MSiA432 Final Project

Image Classification and Style Transfer

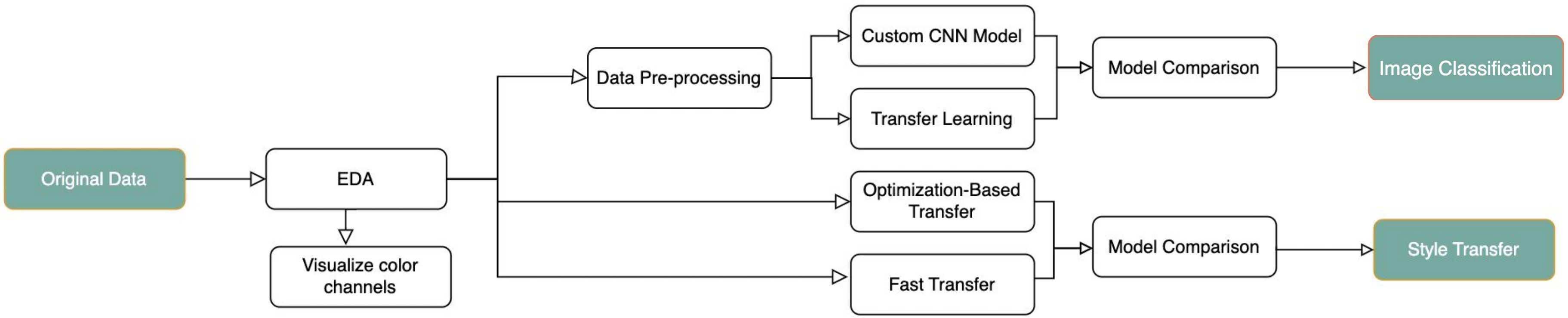
Cindy Chen, Michelle Liu, Yifei Wang, Vivian Zhou

Abstract

Project Objective

- Target users
  - Wildlife conservation institutions
  - Animal-loving volunteers
- Goals
  - Identify and document animal species to support monitoring and conservation efforts
  - Create visually appealing posters to attract public attention

Flow Chart


Original Data
EDA
Visualize color channels
Data Pre-processing
Custom CNN Model
Transfer Learning
Model Comparison
Image Classification
Optimization-Based Transfer
Fast Transfer
Model Comparison
Style Transfer

Exploratory Data Analysis

Data

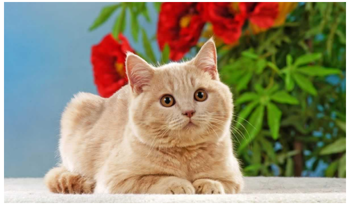
- Obtained from [Kaggle]
- 10 animals
  - Bear, cat, crab, dog, dolphin, duck, hamster, jellyfish, leopard, and panda
- 60 colored images per animal
  - 80-20 train-test split



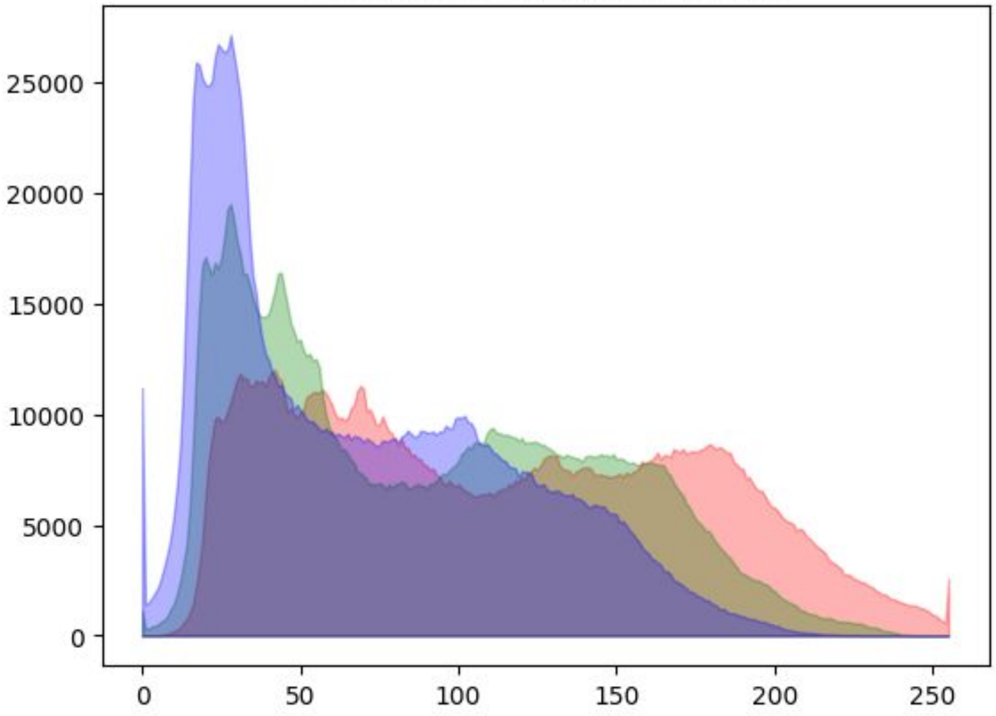




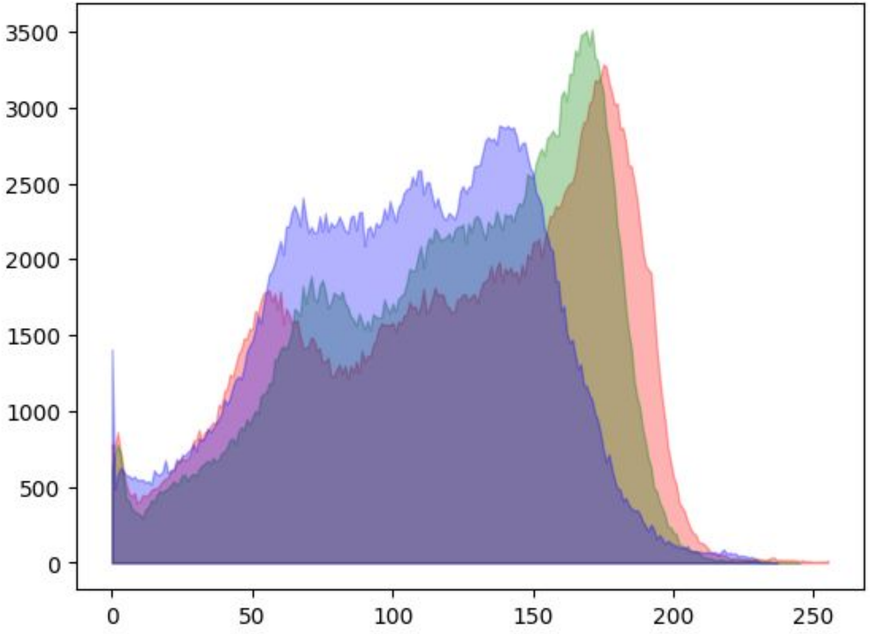




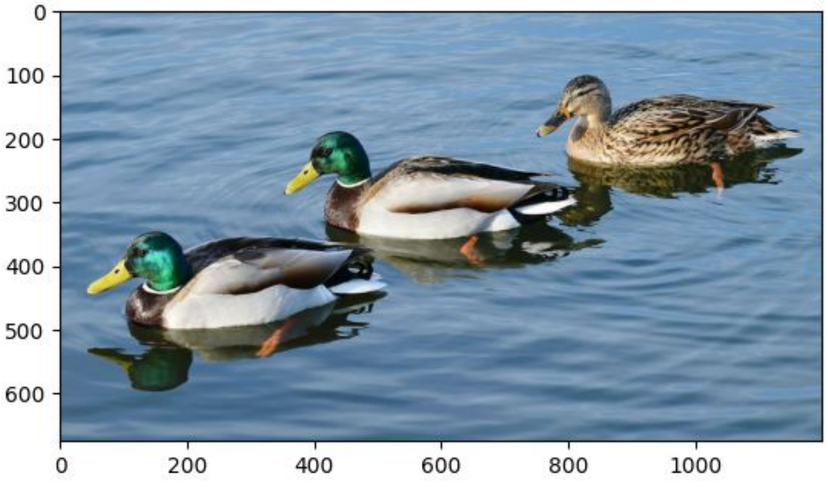

Color Histogram - Original Images

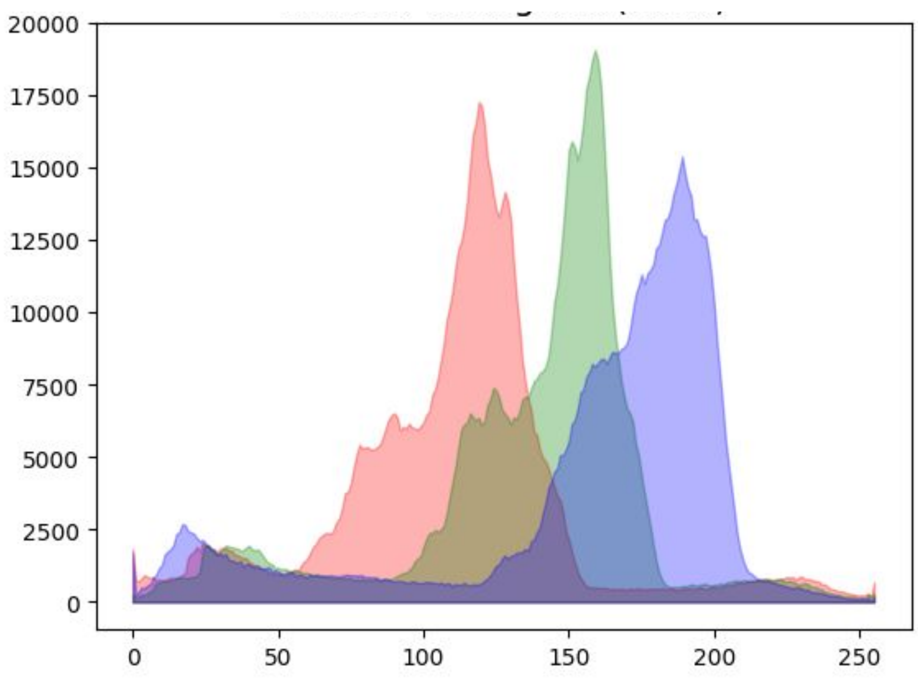


Color Histogram - Original Images



Color Histogram - Original Images

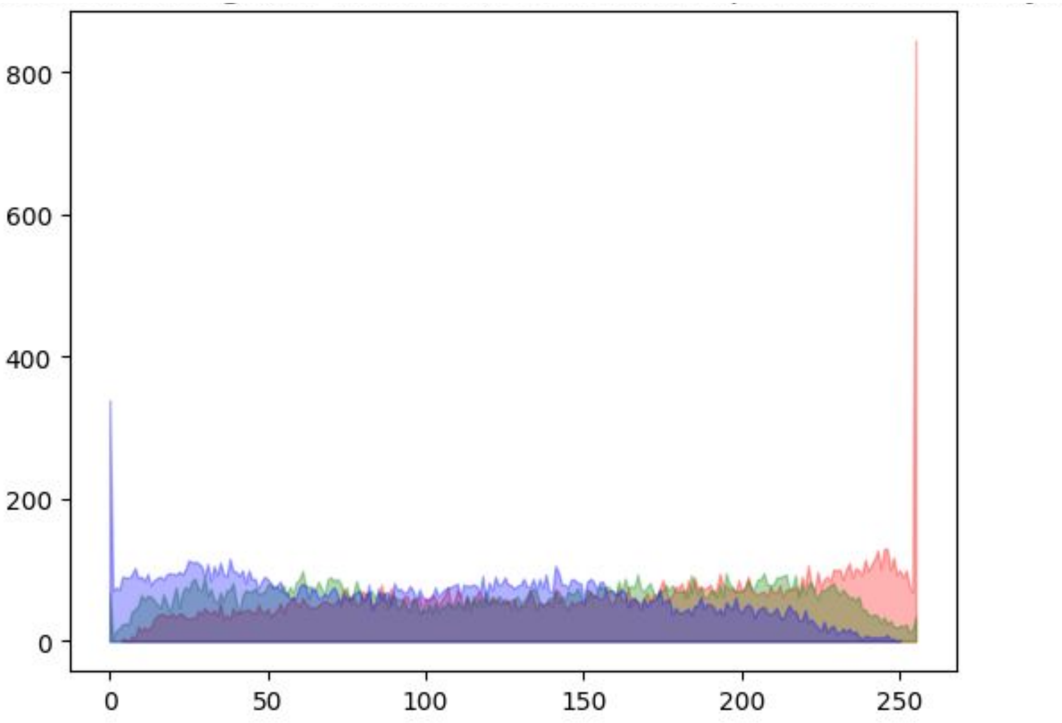




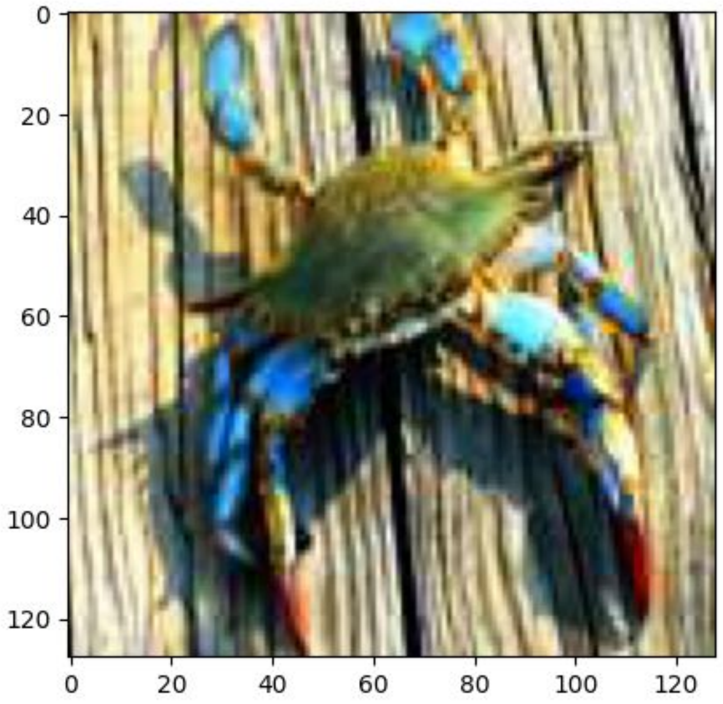
Data Preprocessing

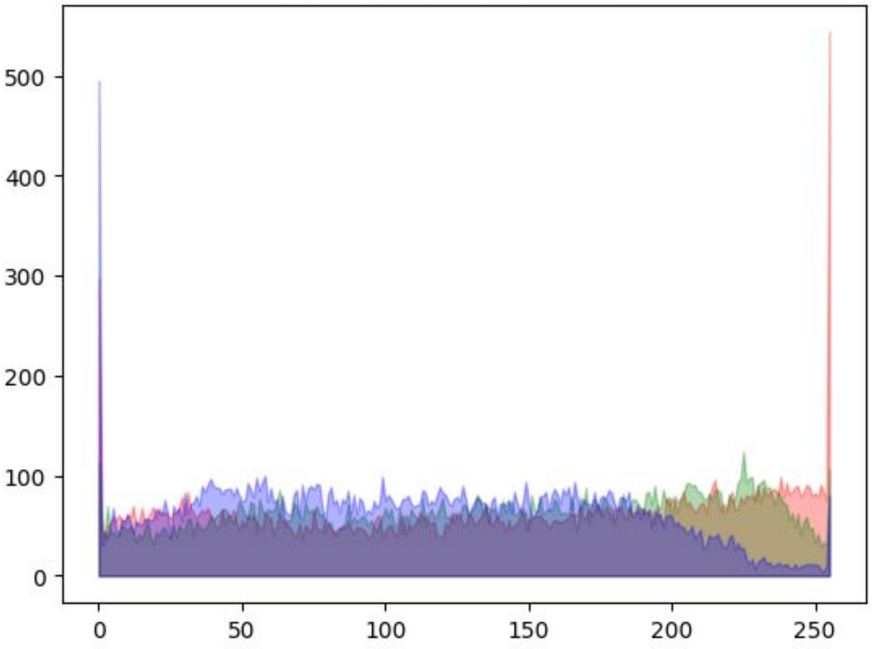
- Resize
  - Original data of various dimensions
  - All resized to (128, 128, 3)
- Intensity equalization
  - To enhance the overall contrast
  - To make the details and features in images more distinguishable

Color Histogram - Preprocessed Images

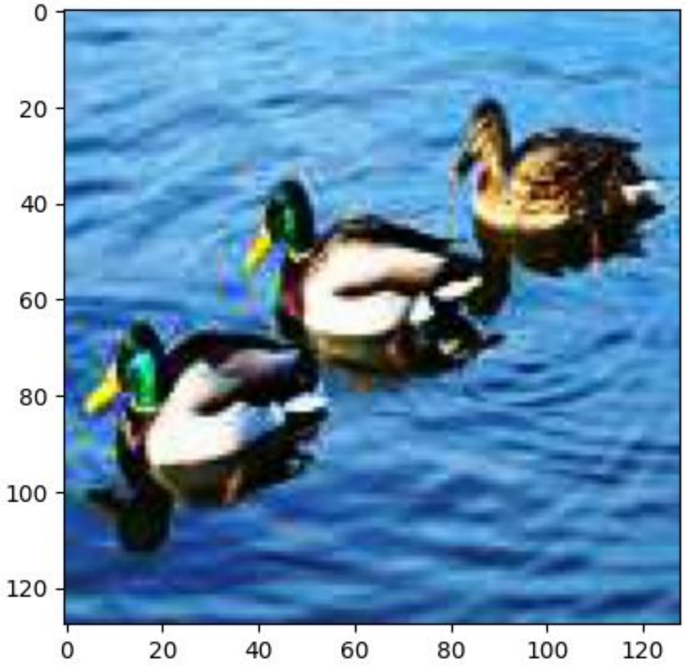


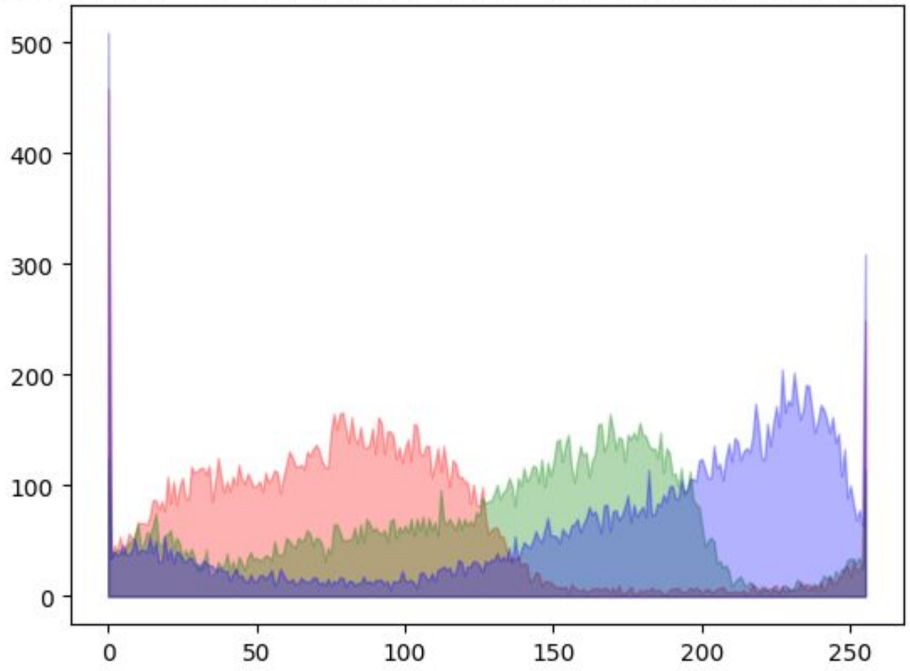
Color Histogram - Preprocessed Images





Color Histogram - Preprocessed Images





Model Training and Evaluation

Image Classification

- Custom CNN Model
- Transfer Learning with EfficientNetB3

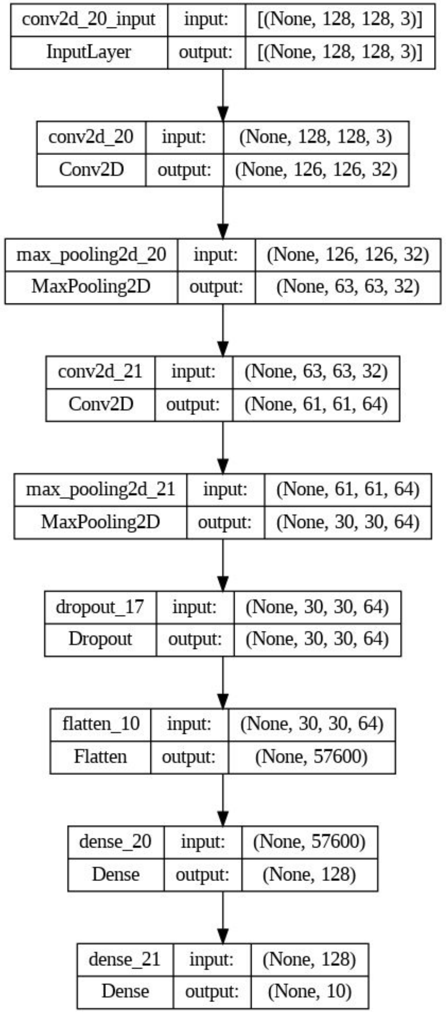
Custom CNN Model

Model architecture

- Sequential model with 2 convolutional and pooling layers
  - ReLU activation
- Dropout layer: Rate of 0.25 to prevent overfitting
- Flattened layer: Converts 2D output to 1D
- Dense layers: 128 units with ReLU activation
- Output layer: Softmax activation with the 10 number of classes
- Loss function: Categorical cross entropy

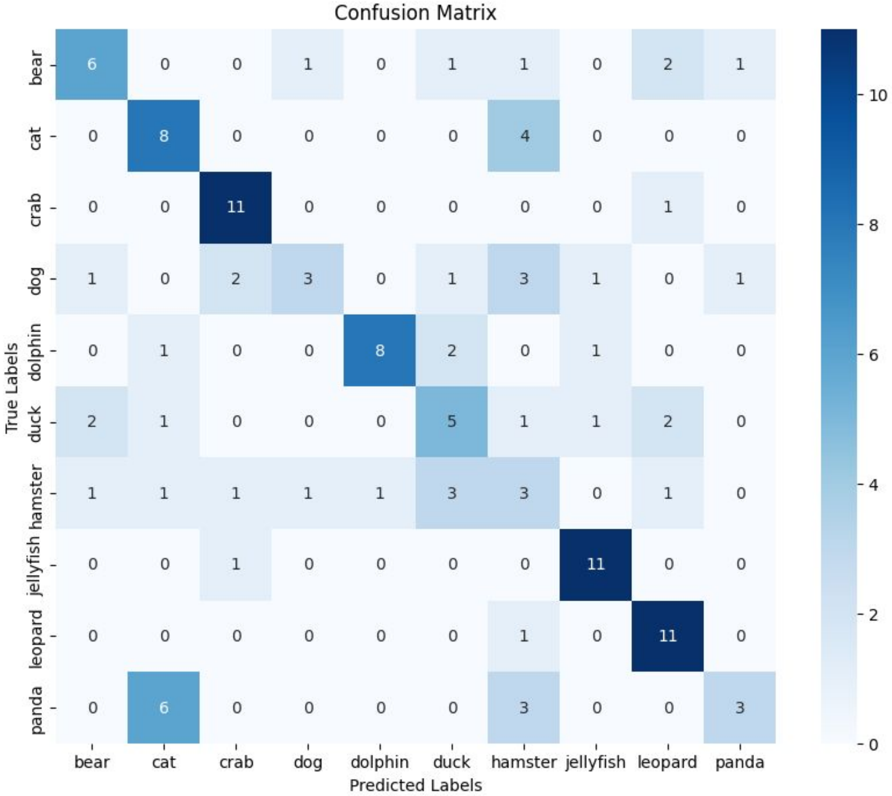
Hyperparameters

- Optimizer: Adam
- Batch size: 32
- Validation split percentage: 0.3
- Epoch: 15



Custom CNN Model (cont.)

- Model performance
  - Test accuracy: 0.575
  - F1 score: 0.562
  - Confusion matrix
    - **Perform well** when classifying crab, jellyfish, and leopard (acc: 0.917)
    - **Perform relatively poor** when classifying dog, hamster, and panda (acc: 0.25)

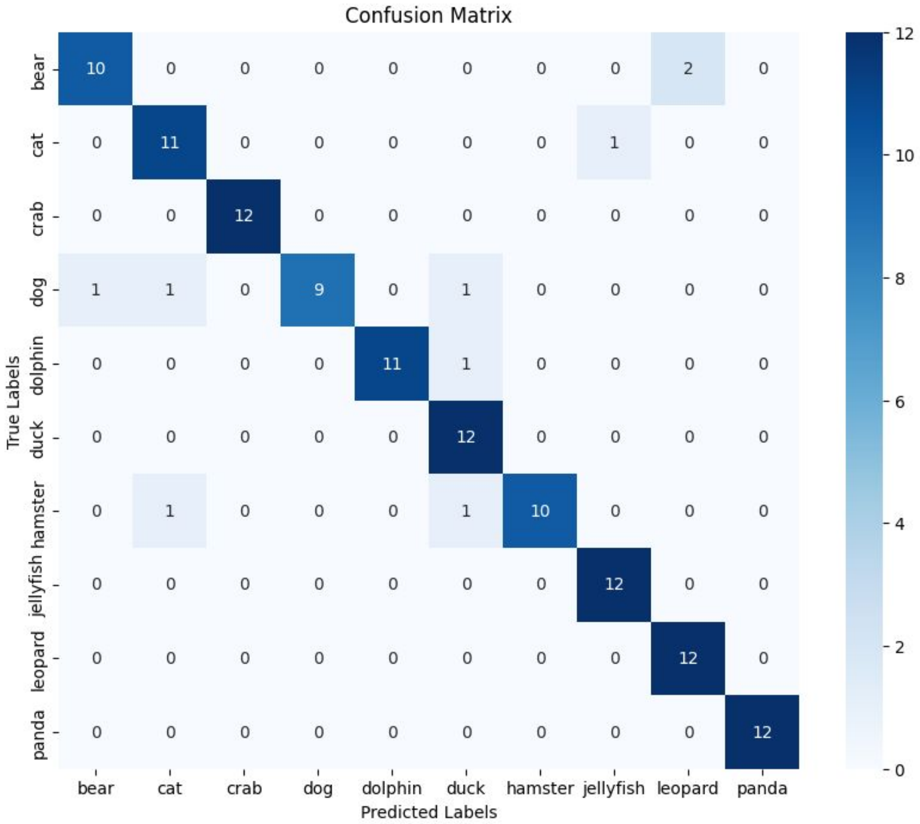


Transfer Learning with EfficientNetB3

- Model structure
  - Freezing all layers of EfficientNetB3 other than the last 10 to create the base model
  - Adding 4 dense layers on top of the base model
    - Activation function: relu
    - Nodes: 512 for the first 2 layers, and 1024 for the last 2 layers
    - Dropout rate: 0.2 and then 0.1
    - Batch normalization applied
  - Optimizer: Adam
  - Loss function: Categorical cross entropy
- Training
  - Batch size: 32
  - Epochs: 15

Transfer Learning with EfficientNetB3 (cont.)

- Model performance
  - Test accuracy: 0.925
  - F1 score: 0.924
  - Confusion matrix
    - **Perfect classification** for crab, duck, jellyfish, leopard, and panda



Model Comparison

	Custom CNN	Transfer Learning (EfficientNetB3)
Accuracy	0.575	0.925
F1 Score	0.562	0.924
Pros	• Flexible and full control over the model architecture • Provide better interpretability and insights by analyzing intermediate layers	• Higher performance • Time and resource-saving
Cons	• Longer training time • Lower accuracy • Require more data to achieve good result	• Risk of overfitting • Less control over the model architecture • Blackbox

Style Transfer

- Optimization-Based Neural Style Transfer
- Fast Style Transfer

Optimization-Based Neural Style Transfer

- Use **VGG19** to extract features from both the style image and the content image
  - **Content Layers**
    - Capturing the semantic content of an image
    - Typically located deeper in the network, where higher-level features (shapes, object parts, combinations) are encoded
  - **Style Layers**
    - Capturing the texture and style of an image
    - Located in the earlier stages of the network, where lower-level features are learned
    - The Gram matrix of the activation maps from these layers represents the style information

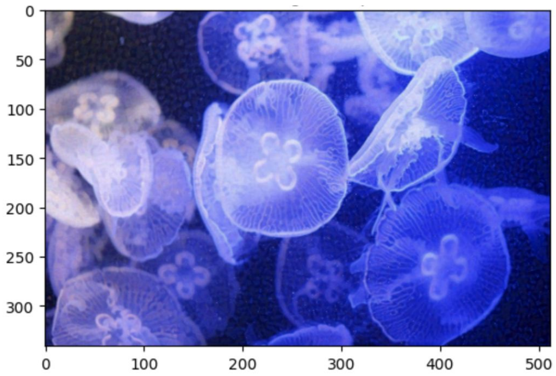
Optimization-Based Neural Style Transfer (cont.)

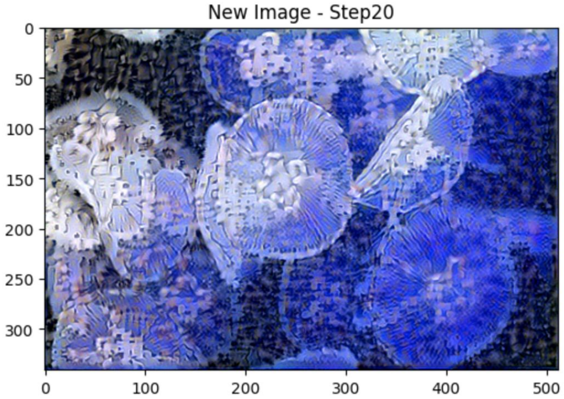
- **Loss Function:** a weighted combination of the content loss and the style loss
  - **The content loss:** measures the similarity between the content image and the generated image
  - **The style loss :**measures the similarity between the style image and the generated image
- **Optimization:** The generated image is iteratively updated by adjusting its pixel values, where the algorithm can use gradient descent to find the optimal image that minimizes the loss function. The original “A Neural Algorithm of Artistic Style” paper recommends LBFGS Optimizer, Adam Optimizer also works

Evaluation with Content Loss and Style Loss

- **Higher weights for the content loss**
  - Preserve more of the original content
  - Output image closely resembled the input image while still incorporating elements of the desired style
- **Higher weights for the style loss**
  - Stronger style transfer effects, with more pronounced artistic characteristics
- Tested different weight combinations to achieve **optimal weight combination**
  - Some combinations resulted in a harmonious blend of content and style, producing visually striking images that struck a balance between the two
  - Some resulted in the style dominated the content or vice versa
  - Subjective and dependent on the desired outcome

Loss Comparison Example

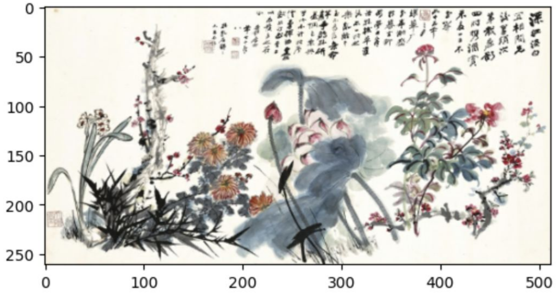


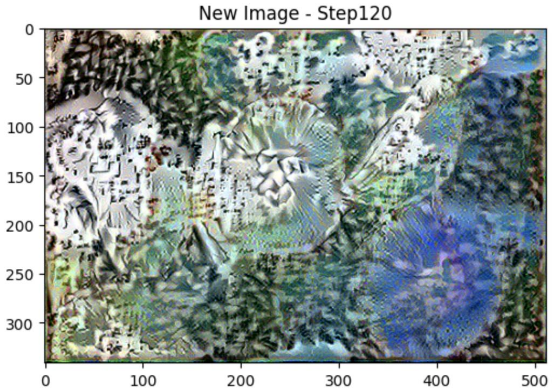


Step 20 Image

Style loss: 139.31

Content loss: 0.027



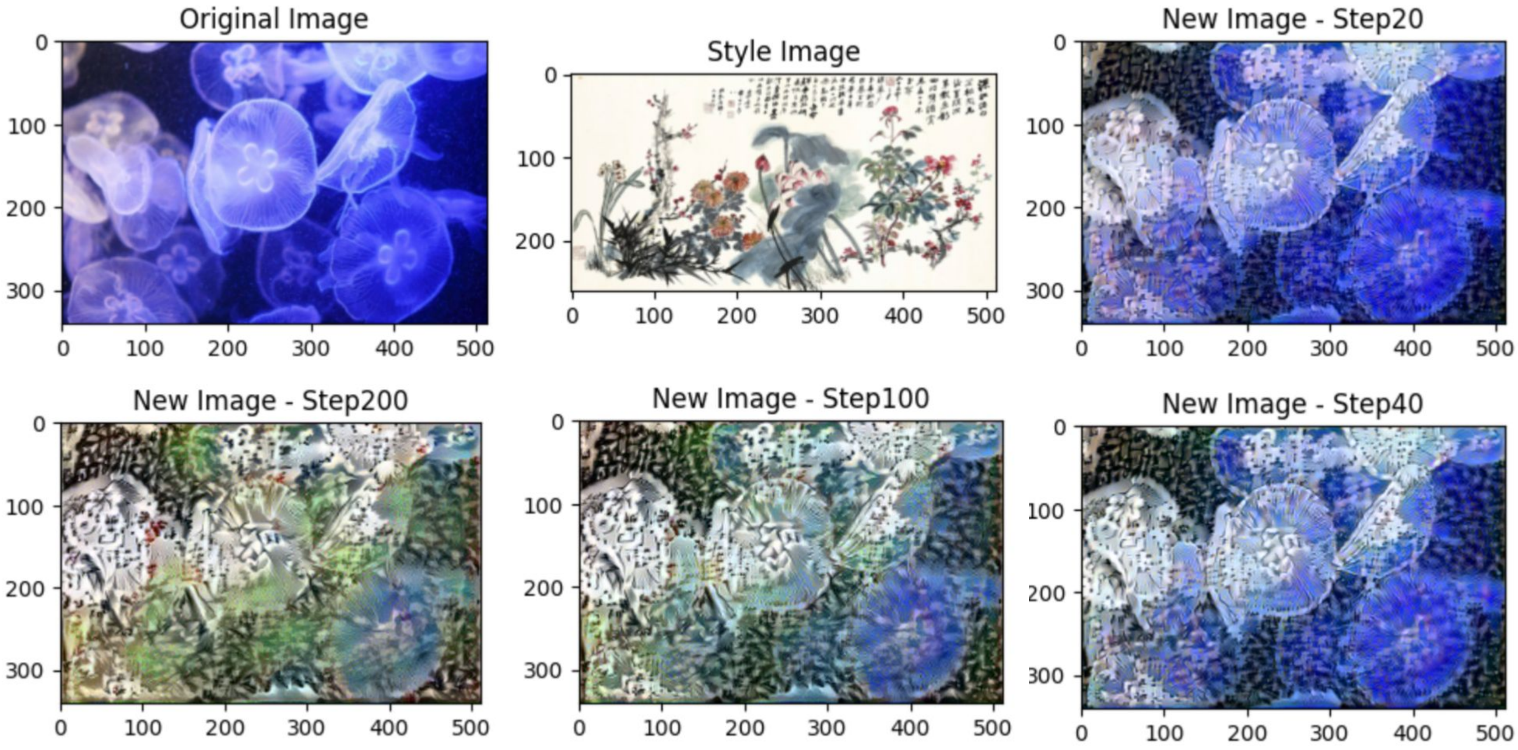


Step 120 Image

Style loss: 83.92

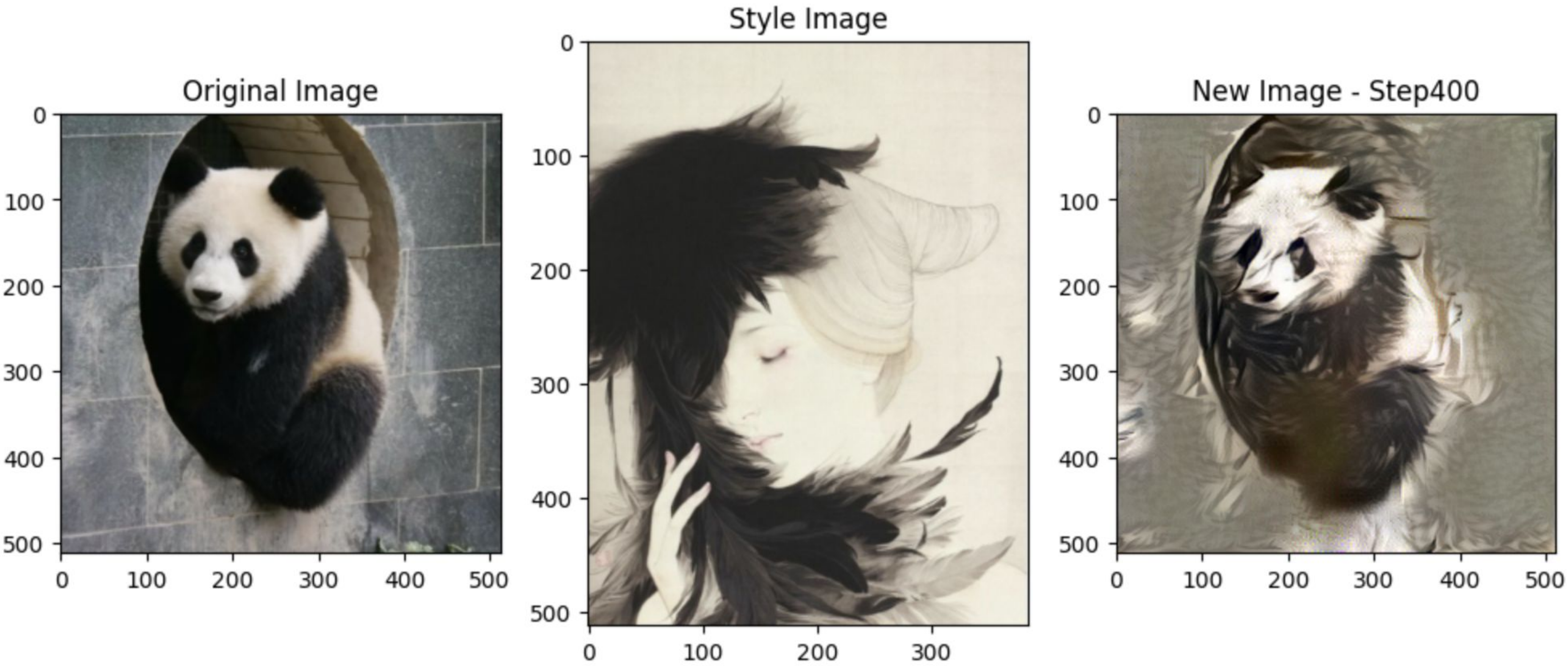
Content loss: 0.039

Zhang Daqian Style Jellyfish



Georgia O'Keeffe Style Cat

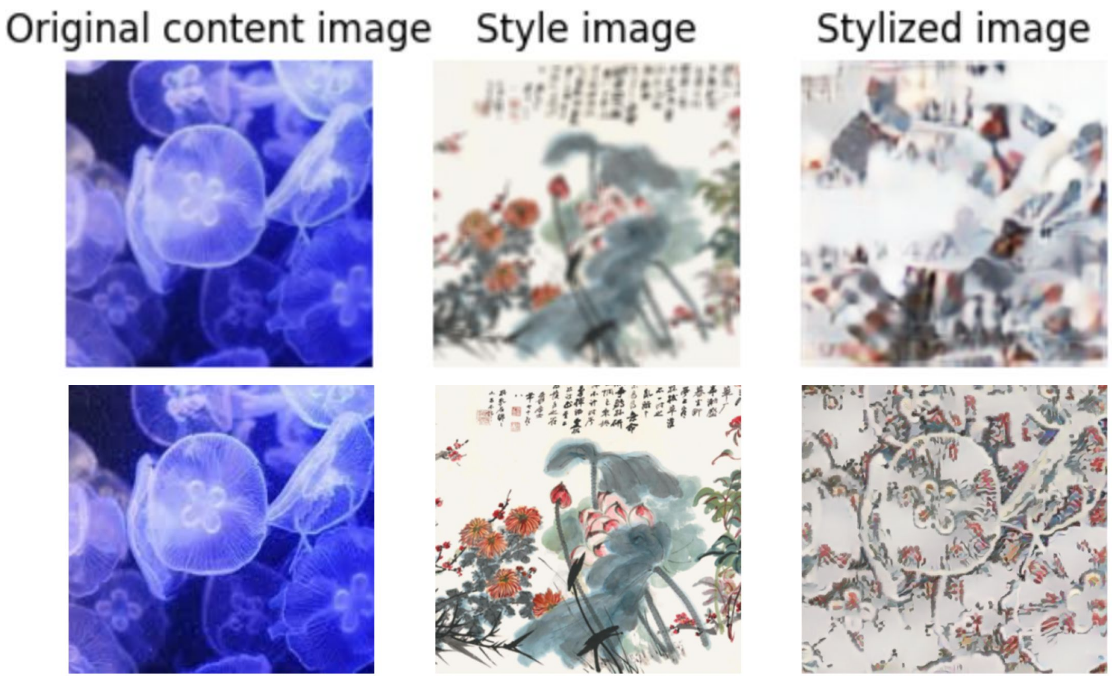
Luo Hanlei Style Panda



Fast Style Transfer

- Rapid application of artistic styles to image or videos
- **Input:** URL
- **Pre-processing:** cropping images to a square shape and maintaining aspect ratio, and then normalizing the pixel values for further processing. (The content & style image (256 recommend) size can be arbitrary)
- **Model:** Tensorflow hub module
    - VGG16 is used to extract content features and calculate the content loss
    - Inception_v3 is used to extract style features and calculate the style loss
    - Adam optimizer to minimize the loss and generate the best output image

Evaluation with different content loss and style loss



Row 1:

Style loss: 2.20

Content loss: 0.10

Row 2:

Style loss: 197.029

Content loss: 0.059

Zhang Daqian Style Jellyfish

Original content image

Style image

Stylized image

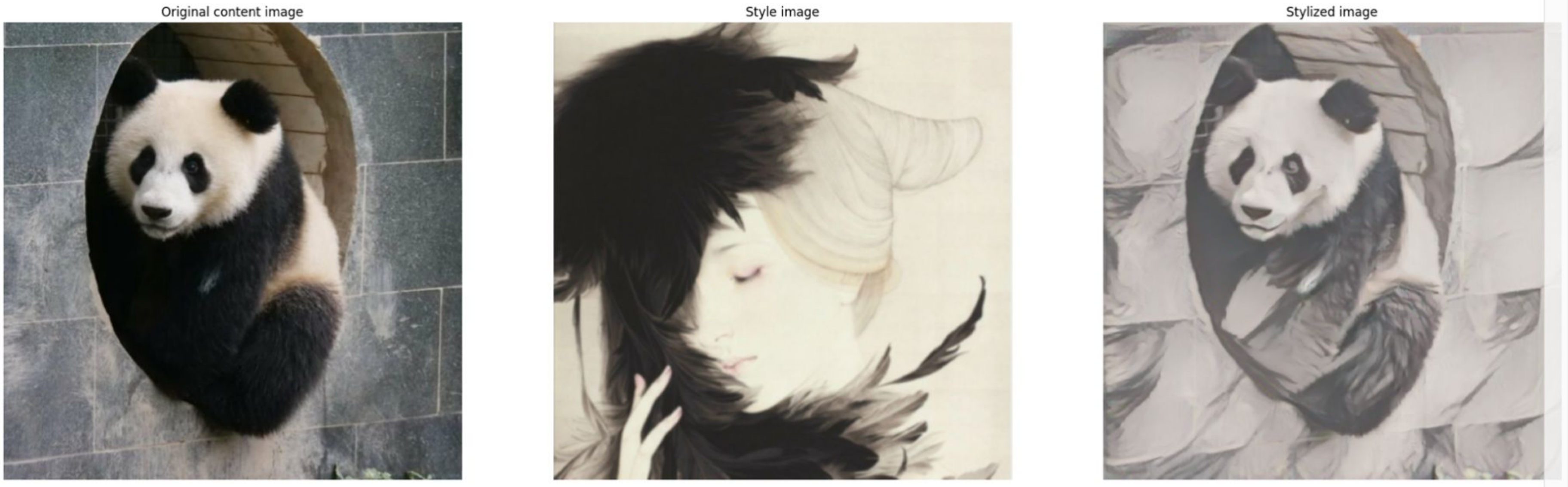
Georgia O'Keeffe Style Cat

Original content image

Style image

Stylized image

Luo Hanlei Style Panda



Model Comparison

	Optimization-Based Transfer	Fast Style Transfer
Pros	• Algorithm easy to understand and simple to implement • Flexible: used to transform any images into artistic representations or merge different artistic styles into a single image	• High performance • Easy to implement • Flexible: applied to different types of media • Time and resource-saving (real-time application)
Cons	• Relatively time consuming • Needs to manually choose the style layers and content layers • Many hyperparameters to tune (layer weights, content loss weight, style loss weight, …) • Confuse content with style	• Computational limitations: requires efficient algorithms and powerful hardware • Lack of control

Model Operations

Model Operations

1. Model Deployment
    a. Train model: Use **Amazon SageMaker**, a fully managed machine learning service that provides a robust and scalable environment for training and testing models
    b. Host model: Use **Amazon Elastic Inference** to host the models, which allows to attach low-cost GPU-powered inference acceleration to **Amazon SageMaker** instances.
    c. API creation: Deploy models as a **REST API**

2. Model Maintenance and Update
    a. Automated Re-training: **Amazon SageMaker** provides features that allow for periodic re-training of the models
    b. Model Monitoring: **Amazon SageMaker** Model Monitor continuously monitors the quality by setting up alerts
    c. Version control: **Amazon S3** provides capabilities for versioning models so that we can always roll back to a previous version if needed

Conclusion

Summary of Findings - Image Classification

- **Transfer Learning EfficientNetB3**
  - Leverage the knowledge learned by pre-trained models, capturing generic features from a broad range of images
  - Then fine-tune on specific task
  - Effectively recognized and distinguished between different animal classes in our dataset
  - Higher accuracy and F1-score
- **Custom CNN**
  - Lacked the initial knowledge and learned representations from scratch
  - Lower accuracy and F1-score

Future Directions - Image Classification

- More efficient and lightweight models
  - For edge devices like smartphones or IoT devices
- Multi-task learning
  - Multiple classifications per input
- Self-supervised learning
  - Using self-supervised learning methods for pre-training
  - Letting the model learns useful representations from unlabeled data
- Explainability and transparency
  - Improving the transparency and interpretability of these models

Summary of Findings - Style Transfer

- **Optimization-based Neural style transfer**
  - Generate highly artistic and visually impressive stylized images
  - Intricate details and faithful style replication
  - Computationally demanding
- **Fast style transfer**
  - Real-time or near-real-time stylization with pre-trained models, such as convolutional neural networks
  - Accessible to individuals without extensive knowledge of deep learning techniques
  - Sacrifices some fine-grained artistic detail compared to optimization-based neural style transfer

Future Directions - Style Transfer

- Real-time style transfer
    - Applications in augmented reality (AR), video streaming, and gaming
- High-quality results
    - Future models for maintaining style consistency and quality in high-resolution outputs
- Style interpolation and multiple style transfer
    - Interpolating between multiple styles or apply multiple styles to one content image at the same time
- Semantic style transfer
    - Understanding and considering the semantic meaning of different parts of the content and style images
- Adversarial training and generative models
    - Improving training stability and preventing mode collapse
    - Google Deep Dream Explore

References

- Chua, L. O., & Roska, T. (1993). The CNN paradigm. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 40*(3), 147-156.
- Gatys, L. A., Ecker, A. S. & Bethge, M. (2015). A Neural Algorithm of Artistic Style (cite arxiv:1508.06576)
- Golnaz G., Honglak L., Manjunath K., Vincent D., & Jonathon S. (2017). Exploring the structure of a real-time, arbitrary neural artistic stylization network. In T.K. Kim, S. Zafeiriou, G. Brostow and K. Mikolajczyk, editors, Proceedings of the British Machine Vision Conference (pp. 114.1-114.12).
- Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning (pp. 6105-6114).
- Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). *A survey of transfer learning. Journal of Big data, 3*(1), 1-40.
- Zhu, J-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2223-2232).