




# GANKIN: generating Kin faces using disentangled GAN

Fady S. Ghatas<sup>1</sup>  · ElSayed E. Hemayed<sup>1</sup>Received: 25 September 2019 / Accepted: 3 January 2020 / Published online: 8 January 2020  
© Springer Nature Switzerland AG 2020

## Abstract

Kin image generation from parents' images is a high-level prediction and generation problem. This study presents a new method to predict and generate a kin face using parents' faces, i.e. Tri-subject prediction or two-to-one prediction. Training one end-to-end conditional GAN from scratch can run into mode-collapse and may not converge, we overcome this by using a modular pipeline of unconditional smaller models. We start by extracting father and mother features using a pre-trained model FaceNet, then we use extracted features to predict kin features. After that, we use a linear regression model to convert the predicted features to a latent vector that can be passed to an unconditioned generative adversarial network (GAN) which generates the required kin face. Also, we present a disentangling method to help choosing the age and gender of the generated kin. The modular nature of the proposed model allows validating and improving each part of the pipeline separately. The model achieves promising results compared to the state-of-the-art.

**Keywords** Kinship synthesis · Generative adversarial network · Features disentangling · Modular GAN

## 1 Introduction

Kin image generation from parents images is a sophisticated problem. The features of the child are deeply connected to the parents. A human mind can imagine how a son or a daughter may look like, but it was impossible for a computer logic to predict it. This problem is hard because there is no single prediction for each father and mother combination which makes this problem nondeterministic.

Advances in kinship generation field can open doors to thorough understating of the deep hidden relations between parents' features and kins' features. Understanding these relations can help in the future predicting offspring features, like genetic disorders, when combined with medical and DNA data. Kinship generation can also help producing better datasets and enhancing existing datasets which can help solving more problems like kinship verification.

In this study, we build a complex modular pipeline using pre-trained models to generate the kin image.

The modular nature of the pipeline changes the way we approach GAN problems and introduces a systematic way that can be easily generalized to more generative problems and it widens the door of possible solutions especially in conditional GAN problems by converting the problem to an unconditional GAN preceded by a feature prediction network and a features to noise network as thoroughly discussed in the study.

We start by extracting features from parent images using Facenet [23], then the features are combined through concatenation and are passed to a neural network to predict the kin features. The kin features are then passed to a linear regression model to convert the features to a latent vector which can be finally passed to the pre-trained PGGAN model [8] to generate the kin face.

We use mainly two datasets: (1) Family101 [5] which has 206 families, with 14,816 images (2) FIW [16] which has 1000 families with 11,932 images.

To verify the output, we calculate cosine similarity between generated kin face and parents using a logistic

✉ Fady S. Ghatas, fady@titrias.com | <sup>1</sup>Faculty of Engineering, Cairo University, Giza 12613, Egypt.



regression model. The average accuracy of the generation is ( $63\% \pm 7\%$ ), this is comparable to manual human accuracy ( $66.75\% \pm 4\%$ ).

This study is organized as follows: We start with an in-depth literature review on face generation using GAN, Kinship verification and kin face generation. After that we discuss the proposed model, then the experimental results. Finally, We conclude the study with final remarks and suggested future work.

## 2 Literature review

In this section, We start by reviewing datasets related to kinship verification and kin face generation, then we discuss kinship verification methods. We then list approaches used to generate faces in general. Finally we review recent researches directly related to kin face generation.

### 2.1 Datasets

#### 2.1.1 Family101

Family101 [5] was the first introduced dataset related to the area of family verification and classification in 2013. The dataset has 101 public families with 1 to 7 generations per family, these 101 families translate to 206 nuclear families containing more than 607 individuals with 14,816 images of resolution  $120 \times 150$  pixel.

#### 2.1.2 FIW: families in the wild

In 2016, the FIW dataset [18] was released which is larger and of better quality than any previous datasets related to family classification or kinship verification.

FIW has 11,932 family photos of 1000 families with a resolution of  $224 \times 224$  pixels.

FIW dataset generally has better postures and higher quality images than other datasets.

### 2.2 Kinship verification

Like any classification and verification problem, the two main methods to approach the kinship verification problem are (1) the handcrafted feature-based approach and (2) the deep learning approach.

Starting from 2017, Northeastern University started to hold the annual challenge RFIW (Recognizing Families In the Wild) [19] to target the problem of kinship verification, ever since a lot of new papers were released that target the problem using deep learning. Previous work before this point was mainly using hand-crafted facial features.

Some feature-based approaches include using LBP features [2] to encode the faces and then use a KNN or SVM to verify kinship. Other researches combined the LBP with HOG features [4]. SIFT was used in [13, 17] to extract features and SVM is used as the classifier.

The other approach is deep learning, one approach (The one used in this research) is using a deep feature extraction network like DeepID [15], ImageNet [21], ResNet [7], FaceNet [23], VGG [24], SphereFace [11] followed by a distance calculation using Cosine similarity or Euclidean distance to calculate the score.

We discuss mainly two papers, SelfKin [1] by Dahan et al which won RFIW in 2018 and Tri-Subject Kinship [17] by Qin et al which is one of the best tri-subject verification papers.

#### 2.2.1 One-to-one verification

SelfKin [1] model won RFIW 2018 challenge and obtained the state-of-the-art accuracy of 70%, SelfKin scores each one-to-one relation independently to find the correct classification (Father–Son, Father–Daughter, Mother–Son, Mother–Daughter).

SelfKin uses VGG [24] as the feature extraction network followed by a Mask Layer then a Local-Global classifier, the result is then passed to a global averaging layer. The result of global averaging layer is passed to a softmax layer to get the label.

#### 2.2.2 Two-to-one classification

Two-to-one classification is the problem of verifying the kinship of a face to both parents together (Father–Mother–Son or Mother–Father–Daughter).

The Tri-Subject Kinship network [17] works as follows: an image of a potential father, an image of a potential mother and an image of the proposed kin are supplied to a SIFT feature extractor. Each image is divided into multiple overlapping patches before passing it to the SIFT feature extractor, each patch produce 128-dimension feature vector. Then the feature selection is used to select the best features representing the problem. Finally the father features are passed along with kin features to Symmetric Bilinear Model (SBM) model. The same is repeated on mother and kin features where it's passed to another SBM, both results are then passed to a classifier, e.g. SVM which finally scores the inputs and give the final score.

RFIW included tri-subject kinship as separate branch in 2019 competition, so we predict faster progress in this track in the future.

### 2.3 Face generation before GAN era

Even before the idea of generative adversarial networks was innovated, multiple artificial intelligence techniques were used to solve the problem of face generation.

Initial trials were targeting the face generation problem a bit differently, generative models weren't available yet, so the solution was to use face retrieval and morphing methods using feature extraction to match a sketch or defined features to a set of faces [22], this method gives a discrete number of outputs equals to the number of labels, so no generation is involved. The output is one of the images the model trained on with some morphing.

Following the face retrieval, Variational Autoencoders [10] were innovated and VAEs offered an interesting solution to generative problems, VAE consists of an encoder and a decoder followed by an optimizer. Initially the VAE learns how to encode and then decode training data, the loss function minimizes the difference between the real image and the decoded image, the last layer of the encoder outputs a latent vector that can be used as a representation of the face. After training the VAE, the decoder is used separately to generate images from different latent vectors. The values of the latent vectors give different outputs similar to the training data but not identical. The continuity of the latent variables is the special nature of VAE that helps generating data never seen in the training set.

### 2.4 GAN face generation

Using GAN to generate faces has been a trending research topic since the generative networks were introduced [6].

#### 2.4.1 Progressive growth of GAN

Progressive Growth of GAN (PGGAN) [8] marked a new era of face generation, with resolutions up to  $1024 \times 1024$ . The results are very realistic that the generated images sometimes can't be recognized as fake by humans.

Although the architecture is similar to other GANs, what makes this method unique is how the network is progressively trained.

The network is trained in steps, starting by training a small generator and discriminator of size  $4 \times 4$ . When the small generator can produce good results, we advance to a higher resolution ( $8 \times 8$ ) by adding new layers to the generator and the discriminator. This allowed the network to reach high resolutions consistently and progressively.

#### 2.4.2 Attribute-guided face generation

The other trend of face generation is attributes-based generation [14]. In Conditional Cycle GAN, the generator takes

two inputs: the original image and an additional attributes vector that represents the main differences needed on the face. Besides generating new faces, Conditional CycleGAN can reconstruct faces using features vector.

### 2.5 Kin face generation

In this section, we discuss the main trends in kin face generation field.

#### 2.5.1 Face synthesis

Before the era of GAN, the main method of kinship generation was mainly through synthesis and picking the best synthesized face or directly picking one of pre-added kin images.

In [3], parent images are provided to a face detection algorithm, then a facial landmark identification algorithm detects main face points like eye boundaries, lips and nose tip. The face is finally provided to a feature extraction algorithm which extracts and combines parents features including: shape and color of the eye, mouth shape, skin color, lips edges. After extracting the needed features, one out of the nine-baby dataset is selected as a base of morphing. The base baby face is morphed based on parents features to generate the needed kin image based on Eq. 1 where  $I_{Baby}$  is the base baby image and  $C$  is a smoothing factor for the baby features ( $F$ ).  $\beta$  is base image to morphing ratio and  $\Sigma$  is how sharp the baby features should be.

$$Feature = \beta I_{Baby} + (1 - \beta)(\sigma F + (1 - \sigma)C') \quad (1)$$

#### 2.5.2 KinshipGAN

Using GAN in Kin face generation is still a new area, a recent research [16] explored the possibility of creating generating kin faces based on one of these one-to-one relation: father–daughter father–son, mother–daughter and mother–son. The generator follows an auto encoder pattern where parent face main features are extracted (encoder) and then, the extracted features alongside a gender parameter ( $c$ ) are used to generate kin face (Decoder).

To enhance the encoder network, and mainly due to the lack of a large scale dataset, a pre-trained VGGFace-16 network is used as feature extractor.

## 3 Proposed model

GANs are hard to train. Moreover, changing the inputs or outputs of the model requires a complete retrain and may require a new dataset as well, depending on the changes.

The complete model architecture is illustrated in Fig. 1. We start by extracting features from father and mother faces, then we concatenate both vectors to a single vector. We pass the concatenated features vector to a kin-features prediction network which predicts the new features. Then

the new features as well as the age and gender are passed to a feature-to-noise model to get the needed latent vector which is finally passed to the PGGAN [8] model to get the kin face. The complete model architecture is explained in detail in the following subsections.

### 3.1 Feature extraction

We start by extracting features from both parents. For this task, we use FaceNet [23]. FaceNet is a deep neural network that takes a  $224 \times 224$  pixels image and produces a compact 128-dimensional vector to represent the features of the face.

### 3.2 Kin features prediction

After extracting parents' features, we concatenate both vectors together in a single 256-dimension vector which is then passed to a custom network that predicts the kin features (128-dimension vector). The kin features prediction network consists of 4 fully-connected layers followed by Adam optimizer with 0.01 learning rate and mean square error loss function. The model is illustrated in Fig. 2. The model is trained on parents features and corresponding kin features using FIW dataset.

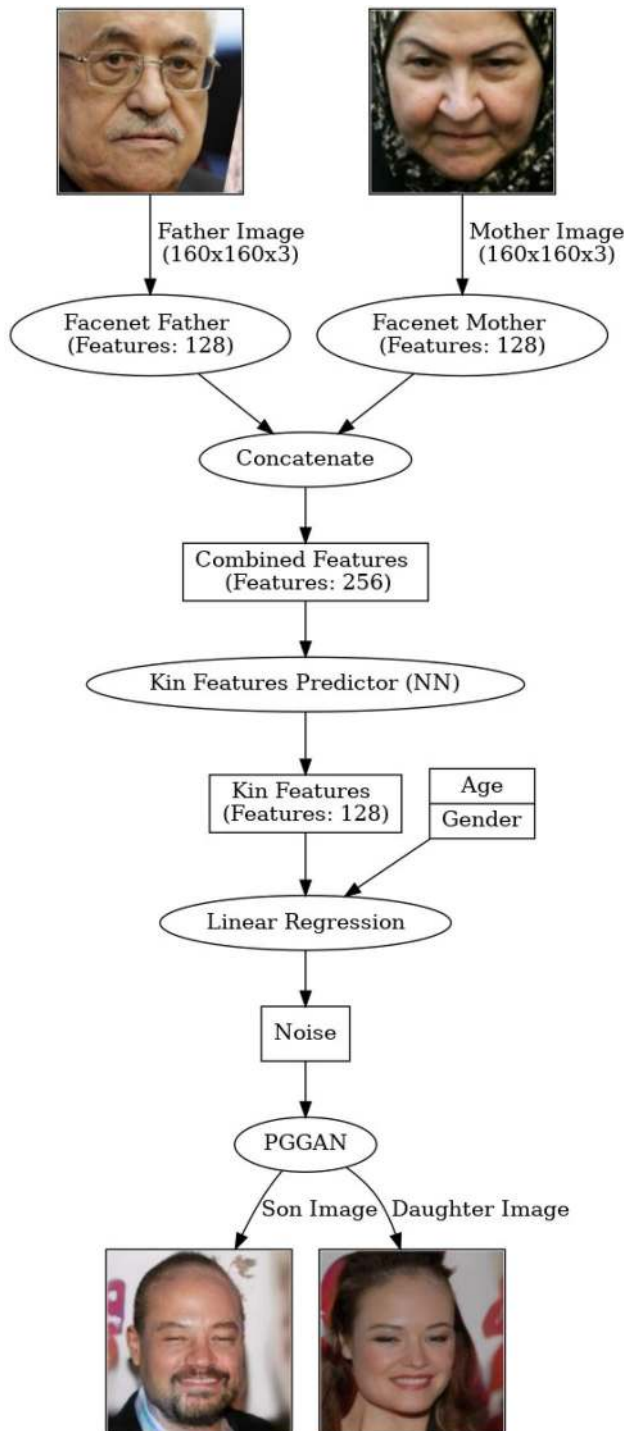


Fig. 1 GanKin network

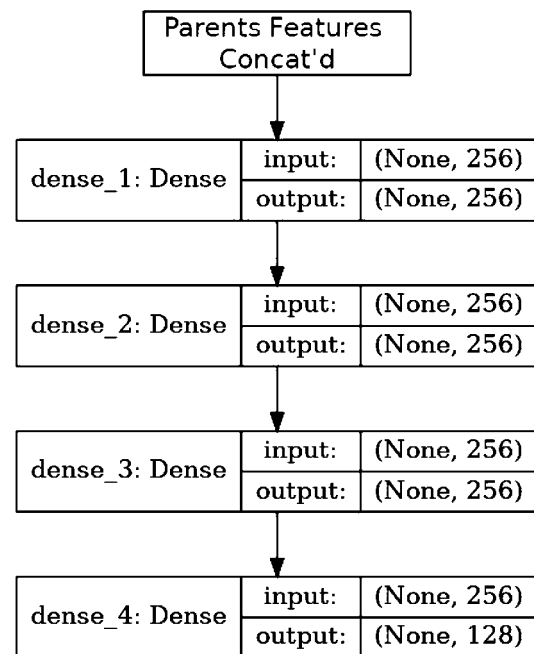


Fig. 2 Kin features predictor

### 3.3 Feature-to-noise model

GAN noise is reverse-engineered through regression, we obtained the needed noise to construct a face with similar features.

We use linear regression to convert the features to latent vector that can be passed to the PGGAN network. Besides the predicted kin features, we provide the needed age and gender of the kin. But we need to make sure the gender and age aren't affected by other features by making age and gender orthogonal on each other and on other features.

This process is called disentangling, disentangling is the process of making some features free or independent from being affected by other features. And that's the main reason for using a linear regression as the feature-to-noise model instead of a neural network, because we can't easily disentangle a neural network regression network. In our example, there is no need to disentangle FaceNet features. So we use FaceNet output as is and we add two more manual disentangled features to represent the age and the gender of the desired kin. Linear Regression main purpose is to find the best relation between each output and each input, so it can later interpolate output values using any input vector. We use least squares fitting function to train the linear regression model.

We faced a challenge that the feature-to-noise model must be trained using randomly generated faces and the randomly generated faces don't have pre-defined age or gender, instead of crafting a hand-made age-gender dataset, we use the age and gender extractor defined in Rothe et al. [20] which uses IMDB-WIKI dataset, we only need to use that model in the training phase of the feature-to-noise model so that the age and gender features can accurately represent the face properties. The final features vector consists of 130 features, 128 from the FaceNet network + 1 feature to represent age + 1 feature to represent the gender.

To be used in our pipeline, the feature-to-noise model takes a single 130-dimension vector and produces the needed 512-dimension latent vector which is passed to the PGGAN network. To train the network in our case, we randomly generate 100,000 sample and extract the features from generated faces. Using the latent vectors as outputs (labels) and the resultant features as inputs, the linear regression model is trained to convert any features vector to a latent vector.

### 3.4 Feature-to-noise model validator

To verify the accuracy of feature-to-noise model, we built a new simple model to measure how accurate can we generate a face from a features vector. To do that, we feed the

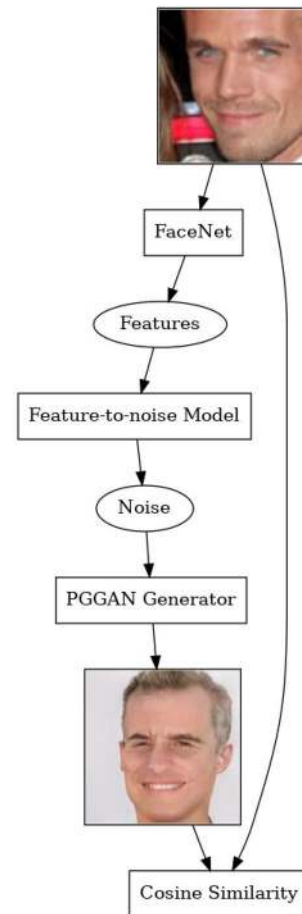


Fig. 3 Feature-to-noise verification

GAN a latent vector to get the corresponding face, then extract the features from the generated face using our feature extractor model, after that the features are passed to the feature-to-noise model, finally the generated noise is passed again to the GAN model and both outputs are compared together as in Fig. 3.

To compare the images, we calculate the cosine similarity between both images.

## 4 Experimental results

In this section, we discuss the results of each part of the pipeline. RAM, GPU and CPU that we use in this study are listed in Table 1

### 4.1 Feature-to-noise validation

We start with evaluating the feature-to-noise model using the model presented in Sect. 3.4. To test the model, we

**Table 1** Machine specifications

Aspect	Value
RAM	12 GB
GPU device	NVidia 1050 Ti
GPU cores	768
GPU memory	4GB
CPU	Xeon L5640, 12 Cores, 2.27GHz

supply the features-to-noise with a pre-generated face features. Features to noise samples can be seen in Fig. 4, the figure shows very good generation accuracy. Each face was re-generated using the features-to-noise model.

These good results lead us to an assumption, GAN can be used to reconstruct data from features. To validate this assumption we need to repeat the same experiment but on real images instead of generated images. Figure 5 has the results of reconstructing images from features. Note that these images are part of FIW dataset not part of CelebA dataset [12] (the dataset used to train the GAN).

The visual aspects of the results are good, but still the generated images have some inaccurate results. In depth analysis of the model are carried out to check the similarity between the indirectly generated images in comparison



**Fig. 4** Generated Image (Top row) and reconstructed image using linear regression (Bottom Row)



**Fig. 5** Generated faces using images to features to noise, real images (Top) and reconstructed images (Bottom)

and the directly generated images. The directly generated images are those images which have been generated directly from latent vectors, and the indirectly generated images are those images which we used features to predict needed noise and then used the resultant noise to re-generate the images.

We extend the previous results to include the added gender and age model, we need to be sure the GAN will be able to handle the generation of the face using face-net features while being restricted by the provided age and gender, the results are shown in Fig. 6 We selected a



(a)



(b)



(c)



(d)

**Fig. 6** Generated images using image to features to noise pipeline after adding Age and gender features to the stack, going from up to down age is increased (y-axis) and going from left to right we move along gender axis from male to female (x-axis), the image on the top row is the real face used to extract needed features

real image and extracted features and then used the linear regression model to generate multiple images with different ages and gender levels for the same face features.

Although the process used is the same, the results here are a bit worse than the previous results in Fig. 4. That's a result of two main reasons:

The first reason is GAN biasing; any unconditional GAN is biased to the dataset used to teach the network how to generate. So in our case the GAN is biased to generate celebrity-like faces. It can try to imitate the output we want, but the output won't be as good as using data similar to GANs domain. So, when we tried to regenerate an image that was originally generated using the same PGGAN network, the output was better than trying to generate an image that wasn't generated using the PGGAN network.

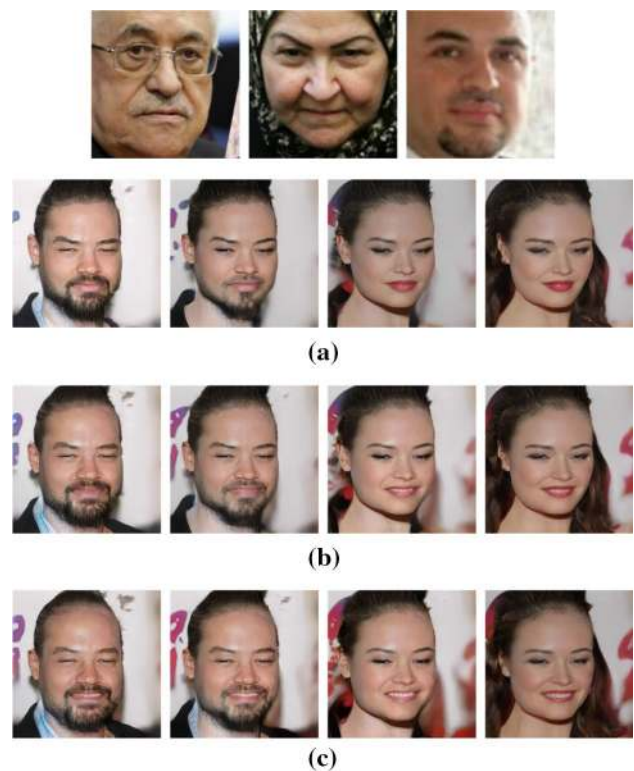
The second reason is the noncontinuous nature of generated samples. Not all latent values generate a real face, to explain this a bit more, let's assume we have Latent vector (A) that generates face (A) and latent vector (B) that generates face (B), when we move from Point (A) to point (B), intermediate latent vectors may not generate a complete face like illustrated in Fig. 7, the middle result represents a very poor face generation in GAN space, moving around this point produces better results.

Figure 8 shows the real father image, the real mother image, the real kin image and the generated kin images. The generated images are very comparable to the correct kin face. Also, moving on gender and age axes yielded good results.

More results are included in Fig. 9, the generated kin's faces have some visual aspects similar to the parents. For example, the first sample, the daughter has the fathers'



**Fig. 7** Intermediate noise values generate partial faces, leftmost and rightmost images are considered true faces however, intermediate values between them -the middle images- are partially corrupted



**Fig. 8** Generated images using image to features to noise pipeline after adding Age and gender features to the stack, going from up to down age is increased (y-axis) and going from left to right gender is increased (x-axis), the 3 images on the first row are the real father, real mother and the real kin

eyes and mothers skin tone as well as face shape. The samples show that the model can preserve the race and skin color very well.

## 4.2 Validation

We use a cosine similarity calculation on parent features to predict if the kin is a truly related to the parents or not.

The Verification model average accuracy is 63%, a more detailed validation table is listed in Table 2

Applying the model on generated samples yielded an average accuracy of 63%, detailed accuracy in Table 3.

To compare our results with KINGAN model [16], we applied the same face retrieval test to check the retrieval accuracy, the retrieval accuracy using NN through cosine similarity between generated kin face and real kin face

**Fig. 9** Generated images, the father on the left, the mother in the middle and the generated kin on the right



**Table 2** Validation model accuracy

Relation	Accuracy (%)
Father–Son	65.16
Father–Daughter	61.75
Mother–Son	61.95
Mother–Daughter	63.31

**Table 3** Model accuracy

Relation	Accuracy (%)
Father–Son	71.38
Father–Daughter	60.99
Mother–Son	57.68
Mother–Daughter	62.65

**Table 4** Retrieval accuracy compared to [16]

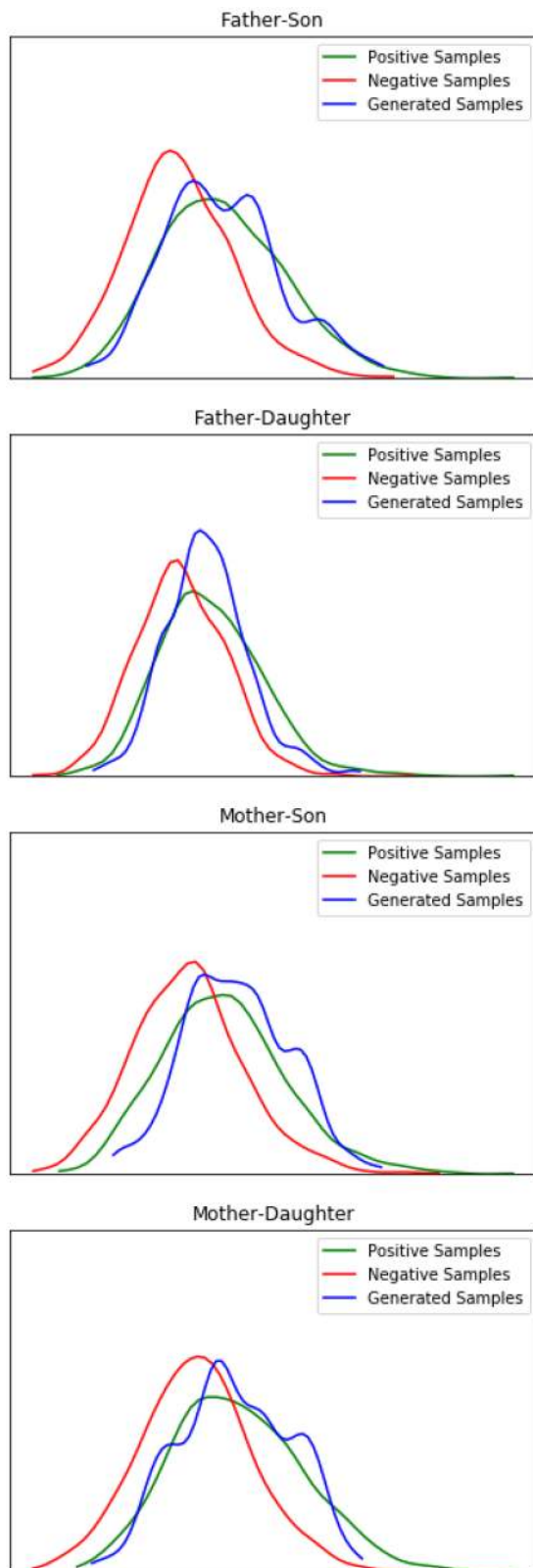
Model	Accuracy
KinshipGAN (w/o deep face)	0.048
KinshipGAN ( $\lambda_c = 1.0$ )	0.063
KinshipGAN ( $\lambda_c = 0.1$ )	0.107
Our model	<b>0.19</b>

Bold value indicates our result

yielded a retrieval accuracy of 0.19. It’s worth to mention that we don’t think retrieval accuracy is the best test for our approach, however, for the sake of comparison, the results are in Table 4

To visualize the results even more, we ran the validation model on 3 sets: Positive Samples (Real Parent, Real Kin), Negative Samples (Parent, not the real kin) and Generated Samples (Parent and the Generated kin using our model). The Generated samples histogram landed very close to the positive samples histogram as seen in Fig. 10 . Where the x-axis represent the confidence of the validation model and y-axis represent the number of samples





**Fig. 10** Data distribution of the validation scores for positive samples, negative samples and Generated samples

## 5 Conclusion and future work

In this study, We introduced a modular neural network model that predicts kin face from parents face images, the used model accuracy is 63%.

There is a huge potential of development due to the modular nature of the proposed solution. One of the possible improvements to the current pipeline is using the approach published in [9]. The study introduces a new generator architecture that learns how to deeply understand the data, i.e. separate high-level features and attributes. We think using this model instead of PGGAN will give much better results. Also, replacing the linear regression with a more powerful regression technique will give better results as well.

**Acknowledgements** I want to thank Youssef Ghatas for his help throughout this research.

### Compliance with ethical standards

**Conflict of interest** The authors declare that they have no conflict of interest.

## References

1. Dahan E, Keller Y (2018) Selfkin: self adjusted deep model for kinship verification. CoRR
2. Dandekar AR, Nimbarde MS (2014) Verification of family relation from parents and child facial images. In: 2014 international conference on power, automation and communication (INPAC), pp 157–162. <https://doi.org/10.1109/INPAC.2014.6981146>
3. Divil S, Shunhavanich P (1998) Baby face generator. <https://pdfs.semanticscholar.org/b85b/8a3c66fcb6b15cb18668244825fe7843525c.pdf>
4. Fang R, Tang KD, Snavely N, Chen T (2010) Towards computational models of kinship verification. In: 2010 17th IEEE international conference on image processing (ICIP), IEEE, pp 1577–1580
5. Fang R, Gallagher AC, Chen T, Loui A (2013) Kinship classification by modeling facial feature heredity. In: Proceedings of IEEE international conference on image processing, pp 2983–2987. <https://doi.org/10.1109/ICIP.2013.6738614>
6. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ (eds) Advances in neural information processing systems 27, Curran Associates, Inc., pp 2672–2680. <http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf>
7. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
8. Karras T, Aila T, Laine S, Lehtinen J (2017) Progressive growing of gans for improved quality, stability, and variation. Preprint [arXiv:171010196](https://arxiv.org/abs/1710.10196)
9. Karras T, Laine S, Aila T (2018) A style-based generator architecture for generative adversarial networks. Preprint [arXiv:181204948](https://arxiv.org/abs/1812.04948)

10. Kingma DP, Welling M (2013) Auto-encoding variational bayes. Preprint [arXiv:13126114](https://arxiv.org/abs/1312.6114)
11. Liu W, Wen Y, Yu Z, Li M, Raj B, Song L (2017) Sphereface: deep hypersphere embedding for face recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 212–220
12. Liu Z, Luo P, Wang X, Tang X (2015) Deep learning face attributes in the wild. In: Proceedings of the IEEE international conference on computer vision, pp 3730–3738
13. Lu J, Hu J, Liong VE, Zhou X, Bottino A, et al IUI (2015) The fg 2015-kinship verification in the wild evaluation. In: Proceedings of 11th IEEE international conference and workshops automatic face and gesture recognition (FG), vol 1, pp 1–7. <https://doi.org/10.1109/FG.2015.7163159>
14. Lu Y, Tai YW, Tang CK (2017) Attribute-guided face generation using conditional cycleGAN
15. Ouyang W, Wang X, Zeng X, , Luo P, Tian Y, Li H, Tang X (2015) Deepid-net: deformable deep convolutional neural networks for object detection. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), pp 2403–2412. <https://doi.org/10.1109/CVPR.2015.7298854>
16. Ozkan S, Orkan A (2018) KinshipGAN: Synthesizing of kinship faces from family photos by regularizing a deep face network. In: 2018 25th IEEE international conference on image processing (ICIP), IEEE, pp 2142–2146
17. Qin X, Tan X, Chen S (2015) Tri-subject kinship verification: understanding the core of a family. *IEEE Trans Multimed* 17(10):1855–1867. <https://doi.org/10.1109/TMM.2015.2461462>
22. Rakesh S, Atal K, Arora A, Purkait P, Chanda B (2012) Face image retrieval based on probe sketch using sift feature descriptors. In: Perception and machine intelligence, Springer
18. Robinson JP, Shao M, Wu Y, Fu Y (2016) Families in the wild (fiw): Large-scale kinship image database and benchmarks. In: Proceedings of the 24th ACM international conference on multimedia, ACM, New York, NY, USA, MM '16, pp 242–246. <https://doi.org/10.1145/2964284.2967219>
19. Robinson JP, Shao M, Zhao H, Wu Y, Gillis T, Fu Y (2017) Rfiw: Large-scale kinship recognition challenge. In: Proceedings of the 25th ACM international conference on multimedia, ACM, New York, NY, USA, MM '17, pp 1971–1973. <https://doi.org/10.1145/3123266.3132059>
20. Rothe R, Timofte R, Gool LV (2015) Dex: Deep expectation of apparent age from a single image. In: Proceedings of IEEE international conference on computer vision workshop (ICCVW), pp 252–257. <https://doi.org/10.1109/ICCVW.2015.41>
21. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M et al (2015) Imagenet large scale visual recognition challenge. *Int J Comput Vis* 115(3):211–252
23. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 815–823
24. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. Preprint [arXiv:13126034](https://arxiv.org/abs/1312.6034)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.