Efficient Image Similarity Modelling using Supervised Contrastive Learning

K5001[1]

[1]**Dept. of Engineering, University of Cambridge**

ABSTRACT

This article delves into the application of supervised contrastive learning to help in training the embedding of a pre-trained network. Training the model on the embeddings as the output helps in clustering similar images in an n-dimensional space. These embeddings can be represented as an n-dimensional vector and the similarity of the images can be calculated through various distance measuring techniques like cosine similarity, euclidean distance, etc. In this article an EfficientNet model was trained on 50 classes of the Tiny ImageNet dataset through supervised contrastive learning and similarity between images were calculated through the cosine-similarity of the embeddings. This methodology achieved 87% accuracy on the Validation images of Tiny ImageNet and 65% accuracy on an unseen dataset of signature forgery detection.

Keywords: Supervised Contrastive Learning, Image Similarity Model, Tiny ImageNet

INTRODUCTION

This article delves into the designing and training of a network whose task is to match two images. It is meant to provide an answer on whether two images are similar or dissimilar. The basis of similarity of two images are derived from the class labels of the Tiny ImageNet Dataset. If two images belonged to the same class then the model is meant to output that the two images are similar and dissimilar otherwise.

One of the main issues with Tiny ImageNet dataset is the fact that the images used in the dataset are too small (64 X 64). This lead to an issue of numerous important fine-grain details that were omitted from the image. Examples of these images are shown in Figure 1.

Figure 1. Example Figures present in the Tiny ImageNet Dataset

Three models were taken into consideration based on the previously implemented papers. The respective discussion around the papers taken into consideration will be expressed in the section Literary Review. The first model that was tested out had a pair of images as inputs that were passed down to a pre-trained model to get the embeddings. These embeddings were then concatenated and forwarded to a fully connected layer with a single output node that held the value 1 if similar and 0 if dissimilar.

The second model was based on the concept of contrastive learning. Image pairs were used as an input for the model whose similarity was used in training the embeddings. The loss function was based on the euclidean distance between the embeddings. This technique has been used in numerous research problems like signature verification and kinship detection.

The third model was based on supervised contrastive learning model. The idea behind the model is to train the embeddings so that inputs belonging to same classes are clustered together uniquely in an n-dimensional space. The embeddings are clustered around an anchor that help create separability

amongst different classes. This performed the best out of the three and achieved an accuracy of 87% on unseen images of trained categories and a 65% accuracy on categories that weren't shown to the model.

LITERARY REVIEW

Krizhevsky et al. (2012) presented an architecture eminently known as the AlexNet. This architecture had sequential convolutional neural networks that were forwarded to a fully connected network. The purpose of convolutional network was to extract features from the images and reduce them to their respective embeddings. These embeddings were then forwarded to a fully connected layer for prediction of the input into it's respective class labels. This research set the blueprint for most of the image classification and object detection techniques to follow like VGG, SSD etc.

Dey et al. (2017) researched on the topic of Signature Forgery Detection. The concepts of using a Siamese network to compare an original signature with another signature and predict whether it was a forgery or a genuine signature. The concept of detecting this forgery was based on how similar the new signature was to the original one and hence the application was similar to that of prediciting whether two images were similar. This type of a network used a contrastive loss function to train the embeddings based on the two inputs. The contrastive loss was based on the similarity of the two images and the quantitative aspect of the similarity was the euclidean distance of pairwise elements in the embeddings. If two images were similar, the euclidean distance would be low, else if they were dissimilar, the euclidean distance would be high.

An improved concept of training the embeddings was brought forth by Khosla et al. (2020). In the research, the author proposed that rather than training embeddings through image pairs, it was imperative for the embeddings to be trained with an anchor point. This way the embeddings could be thought of as an n-dimensional vector and every embedding of an image could have its respective anchor point based on the class label. This helps in visualizing similar input images as clusters in the n-dimensional space around the anchor points. Pre-training the embeddings has lead to significant gains even with respect to augmented images as it is demonstrated in the article itself where this methodology had an overall 1% improvement than previously implemented state of the art models.

Based on these literature, the idea of the model presented in this report was formed. Three models were implemented, each improving upon the shortcomings of the previous one.

MODEL SELECTION

The specifications of the machine that was used to conduct this research are given in Table 1. Please note that this research was conducted on the author's personal machine.

Component	Specification
CPU	Ryzen 3600 (6 core CPU)
GPU	NVIDIA RTX 2060 Super (8GB VRAM, 2176 Cuda Cores)
RAM	24 GB

Table 1. Specifications of the machine used to conduct this research

Datasets

There were two datasets that were used in this study. One of the datasets was used for training and validating the model on seen categories of data and the other dataset was used entirely for validation purposes.

Tiny ImageNet Dataset

This dataset is a subset of the ImageNet dataset. It contains 200 classes where each class contains 500 training images for a total of 100000 images. These images are also downsized to 64X64 as compared to the original size of 299X299. This poses a challenge while training the model as a lot of fine-grain features are lost in downsizing the image. This forces the model to learn on the coarse-grain and structural features in the images. For the purpose of the study, the original train-test-validation split was maintained so that the unseen images would be remain unseen by the model.

ICDAR 2011 Signature Dataset

This dataset consists of signatures of dutch users that were both genuine and fraudulent. This dataset had pairs of signatures and their respective labels annotated hence it was tailored to the research that was conducted in this report. The idea behind using this dataset was that the similarity metric between genuine and forged signatures would be high, hence, the trained model (if trained correctly) would be able to distinguish between genuine and forged signatures based on a particular given threshold. This tests the fine-grain applicability of the model.

Model 1

Data Augmentation for Model 1

For the task of training Model 1, the Tiny ImageNet dataset was modified to consist of pair of images that were similar and dissimilar. From each class of Tiny ImageNet dataset 200 unique pairs were generated so that they can be labeled as a similar class. It was also enforced that these pairs have not been taken into consideration so that only unique pairs were considered into training the model. This lead to 40,000 image pairs that were similar. The rest of the 40,000 images were selected at random from the pool of the training image files and it was enforced that these images not have the same class so that these can be labeled as the dissimilar class. This lead to a total of 80,000 image pairs for training.

Validation images were generated along the same blueprint of training images but these images were taken up from the val folder of the dataset. To maintain 80-20 split of training and validation, a total of 20,000 image pairs were generated that had 10,000 similar and 10,000 dissimilar image pairs.

The intuition behind keeping the training and validation folders separate is to have zero overlap between the image pairs. Splitting the already existing training pairs can have multiple cases where one of the images used in the pairs be present in both validation and training split. Hence to avoid this scenario entirely the training and validation images were kept entirely separated in the original dataset.

The basic augmentation following after generating these image pairs are given as follows:

- Resize: Since the images were tiny as compared to other benchmark datasets that pre-trained models have been trained on, it was imperative to resize the images to dimensions that were comparable to the standard image dimensions for Deep Learning. For the first model, the images were resized to 384X384 using the bicubic interpolation to fill in the missing pixel values.
- RandAugment: This is a functionality in PyTorch that applied a random number of augmentations derived from the research by Cubuk et al. (2020). The number of random augmentations were 2 and the magnitude of them were chosen to be 7. The reason to choose these parameters as these were the parameters chosen in the original research by Cubuk et al. (2020). As shown in Figure 2, the augmentations included rotation, translation and centre crop augmentations to name a few.

The sample of this dataset is shown in Figure 2.

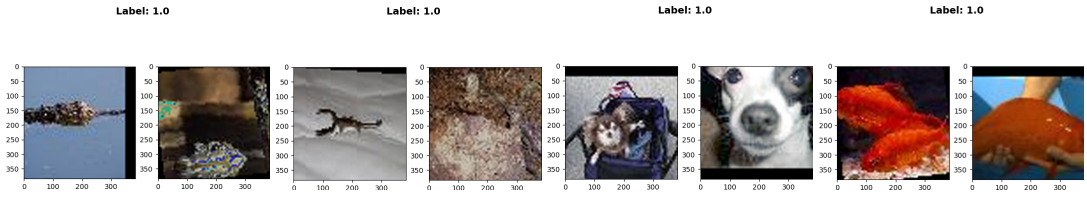

Figure 2. Sample of the augmented dataset

Model Architecture for Model 1

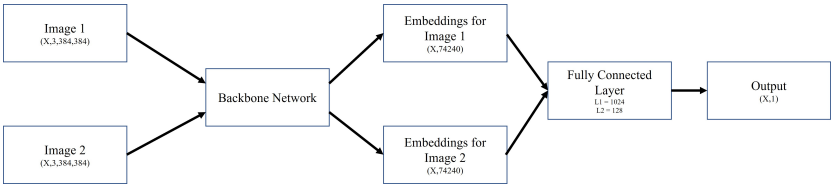

Figure 3. Basic Outline of Model 1

This model was a naive approach in tackling the Image Similarity problem. It can be intuitive to think that since a pre-trained model will provide embeddings for an image, the embeddings could be forwarded to a Multi-Layer Perceptron or a fully connected network to able to map the function that would be required to compute the similarity between the embeddings.

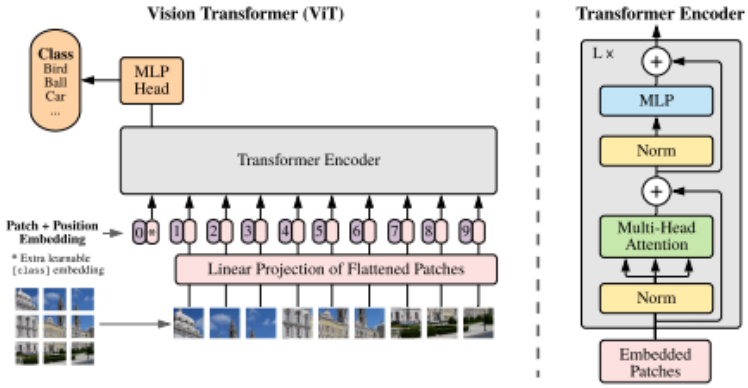

Figure 4. Backbone architecture used for Model 1 (Vision Transformer Model). Image Sourced from Kolesnikov et al. (2021)

The pretrained model used for this architecture was based on the Vision Transformer model proposed by Kolesnikov et al. (2021). The architecture of the model is shown in figure 4. In this model, the image is split into fixed-size patches that are linearly embedded and then proceeded to have position embeddings added to them. The resulting vectors are then fed to a transformer encoder who's architecture is also provide in the given figure. This encoder makes use of a multi-head attention block that is fowarded to a Multi-Layer Perceptron. The outputs of these intermediate layers also have the original input added on top of it that is similar to the functionality of a residual block.

In this model, the embeddings generated from the model were concatenated together to given a resultant (X,148480) dimension embedding. Here X denotes the batch size. This embedding was forwaded to a Dense layer with 1024 nodes with an activation function of ReLU, which was then forwarded to another Dense layer with 128 nodes and a ReLU activation function. The final layer was then forwarded to a single output node with an activation function of sigmoid. This helped in converting the resultant output to be in the range of [0,1]. Here 0 signified that the images were dissimilar and 1 signified that the images were similar.

The loss function used for training the architecture was the binary cross entropy loss as the entire task was reduced down to a binary classification using two input images. The loss function is given in the equation 1.

$$L_{BCE} = -\frac{1}{n}\sum_{i=1}^{n}(Y_i.log\hat{Y}_i + (1-Y_i).log(1-\hat{Y}_i)) \tag{1}$$

In Equation 1, Y_i denotes the ground truth and $\hat{Y}_i$ denotes the predicted value for the given input X_i. This model was trained for 10 epochs. The summary of other components used while training Model 1 is given in table 2

Components	Value
Epochs	10
Learning Rate	0.01
Output Function	Sigmoid
Loss	Binary Cross Entropy
Optimizer	Adam

Table 2. Components of Model 1

Remarks for Model 1

This approach wasn't able to train the model to generalize to unseen images and it struggled to perform well. It achieved a validation accuracy of 50% which can lead to the inference that the model isn't performing any better than randomly guessing the output. Moreover, due to the high volume of image pairs used the training time took was high as well. The training finished taking approximately 12 hours. This immediately lead the research into reducing the data size used for training the model and also retrain the model. The difficulty in finding the similarity between two images may have been due to improper representation of the images in the embeddings. Hence, the idea of Model 2 was implemented.

Model 2

Data Augmentation for Model 2

The data augmentation techniques used for model 1 have been carried forward for this model as well. The only difference being is the reduction in the size of the dataset. Instead of 80,000 training image pairs and 20,000 validation image pairs, only 8,000 training image pairs and 2,000 validation image pairs were taken into consideration.

Except for these changes the same augmentation techniques that were resizing the images and performing the RandAugment operation on it.

Architecture for Model 2

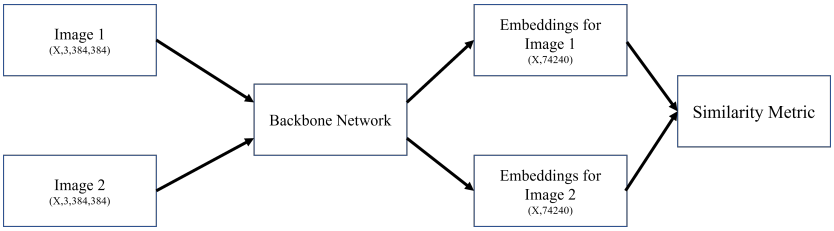

Figure 5. Outline of the Network for Model 2

The outline of the network for model 2 is shown in Figure 5. Instead of training a fully connected layer, the entire network was trained so that the embeddings could be better represented. This intuition was implemented through the research implemented by Dey et al. (2017). In his article about Signature Forgery detection using a Siamese Network.

In a Siamese network, the embeddings are trained by using a loss function that encapsulates the similarity metric between two embeddings. This similarity metric can be the euclidean distance or cosine similarity. A pair of inputs are forwarded to a CNN network that yields these embeddings. To simplify the task of training the CNN, a pre-trained network is used as a backbone.

For this model implementation the same Vision Transformer described in Model 2. The loss function to train the model is given by equation 2.

$$L_{siamese} = (Y)(-log(T)) + (1-Y)(-log(1-T)) \tag{2}$$

The variable T in equation 2 refers to the Similarity metric that is used to measure the similarity between the two embeddings. For this model, the euclidean distance was taken into consideration. Similar images were meant to have a euclidean distance of 0 and dissimilar images were meant to have it as 1.

For predicting the similarity, once the embeddings have been received for the images, euclidean distance can be applied on the vectors. The more dissimilar the embeddings were the higher the euclidean distance would be. A threshold can be applied on this distance that would define dissimilar images if the distance was above that threshold. This model was trained for 10 epochs. The other components used while training Model 2 is given in table 2.

Components	Value
Epochs	10
Learning Rate	0.01
Output Function	Euclidean Distance
Loss	Contrastive Loss
Optimizer	Adam

Table 3. Components of Model 2

Remarks for Model 2

The model seemed to struggle in generalizing the embeddings for unseen images as well. Having too many image pairs and not enough epochs to train doesn't help in being able to train the embeddings. Even though the training time was reduced to 4 hours, the model was only able to get a validation accuracy of 56% on unseen images. To improve upon this model it was important represent the embeddings in the form of a cluster. This way embeddings that belong to the same class could be represented closer to each other in an n-dimensional space. Also, the number of parameters to train in a Vision Transformer are around 768 million. This was too high and contributed to a large amount of training time. Reducing the number of trainable parameters was also an important factor for feasiblity of training the model.

Model 3

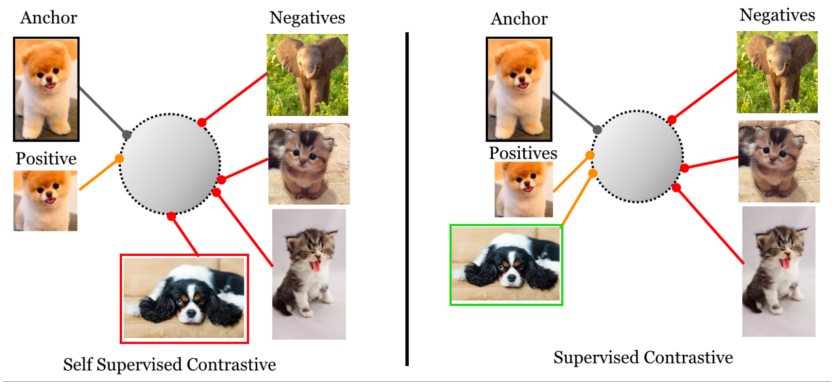

Figure 6. Difference between contrastive learning and supervised contrastive learning. Image sourced from Khosla et al. (2020)

The main reason why Model 2 failed is that the model tends to cluster together absolutely identical images together and represent them together in the cluster. Hence, it is important for the model to cluster the images belonging to the same class to classify them as similar images. This was the intuition behind supervised contrastive learning where images belonging to same classes would be clustered together as shown in figure 6.

Data Augmentation for Model 3

For this particular model only the first 50 models were taken into consideration while training. This was done to reduce the computation time taken while training the model. The validation pairs were also kept separate. Except for reducing the number of classes everything else was kept exactly the same. The same augmentations were applied to the images as described in the previous models.

Architecture for Model 3

In a supervised contrastive learning scenario, the similar images are trained so that embeddings would cluster around their respective anchor points. It still relies on a CNN/transformer network to be trained for the resulting embeddings. For reducing the number of trainable parameters, a different backbone network was taken into consideration. The backbone architecture taken into consideration is the EfficientNet. The number of parameters was reduced by a factor of 40, resulting in 19 million trainable parameters. The architecture of EfficientNet is described in Figure 7.

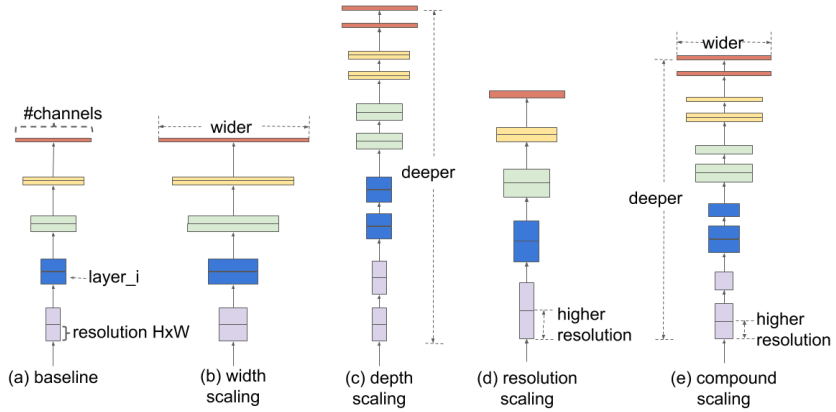

Figure 7. Architecure of EfficientNet. Image sourced from Tan and Le (2019)

EfficientNet is a convolutional neural network design and scaling technique that uses a compound coefficient to consistently scale all depth, breadth, and resolution parameters Tan and Le (2019). The EfficientNet scaling approach evenly scales network breadth, depth, and resolution using a set of preset scaling coefficients, in contrast to standard practise, which scales these variables arbitrarily. The rationale behind the compound scaling approach is that larger input images require more layers in order to expand the network's receptive area and more channels in order to catch more fine-grained patterns on the larger picture. In addition to squeeze-and-excitation blocks, the foundational EfficientNet-B0 network is built upon the MobileNetV2 inverted bottleneck residual blocks.

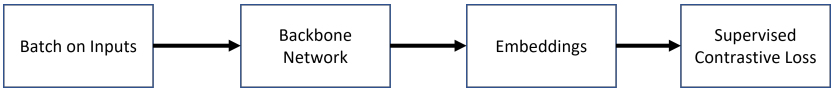

Figure 8. Stage 1: Training the model

Initially a batch of images is forwarded down this network one by one. This results in a batch of embeddings for each image. Within this batch of images the loss between similar and dissimilar images is calculated through the following equation 3.

$$L_i^{sup} = -\frac{1}{2N_{y_i} - 1} \sum_{j=1}^{2N} \mathbb{1}_{i \neq j} . \mathbb{1}_{y_i = y_j} . log \frac{exp(z_i . z_j / \tau)}{\sum_{k=1}^{2N} \mathbb{1}_{i \neq k} . exp(z_i . z_k / \tau)} \tag{3}$$

In Equation 3, the variable z represents the embedding vector of the i^{th} input in the batch. τ is a parameter that is defined as the temperature. N_{y_i} refers to the number of embeddings having the same label as y_i in the given batch of input. If the vector embeddings are similar then the dot product of the embeddings would be high. This would lead to the exponent function being high as well. Since, the concept while training a model is to minimize the loss function, hence the negative of this function is taken into consideration. If the embeddings are similar, the loss would be as low in the negative quadrant.

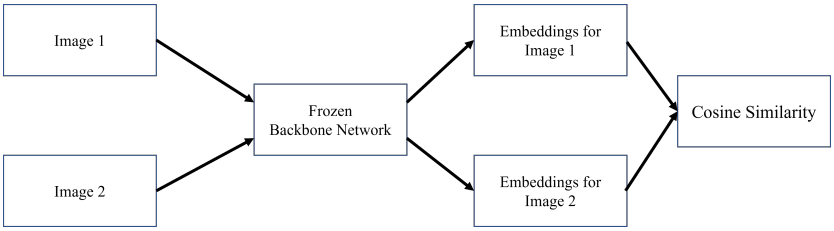

Figure 9. Stage 2: Performing inference on the model

After training the embeddings through supervised contrastive learning, the embeddings will be a representation of the input image in an n-dimensional space. To measure the similarity between two embeddings, cosine similarity was taken into consideration. This helps in analyzing the angle between these two vectors. If the images are similar, the angle would be low and the cosine of this angle would be closer to 1. The more dissimilar the images will be the more they will tend to go towards -1.

This model was trained for 30 epochs. The other components used in training Model 3 is given in table 4

Components	Value
Epochs	30
Learning Rate	0.01
Output Function	Cosine Similarity
Loss	Supervised Contrastive Loss
Optimizer	Adam

Table 4. Components of Model 3

RESULTS

Model	Type of Model	Val Accuracy
Model 1	CNN-FCN	50%
Model 2	Siamese	56%
Model 3	**Supervised Contrastive Learning**	**87%**

Table 5. Validation accuracies on unseen images of seen categories across different models

Out of the three types of models implemented the best performing model was Model 3 that used the technique of supervised contrastive learning. These validations were performed on images that were part of the original train-val split, hence there are no images that might have been reused while training the models. Hence these metrics are defined on unseen images of categories that have been seen by the model. For Model 1 the threshold for the output was chosen to be at the optimum value of 0.5 and for Model 2 the threshold was 0.2. For Model 3 the optimum threshold was 0.21, which translates to an angle of 77.9 degrees. The loss function of Model 3 is given in figure 10.

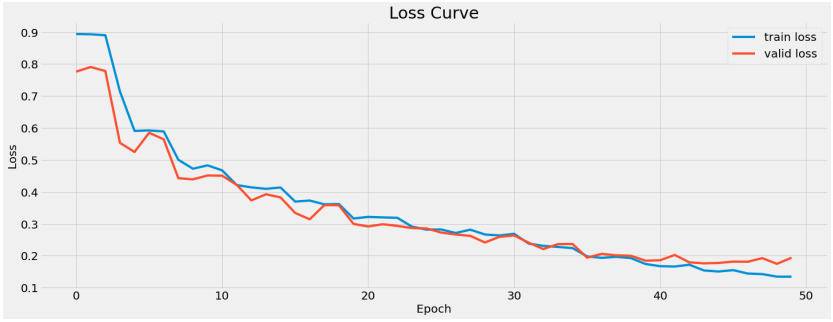

Figure 10. Training and Validation Loss for Model 3

To visually validate the metrics, a couple of inputs from the validation dataset was analyzed. The correct predictions that the model had are shown in Figure 11. Also the wrong predictions are shown in Figure 12. The confusion matrix for Model 3 is given in figure 14.

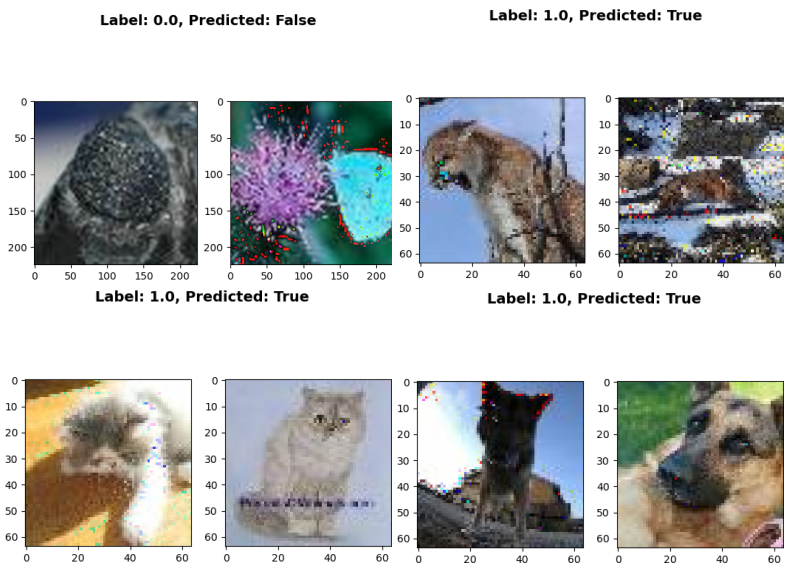

Figure 11. Correct Predictions made by Model 3

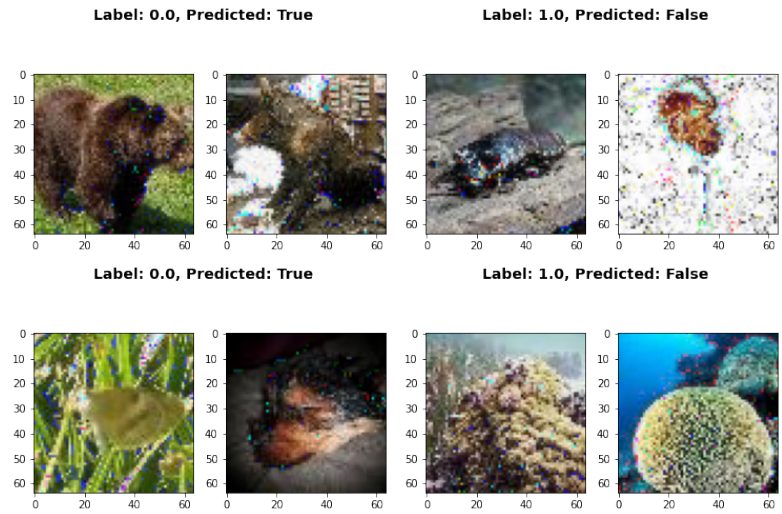

Figure 12. Wrong Predictions made by Model 3

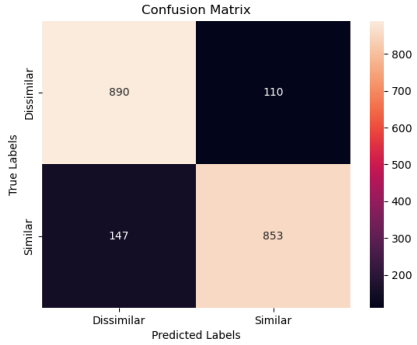

Figure 13. Confusion Matrix for Model 3

From the confusion matrix numerous other important metrics can be calculated. These metrics are given in table 6.

Class	Recall	Precision	F-score
Dissimilar	85.82%	89%	0.873
Similar	88.57%	85.3%	0.869

Table 6. Class-wise performance metric of Model 3

Validation on Unseen Categories

From the previous experiments, it was concluded that Model 3 performed the best out of the three models, hence this model was selected to test it's applicability further.

To validate the model on unseen categories the model was tested on the signature verification dataset. The model achieved an accuracy of 65% without any training on this dataset. A batch of 128 image pairs were taken as test input and the performance metrics were measured on it. A high threshold of 0.7 was taken into consideration here as the intuition was that signatures are gonna be closely clustered in a embedding space and to distinguish between the forged and genuine signs, a really narrow angle had to be chosen which roughly translates to 45.5 degrees. They are summarized in the table 7.

Class	Recall	Precision	F-score
Dissimilar	55.22%	71.53%	0.623
Similar	75.40%	60.52%	0.671

Table 7. Performance metrics for Signature verification dataset

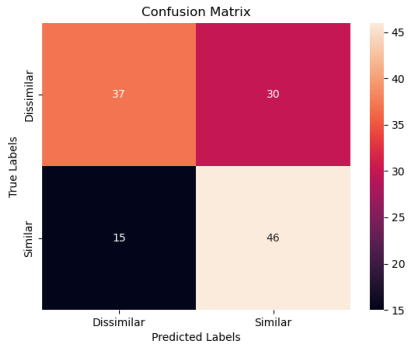

Figure 14. Confusion Matrix for the signature verification dataset

Some of the correct predictions done by the model on the signature verification dataset are shown in Figure 15. Similarly a couple of wrong predictions done by the model is shown in Figure 16.

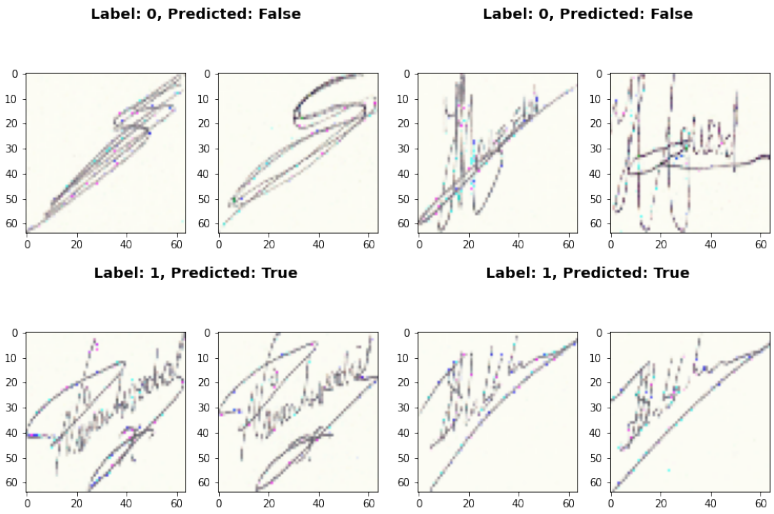

Figure 15. Correct Predictions made by Model 3 on Signature Verification Dataset

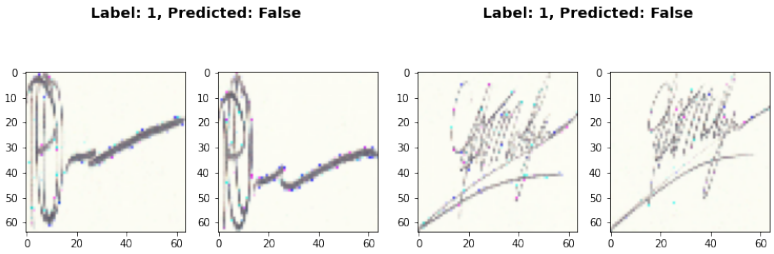

Figure 16. Wrong Predictions made by Model 3 on Signature Verification Dataset

DISCUSSION/CONCLUSION

From the experiments conducted using various models it was evident that this problem wasn't a simple classification task. This task needed to be extended to learn about the embeddings that a model produces. This was evident from Model 1 where the model failed to improve upon the training methodologies and presented a constant 50% validation accuracy.

Even though representation learning was emphasized in Model 2, it was hard to describe the embeddings for multiple classes. It was evident that the model was having a tough time representing a unified embedding for a given class. This was shown by a slight improvement in the overall validation accuracy by 6% but wasn't enough to be classified as a well performing model. If the embeddings of a class had a particular point to cluster around, it would've been much easier for the model to provide similar embeddings for similar classes.

Hence the final model was meant to tackle this issue by keeping anchor points for embeddings to cluster around. It would've been interested to visualize these clusters.

One of the main reasons why the results on the signature dataset can be considered significant is due to the fact tat the difference between these images tend to be really fine and small. Representing these difference without training a model specifically for this is extremely tough. Moreover as discussed previously the Tiny ImageNet dataset contains images that are way too small, hence the only meaningful features it can extract are mostly coarse-grained. Having a model trained on a coarse-grained dataset performing reasonably well on a fine-grained dataset shows the merit of this methodology.

Improvements on Model 3

The Model 3 can be improved further in a variety of ways. There were numerous cut downs that were done to reduce computation time and feasibility of training. Some of them can be described in the following points below:

- Improved Model: For ease of conducting these experiments a smaller and relatively faster model named EfficientNet was chosen. Based on the ImageNet leaderboard there are numerous other state-of-the-art models that perform better than EfficientNet as shown in figure 17. Models like ViT or CoCa could be considered as a backbone architecture as well.

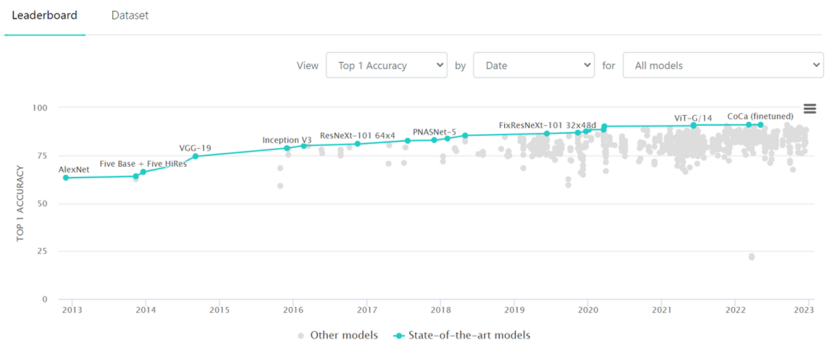

Figure 17. ImageNet Leaderboard sourced from [https://paperswithcode.com/sota/image-classification-on-imagenet]

- Increase Datasize: For faster model runtime, the datasize was clipped to 50 classes. An improvement while training the model would be to include more classes to be able to define more anchor points and clusters in an n-dimensional space.
- Better Quality Images: To improve upon fine-grain image classification attribute of the model, it is imperative to have higher quality images through which more fine-grain features can be extracted.

NOTE

The code to run Model 3 is properly documented in the Notebook named 'Report.ipynb'. To reproduce the results please make sure that the data and generated labels are present in the 'data/' folder. Also place the notebook in the 'src/models/' folder as it contains all the supporting scripts that are needed to run the results. A copy of the weights of the model will be present in the same folder. Model 1 and Model 2 will be present within the packages inside the 'src/models/' folder. Model.py consists of Model 1 and Model_v2.py consists the Model 2. Their respective dataloaders and loss functions are described in their respective .py files.

REFERENCES

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. (2020). Randaugment: Practical automated data augmentation with a reduced search space. 33:18613–18624.

Dey, S., Dutta, A., Toledo, J., Ghosh, S., Lladós, J., and Pal, U. (2017). Signet: Convolutional siamese network for writer independent offline signature verification.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. (2020). Supervised contrastive learning. 33:18661–18673.

Kolesnikov, A., Dosovitskiy, A., Weissenborn, D., Heigold, G., Uszkoreit, J., Beyer, L., Minderer, M., Dehghani, M., Houlsby, N., Gelly, S., Unterthiner, T., and Zhai, X. (2021). An image is worth 16x16 words: Transformers for image recognition at scale.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. 25.

Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks.