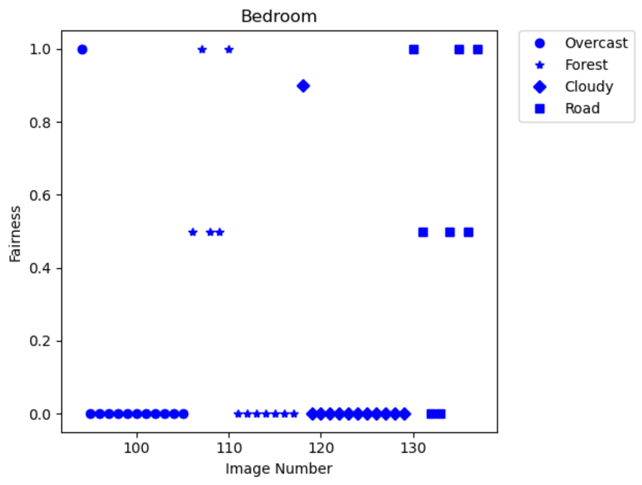


Figure 14: Jain Index of the bedroom model

Based on Figure 14, the bedroom model struggled with all the image groups, only a few of the images had a Jain index of 1. Moreover, even less images had a Jain index of 0.5 and many images had a Jain index of 0. Though this was not too surprising since the model was trained on images based on indoor sceneries and feature.

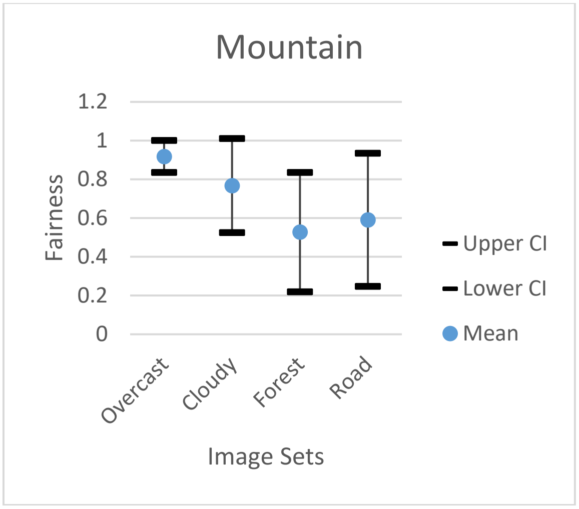


Figure 15: 95% Confidence Interval Graph for the Mountain model

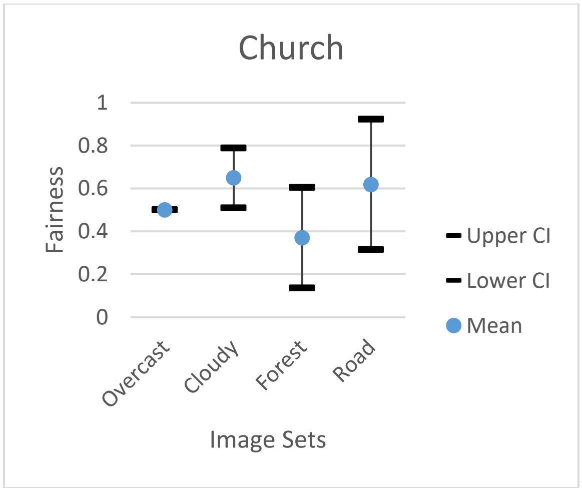


Figure 16: 95% Confidence Interval Graph for the Church model

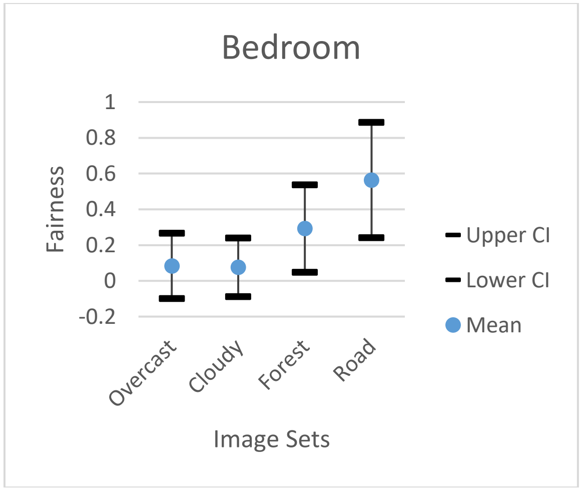


Figure 17: 95% Confidence Interval Graph for the Church model

Figure 15, Figure *16*, and Figure *17* show a 95% confidence interval graph for each individual model. Each model has the image sets of overcast, forest, cloudy, and road. Based on Figure 15, Figure *16*, and Figure *17* the bedroom model showed the worst performance since the overcast and cloudy image sets had most of its fairness data points at 0 and the forest and road image sets had a wide range of data points. The mountain model had performed the best compared to the forest and bedroom models. The mountain model's overcast image set had most of its data points above 0.8. While the mountain model did not perform well with the forest image set showing a wide range of data points the cloudy image set from the mountain model performed better than the forest model by having a better average of data points above 0.8.

VI. Challenges

Because of the ongoing research with deep learning neural networks. New information is being updated at an infrequent pace.

Google Street View was originally proposed to be used because of its wide use, images, and coordinates associated with an image. However, as mentioned before, Google Street View was not updated often. Infrequent updates meant the model would become outdated. The autoencoder may be trained on images of an area in 2012 and images from the same area in 2022 may be different images.

New functions that were proposed did not have enough time to be used. The Swapping Autoencoder model released its official code in April 2021. The Swapping Autoencoder model is a new model that was released to show the changes in images [8]. The Swapping Autoencoder model would be able to show what a city looked like in the rain, snow, summer, fall, night, day, etc. By using the Swapping Autoencoder model the images from the Flickr database could be used to artificially update the images from Google Street View. However, as mentioned before the model was released in April 2021.

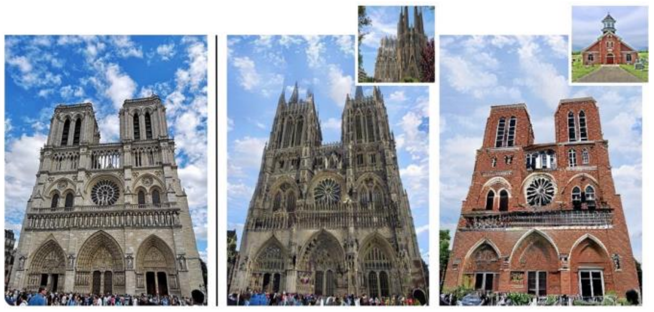


Figure 18: Swapping Autoencoder Model changing the style and texture of the same building [8].

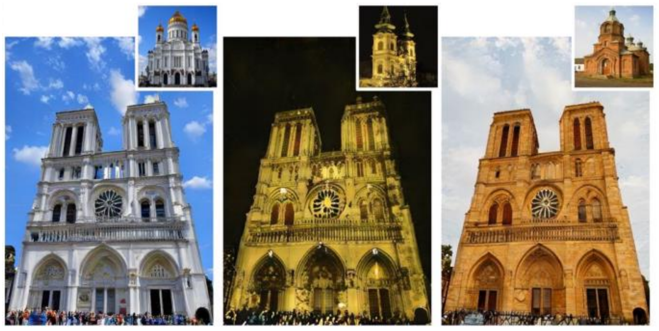


Figure 19: More examples of the Swapping Autoencoder Model with an emphasis on the lighting change [8].

With a new model it was not tested for different environments. Any specific issue a user had would need to be fixed by the user or the issue had to be sent to the creator of the model to fix in an unknown amount of time. This model was found to only be compatible in a Linux environment. The parameters used were not compatible with the Windows 10 parameters. Moreover, a GPU with CUDA version 10.1 is required to run the Swapping Autoencoder model.

Though later throughout the research a correct environment was used to run the Swapping Autoencoder model it had one flaw of not being able to choose the areas that were swapped. The Swapping Autoencoder research paper and website showed a graphical user interface, or GUI, that manipulated the environment and change the image in real-time [8]. However, that GUI was not in the source code and was not able to be used in this research. To circumvent the issue the Swapping Autoencoder model was built using StyleGAN2 another deep learning neural network that also changed the images of different images that were fed to the neural network [6].

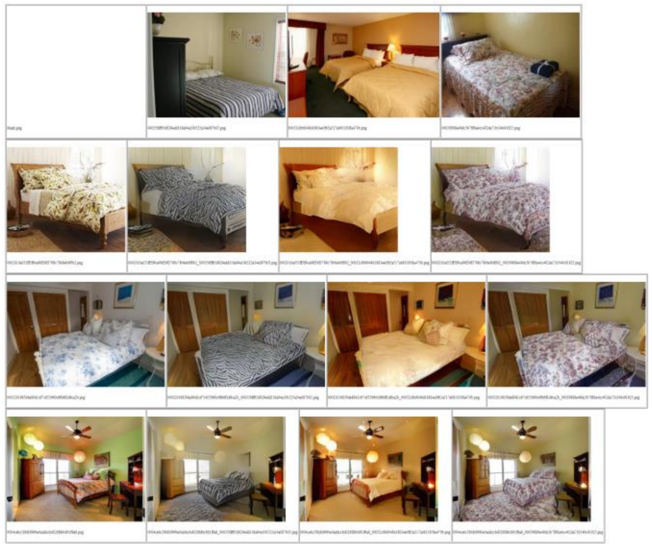


Figure 20: Changing the bedroom images without choosing the textures to change [8].

VII. Discussion

Given more time and more data an autoencoder trained in changing the features of cities is going to perform much better in feature swapping with images from Google Street View. More data includes even more weather variations with different amounts of lighting. This also includes using more images from Google Street View.

Google Street View has also adopted a program for trusted users to upload images they have taken. This may lead to the opportunity to provide Google Street View the most up to date images of a local area. This would allow an autoencoder the opportunity to be trained on the most up to date image of a certain area. This autoencoder may be able to change the features of a specific town or city with these updated images. However, the same problem persists that Google Street View updates its images at an unknown time and the process to getting user submitted images on Google Street View. The upfront cost of the cameras used to take Google Street View compatible photos may also dissuade users from doing personal projects.

Another aspect that could be compared is how humans interpret the images. While the autoencoder or the CNN model can pick out the different features and assume what the features are and what needs to be changed, humans can also do the same. Moreover, the human group could be tested on identifying the autoencoder generated images versus the images taken from Google Street View. The human group could also be given a small survey about how accurate they believe the images to be and if it would help them in their day to day lives.

VIII. Conclusion

Images online are useful for users to look for different landmarks and find out more about a specific location. Using tools like Google Street View helps users correlate an image to the location chosen. However, Google Street View still limits the user with its outdated images, unknown update times, and infrequent update times. By combining an updated image database, Flickr, and a deep learning neural network this paper showed that a conversion is possible to use a deep learning neural network to artificially update the images provided by Google Street View. In conclusion, the mountain model from the swapping autoencoder [6] performed the best. The swapping autoencoders still need to be trained for a long period of time with a wide variety of image sets to create more accurate and concise results.

REFERENCES

[1] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva and A. Torralba, "Places: A 10 Million Image Database for Scene Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 6, pp. 1452-1464, 1 June 2018, doi: 10.1109/TPAMI.2017.2723009. Verma, A. (2020, June 30). Pytorch [Basics]-Intro to CNN. Retrieved August 06, 2020, from https://towardsdatascience.com/pytorch-basics-how-to-train-your-neural-net-intro-to-cnn-26a14c2ea29

[2] Brock, A., Lim, T., Ritchie, J. M., & Weston, N. (2016). Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093*.

[3] Flickr API. (2021). Retrieved April 26, 2021 from https://www.flickr.com/services/api/

[4] Gatys, L.A., Ecker, A.S., & Bethge, M. (2016). Image Style Transfer Using Convolutional Neural Networks. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2414-2423.

[5] Jiang, P., Wu, H., & Xin, C. DeepPOSE: Preventing GPS Spoofing Attack via Deep Neural Network.

[6] Karras, T., Aittala M., Janne H., Laine S., Lehtinen J., & Aila T. (2020) Training Generative Aderversarial Networks with Limited Data. ArXiv, abs/2006.06676v2.

[7] Matthew Stewart, P. D. R. (2020, July 29). *Comprehensive introduction to autoencoders*. Medium. Retrieved November 2, 2021, from https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368.

[8] Park, T., Zhu, J., Wang, O., Lu, J., Shechtman, E., Efros, A.A., & Zhang, R. (2020). Swapping Autoencoder for Deep Image Manipulation. ArXiv, abs/2007.00653.

[9] Pulkit S. (2019, October 1). Build an Image Classification Model using Convolution Neural Networks in PyTorch Retrieved April 26, 2021 from https://www.analyticsvidhya.com/blog/2019/10/building-image-classification-models-cnn-pytorch

[10] Pulkit S. (2019, September 17). A Beginner-Friendly Guid to PyTorch and How it Works from Scratch. Retrieved April 26, 2021 from https://www.analyticsvidhya.com/blog/2019/09/introduction-to-pytorch-from-scratch/?utm_source=blog&utm_medium=building-image-classification-models-cnn-pytorch