# COMP90086: Final Project Report

Tanzid Sultan
tanzids@student.unimelb.edu.au

## I. Introduction

In this project, we take on the task of designing a computer vision algorithm to solve the Totally-Looks-Like (TLL) challenge. This task is derived from a dataset containing pairs of images that are perceived to have semantic and visual similarities [5]. In particular, the main objects in each image is either a person (celebrity), a fictional character or an inanimate object that resembles a person/fictional character. The criteria for judging similarity in these image pairs are diverse and span a vast range of features such as color, texture, shape and more intricate contexts and concepts, specifically cultural concepts, as can be seen for instance in the similarity between the person's hairstyle and the banana with nutella on top in the example pair from Figure 1. In order to capture such a rich set of features and quantitatively measure such similarity between images, we have chosen to implement an end-to-end deep-learning model, which circumvents the need for designing hand-engineered features and is a data-driven approach.

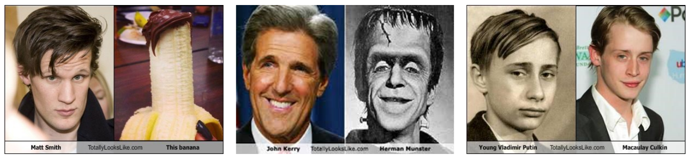

Fig. 1. Some example left-right image pairs from TLL dataset.

## II. Related Work

A common application of image similarity learning is for face recognition tasks, in which a machine learning model is trained on just a few template images of a person and is then able to detect that person in other previously unseen images. This type of task is also referred to as *One-shot Learning* because the model has to learn from only one or few similar pairs of images. A highly effective technique for solving this task is with the use of a *Siamese Neural Network* [3] to learn the feature embeddings which are able to capture information about a person's appearance and can robustly identify them in unseen images.

## III. Method

Our dataset consists of a set of "left-right" image pairs and the goal of our algorithm is to be able to find the most likely "right" image from a given set of candidates for a given "left" image. We can frame this task as a similarity measurement problem in which we compute the similarity between the left image with each possible candidate and select the one with the highest similarity score. To solve this task, our approach centers around training a feature extractor model which maps images from pixel-space into a low-dimensional feature embedding space in which we would expect similar images to be in close proximity. Inspired by state-of-the-art face recognition systems such as *FaceNet* [6], our algorithm utilizes the following key components for this task: a (deep-convolutional network based) feature extractor model and a siamese triplet model with a triplet loss function.

### A. Siamese Network and Triplet Loss Function

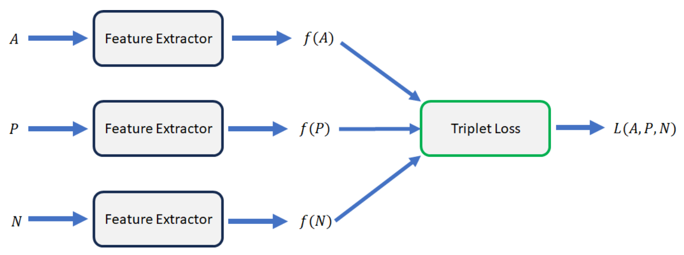

Fig. 2. High-level overview of the siamese network architecture.

The siamese network has a simple architecture designed specifically for the task of supervised similarity learning, which it accomplishes with the use of a triplet loss function. As shown in Figure 2, this network takes in three input images, i.e. a triplet of images: an anchor image (A), a positive image (P) and a negative image (N). P represents a ground truth image which is similar to A and N represents a foil which is dissimilar to A. For a left-right image pair from the TLL dataset, the left image would be assigned as A and the right image would be assigned as P. Each image is then mapped into an embedding vector by the feature extractor and the resulting embeddings are used to compute the triplet loss which is defined as follows:

$$L(A, P, N) = |f(A) - f(P)|^2 - |f(A) - f(N)|^2 + margin \tag{1}$$

where $f(\cdot)$ denotes an embedding vector and $|f(\cdot) - f(\cdot)|$ denotes euclidean distance between embedding vectors. We can see that minimising this loss function is equivalent to simultaneously minimizing the first term, which is the distance between the A and P embeddings, and maximizing the second term which is the distance between the A and N embeddings. A positive margin is also added to ensure a minimum non-zero separation between the P and N embeddings (which also prevents $f(P) \approx f(N)$). By training the network on a large set of triplets and minimizing the triplet loss function (by optimizing the parameters in the feature extractor model), it is then possible to learn feature embeddings which can be used for similarity measurement tasks.

The selection of negative images for the triplets can have an appreciable impact on the learning process. Using harder negative images, i.e. negatives which are closer to the positive and anchor in terms of similarity (such as nearest neighbors in embedding space), tends to yield more robust embeddings (e.g. [6]). However, due to limited compute resources and for sake of simplicty, in our approach we chose to select negative images randomly for each triplet, so there is scope for future improvement.

## *B. Transfer Learning with Pre-trained Embeddings*

Due to the small size of our dataset, training a feature extractor model from scratch is not a feasible option, because there would be high risk of overfitting to our limited training data resulting in the model not learning more general feature representations that may be useful for unseen images. Therefore, we have chosen to bootstrap from an existing deep-convolutional network model, the *ResNet50V2* [2] which has been pre-trained on the large ImageNet dataset and is often used as a high-accuracy benchmark. Even though the ImageNet dataset is more centered around natural and inanimate objects/scenes while our dataset centers around highly contextual objects such as people and characters (or inanimate objects that look like characters), there is a large degree of feature overlap between the two datasets and therefore the features learned by the pre-trained model can be transferred to our task. We also want our model to learn features that are more unique and relevant to our dataset, so we choose to replace the top part (consisting of fully-connected layers for image net classification task) of the ResNet50V2 network with a set of different layers with trainable weights which can be fine-tuned to adapt the features to our dataset accordingly.

## *C. Data Augmentation*

To help our model achieve better generalization capability and make it less prone to overfitting to our small dataset, we have chosen to artificially increase the size of our dataset using the following random image augmentation techniques (mostly affine transformations): zooming (by up to factor of 0.1), horizontal and vertical translations (by up to factor of 0.1), rotations (by up to 10 degrees), horizontal flipping and uniform brightness and contrast adjustments (by up to a factor of 0.1). These particular image augmentation techniques were found to be most suitable for our dataset. Including zoomed images in our training data makes the model more robust to learning scale-invariant features, similarly rotating the images allows the model to detect the same feature at different orientations. We also incorporated horizontal and vertical translation of the images, because this would make the model more robust to variations in positions of objects in the image. We also included horizontal flipping of images as this may help the model focus on features that are invariant under reflections (Although we initially assumed that horizontal flipping may interfere with our model's ability to learn features that are relevant for finding images of objects with very similar pose such as images containing persons looking in a certain direction, in implementation, it did not have any adverse effect.)

## *D. Training Experiments and Evaluation*

To train our siamese network, we have used the following strategy to generate triplets. First we hold out a portion (20%) of the left-right pairs for validation purposes and use the remaining pairs for training. For each left-right training pair, we select $N$ different foils randomly from the combined set of all training images, making sure to exclude any duplicates and repetition of the left or right image. This yields $N$ different combinations of triplets using the same left-right pair but a different foil (making $N$ a model hyperparameter). Ideally, we would want $N$ to be as large as all images in the training set, however this is too computationally prohibitive. We also use a similar method to generate triplets from the validation set.

Then we run two phases of experiments in which the complexity of the feature extractor is varied, however the dimensions of the embedding vectors is fixed at 256. In Phase 1, we freeze all layers in the Resnet50V2 pre-trained model, and add three trainable fully-connected (FC) layers (the size of each of these FC layers is tuned) which yield our embedding vectors. Then we train this model with our prepared triplet data. In Phase 2, we unfreeze some of the higher convolutional blocks of the ResNet so that their parameters can be trained, followed by the same extra fully connected layers which we had in the previous phase. Since the higher layers of the pre-trained ResNet contain high-level features which are specifically relevant for the original Imagenet classification task which it was trained on, by allowing these weights to be re-trained, we can fine-tune the higher level features such that they can be adapted to our similarity task. By unfreezing these layers, we also substantially increase the number of trainable parameters in our model and hence the potential for greater expressive power of the learned features. However, we were cautious about tuning the learning rate to a small value to avoid quickly overfitting to our small training dataset. Finally, within each phase, we run several sub-phases in which we vary our hyperparameter $N$ to investigate the effect of having more foils for each anchor-positive pair.

In each sub-phase, we train our models close to convergence and use early stopping when the mean triplet loss on the validation set triplets stop decreasing and/or starts increasing. To evaluate the performance of our trained models, we have chosen to use the Top-1, Top-2 and Top-5 accuracy metrics computed for the validation dataset. For each left image from the validation set. we create a set of 20 candidate images which include the ground truth right image and 19 other images drawn randomly (without replacement) from all images in the validation set. Then we compute the squared euclidean distance between the left image and each candidate image, re-scale the distances to be in the range [0,1] and assign similarity scores (which we define as 1 minus the re-scaled euclidean distance) and compute the top-1, top-2 and top-5 prediction accuracy based on whether the ground truth right image is among the images with highest similarity scores. In addition, we visually inspect a random subset of the validation images and their candidate sets to see whether the similarity predictions of our model make sense, qualitatively.

## IV. RESULTS

### A. Phase I

TABLE I

| N | Top-1 Acc | Top-2 Acc | Top-5 Acc |
|---|---|---|---|
| 20 | 0.27 | 0.43 | 0.67 |
| 50 | 0.27 | 0.43 | 0.65 |
| 100 | 0.28 | 0.40 | 0.65 |
| 100[a] | 0.28 | 0.40 | 0.66 |

[a]with data-augmentation

Table 1 shows the Top-1,2 and 5 accuracies for the validation dataset from our Phase 1 experiments. We see that the model performs significantly above a random chance baseline. In particular, the Top-5 accuracy is over 60% which indicates that the features learned by the model are indeed useful for our similarity task. The range of N values that we have chosen to explore did not have any significant impact on the accuracies, which may indicate that the value may need to be substantially larger than 100 to see any performance boost, however the improvement may only be marginal and will be offset by the much larger compute resource requirement. Additionally, our choice of 256-dimensional embedding vectors may not be sufficient to capture the broad range of features needed for the task. To also get a sense of what kind of features that are being learned well, we found it useful to visually inspect some of the instances from the validation set.

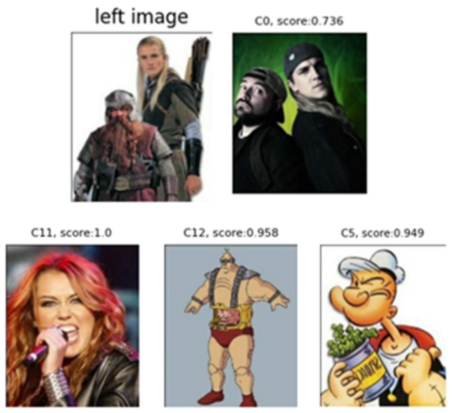

Fig. 3. Phase I validation instance which fails Top-5 predictions. Top row shows left-right pair. Bottom row shows top 3 predictions.

Figure 3 shows an instance for which the model fails to predict the ground truth image in the Top-5. Note that the colors and brightness levels drastically differ across the two images in the pair. The left image is much brighter and has a plain white background, while the right image has a dark background, shadowy textures and an overall greenish tint. Simpler features such as color, general shape and texture of the image will not be sufficient to capture the similarity between the images in this case. However, we may note that the top 3 predictions share a number of common features with the left image, for example the third best prediction (Popeye character) has a similar white background as the left image.

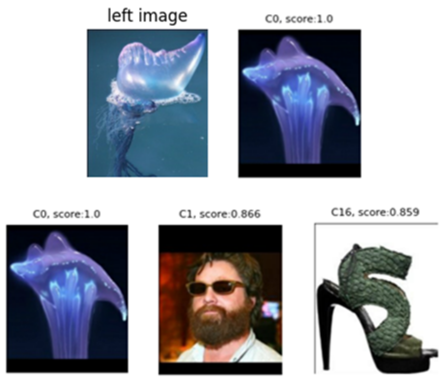

Fig. 4. Phase I validation instance which succeeds Top-1 predictions. Top row shows left-right pair. Bottom row shows top 3 predictions.

Figure 4 shows an instance in which the model is able to make a Top-1 prediction of the right image. Note that in this case, the most prominent features which encode the similarity between the left and right image are the colors and the overall shape and our model seems to be able to learn these simpler features with relative ease.

### B. Phase II

TABLE II

| N | Top-1 Acc | Top-2 Acc | Top-5 Acc |
|---|---|---|---|
| 20 | 0.43 | 0.565 | 0.76 |
| 50 | 0.43 | 0.59 | 0.79 |
| 50[a] | 0.41 | 0.54 | 0.77 |

[a]with data-augmentation

Table 2 shows the results from our Phase 2 experiments, in which we unfroze some of the higher covolutional layers in our pre-trained ResNet feature extractor to allow for fine-tuning of the weights. Even with only 20 foils, performance across all three accuracy metrics have improved by over 20% compared to the best result from the previous phase of experiments. In particular, we note that the Top-5 accuracy is near 80% which is quite high. Figure 5 shows an example of a validation instance with Top-1 prediction. The features which the model is using to match this image are the hair texture and colors, as evident from the 3 best candidates predicted.

Overall, the Phase 2 experiments highlight the significant impact from fine-tuning the feature extractor model. As an aside, we also note that the model which we used in our best performing submission to the Kaggle competition was

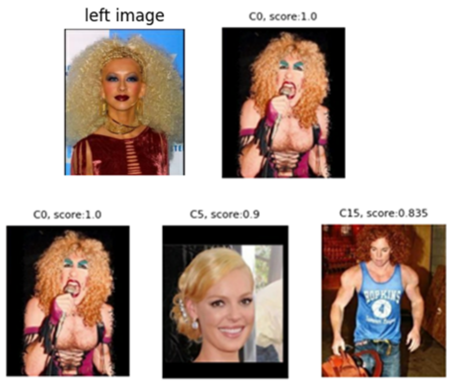

Fig. 5. Phase II validation instance which succeeds Top-1 predictions. Top row shows left-right pair. Bottom row shows top 3 predictions.

the same model which we used in our Phase 2 experiments. The Top-2 prediction accuracy of our model on the Kaggle test set is almost the same as the results obatined on our validation set which supports the reliability of our evaluation strategy.

## V. Conclusion and Future Improvements

Even though fine-tuning the higher level layers of our feature extractor model demonstrably improves performance, our model still falls short of acheiving anywhere close to human-level visual perception, primarily due to the insufficient expressive power of the learned features embeddings. Getting the feature extraction correct may be the single most important component for solving a similarity task such as ours.

In order to enable our model to learn more complex features, it may be necessary to use a more sophisticated method for selecting the foil image in the training triplets. Clearly, simple random selection of foils is not the best approach. A better approach may be to iteratively replace the foils in each training triplet (e.g. at the end of each training epoch) such that only the hardest foils are selected. This can be done, for instance, by computing the nearest neighbor images to the left image from the entire training set.

There is also scope for improvement in the fine-tuning process of the pre-trained feature extractor model. In particular, a more sophisticated approach that we would like to have investigated was the effect of iteratively unfreezing more of the lower layers in the ResNet coupled with learning rate scheduling such that the rate decays gradually over training epochs and this process could be continued over a large number of epochs. This approach may have allowed the model to learn more expressive features and achieve higher accuracy for our similarity task. Transformer-based pre-trained computer vision models which are the current state-of-the-art would also be another promising and more computationally feasible alternative to using convolution-based models such as ResNet. E.g. it has been demonstrated [1] that it is possible to design a purely self-attention based architecture for vision tasks which can perform comparably to the state-of-the-art convolutional networks while requiring only a fraction of the compute resources. Yet another option to explore would be the use of higher dimensional embedding vectors which may enable the model to learn a substantially larger and more nuanced set of feature representations. Multi-stage models and ensemble techniques that can combine specialized models which have been trained to learn different sets of features may be another viable strategy for solving our task.

Finally, instead of targeted efforts into improving the feature extraction ability of the model, we may also look into techniques for cleaning and pre-processing the dataset to ensure that we are not sabotaging the supervised training of the feature extractor by unintentionally supplying it with incorrect/duplicate data. We may also be able to improve performance by using more sophisticated techniques for image augmentation/synthesis such as Generative Adverserial Network-based methods, non-affine geometric transforms, pixel-level transformations and even photo-metric methods [4].

## References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[3] Gregory R. Koch. Siamese neural networks for one-shot image recognition. 2015.

[4] Alhassan Mumuni and Fuseini Mumuni. Data augmentation: A comprehensive survey of modern approaches. *Array*, 16:100258, 2022.

[5] Amir Rosenfeld, Markus D. Solbach, and John K. Tsotsos. Totally looks like - how humans compare, compared to machines. In *Computer Vision ACCV 2018*, pages 282–297, Cham, 2019. Springer International Publishing.

[6] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.