Task 2: Similarity

Ameya Rathod
International Institute of Information Technology, Hyderabad
ameya.rathod@research.iiit.ac.in

March 24, 2025

Abstract

This report presents the methodology and experiments performed to build a model for finding similarities in paintings using a Siamese network architecture. The approach leverages the National Gallery of Art open data set to integrate metadata and images, and employs both contrastive and triplet loss functions for training. Detailed discussion is provided on data integration, model architecture, and training strategies, leading to an effective system for identifying similar portraits based on facial features and poses.

1 Introduction

The objective of this work is to develop a deep learning model that can identify similarities in paintings, such as portraits sharing similar faces or poses. By leveraging the National Gallery of Art open data set[1], the work integrates metadata with image data to form a comprehensive dataset. A Siamese network is utilized to learn an embedding space where similar paintings are close together and dissimilar ones are far apart. Two loss functions, contrastive loss and triplet loss, were experimented with to improve the quality of the learned representations.

2 Data Integration

The initial phase involved integrating multiple data sources from the National Gallery of Art open data set. Three primary tables are used:

- **objects.csv:** Contains metadata about the paintings.
- **media_relationships.csv:** Defines the relationships between objects and media.
- **media_items.csv:** Provides URLs for the images.

[1] https://github.com/NationalGalleryOfArt/opendata

The tables are joined using common keys (e.g., `objectid`, `relatedid`, and `mediaid`) with an inner join strategy. This operation links each painting with its corresponding image, resulting in a unified dataset that enables efficient mapping of object identifiers to image URLs. An image dictionary is then created from this unified data to facilitate fast access during training, particularly for generating positive and negative image pairs or triplets.

3 Model Architecture

The Siamese network architecture is designed to learn meaningful embeddings from painting images. It consists of convolutional layers for feature extraction followed by fully connected layers to produce the final embeddings.

3.1 Siamese Network with Contrastive Loss

The model uses a twin network structure where each branch processes an image. The embeddings from both branches are compared using a contrastive loss function that minimizes the distance between similar images and maximizes the distance for dissimilar images.

```
import torch
import torch.nn as nn

class SiameseNetwork(nn.Module):
    def __init__(self):
        super(SiameseNetwork, self).__init__()
        self.cnn = nn.Sequential(
            nn.Conv2d(3, 32, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2),
            nn.Conv2d(32, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2),
            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(2)
        )
        self.fc = nn.Sequential(
            nn.Linear(128 * 28 * 28, 512),
            nn.ReLU(inplace=True),
            nn.Linear(512, 128)
        )

    def forward_once(self, x):
        output = self.cnn(x)
        output = output.view(output.size(0), -1)
        output = self.fc(output)
        return output

    def forward(self, input1, input2):
        output1 = self.forward_once(input1)
        output2 = self.forward_once(input2)
```

```
        return output1, output2
```

The model was trained for 7 epochs using pre-generated positive and negative pairs, with the final model selected after 5 epochs based on loss stabilization.

3.2 Siamese Network with Triplet Loss

To further refine the embedding space, the model was later trained using triplet loss. This approach considers an anchor, a positive, and a negative sample simultaneously, ensuring that the anchor is closer to the positive than to the negative by a specified margin. The training for triplet loss was conducted for 11 epochs, which further enhanced the quality of the learned representations.

4 Evaluation

The evaluation of the trained models on the test set provides insights into their effectiveness in learning meaningful embeddings for paintings. The performance results for the contrastive and triplet loss models are summarized in Table 1.

Loss Function	**PP Cosine Similarity**	**NP Cosine Similarity**
Contrastive Loss	0.9815	0.2679
Triplet Loss (Anchor-Positive)	0.7788	-
Triplet Loss (Anchor-Negative)	-	0.2432

Table 1: Evaluation results for models trained with contrastive and triplet loss (PP stands for positive pair, NP stands for negative pair).

The Siamese network trained with contrastive loss demonstrated a strong ability to distinguish between similar and dissimilar pairs, achieving an average cosine similarity of **0.9815** for positive pairs and **0.2679** for negative pairs. This indicates that the model successfully clusters similar paintings while pushing apart dissimilar ones.

On the other hand, the triplet loss model produced embeddings where the average cosine similarity between anchor and positive pairs was **0.7788**, while the similarity between anchor and negative pairs was **0.2432**. This result suggests that triplet loss effectively structures the embedding space by ensuring that similar paintings remain closer while maintaining a clear margin from dissimilar paintings.