Image-to-Image Translation for the Cityscapes Dataset

Valentina Isabel Ortega Pinto
University of Santiago de Compostela
Galicia, Spain
valentinaisabeldelamilagrosa.ortega@rai.usc.es

Martín Romero Romero
University of Santiago de Compostela
Galicia, Spain
martin.romero.romero@rai.usc.es

***Abstract*—This study explores various strategies to enhance image quality in image-to-image translation tasks, focusing on the cityscapes dataset. We investigated the effectiveness of baseline models, pre-trained models, and three specific improvement techniques: data augmentation, modification of loss functions, and architectural adjustments of the baseline model. Among these, increasing the complexity of the baseline model's architecture proved most effective, significantly improving image translation quality. Despite these advancements, the highest Structural Similarity Index (SSIM) achieved on test images was 53%, indicating substantial potential for further refinement. This paper details the methodologies applied, evaluates the impact of different enhancement strategies, and discusses the implications of our findings for future research in the field of image translation**

***Index Terms*—pix2pix, image translation, cityscapes, computer vision**

I. Introduction

The primary objective of this project is to employ image-to-image translation models, notably Pix2Pix [1], for generating realistic images from input label maps. Pix2Pix utilizes adversarial training to learn a mapping function that translates an input image domain into a corresponding output image domain. This involves the simultaneous training of a generator network, which produces realistic images, and a discriminator network, which discerns between generated and authentic images.

Specifically, this project focuses on the Cityscapes dataset [2] , which comprises urban cityscape views. The goal is to develop a foundational model based on existing literature [1] and enhance its performance through innovative approaches such as data augmentation and transfer learning. By leveraging these techniques, the project aims to refine the quality and applicability of the generated images, pushing the boundaries of current image-to-image translation technologies.

II. Technical Approach

A. Base Model

For the base architecture, we implemented a Pix2Pix model using the Pix2Pix-PyTorch repository [3], which is established on recommended literature for this work. The base model adheres to the classical Pix2Pix architecture, featuring a generator based on U-Net models. The generator comprises an encoder and decoder, each composed of 8 layers of convolutional layers with batch normalization. Additionally, the generator incorporates skip connections, following the general structure of a U-Net model. These skip connections play a crucial role in preserving fine-grained spatial information from the input to the output.

The discriminator is structured according to the recommended PatchGAN architecture. A PatchGAN discriminator is a convolutional neural network (CNN) specifically designed to assess the realism of image patches rather than the entire images. It consists of multiple convolutional layers followed by downsampling operations, such as strided convolutions, aimed at extracting hierarchical features from the input patches. Leaky ReLU activation functions introduce non-linearity to the network, facilitating feature learning and gradient propagation. Instead of producing a single output for the entire image, the discriminator outputs a classification score for each patch in the input image. Finally, a sigmoid activation function is applied to the last convolutional layer to produce probability scores indicating the likelihood that each patch is real or fake. This patch-wise evaluation mechanism provides small-detailed feedback to the generator, thereby aiding in the generation of high-quality, visually consistent outputs.

For training the base model, we conducted image normalization to scale pixel values to the range of 0 to 1. The available image datasets were partitioned into distinct sets for training, validation, and testing purposes. Following the recommendations outlined in the paper, we adopted specific hyperparameters for training. This included training the model for a minimum of 200 epochs, employing a learning rate of 0.0002, and utilizing a batch size of one. We opted for the Adam optimizer, a widely-used choice for deep learning tasks.

Regarding the loss functions employed during training, we utilized Binary Cross Entropy coupled with an additional L1 term for the generator loss. This L1 term was weighted by a factor of alpha = 100, as suggested in the literature. The incorporation of the L1 term alongside the binary cross-entropy loss aims to promote both realism and fidelity in the generated images. By combining these loss functions, the model is incentivized to generate outputs that not only match the target distribution but also preserve fine-grained details and structures from the input images. This comprehensive loss formulation contributes to the training stability and effectiveness of the Pix2Pix model, hoping it produces high-quality results.

B. Selected Metric

We chose to work with the Structural Similarity Index Measure (SSIM), which is a widely-used metric in image processing that quantifies the similarity between two images. It evaluates not only pixel-wise differences but also takes into account perceptual features such as luminance, contrast, and structure. SSIM operates by comparing local patterns of pixel intensities between corresponding image patches and computing a similarity score based on luminance similarity, contrast similarity, and structure similarity. The resulting index ranges from -1 to 1, where a value closer to 1 indicates a higher similarity between the images.

SSIM is a good choice of metric to evaluate the performance of Pix2Pix models for several reasons. Firstly, Pix2Pix aims to generate realistic images that preserve the structure and details of the input images while translating them into a different domain. SSIM's consideration of both structural information and perceptual features aligns well with this objective, as it captures both the global and local similarities between the generated and ground truth images. This makes SSIM more robust than traditional pixel-wise metrics like Mean Squared Error (MSE) or Peak Signal-to-Noise Ratio (PSNR), which only focus on pixel-level differences and may not correlate well with human perception.

Furthermore, SSIM is particularly suited for evaluating the quality of image-to-image translation models like Pix2Pix because it accounts for variations in luminance and contrast, which are essential for maintaining the visual fidelity of the generated images. By penalizing differences in structure while allowing for variations in luminance and contrast, SSIM provides a more comprehensive assessment of image quality.

Additionally, SSIM's ability to operate at multiple scales and consider local image patches makes it sensitive to both large-scale structural changes and fine-grained details. This makes it a suitable metric for evaluating the performance of Pix2Pix models across different spatial frequencies and textures. Overall, SSIM offers a holistic and perceptually meaningful measure of image similarity, making it a valuable tool for assessing the effectiveness of Pix2Pix models in generating realistic and visually pleasing results.

C. Potential Improvements

1) Different loss functions: Our initial approach involved exploring alternative loss functions beyond Binary Cross Entropy (BCE) and varying the alpha values associated with the L1 term in the loss calculation. To this end, we trained the base model using Mean Squared Error (MSE) as the loss function. Additionally, we conducted experiments with different alpha values for the L1 term, specifically testing values of 50 and 75 in addition to the original value of 100. By systematically varying these parameters, we aimed to gain insights into their impact on model performance and identify optimal configurations for achieving the most desirable outcomes.

2) Pre-trained models: We implemented some of the pytorch pre-trained models, but we also wanted to try implementing pre-trained models from other related works and repositories.

For the pytorch pretrained models, we chose to implement a generator based on a DeepLabV3. the DeepLabV3 model is used as a feature extractor (backbone) to capture high-level features from the input image. The pretrained DeepLabV3 model is loaded, and the backbone is extracted. A final convolutional layer is added to the backbone to generate the output images. During the forward pass, the input image is passed through the base model to extract features, and then the final convolutional layer is applied to generate the output images. If necessary, the size of the output image is adjusted to match the size of the input image using bilinear interpolation. This architecture leverages the powerful feature extraction capabilities of the DeepLabV3 model for image generation tasks.

On the other hand, the discriminator architecture is based on the ResNet18 model, a popular deep learning architecture known for its effectiveness in various computer vision tasks. In this implementation, a pretrained ResNet18 model is utilized as the core for feature extraction. The input image, along with any conditional information (if provided), is first combined, with the input image being concatenated with the conditional information along the channel dimension. The concatenated image is then passed through a modified first convolutional layer to adjust for the increased number of input channels. Subsequently, the image is forwarded through the remaining layers of ResNet18, excluding the final classification layer. After passing through the ResNet layers, the feature map is flattened and fed into a fully connected layer (linear layer) for classification. The output of the linear layer is passed through a sigmoid activation function to produce the final probability score indicating the likelihood that the input image is real. This architecture leverages the powerful feature extraction capabilities of ResNet18 for discriminating between real and fake images while accommodating conditional information if provided.

This model was trained using identical hyper-parameters to those employed in training the base model, including Binary Cross Entropy loss functions and training for 200 epochs.

The other pretrained model implemented was sourced from the CycleGAN and Pix2Pix in PyTorch repository [4], based on the research outlined in ”Image-to-Image

Translation Using Cycle-Consistent Adversarial Networks" [5]. This pretrained model employs a U-Net generator with skip connections and a typical PatchGAN discriminator. After uploading the weights, we further trained the model using the Mean Squared Error (MSE) loss function, augmented with an additional L1 term specifically for the generator.

3) Changes in architecture: The main changes implemented were centered around the design of the generator. The UnetGenerator from the improved design and the Generator from the base model showcase several significant distinctions in their architectural composition and design principles. The enhanced UnetGenerator adheres to a U-Net-like structure, although with the encoder-decoder components separated to enhance complexity. Nonetheless, we retained the crucial skip connections between corresponding encoder and decoder blocks. Furthermore, we introduced batch normalization after each convolutional layer and applied dropout regularization specifically to the first three layers of decoder blocks to mitigate overfitting risks. In contrast, the Generator from the baseline model adopts a simpler architecture devoid of explicit encoder-decoder stages. It relies on a sequence of convolutional and transpose convolutional layers, with batch normalization integrated throughout the network. Despite these disparities, both generators leverage the hyperbolic tangent (tanh) activation function for output scaling, with the shared goal of translating input images into corresponding output images. These alterations were implemented with the aim of enhancing the robustness of the generator to produce higher-quality images.

4) Data augmentation: We also experimented with various data augmentation techniques, including horizontal flips and custom color jitter, across all the previously mentioned models to assess their impact on performance. However, contrary to expectations, it was found that applying these transformations consistently led to a deterioration in both metric scores and loss functions for each model trained. The results of these experiments were consistently poor, prompting us to exclude them from the results section of this report. While it was hoped that data augmentation would enhance model performance by introducing additional variability and robustness, the observed outcome suggests otherwise. Consequently, we chose not to include these unsuccessful experiments in our final analysis.

III. Results

The base model yielded a training Structural Similarity Index (SSIM) of 36% and test SSIM of 30%, exhibiting the expected behavior in terms of loss functions. Throughout training, the generator loss consistently decreased, reflecting improved performance over iterations. Conversely, the discriminator loss decreased initially but eventually stabilized. This baseline model employed Binary Cross Entropy (BCE) loss. Subsequently, exploring the efficacy of Mean Squared Error (MSE) loss, we observed an overall performance boost. The baseline model trained with MSE attained an impressive training SSIM of 80% and a testing SSIM of 49%.

Further experimentation involved leveraging pre-trained models to potentially enhance results. Utilizing a PyTorch pre-trained model with BCE loss, we achieved a remarkable training SSIM of 64% and a testing SSIM of 50%. Transitioning to a model based on the available literature [5] and other repositories [4], we opted for the MSE loss due to its proven performance enhancement in baseline models. The pre-trained model with MSE exhibited similar behaviour as the previous model, yielding a training SSIM of 62% and a testing SSIM of 51%.

In our pursuit of superior results, we opted to augment the complexity of the base models. This approach resulted in a notable performance escalation, with the enhanced model boasting an impressive training SSIM of 82% and a respectable testing SSIM of 53%, the highest values obtained so far. Initially employing BCE loss, we explored the potential benefits of switching to MSE loss. However, contrary to expectations based on previous models, the MSE-based approach did not yield further improvements for the more complex models. Instead, the model trained with MSE reported a training SSIM of 48% and a testing SSIM of 46%.

Table I shows the comparison of training, validation and test SSIM across all tested models.

TABLE I
Structural Similarity Index Measure (SSIM) comparison across trained models

	SSIM		
Models	Train	Validation	Test
Baseline Model	0.369	0.324	0.301
Baseline Model with MSE	0.808	0.512	0.499
Pre-trained from pytorch	0.641	0.535	0.505
Pre-trained from literature	0.621	0.510	0.511
Enhanced baseline model	0.820	0.536	0.535
Enhanced baseline model with MSE	0.487	0.455	0.462

In evaluating the quality and appearance of the generated images, we note that the overall structure of the image is effectively replicated, with the main components accurately captured. However, there is still room for enhancement in terms of image details. This indicates that further refinements and adjustments most likely will achieve greater similarity between the target and generated images. Figures 1 and 2 show the the generated image and correspondent target image for the best-performing model.

IV. Conclusions

Overall, the project objectives were successfully met, as we were able to develop and train pix2pix models for an image-to-image translation task, generating realistic images from input label maps:

- Establishing baseline models was crucial for a comprehensive understanding of these architectural frameworks.

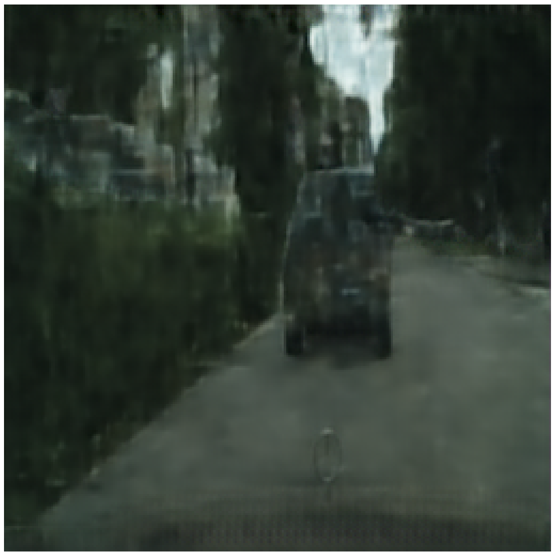
Fig. 1. Generated image from test of the best performing model

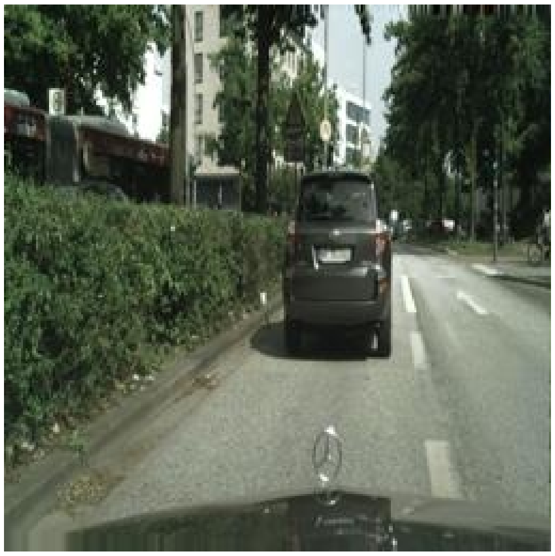
Fig. 2. Target image from test set of the best performing model.

These models provided a foundation for comparison and further experimentation.

- The choice of loss function significantly influenced model performance. Although our top-performing model utilized BCE, it is noteworthy that baseline and pre-trained models achieved superior results with MSE loss functions. This suggests that the optimal choice of loss function can vary depending on the model and the specific task.
- Contrary to expectations, data augmentation consistently impaired the performance of our models. This outcome challenges the conventional wisdom that image transformations universally enhance the performance of image translation tasks, indicating a potential need for more targeted augmentation strategies.
- Modifying the model architecture to increase complexity was the most effective approach to enhance performance. This approach enabled us to achieve a peak SSIM of 53% on the test set and 82% during training, underscoring the benefits of architectural adjustments.

V. Recommendations

Considering the potential for further enhancements in our project, we recommend the following strategies to improve the efficacy and efficiency of these pix2pix models for this image-to-image translation task:

- Due to hardware limitations, our project was constrained in its ability to extend the number of training epochs for the models. However, we recommend future studies consider training the models over more epochs as this could potentially reduce generation loss further, thereby achieving higher SSIM values.
- Continuous review of existing literature and databases to identify and adapt pre-trained models that are more specifically aligned with our objectives that the ones already mentioned in this work. This can leverage the advances in similar tasks to potentially boost performance without the extensive need for training from scratch.
- Implement systematic hyper-parameter tuning to refine the models' performance. Techniques such as grid search, random search, or Bayesian optimization could be employed to identify the optimal settings that maximize the SSIM value,
- Given that conventional data augmentation methods did not yield positive results, it is advisable to experiment with more tailored augmentation techniques. This might include less aggressive transformations or domain-specific augmentations that preserve critical features necessary for effective image translation.
- Incorporate additional evaluation metrics, beyond SSIM,like Peak Signal-to-Noise Ratio (PSNR) and perceptual image patch similarity (LPIPS) for a more comprehensive assessment of image quality and similarity. This could provide deeper insights into the model's performance nuances.

References

[1] P. Isola, J. -Y. Zhu, T. Zhou and A. A. Efros, ”Image-to-Image Translation with Conditional Adversarial Networks,” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 5967-5976, doi: 10.1109/CVPR.2017.632. keywords: Gallium nitride;Generators;Training;Image edge detection;Force;Image resolution

[2] Cityscapes Dataset. Available at: https://www.cityscapes-dataset.com/

[3] Pytorch-pix2pix repository available at: https://github.com/znxlwm/pytorch-pix2pix/tree/master

[4] CycleGAN and pix2pix in PyTorch repository available at: https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix/tree/master

[5] J. -Y. Zhu, T. Park, P. Isola and A. A. Efros, ”Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks,” 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 2242-2251, doi: 10.1109/ICCV.2017.244. keywords: Training;Painting;Training data;Semantics;Extraterrestrial measurements;Graphics,