TRANSFORMING SATELLITE IMAGES INTO GOOGLE MAPS FORMAT

Salma Elmalky
Student# 1010855371
sal.elmalky@mail.utoronto.ca

Muhammad Ahmed
Student# 1010603839
muhd.ahmed@mail.utoronto.ca

Mohammed Suwan
Student# 1009842093
mohammed.suwan@mail.utoronto.ca

Chenhan Xu
Student# 1010874047
chenhan.xu@mail.utoronto.ca

ABSTRACT

This research project aims to train a conditional GAN to turn satellite images into Google Maps style images. Deep neural networks enable the extraction and classification of information from images. Although maps continue to play a vital role in society, researchers have explored using image-to-image translation techniques to convert aerial and satellite imagery into map representations. The developed generator uses a U-Net: it down-samples the input image to capture context, then up-samples while skipping connections to keep details. The discriminator is a PatchGAN: it looks at 70X70 pixel patches and learns to tell real maps from fakes. For data, we collected paired satellite-map tiles, resized them to a common resolution, normalized pixel values, and augmented the data. Early results show sharp roads and buildings with clear edges. The research and project aim to showcase the effective methods of artificial intelligence in transferring satellite data into user-friendly maps.

1 INTRODUCTION

Accurate, up-to-date maps support everything from urban planning and disaster response to everyday navigation. Yet manually annotating or redrawing map style images from satellite imagery is time-consuming and expensive. At the same time, vast archives of high-resolution satellite images are becoming widely available. Our project aims to bridge this gap by automatically converting raw satellite images into clear, readable Google-style map renderings using deep learning.

With the growing availability of satellite imagery, automating the conversion of these images into usable map formats is a practical and effective use of deep learning. Image-to-Image translation is an ideal deep learning technique that can be used in our application, and has been previously used in similar applications, see Figure 1 for an example of the input and output generated from an article which covered the same topic Giri et al. (2023).

The team's proposed Pix2Pix neural network architecture is based on Conditional Generative Adversarial Networks (cGANs), which is known for image-to-image translation Isola et al. (2017). Our model comprises a U-Net Generator and a PatchGAN Discriminator, both built on Convolutional Neural Networks (CNNs); inputting paired supervised data of maps to train our model to properly output the correct map of a given location through data augmentation. The neural network should facilitate accurate satellite-to-map conversion, enabling an effective methodology of updating and producing global maps. This report underlines the current progress of our project, including the implementation of data processing for the Pix2Pix dataset, a baseline simple U-net generator, and our primary cGAN model.

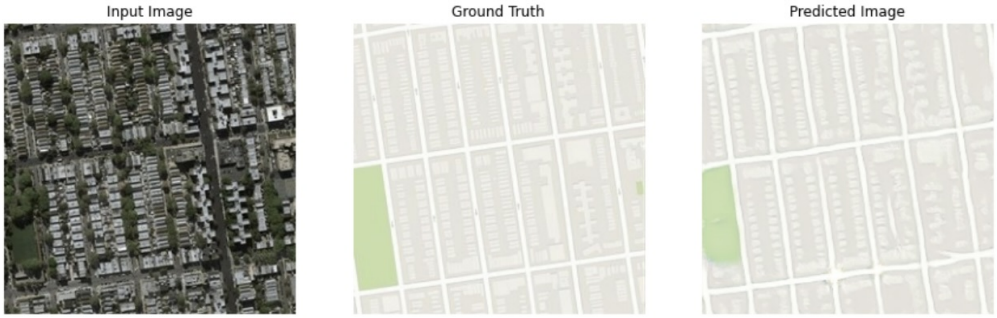

Figure 1: Input and output example of our model expected after implementing the cGAN from a 2023 paper. Giri et al. (2023)

2 Individual Contributions and Responsibilities

2.1 Team Coordination

To ensure that each team member adheres to deadlines and project objectives, the team holds weekly hybrid meetings every Saturday morning via Google Meet at approximately 11:30 AM ET (Toronto Time). These meetings serve as a checkpoint to review progress, address challenges, redistribute tasks when needed, and offer support across responsibilities. If obstacles arise, team members can step in to assist. For daily communication, the team uses Discord to coordinate asynchronously and share updates.

2.2 Project Management

Task tracking and progress monitoring are managed through a Gantt Chart shared via Google Sheets. This dynamic document allows all team members to update their assigned subtasks, signal completion status in real time, and stay aligned on upcoming milestones. Github is used for version control and code collaboration alongside Google Colab.

2.3 Completed Tasks

This section outlines the tasks completed by each team member, demonstrating that the team is currently on track with the project timeline. The division of responsibilities and task progress has been tracked using our shared Google Sheets, which provides a detailed breakdown of each individual's contributions. These include data preprocessing, dataset splitting, baseline model implementation, code generation for the U-Net generator and PatchGAN discriminator, training scripts, initial visualization outputs, and progress report writing shown in Figure 2 below.

3	**Preprocessing Data**					
3.1	Data Collection	Salma	7/1/25	7/3/25	2	100%
3.2	Preprocessing and Splitting	Salma , Ahmed	7/4/25	7/7/25	3	100%
3.3	Data Augmentation	Muhammed Ahmed	7/7/25	7/8/25	1	100%
3.4	Expanding Dataset	Salma	7/8/25	7/11/25	3	60%
4	**Baseline Model**					
4.1	Generator Code	Daisy	7/7/25	7/8/25	1	100%
4.2	Training and Validation	Daisy	7/8/25	7/9/25	1	100%
4.3	Visualizing Results	Daisy	7/10/25	7/10/25	0	100%
5	**Primary Model**					
5.1	Generator Code	Salma	7/7/25	7/9/25	2	100%
5.2	Discriminator Code	Mohammed	7/7/25	7/9/25	2	100%
5.3	Training and Validation	Muhammed Ahmed	7/9/25	7/10/25	1	75%
5.4	Visualizing Results	Muhammed Ahmed	7/10/25	7/10/25	0	80%
6	**Developing Report**					
6.1	Abstract	Mohammed	7/7/25	7/8/25	1	100%
6.2	Introduction	Mohammed	7/8/25	7/9/25	1	100%
6.3	Individual Contributions	Mohammed	7/8/25	7/9/25	1	100%
6.4	Baseline Model	Daisy	7/8/25	7/9/25	1	100%
6.5	Architecture	Salma	7/7/25	7/8/25	1	100%
6.6	Visualizing Results	Muhammed Ahmed	7/10/25	7/10/25	0	100%

Figure 2: Progress Track

2.4 Division of Tasks

Table 1: Upcoming Initial Division of Tasks and Internal Deadlines

Milestone	Subtask	Assigned	Internal Deadline
Final Report	Introduction	Mohammed	1/8/25
	Illustrations	Muhammed Ahmed	3/8/25
	Background and Related Work	Salma	3/8/25
	Data Preprocessing	Salma	3/8/25
	Architecture	Mohammed	5/825
	Baseline Model	Daisy	5/8/25
	Quantitative Results	Muhammed Ahmed	5/8/25
	Qualitative Results	Mohammed	5/8/25
	Evaluate on new data	Salma	5/8/25
	Discussion	Daisy	7/8/25
	Ethical Considerations	Salma	7/8/25
	Project Quality	Muhammed Ahmed	9/8/25
	Editing	Mohammed	10/8/25
Final Presentation	Problem	Mohammed	10/8/25
	Data	Salma	10/8/25
	Data Processing	Salma	10/8/25
	Model	Mohammed	10/8/25
	Demonstration	Muhammed Ahmed	10/8/25
	Quantitative Results	Muhammed Ahmed	10/8/25
	Qualitative Results	Daisy	10/8/25
	Takeaways	Daisy	10/8/25

As the project progresses toward its final milestone, the team remains committed to maintaining strong communication, ownership of deliverables, and consistent collaboration to ensure successful completion

3 Data Processing

For the data, we used the Pix2Pix Kaggle "Maps" dataset Cijov (2020), which is a collection of 1096 pairs of training images and 1098 pairs of validation images. The pairs include a satellite image on the left and a map image on the right. Each image has a 1200x600 pixel resolution with 3 channels for RGB.

3.1 Processing and Splitting Data

The images were combined into 2194 pairs in one folder. These pairs were then loaded onto Python, for which we created a function that processes the images into cropped images of 256x256 so that the satellite was separate from its map counterpart. Within the function for processing the data was the BICUBIC method to enhance the quality by smoothing the images. The resized pair can be seen below in Figure 3.

Figure 3: Resized pairs to 512x256 pixels.

Once the images were cropped, the following process was:

1. Each image in this folder was made into a numpy array, labeling the satellite image as input and its pair, the map, as a target for the GAN model.
2. The input and target array were transposed to be in a CxHxW format and normalized into the standard image [-1,1].
3. After that was randomized data splitting and data augmentation of our training data. An 80,10,10 ratio for training, validation, and testing, respectively.

Resulting in 1755 training images, 219 validation images, and 220 test images, each placed in separate folders. This ratio was chosen to maximize the amount of training data to help the model learn better and minimize the potential of overfitting.

3.2 Data Augmentation

An augmentation class was established, which includes horizontal flipping, random rotation, and random jittering. The jittering process resizes images to 286x286 and then randomly crops them to 256x256. These augmentations are applied to the satellite map images, following a similar approach to that used in Isola et al. (2017), in order to improve the model's generalization and prevent overfitting to the training dataset. See Figure 4 for an example of the augmented data.

Figure 4: Comparison of original map/satellite pairs (on the top) with augmented pairs seen in the train loader (on the bottom).

3.3 Challenges

Expanding our dataset using data augmentation helps overcome some of the bias and diversify our dataset, While we are aware of the limitations and potential biases of this dataset, the current access available to map and satellite pairs is small. Therefore, further research is still to be done in hopes of potentially expanding our test set with newly created pairs to help our model generalize better. We will be freezing our test set and following the plan below to expand on our dataset:

- Generate synthetic pairs with the use of public map-tile APIs, which will be accessed from services like CartoDB and ArcGIS. The plan would be to write code that generates satellite-to-map pairs similar to the images in the maps Kaggle dataset.

- The cities in which we will be looking into doing this for includes: Auckland, Shanghai, Seoul, Cairo, Barcelona, Lima, Montreal and Toronto to allow for a more global perspective. The cities are selected for their diverse coastal and urban features to broaden geographic variety

4 Baseline Model

A U-Net model Ronneberger et al. (2015) will be used for comparison as it is able to identify major features in a satellite map and construct a simpler map version. This architecture has an encoder and a decoder of 5 blocks with skip connections to process images. Each block in the encoder has 2 convolution layers and max pooling activated by ReLu and a dropout to increase depth and decrease resolution. Each decoder block has 2 up-convolution layers with skip connections from the corresponding encoder block to recover dimensions Alejandro I. Armendia (2024). This model should be able to reconstruct major features from the satellite map.

4.1 Training the Model

This model is trained with the same test and validation set used in the primary model. The figure below is the training curve with batch size 64, learning rate 0.001 and 30 epochs. The training loss and validation loss are both decreasing with the presence of some noise.

Figure 5: Training and Validation Loss curves for the baseline model after 30 epochs.

4.2 Qualitative and Quantitative Results

As for qualitative results, the figure below shows 2 samples randomly chosen from the test set. The predicted map contains some of the major features such as a bit of green area in the first sample (top left corner), and the outline of the building in the second sample (top left corner)

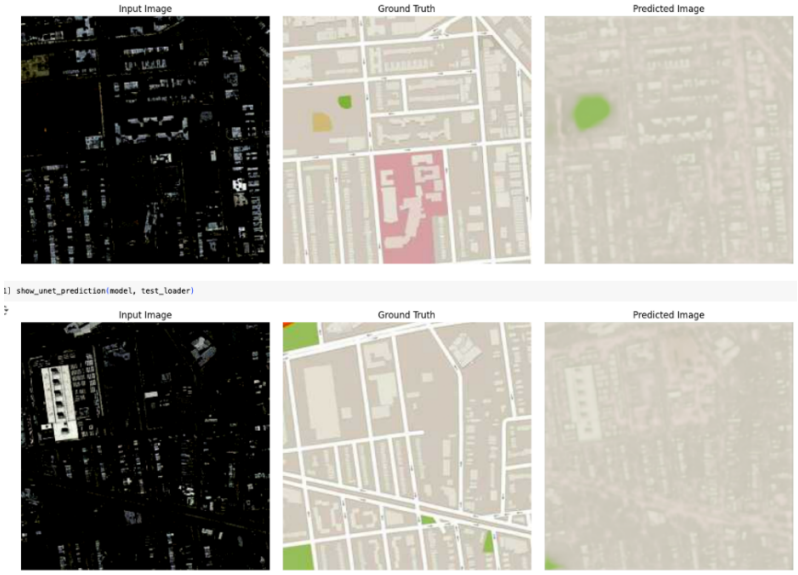

Figure 6: Visualizing predictions on random samples in the test set.

In order to measure accuracy for image results, the IOU (intersection over union) is a metric used to obtain quantitative results Standfsk (2024). This is calculated by dividing the overlapped area of the predicted image and original map by their union area. The average IOU on the test set for this model was 95.80

L1 loss is another metric we used in the primary model to evaluate performance. In image segmentation, L1 loss detects the absolute differences between labels for pixels. We will also be using the same metric in the primary model for comparison. In the baseline model, the L1 loss was calculated to be 0.0712. It had a higher loss compared to the current primary model as the U-Net itself generates vague borders and weights are not updated with the discriminator.

4.3 Challenges

As mentioned before, the data used for training may be biased because all images are from New York City. We plan on increasing the dataset as this project advances. Another challenge was trying out different U-Net structures and training the appropriate model so it doesn't learn noises only.

5 Primary Model

The chosen architecture was the conditional generative adversarial network (cGAN). Comprising two bodies, including a U-net Generator and a PatchGan Discriminator, as displayed in Figure 7. The model works by training the network to be able to generate map images from satellite images by learning how to differentiate between real and fake images and generate accurate representations of the input satellite images.

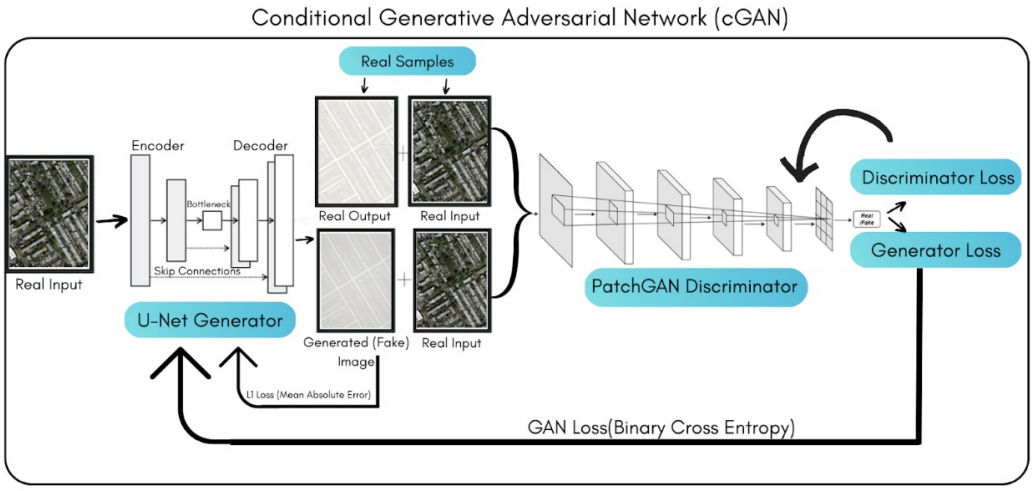

Figure 7: Primary Model Architecture

5.1 U-Net Generator Architecture

The U-net Generator is fed an image into an encoder, until at a bottleneck where the most important features are learned. This bottleneck of high-level information is then fed into a decoder, which then up-samples the information, attempting to reconstruct a map-like replica of the satellite input. The U-net Generator comprises of CNN blocks in addition to skip connections, creating the U-net shape.

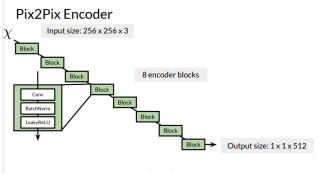

Figure 8: Encoder blocks which down-sample the image into a 1x1x512 image. Arya M. Pathak (2024)

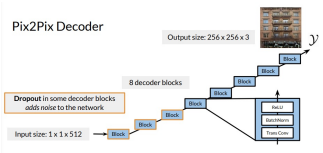

Figure 9: Decoder diagram which up-samples back to the reconstructed 3x256x256 image. Arya M. Pathak (2024)

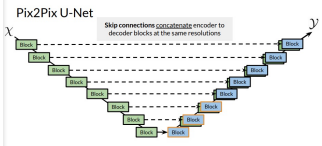

Figure 10: Generator class U-net structure with skips connecting the encoder and decoder blocks. Arya M. Pathak (2024)

As seen in Figure 8, the generator's encoder consists of eight sequential 4×4, stride-2 convolutional blocks with LeakyReLU(0.2); batch normalization is applied to all but the first block to improve generalization. Its decoder mirrors this structure with eight 4×4, stride-2 transposed-convolution blocks, each followed by instance normalization and ReLU; the first four blocks also include 0.5 dropout to reduce overfitting. Skip-connections between matching encoder and decoder layers preserve spatial detail. See Figure 9 for a diagram of the decoder class.

Finally, the Generator class simply processes the encoder and decoder blocks to allow for the skip connections and for the model to take the inputted satellite input and output a 3x256x256 generated map counterpart. Applying the skip connections to all the layers except the bottleneck to produce the U-net shape, as seen above in Figure 10. By concatenating the parallel layers.

5.2 PatchGAN discriminator Architecture

The discriminator in our conditional GAN is a PatchGAN, it judges the realism of the generated image using 70×70-pixel patches rather than the entire image, encouraging sharp outputs. Its input is the channel-wise concatenation of a real satellite image and either a real or generated map image.

At the core of the PatchGAN are four sequential down-sampling blocks. Each block applies a 4×4 convolution with stride 2 and padding 1, halving the spatial dimensions while expanding the number of feature channels. The architecture generates the following channels in sequence: 64-128-256-512. After every convolution and batch normalization (where used), a LeakyReLU activation with a negative slope of 0.2 introduces non-linearity.

Following these four down-sampling stages, we apply a fifth 4×4 convolution with stride 1 and padding 1, maintaining spatial dimensions but enlarging the receptive field so that each output element corresponds exactly to a 70×70 patch in the input. This layer is followed by batch normalization and another LeakyReLU. Finally, a last 4×4, stride-1 convolution reduces the feature map to a single channel, and a Sigmoid activation converts each patch score into a probability between 0 and 1. This architecture both enforces realism and encourages the generator to produce sharp, detailed map features.

5.3 Hyper-parameter Tuning

The model is trained using the Adam optimizer with a learning rate of 0.0002 and a batch size of 1. Both the generator and discriminator use Adam. The discriminator is trained using Binary Cross-Entropy (BCE) loss, while the generator is trained using a combination of L1 loss and BCE loss.More specifically, using this equation generatorloss = criterionGAN(discr output, real images) + L1lambda * criterionL1(generated image, ground truth image).

Figure 11 shows the results of initial training with hyperparameter tuning, varying batch size and learning rate. We used beta as 0.5 and 0.9 with learning rate from 0.0002-0.005. Lambda value is taken as 100 recommended in Isola et al. (2017). The generator and discriminator losses displayed are raw loss values recorded during training, both losses plateaued.

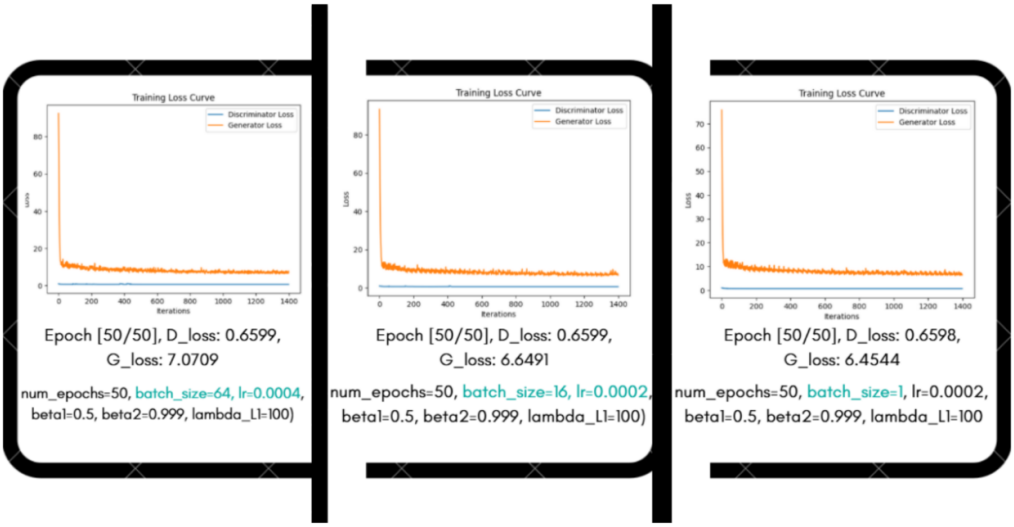

Figure 11: Generator and discriminator training loss curves

5.4 Model Performance

This section details the evaluation of model performance in the validation dataset.

5.4.1 Qualitative Results

The figure below presents two examples of the qualitative results obtained of our model on validation dataset. Each row consists of the input satellite image, the corresponding ground truth map, and the model's generated map output.

Figure 12: Initial Qualitative Results from our model.

5.4.2 Quantitative Results

We evaluated the model's performance on the validation set using pixel-wise metrics. The average L1 loss (Mean Absolute Error) across the validation set was 0.0691, indicating good pixel-level similarity between the generated and ground-truth images. However, this suggests the blurriness observed in the qualitative results.

We also computed the Peak Signal-to-Noise Ratio (PSNR), a widely used image fidelity metric for evaluating pixel-level differences Wang et al. (2004). The model achieved a PSNR of 23.51 dB on the validation set. Additionally, we plan to utilize different quantitative metrics such as the Structural Similarity Index (SSIM).

5.5 Challenges

Training this conditional GAN presents several practical hurdles. Training time was substantial with each epoch took upwards of 15–20 minutes on a single GPU. On Google Colab, sessions can time out or disconnect before training finishes, forcing frequent checkpoint saves.

After an initial drop, both generator and discriminator losses plateau. After training for several epochs, the generated images remain noticeably blurred, indicating the model is struggling to learn sharp features early in training; suggesting a need to review the adversarial loss and model's capacity.

6 Link to GitHub & Google Colab

The Project's GitHub repository

Google Colab

Dataset Drive Folder

REFERENCES

Alejandro I. Armendia. Decoding the U-Net: A Complete Guide, 2024. URL https://medium.com/@alejandro.itoaramendia/decoding-the-u-net-a-complete-guide-810b1c6d56d8. Accessed on [2025-07-04].

Arya M. Pathak. Image-to-Image Translation (Pix2Pix) - Technical Guide, 2024. URL https://arya2004.hashnode.dev/image-to-image-translation-pix2pix-technical-guide. Accessed on [2025-07-08].

Sunil Belde. Noise removal in images using deep learning models, 2021. URL https://medium.com/analytics-vidhya/noise-removal-in-images-using-deep-learning-models-3972544372d2.

Sreenivas Bhattiprolu. Simple u-net model for image segmentation. https://github.com/bnsreenu/python_for_microscopists/blob/3d1f3e52ca5240fe0e6157b3fc8e41dfd10c8e81/204-207simple_unet_model.py, 2020. GitHub repository, commit 3d1f3e52ca5240fe0e6157b3fc8e41dfd10c8e81.

Canva. Canva, 2025. URL https://www.canva.com/.

A. Cijov. Pix2pix maps, 2020. URL https://www.kaggle.com/datasets/alincijov/pix2pix-maps.

R. Giri, B. R. Lamichhane, and B. Pokhrel. Sketch to image translation using generative adversarial network. *Journal of Engineering Sciences*, 2(1):70–75, 2023. URL https://nepjol.info/index.php/jes2/article/view/60397.

IARJSET. Satellite image to map conversion and land cover analysis using deep learning. *IARJSET International Advanced Research Journal in Science*, 12(5), 2025. doi: 10.17148/IARJSET.2025.125343. URL https://iarjset.com/papers/satellite-image-to-map-conversion-and-land-cover-analysis-using-deep-learning/.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *Berkeley AI Research (BAIR) Laboratory, UC Berkeley*, 2018. URL https://arxiv.org/pdf/1611.07004.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. URL https://openaccess.thecvf.com/content_cvpr_2017/papers/Isola_Image-To-Image_Translation_With_CVPR_2017_paper.pdf.

P. Kashyap. Image normalization in pytorch: From tensor conversion to scaling, 2024. URL https://medium.com/@piyushkashyap045/image-normalization-in-pytorch-from-tensor-conversion-to-scaling-3951b6337bc8.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241. Springer, 2015. URL https://arxiv.org/abs/1505.04597.

Standfsk. IoU and Variants overview, 2024. URL https://medium.com/@cshyo1004/iou-and-variants-overview-a328acf177cd. Accessed on [2025-07-11].

A. Taparia, A. K. Bashir, Y. Zhu, T. R. Gdekallu, and K. Nath. Transforming satellite imagery into vector maps using modified gans. *Alexandria Engineering Journal*, 109:792–806, 2024. URL https://www.sciencedirect.com/science/article/pii/S1110016824010986?via%3Dihub.

V. Tiwari. Pix2pix dataset, 2018. URL https://www.kaggle.com/datasets/vikramtiwari/pix2pix-dataset/data.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. https://ieeexplore.ieee.org/document/1284395, 2004. IEEE Transactions on Image Processing, Vol. 13, No. 4, pp. 600–612.

Hao Zhang and Wenlei Wang. Imaging domain seismic denoising based on conditional generative adversarial networks (cgans). *Energies*, 15(18):6569, 2022. URL https://www.researchgate.net/publication/363389820_Imaging_Domain_Seismic_Denoising_Based_on_Conditional_Generative_Adversarial_Networks_CGANs.

X. Zhao, L. Gao, Z. Chen, and B. Zhang. An cnn-based operational approach for land cover mapping in china. In *IOP Conference Series: Earth and Environmental Science*, volume 502, 2020. doi: 10.1088/1755-1315/502/1/012036. URL https://iopscience.iop.org/article/10.1088/1755-1315/502/1/012036.