COMP90086 Computer Vision
Final Project: Totally-Looks-Like Challenge

Zhaoning Lu
1316543
zhaoningl1@@student.unimelb.edu.au

Jingyuan Liu
1402044
jingyuanl9@@student.unimelb.edu.au

I. Introduction

Image similarity matching is a complex and multi-dimensional problem, aiming to use specialized processing methods or models to compare and evaluate the similarity between two images[1]. Such comparisons and evaluations are not only based on traditional factors such as color, texture, and spatial location, but may also involve deeper semantic and emotional similarities. Unlike other computer vision tasks, the real challenge in image matching is, apart from image representation, how to capture and compute the more abstract and subtle similarities between two images. Their deep abstract meanings and even emotional resonance are challenging for existing models[2]. The "Totally-Looks-Like" dataset contains a large number of such images.

Therefore, this project aims to provide a comprehensive set of solutions for the "Totally-Looks-Like" challenge. By delving into various aspects of visual similarity, we hope to design a model that can approach the subtle human perception of similarity and rigorously evaluate and analyze it using scientific methods. This report aims to comprehensively demonstrate the successes and challenges of our experimental methods and models in bridging the gap between computer and human visual perception.

II. Literature Review

Although the development history of computer vision is not very long, numerous methods have been introduced to address image similarity or matching tasks. In the early stages, one of the notable methods introduced was the Scale-invariant feature transform (SIFT)[3]. The essence of the SIFT algorithm is to search for keypoints (feature points) in different scale spaces and compute their orientations. The keypoints it identifies are particularly distinctive and can remain consistent across different scales, rotations, and viewpoints. Subsequently, the Speeded-Up Robust Features (SURF)[4], an improvement based on SIFT, also emerged. Both emphasize the extraction and matching of local features. However, these traditional methods find it challenging to adapt to complex image patterns and details, especially when facing images with abundant similarities and abstract characteristics.

With the advent of deep learning and CNNs, traditional image processing methods were gradually surpassed by deep neural networks, especially in dealing with complex and abstract tasks. Among them, the two-branch network, particularly the Siamese network, stands out in such tasks[5]. Unlike traditional methods, Siamese networks do not directly extract local features from raw images but learn to extract discriminative deep features from images. It consists of two identical branches sharing weights, each branch receiving different image inputs and outputting feature vectors to the same domain. Then, one or more fully connected layers process these vectors, and a contrastive loss function computes the similarity between feature vectors for image matching. This method has stronger generalization capabilities and accuracy, especially when dealing with images having complex patterns and details.

III. Methodology

To solve the image similarity matching problem, we proposed a pipeline containing 3 blocks, which are data preprocessing block, feature extraction block, and a similarity learning block. Figure 1 shows the structure of our proposed pipeline.

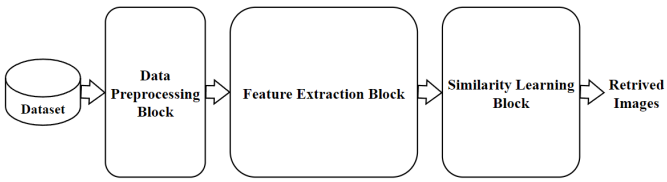

Fig. 1. Image Similarity Matching Pipeline

From a high hierarchy, the data preprocessing block is designed to handle data loading, preprocessing, and mining hard negative samples. The feature extraction block takes images as input and converts them to vectors/tensors containing features extracted. The similarity learning block is used to make predictions based on features we extracted. There are various designs and considerations exist in each of these blocks, which will be covered in detail.

A. Data Preprocessing

Data preprocessing is an essential stage before model training, and the quality of the data directly affects the final performance of the model.

1) Data augmentation: One of the challenges we faced in the initial stages was the lack of training samples, especially for the complex challenge posed by this experiment. To alleviate this problem, our team utilized various data augmentation techniques, including a certain range random horizontal flipping, vertical flipping, rotation, and affine transformations. Through this method, we introduced diversity in our dataset, ensuring that our model does not merely memorize specific pixel values but also attempts to learn more general features.

2) Grayscale conversion: Next, another aspect we considered was converting RGB images to grayscale. During the dataset analysis phase, we noticed that many positive image pairs seemed to have their primary similarities more in terms of structure. As mentioned in[6][7], grayscale images focus more on luminance and edge information. Additionally, after graying, the matrix dimension drops, significantly speeding up operations, while gradient information is still retained. In this way, the model can more efficiently test various methods in the early stages.

3) Removed background: On another note, we found that some images' complex backgrounds might still bias our model. Therefore, our group also used an open-source library [1] to create a dataset without image backgrounds. This way, the model will focus solely on the image's subject, ensuring that, even if a pair have different backgrounds, the model can still capture its main features, ensuring feature consistency and enhancing model robustness. The subsequent results improved the model's ability to detect and understand intrinsic similarities, while reducing computational load.

4) Negative sample: Following this, we hit a certain bottleneck in terms of improving our model. Upon re-examining the training process, we believed that the random negative sample originally used in training had certain shortcomings. These random pairs could not guarantee the model's training intensity. Negative samples should be image pairs that are mistakenly considered positive due to some similarities, yet upon closer observation, have evident differences, or the objects or concepts they represent are not the same. Therefore, using the previous model, we test the training set and randomly selected 200 rights for left (excluding truth pairs) and used the model to select the image with the highest confidence as a hard negative sample.

B. Loss Function

There are two types of Loss Functions that have been implemented and utilized in this project: Hinge Loss and Triplet Loss.

1) Hinge Loss: The primary objective of Hinge loss is to maximize the distinction between positive and negative classes. It is less sensitive to outliers. If a data point is far beyond the margin, it won't affect the loss, giving the model better robustness, especially when faced with complex and messy information.

$$\text{Hinge Loss} = \max(0, 1 - y \times f(x))$$

[1]Daniel Gatis: https://github.com/danielgatis/rembg

$$\text{In Our Model} = \max(0, \text{margin} - y \times \text{Euclidean distance})$$

In our model, by calculating the Euclidean distance between the feature vectors of the two input images, and determining through labels of -1 and 1 whether the distance between each pair of images should be greater or less than a predetermined margin, the model learns and improves. We determined the appropriate value for the margin through continuous debugging, aiming to precisely segregate positive and negative samples. Using targeted training with negative samples, the model is better equipped to delineate a clear boundary between these two types of images for predictions.

2) Triplet Loss: Triplet loss is also very suitable for this task. It learns and distinguishes inter-class differences using three given images–an anchor (A), a positive sample (P), and a negative sample (N).

$$L(A, P, N) = \max(\|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \alpha, 0)$$

Thanks to the loss defined by this triplet, it may perform better in terms of fine-grained similarity metrics. Moreover, to ensure the model learns more from this setup, the selection of N should be as unrecognizable as possible. This was also one of the reasons we mined hard negative samples.
Overall, both loss modes are very suitable for this type of task and its output results, so we experimented with both methods.

C. Simple-CNN-Based Siamese Net (Baseline)

The first method (baseline) we implemented is to use a very shallow CNN network that follows a Siamese structure for feature extraction. The extracted feature embeddings were directly used to calculate Euclidean distance, which can be treated as a similarity measurement. The motivation for implementing this model is to verify the feasibility of Siamese structure on our specific image similarity matching task. As mentioned in the literature review, the core idea of the Siamese network is to use a shared branch to process two distinct input simultaneously and gives their embeddings, which can be used to do similarity measurement. For this baseline model, we compacted 4 Conv2D layers, each connected with a ReLU activation function and a MaxPooling layer. Figure 2 illustrates the structure and parameters of the model.

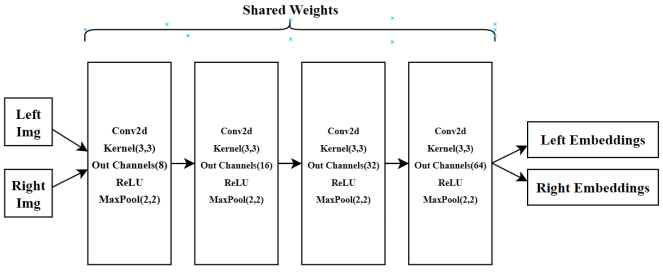

Fig. 2. Structure for Feature Extraction Net

For this model, we have implemented and tested on both grayscale and RGB datasets separately. The loss function used for the Simple-CNN Based Siamese Net was hinge loss with L2 regularization (L2-Lambda=0.01). Considering the size of

the model, there is no dropout layer involved. The model is trained for 20 epochs with Adam Optimizer, the learning rate used is 0.001. The extracted feature embeddings then are directly used to calculate the Euclidean distance. The inverse of the Euclidean distance is treated as a 'confidence score' for an image pair, as similar image pairs would have smaller Euclidean distances. In terms of prediction, the model will calculate the 'confidence score' for each of the image pairs, using the feature embedding obtained, and save it to a CSV file.

D. ResNet-Based Siamese Net with Similarity Learning

The second method we used is using a residual network-based Siamese structure to extract image feature embeddings and then pass them into the similarity learning network. The motivation for implementing this method was to have more parameters to learn deep and shallow image features. Our group have made two variations for the residual net-based Siamese structure, which are based on Res50 and Res101. Both networks use pre-trained weights for ImageNet[8]. To better capture both high-level and low-level features like edges and textures, we extracted the first four layer's outputs to combine with the final outputs as a new feature. After fine-tuning the model, the extracted features then were used as inputs to a separate similarity learning network. The similarity learning network is a simple model with just three fully connected layers compacted together, besides that, there is a dropout layer with a probability of 0.1 connected with the first fully connected layer to prevent overfitting. Figure 3 shows the structure and parameters of our similarity learning model. The output of the similarity learning model is a probability score range from [0 1]. The loss function used for the similarity learning network is triplet loss, which was introduced in the previous section.

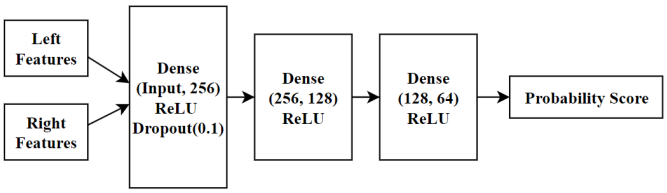

Fig. 3. Structure for Image Similarity Learning Net

Our group used two dataset variations in this method, which are the augmented RGB dataset and the augmented RGB dataset without background. The training data and validation data were split to an 8:2 ratio, which is loaded and controlled by a dataloader. During the fine-tuning phase of the ResNet, the loader will load a batch of 64 image pairs with equal amounts of positive and negative image pairs. The negative samples follow both random choice and hard negative mining mechanisms, as we introduced in the previous part. For the ResNet-Based Siamese Net, we used Adam as an optimizer with a learning rate set to 0.001 and trained for 5 epochs. The Similarity Learning Net used triplet loss as the loss function, it was trained for 40 epochs with a learning rate set to 0.001. The validation method for the similarity learning model is based on top1 and top2 accuracy, we obtain this following the same mechanism as the real competition rules. During prediction, we use the similarity learning model to calculate the similarity for features of each of the image pairs and save the similarity score to a CSV file.

IV. Evaluation Metrics

A. Top-1 and Top-2 accuracy

Top-1 accuracy is the standard classification accuracy, which takes the image with the highest confidence as the actual predicted result. In contrast, top-2 accuracy is more in line with the testing approach of this experiment and will serve as the primary basis for evaluating the model's final performance. Analysis and comparison of top-1 and top-2 can help us ascertain the model's actual performance and identify some potential issues. For instance, if there's little to no difference between top-1 and top-2 accuracies, it might suggest that the model has some overfitting problems.

V. Result & Discussion

A. Simple-CNN-Based Siamese Net (Baseline)

The baseline model has been tested on both our Grayscale and RGB augmented datasets. As discussed before, the motivation for using Grayscale images is to compare the image similarity matching performance with RGB images. The Top 1 and Top 2 accuracy on the validation set of the model are shown in Table I.

TABLE I
Top1&Top2 Accuracy for Baseline

Metrics	Grayscale	RGB
Top 1 Acc	0.21	0.21
Top 2 Acc	0.34	0.39

Since the similarity matching task is to retrieve the top 2 most similar image pairs in a set of 20 image pairs, the performance based on random choice should be 0.1 for Top 2 and 0.05 for Top 1 Accuracy. Based on our experimental results, we observed a significant improvement compared with random choice, which means that using simple CNN as Siamese branches is indeed suitable for this specific task. This verified our initial research on the Siamese structure.

It is also noted that there always exists a performance gap between the Top 2 Accuracy on the Grayscale dataset and the RGB dataset. To better understand this observation, we performed hard wrong prediction mining on both trails. To do this, we collected a bunch of wrong-predicted image pairs and sorted them according to their 'confidence score'. Intuitively, we are trying to find the image pair that shows the highest confidence to be matched but indeed is not (False Positive). Upon examining these samples, we identified some potential issues. Figure 4 showcases one popular type of wrong prediction by our model.

Fig. 4. Problem of Grayscale

As the figure shows, the correct pair for the image on the far left should be the one in the middle. However, the model identified the image on the right as the negative, and it's challenging to pinpoint their similarity. By examining their grayscale versions, the problem became evident: the edge patterns of the waves in the grayscale images were very similar to the one on the right. This revelation prompted us to re-evaluate the trade-offs between grayscale and RGB modes and further test them.

B. ResNet-Based Siamese Net with Similarity Learning

After seeing promising results from our baseline model, we decided to further investigate ResNet-based Siamese structures. As discussed earlier, models trained on RGB images tend to have higher performance, therefore, we only used RGB dataset and RGB without background dataset to train and evaluate the model. The motivation for removing the background was considering some sophisticated background would bias the model. Figure 5 demonstrates the outcome of performing background removal on our RGB dataset.

Fig. 5. Background Remove Outcome on RGB Datasample

After using the Residual Network to extract feature embeddings, and passing them through the similarity learning network, we obtained the top 1 and top 2 accuracy of retrieving similar image pairs.

The Top 1 and Top 2 validation accuracy on the RGB dataset are shown in TableII, while the Top 1 and Top 2 validation accuracy on the RGB without background are shown in TableIII.

TABLE II
TOP1&TOP2 ACCURACY FOR RESNET-SIAMESE WITH SIMILARITY LEARNING (RGB)

Metrics	Res50	Res101	Res152
Top 1 Acc	0.28	0.26	0.27
Top 2 Acc	0.49	0.46	0.45

TABLE III
TOP1&TOP2 ACCURACY FOR RESNET-SIAMESE WITH SIMILARITY LEARNING (RGB NO BACKGROUND)

Metrics	Res50	Res101	Res152
Top 1 Acc	0.31	0.26	0.28
Top 2 Acc	0.51	0.47	0.45

Based on the above results, the first thing we noticed is smaller ResNet tends to give better performance. Under this specific scenario, one of the most likely reasons would be larger networks normally require more training data to work well. In addition to that, on the background removal level, models trained on RGB images without background generally give better retrieval accuracies. This suggests that the background might introduce some noise or irrelevant features, which matches our intuition.

The independent results from all these variations of ResNet were also used to perform a majority voting, where each of the probability elements is multiplied with its own model's accuracy and added together in the end. It can be observed that the final ensemble of models can handle these situations more comfortably. Different ResNet variants, with varying depths, will capture different features. By combining the decisions of these models, the ensemble can capture a richer similarity. Furthermore, by weighting the decisions based on the accuracy of each model, the reliability of the results is further ensured.

TABLE IV
TOP1&TOP2 ACCURACY FOR MAJORITY VOTING

Metrics	RGB	RGB no background
Top 2 Acc	0.50	0.54

VI. CONCLUSION&FUTURE WORK

In summary, for this image similarity matching task, due to the presence of very complex or ambiguous samples, it's challenging to determine whether the two images are similar. Our group proposed a promising pipeline with different block variations to handle the task, where each of the variations has its strengths and shortcomings. Therefore, an ensemble of models can handle these cases more gracefully, which gives a final Top 2 accuracy of 0.54.

For future development, we could investigate more on extracting intermediate layers output for fusion, as the current implementation hasn't yielded the anticipated advancements observed in the industry.

REFERENCES

[1] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[2] M. Chen, L. Zhang, and J. P. Allebach, "Learning deep features for image emotion classification," in *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2015, pp. 4491–4495.

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[4] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer, 2006, pp. 404–417.

[5] I. Melekhov, J. Kannala, and E. Rahtu, "Siamese network features for image matching," in *2016 23rd international conference on pattern recognition (ICPR)*. IEEE, 2016, pp. 378–383.

[6] C. Kanan and G. W. Cottrell, "Color-to-grayscale: does the method matter in image recognition?" *PloS one*, vol. 7, no. 1, p. e29740, 2012.

[7] A. Güneş, H. Kalkan, and E. Durmuş, "Optimizing the color-to-grayscale conversion for image classification," *Signal, Image and Video Processing*, vol. 10, pp. 853–860, 2016.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.