KINSHIP RECOGNITION

Anusha Kumar, Eeshapriya Gutta, Tanya Shourya

ABSTRACT

The kinship verification system aims to build a machine learning algorithm that determines if two individuals are related. Facial appearance is a crucial part to determine if two people are related. There are various challenges when trying to analyse the images for prediction. Factors such as facial expression, angle of an image, pose, lighting, quality of the image are some of the many challenges faced. Adding to these, a person with hair dyed or wearing a hat would make the prediction and analysis even more challenging. There have been multiple researches done in this field with a wide range of open datasets available.The application to the algorithm include finding missing children, identifying family members of criminals to trace them, social media analysis to name a few.

1 INTRODUCTION

Based on anthropology, Kinship refers to sharing certain features or having common origins between two people, such as the relationship between a child and parent/grandparent or siblings. Kinship recognition is one of the most challenging topics having a wide range of applications in multiple domains.

The following relationships are available in the dataset being used: Father-Daughter (fd), Mother-Daughter(md), Father-Son(fs), Mother-Son(ms), Spouse-Spouse (ss). A more sophisticated classification would include Grandparent-grandchild relationship as well.

In this project we suggest the following parts to solve this problem of verifying kinship: Reading dataset, identifying related and unrelated pairs of images, data cleaning, using VGGFace2 pretrained model to input into the Siamese neural network. Since a Convolutional Neural Network only takes one input, we use Siamese neural network to compare the two images and find similarity.

2 DATASET

The Families in the Wild dataset from SMILE Lab, Department of Computer and Electrical Engineering, Northeastern University contains a dataset of 11,932 natural family pictures of 10k families with an average of 10 photos per family. This data is diverse and not constrained to a domain. It has images across diverse family trees with different relations. It also contains images of the same person during different age groups. Images also include negative samples to improve the efficiency of the algorithm.

The Training set contains images of 500 families from 50 different families. Images include pictures of unrelated people, which makes for a good negative sample. The dataset is divided to train and validation in 80:20 ratio to evaluate different performance metrics

The 'train-pairs' CSV file contains classification of family members based on whether a relationship exists or not. This helps define the target variable for model input.

3 DATA PREPARATION

The train-pairs.csv file contains the pairs of members in the first two columns and their relationship in the subsequent columns. For a baseline model, classification needs to be made whether the two input images are related or not. For this, the existing output columns are dropped from the dataset.

An additional column containing value 1 is added for each pair for which a relationship exists in the train-pairs.csv indicating some relation between the two.

The file paths are re-iterated for each pair of images and the output column populates a 0 value if the pair value is not present in the train-faces.csv file. All combinations for the pairs in the files are created and assigned values in the third column to show if they are related or not. The final data file with 17299 rows is used as an input for prediction.

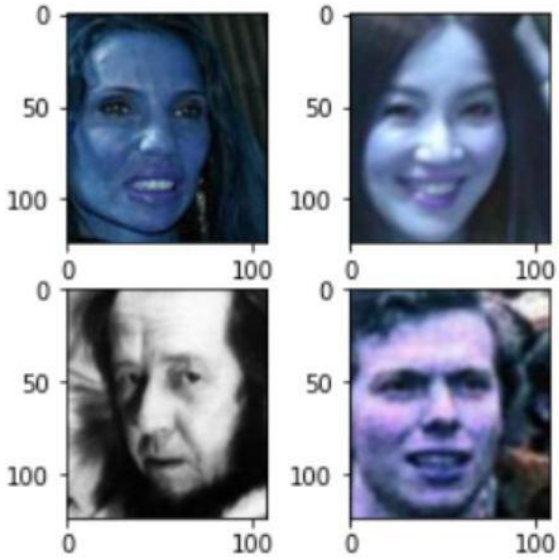

Figure 1: Plotting the data read for the first 2 image pairs

The data is then divided into training and validation sets. The folders with F09 series are taken as the validation set. It is a set of 61 folders with 3114 images. The training set contains 14185 images after the split. The model is then trained and the 'train' set and accuracy calculated against the 'val' set.

4 MODELLING APPROACH

We propose a Siamese convolution neural network, experimenting using different parameters on the VGGFace pre-trained model with a binary cross-entropy loss function used at the end.

4.1 VGGFACE

The VGGFace2 pre-trained model is being used in this project. This model was developed by members of Visual Geometry Group which is trained on 3.1 Million images of 9k subjects. The RestNet-50 and SqueezeNet-ResNet-50 model is trained to identify and verify faces from the given image. It has also been tested on benchmark datasets for face recognition proving it's efficiency. The classifier uses Softmax activation function to classify output as people's faces. The model is trained further to increase the Euclidean distance between pairs of images of different people and decrease the distance between pairs of images of similar people. Triplet loss function is used to achieve this.

4.1.1 TRIPLET LOSS

Triplet Loss function accepts three inputs which works well with Siamese network. A triplet loss function is a loss function which takes three inputs, namely anchor, true input and false input. The anchor is compared with true and false value such that the Euclidean distance between anchor and

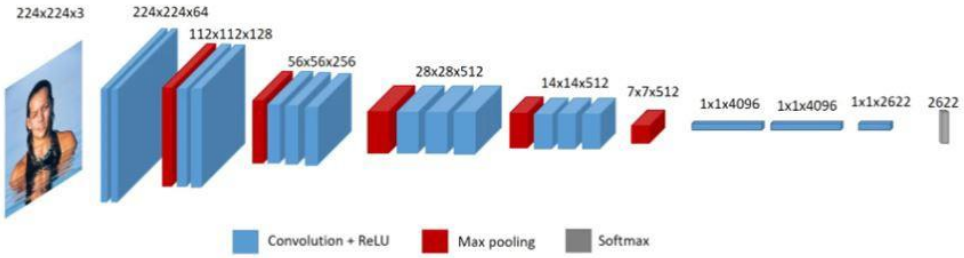

Figure 2: Layers in VGG Face Pre-trained Model

true input is decreased and between anchor and false value is decreased. This helps in comparing the 2 images and find similarity metric between them.

The loss function is given by:

$$Loss = max(dist(a, p) dist(a, n) + margin, 0)$$

where dist(a,p) is the distance between anchor image and positive image, also dist(a,n) is the distance between anchor and negative image.

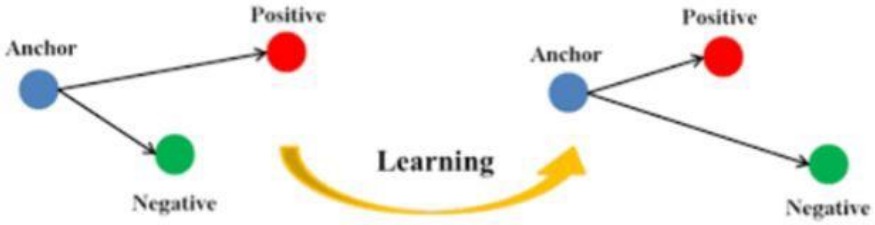

Figure 3: Triplet Loss

The preprocessinput() function is used in the keras-vggface library to scale the images using mean values in VGGFace 2. 'Version = 2' is specified to scale and centre the pixel values in the same way as data which was used to fit the VGGFace model.

VGGFace2 provides 2 models, resnet50 and senet50. Training the model using senet50 resulted in a 46% accuracy. When the resnet50 model is used, the accuracy comes around 60%. We hence use resnet50 for training. Resnet50 is a convolutional neural network model with 50 layers. It takes an input an image with dimension 244x244.

4.2 IMPLEMENTING SIAMESE NETWORK

Siamese network consists of a unique architecture that determines similarity or dissimilarity between inputs. Its key feature is that, unlike conventional CNN techniques that take only a single input image, a siamese network takes two images as model input to check for similarity. These input pairs are passed through similar sub-networks having the same parameters, which produce an encoded vector feature for each image. These are further passed through the differencing layer to classify if the inputs are similar or not based on the score calculated using Binary cross entropy.

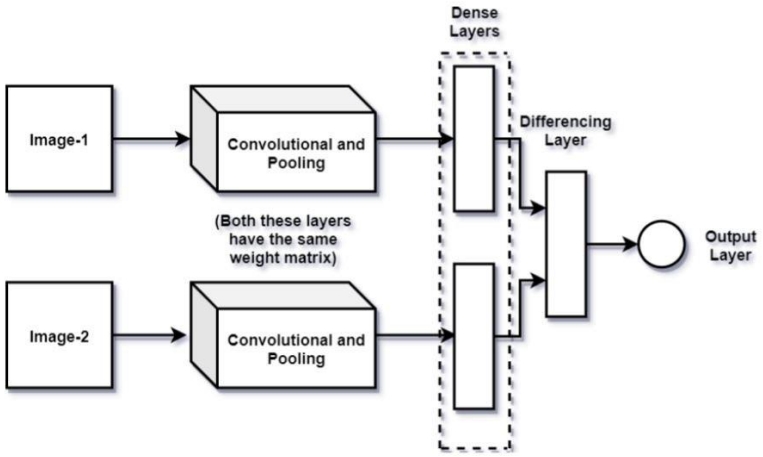

Figure 4: Siamese network

The images are converted into a numpy array format. The model input is a pair of two images(person 1 and person 2). The task is to identify whether these two persons are related or not. Prior to model input, each pair is split into a vectorized left and right input. These image representations are fed to a common architecture i.e VGGFACE pre-trained model. Their output consists of features representing the two images. These two vectors are then sent through Global Max Pool and Global Avg Pool. Two variables are calculated 1) The square of the difference of the two vectors. 2) Product of the two vectors. These two variables are concatenated and fed to a dense layer followed by a Dropout layer. The final layer is a fully-connected layer with a single node using the sigmoid activation function to output the Similarity score.

To avoid intermittent system crashes due to large volume and complexity of data, Keras callback functions are used while model training. These callbacks are designed to be able to monitor the performance of the metrics at given intervals in the training run and perform certain actions depending on the performance in metric values. There are three callbacks used for this model, firstly is 'Early Stopping' where the loss value is used as a performance metric to terminate training. The training will stop if the loss does not decrease further after 20 epochs. Second is 'model checkpoint' that stores the Keras model or weight values during training, so the model or weights can be loaded later to continue the training run. Thus for each epoch if the accuracy increases, the weights get automatically saved to the defined file. Lastly, 'ReduceLROnPlateau' callback function is used to reduce the learning rate if the loss value has stopped decreasing. This monitors the loss value and if no improvement is seen in every 6 epochs then the learning rate is reduced by a factor of 0.1. These callbacks are passed as model parameters while fitting the data.

5 RESULTS

Two evaluation metrics are employed in order to display results, namely Model accuracy and AUC scores. On evaluating against the 'resnet50' and 'senet50' , we observe resnet50 to perform better against the performance metrics on validation set.

Parameter	Accuracy	Loss	AUC
resnet50	60.08	70.22	64.27
senet50	55.42	72	62.20

Table 1: Performance Metrics

Training the model on Senet50 gives the following result:

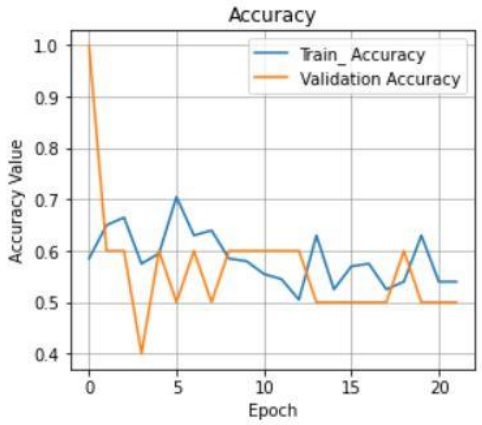

Figure 5: Accuracy graph using senet50

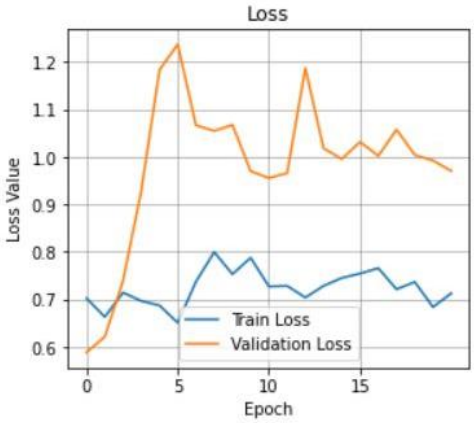

Figure 6: Loss graph using senet50

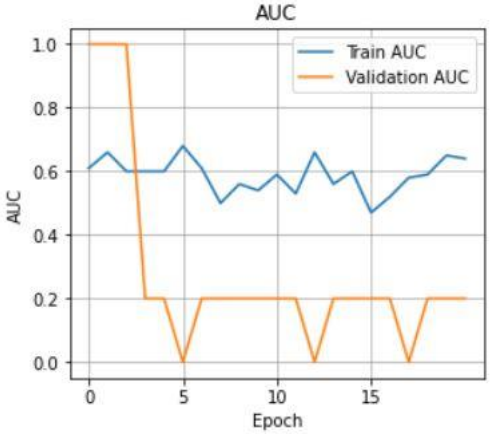

Figure 7: AUC graph using senet5

The accuracy in this case is 55.42%

Training the model on Resnet50 model gives the below result:

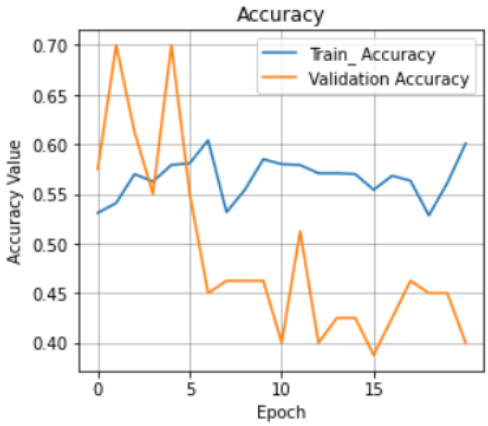

Figure 8: Accuracy curve for resnet50 model

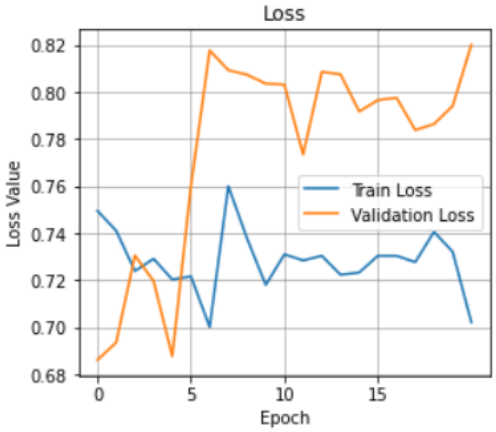

Figure 9: Loss curve using resnet50 model

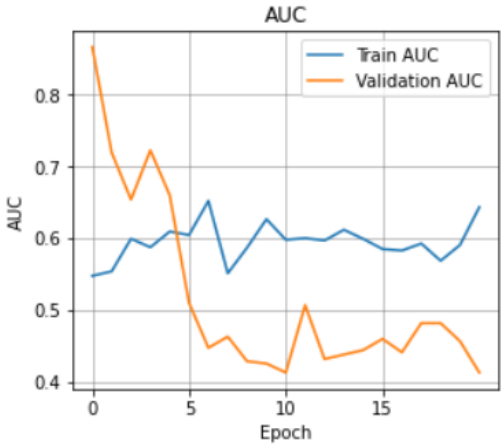

Figure 10: AUC curve using resnet50 model

The accuracy in this case is 60.08%.

6 CONCLUSION

We have presented a baseline model for performing kinship recognition using Siamese neural networks on VGGFace pre-trained model. We experimented and compared performance by fitting on different model parameters and evaluated against various loss functions to obtain an ideal baseline model.

As a further development, this project can be extended to build search and retrieval applications by identifying the type of relationship between the images classified as related.

7 REFERENCES

Figure 2: Layers in VGG Face Pre-trained Model

https://sefiks.com/2018/08/06/deep-face-recognition-with-keras/

Figure 3: Triplet Loss

https://medium.com/@prabhnoor0212/siamese-network-keras-31a3a8f37d04

Figure 4: Siamese network

https://towardsdatascience.com/siamese-networks-line-by-line-explanation-for-beginners-55b8be1d2fc6