Northwestern

DeepCanvas: A Deep Dive into Celebrity Image Classification and Style Transfer

MLDS 432 Deep Learning Group 5
Jiayue Tian, Yidan Wang, Tianyu Wu, Zhiyi Zhu

Abstract

Facial image classification and style transfer are two compelling tasks in computer vision. This project explores deep learning approaches to recognize and classify facial images and transform these images into oil painting styles. For facial recognition, a convolutional neural network (CNN) was employed, achieving an accuracy of 90.48%. In contrast, the YOLOv5s model was used for its speed and efficiency in object detection, with an average confidence score of 0.55. For neural style transfer (NST), both Classical NST using VGG16 and Adaptive Instance Normalization (AdaIN) were implemented. While Classical NST provided high-quality results, AdaIN excelled in speed and simplicity. These methods demonstrate the potential of deep learning for advanced image processing tasks.

Agenda

- Our Cognitive Questions
- Exploratory Data Analysis
- Model Training and Evaluation
- Model Operations
- Conclusion

Our Cognitive Questions

- How can we effectively recognize and classify facial images using deep learning models?

- How can we transform these facial images into an oil painting style while preserving their original features?

Exploratory Data Analysis

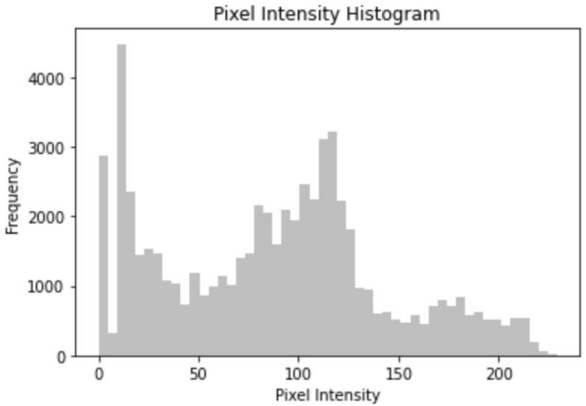
LFW - People (Face Recognition) [Kaggle]

Sample Picture: Ziyi Zhang

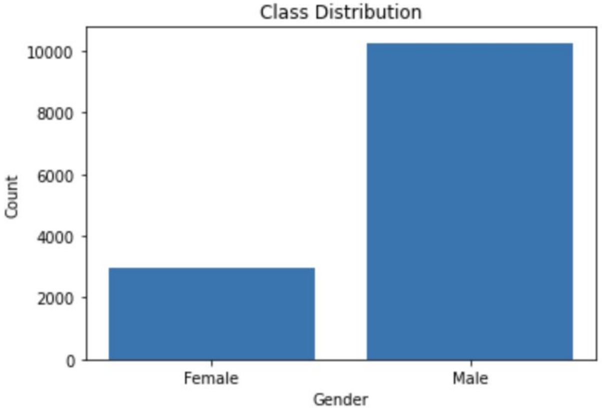
- **Amount:** 13233 face Images, all labeled with the name of the person pictured.
- **Variety:** 1680 individuals have two or more distinct photos in the dataset, allowing for diverse training and testing scenarios.
- **Encoding:** Images are in JPEG format, with each pixel in each color channel (RGB) encoded as a float in the range of 0.0 - 1.0.

Exploratory Data Analysis (Contd.)

- **Resolution:** Original images are 250 x 250 pixels, but the dataset also provides resized images at 62 x 47 pixels by default.



- **Imbalanced Gender Distribution:** With manual labelling, this dataset contains 10266 male faces and 2967 female faces.



Model Training and Evaluation

TASK 1: FACIAL RECOGNITION & CLASSIFICATION

Facial Recognition and Classification involves developing algorithms to identify and categorize facial images based on unique features. We will focus on using deep learning techniques, such as convolutional neural networks (CNNs) and a pre-trained model, YOLOv5s, to detect faces in images, extract and encode key facial features, and classify these images based on their names.

Method 1: CNN

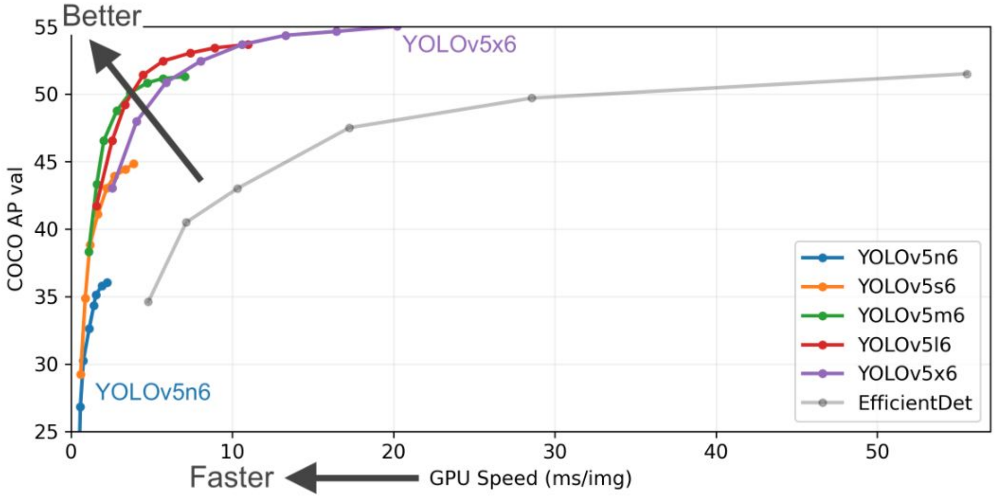
Architecture:

```
Input (1, height, width)
  |
  | -> Conv1 (32 filters, 3x3)
  |     |
  |     | -> ReLU
  |     | -> Max Pooling (2x2)
  |
  | -> Conv2 (64 filters, 3x3)
  |     |
  |     | -> ReLU
  |     | -> Max Pooling (2x2)
  |
  | -> Dropout1 (p=0.2)
  |
  | -> Flatten
  |
  | -> FC1 (200 neurons)
  |     |
  |     | -> ReLU
  |
  | -> Dropout2 (p=0.5)
  |
  | -> FC2 (200 neurons)
  |     |
  |     | -> ReLU
  |
  | -> Dropout3 (p=0.5)
  |
  | -> FC3 (2 neurons)
  |
Output (2 classes)
```

Result: Accuracy Score = 90.48% in the test dataset

Method 2: YOLO

We are using the YOLOv5s model, which has a smaller size and param (suitable for a relatively small dataset).

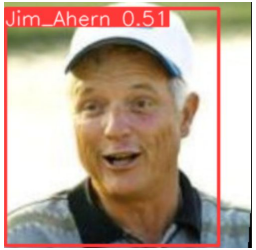


DL algorithm for object detection:

1. Unified detection
2. Grid system
3. Train & Loss function

Results:

Average confidence: 0.55



Task 1 Model Comparison

	Pros	Cons
CNN	• Flexibility • High Accuracy (for this task)	• Training Complexity • Overfitting • Scalability Issues
YOLO	• High Speed • Less background errors • Generalization	• Struggles w/ small objects • Localization errors

TASK 2: NEURAL STYLE TRANSFER (NST)

Neural Style Transfer is a technique of composing images in the style of another image. Neural Style Transfer takes two images as input, namely the image you want to stylise: the Content Image and a Style image. The technique blends their Combination Image such that it resembles the Content Image painted in the style of Style Image.

Method 1: Classical NST

1. Load the pre-trained VGG16 model

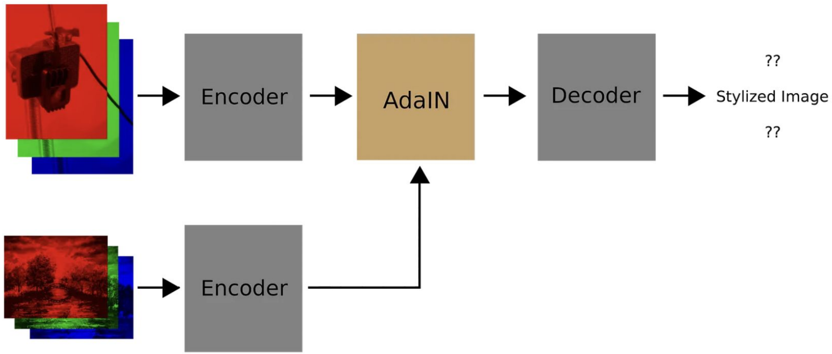
2. Define the Content and Style Representations

3. Compute the Loss and Optimize
   - Content loss
   - Style loss
   - Total variation loss

Method 2: Adaptive Instance Normalization (AdaIN)

What is AdaIN?

$$AdaIn(x, y) = \sigma(y)\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \mu(y)$$



- Paper: *Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization*
- Mean and variance of feature maps tell us about the style of the image.
- Normalize each feature map of the photograph, and add the style of the art piece
- Adjust the encoding of a photograph so that each feature map matches the statistical properties of the corresponding feature map in the painting's encoding.

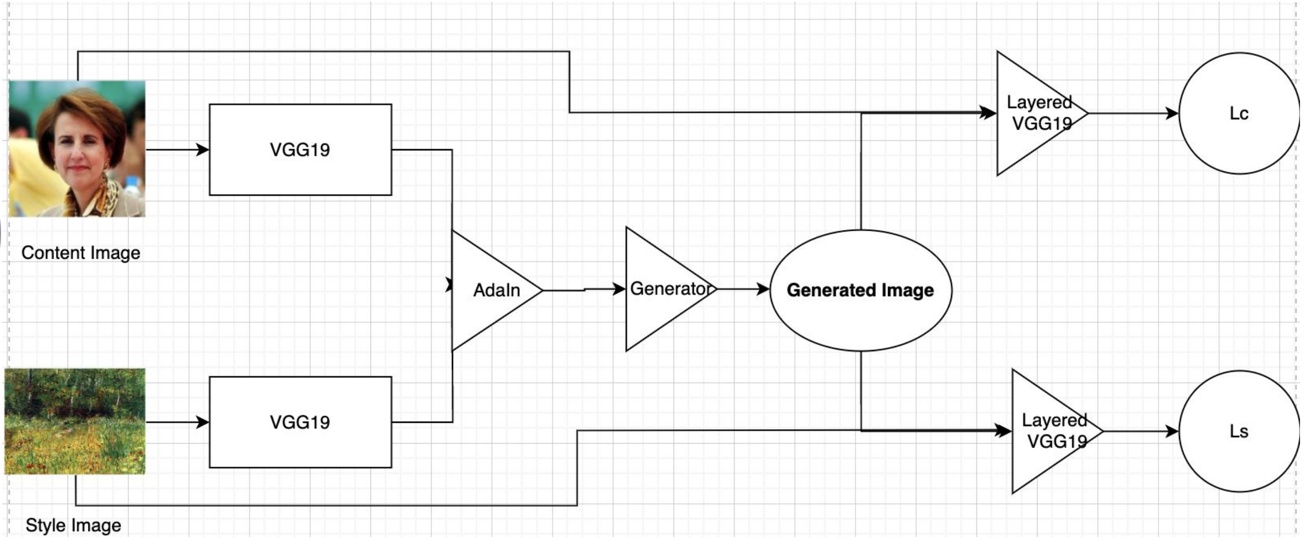
Method 2: AdaIN (Contd.)

Style Transfer Process

$$\mathcal{L}_c = ||f(g(t)) - t||_2$$

$$\mathcal{L}_s = \sum_{i=1}^{L} ||\mu(\phi_i(g(t))) - \mu(\phi_i(s))||_2 + \sum_{i=1}^{L} ||\sigma(\phi_i(g(t))) - \sigma(\phi_i(s))||_2$$

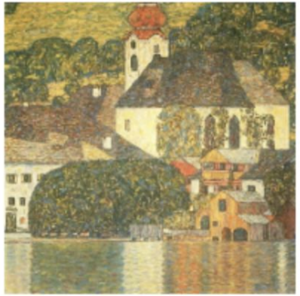
$$\mathcal{L}_t = \mathcal{L}_c + \lambda \mathcal{L}_s$$



Example

AdaIN NST:

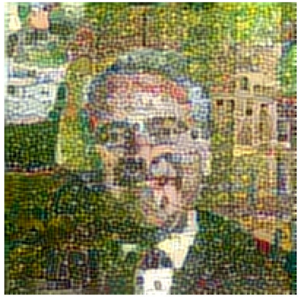
Style Image



+

Content Image

Classical NST:



Task 2 Model Comparison

	Pros	Con(s)
Classical NST	• High Quality • Flexibility	• Computational Intensity
AdaIN	• High Speed • Simplicity	• Generalization • Lack of Fine Control

Conclusion

Facial Recognition and Classification:

- The CNN model achieved an accuracy of 90.48%, indicating its effectiveness in facial image classification.

- The YOLOv5s model demonstrated a balance between speed and accuracy, with an average confidence score of 0.55.

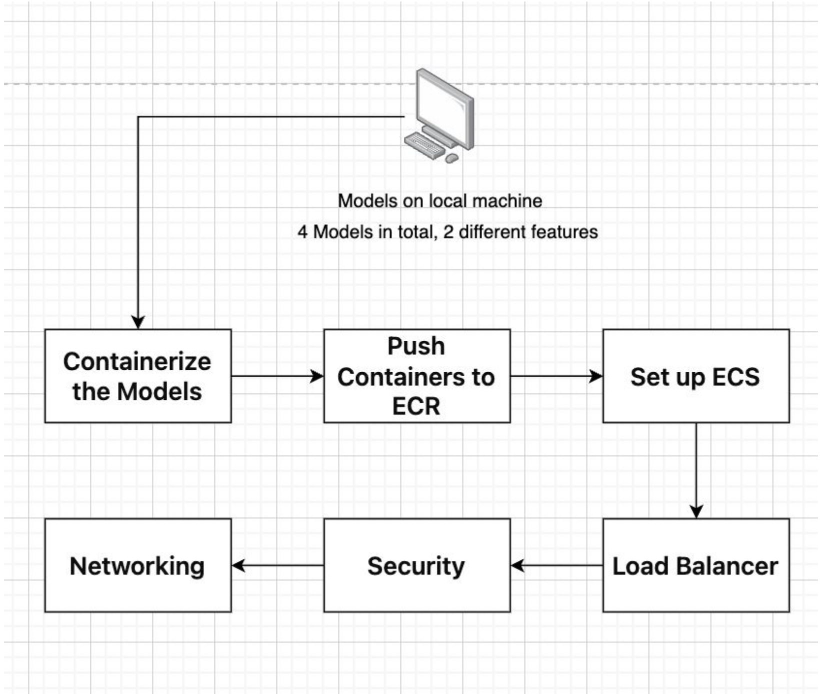
Neural Style Transfer:

- Classical NST provided high-quality results by leveraging the VGG16 model, showing excellent flexibility and detail preservation.

- AdaIN offered a faster and more straightforward approach, suitable for real-time applications, but with less fine control compared to Classical NST.

Appendix

Model Operations - Deployment



Model Operations - Maintenance

- **Monitoring and Logging:**
  - Use Amazon CloudWatch for ECS health and performance.
  - Set CloudWatch alarms for high CPU, memory leaks, and downtime.

- **Continuous Integration/Continuous Deployment (CI/CD):**
  - Establish CI/CD with AWS CodePipeline.

- **Parameter Updates:**
  - Create API-triggered endpoints for real-time parameter adjustments.

- **Version Control:**
  - Maintain model versions in separate ECR repositories or tags.

- **Data Backup and Rollback:**
  - Back up data and configurations in AWS S3.

- **Scale and Optimization:**
  - Leverage ECS auto-scaling based on demand.

References

https://www.kaggle.com/datasets/atulanandjha/lfwpeople/
https://github.com/ultralytics/yolov5/releases
https://antiochsanders.medium.com/real-time-style-transfer-with-adain-explained-f9fa185959aa
https://keras.io/examples/generative/adain/
https://www.kaggle.com/datasets/ikarus777/best-artworks-of-all-time
https://medium.com/ai-techsystems/neural-style-transfer-742dca137976

Source Code:
https://github.com/TianyuWu-Henry/mlds432-group5

Thanks for Listening!