Pattern Recognition Letters
journal homepage: www.elsevier.com

Residual Codean Autoencoder for Facial Attribute Analysis

Akshay Sethi, Maneet Singh, Richa Singh, Mayank Vatsa**

IIIT-Delhi, New Delhi, India

Article history:
Received 29 March 2017

Keywords: Attribute Prediction, Cosine Similarity, Residual Learning, Deep Learning

ABSTRACT

Facial attributes can provide rich ancillary information which can be utilized for different applications such as targeted marketing, human computer interaction, and law enforcement. This research focuses on facial attribute prediction using a novel deep learning formulation, termed as R-Codean autoencoder. The paper first presents Cosine similarity based loss function in an autoencoder which is then incorporated into the Euclidean distance based autoencoder to formulate R-Codean. The proposed loss function thus aims to incorporate both magnitude and direction of image vectors during feature learning. Further, inspired by the utility of shortcut connections in deep models to facilitate learning of optimal parameters, without incurring the problem of vanishing gradient, the proposed formulation is extended to incorporate shortcut connections in the architecture. The proposed R-Codean autoencoder is utilized in facial attribute prediction framework which incorporates patch-based weighting mechanism for assigning higher weights to relevant patches for each attribute. The experimental results on publicly available CelebA and LFWA datasets demonstrate the efficacy of the proposed approach in addressing this challenging problem.

1. Introduction

What Does Your Face Shape Say About You? [1]
Is Your Personality Written All Over Your Face? [2]

These are some questions among all common questions related to facial attributes. A face image can provide multitude of information such as identity, gender, race, apparent age, and expression. While gender, race, and age estimation from face images are well explored research problems, estimating other attributes such as hair color, nose shape, eye shape, and attractiveness is also getting attention. Predicting facial attributes has several unique applications starting from focused digital marketing to law enforcement. Facial attribute prediction has an important application in the domain of online marketing, where targeted advertisements or products can be shown to a user based on his/her physical features and appearance. The increasing usage and access of computing devices (laptops, mobile phones) and Internet facilities has also led to the availability of abundant unlabeled data, which requires tagging in order to utilize facial attributes for meaningful tasks. Further,

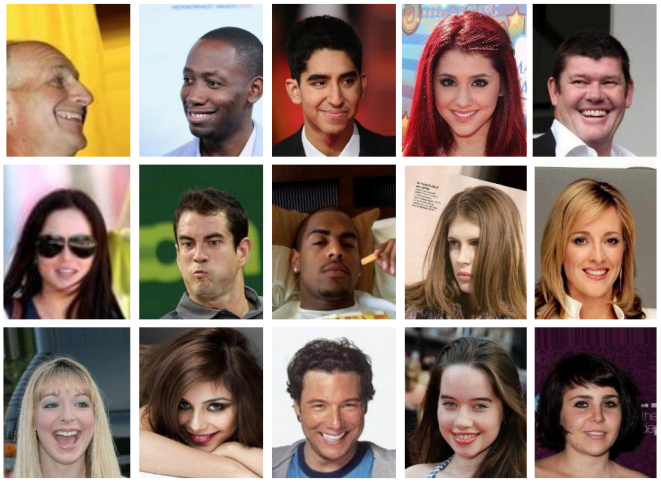

Fig. 1: Sample images from the CelebA dataset (Liu et al., 2015) demonstrating intra-class variations for different attributes. The first row corresponds to the *smile*, second to *young*, and third to *attractive*.

**Corresponding author: Tel.: +91-9654653404; fax: +91-11-26907410;
e-mail: mayank@iiitd.ac.in (Mayank Vatsa)

[1] http://tinyurl.com/k5t6rh7
[2] http://tinyurl.com/lpcz5jb

Table 1: Literature review of techniques used in facial attribute prediction.

Authors	Algorithm	Datasets Used	Classification Accuracy (Avg.)
Kumar et al. (2009)	Hand crafted features and RBF SVMs as classifiers	LFW	83.62%
Chung et al. (2012)	Supervised DBN for attribute classification	LFW	86.00%
Berg and Belhumeur (2013)	Local part based features termed as POOF and linear SVMs	LFW	83.00%
Luo et al. (2013)	Sum Product Network for prediction of attributes	LFW	87.90%
Liu et al. (2015)	Image localization deep CNN, followed by another deep CNN for prediction	CelebA, LFWA	87.00%, 84.00%
Huang et al. (2016)	CNN based model to address the problem of class imbalance during training.	CelebA	84.00%
Ehrlich et al. (2016)	Multi Task RBMs for learning a joint feature representation for attribute classification	CelebA	87.00%
Wang et al. (2016)	Siamese Network minimizing verification loss for face verification. Network is then fine-tuned on CelebA dataset	CelebA, LFWA	88.00%, 87.00%
Zhong et al. (2016a)	Features extracted from off-the shelf deep CNN and classified using linear SVMs	CelebA, LFWA	86.60%, 84.70%
Zhong et al. (2016b)	Features extracted from intermediate layers of deep CNN and classified using linear SVMs	CelebA, LFWA	89.80%,85.90%
Rozsa et al. (2016)	Treats attribute classification as a regression problem and uses MSE loss to train a VGG-16 topology CNN	CelebA	90.80%
Rudd et al. (2016)	CNN based model to perform multi task optimization with class imbalanced training data.	CelebA	90.94%
Hand and Chellappa (2017)	Multi-task deep CNN which also models attribute relationships.	CelebA, LFWA	Avg. accuracy not reported

facial attributes can be used as ancillary (or soft) information for face recognition systems, and can help in reducing the identity search space. Existing research in the literature has shown substantial improvement upon incorporating ancillary information for the task of person identification (Kumar et al., 2009; Samangouei et al., 2015; Mittal et al., 2017). Fig. 1 presents some sample images from the Celeb Faces Attributes (CelebA) Dataset (Liu et al., 2015). Each row contains images corresponding to a single attribute. The challenging nature of the problem in terms of high intra-class variations can be observed from these sample images.

1.1. Related Work

Kumar et al. (2009), in one of the initial works on facial attribute analysis, extracted hand crafted features such as edge magnitudes and gradient directions from facial regions and used the feature vector as input to a Support Vector Machine for attribute classification. Later, Berg and Belhumeur (2013) proposed to extract local part based features termed as POOF and train linear SVMs for each attribute. In 2012, deep learning was explored by Chung et al. (2012) where a deep attribute network was built over Deep Belief Networks trained in supervised manner. Later, Liu et al. (2015) proposed a two stage training approach for addressing the task of facial attribute prediction. Given an unconstrained face image, localization was performed using a deep Convolutional Neural Network (CNN) trained in a weakly supervised manner, followed by another CNN for learning feature representation and classification. In 2016, Wang et al. (2016) proposed a Siamese network which aimed at minimizing the error for the task of face verification. Inspired by the advancements in deep learning models, Zhong et al. (2016a,b) also demonstrated that off-the-shelf features learned by CNN models, pre-trained on massive facial identities, can effectively be adapted for attribute classification. The learned representations are provided as input to a binary linear SVM trained directly for all levels of representations to classify face attributes. The task of facial attribute prediction has also been posed as regression problem (Rozsa et al., 2016), where the authors adopt a 16 layer VGG topology while minimizing the mean squared error loss. Huang et al. (2016) presented an approach for learning deep representation of class imbalanced data by incorporating triplet-header hinge loss in CNNs. Parallely, Rudd et al. (2016) proposed a custom loss function for multiple attributes using a single Deep Convolutional Neural Network. The authors utilized the VGG-16 topology and obtained state-of-the-art classification results. Recently, Hand and Chellappa (2017) also proposed deep multi task CNNs for performing attribute classification. Some existing techniques for facial attribute prediction have been summarized in Table 1.

1.2. Research Contributions

In this research, we propose a novel deep learning formulation for facial attribute prediction in the wild. The key contributions of this research are as follows:

- A novel Residual Cosine Similarity and Euclidean Distance based autoencoder, termed as **R-Codean** autoencoder is proposed. Unlike traditional autoencoders, the proposed formulation incorporates both direction and magnitude information at the time of feature extraction along with shortcut connections, thereby resulting in a residual network,
- The proposed R-Codean autoencoder is utilized to present a facial attribute prediction framework. The framework incorporates a patch-based weighting mechanism for providing higher weight to certain face patches for a given attribute,
- Experimental results on the Celeb Faces Attributes (CelebA) and Labeled Faces in the Wild Attributes (LFWA) datasets (Liu et al., 2015) illustrate the efficacy of the proposed model by achieving comparable results to existing deep Convolutional Neural Network (CNN) models.

The remainder of this paper is organized as follows: the following section presents the proposed R-Codean autoencoder, followed by the proposed framework for facial attribute prediction in Section 3. Section 4 provides the details about the experimental protocol and evaluations, which is followed by the conclusions of this research.

2. Proposed Residual Cosine Euclidean Autoencoder

The proposed R-Codean autoencoder is built using a custom loss function which combines the traditionally used Euclidean distance measure with the Cosine similarity. The proposed loss function ensures that the model minimizes the loss between the input and the reconstructed sample not only in terms of *magnitude*, but also in terms of *direction*. To the best of our knowledge, this is the first work which incorporates cosine loss into the feature learning process of an autoencoder. The proposed model also incorporates residual shortcut connections (He et al., 2016) of two kinds (*symmetric* and *cross*) into an autoencoder.

2.1. Euclidean Distance based Autoencoder

The autoencoder model is an unsupervised deep learning architecture used for learning representations of the given data (Vincent et al., 2010). It is a special kind of neural network, where the input is also the target output. The objective of the model is to learn meaningful representations, such that the model is able to reconstruct the input from the learned representation. A single layer autoencoder consists of an encoding weight matrix ($\mathbf{W_e}$) and a decoding weight matrix ($\mathbf{W_d}$). For an input sample x, the loss function of an autoencoder can thus be formulated as:

$$\underset{\mathbf{W_d},\mathbf{W_e}}{\arg\min} \; \| x - \mathbf{W_d}\, \phi(\mathbf{W_e} x) \|_2^2 \tag{1}$$

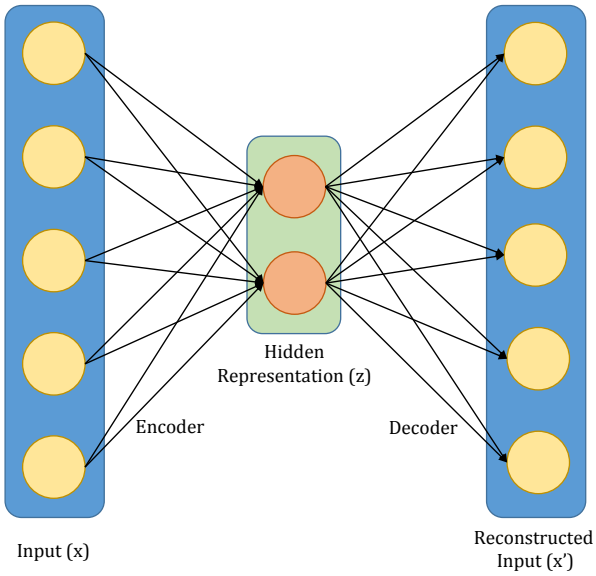

Fig. 2: Diagrammatic representation of a single layer autoencoder with input x, hidden representation z, and reconstructed input x'.

where, ϕ corresponds to a non-linear activation function such as *sigmoid* or *tanh*. Here,$\mathbf{W_e}x$ corresponds to the learned representation for the input x. The decoding weight matrix $\mathbf{W_d}$ can optionally be constrained by $\mathbf{W_d} = \mathbf{W_e}^T$, in which case the autoencoder is said to have tied weights. The above loss function corresponds to the mean squared error, or the Euclidean distance based loss for the autoencoder. Based on the above loss function, the model aims to minimize the reconstruction error (i.e. the error between the input x and the reconstructed sample ($\mathbf{W_d}\phi(\mathbf{W_e}x)$) in order to learn meaningful representations. Fig. 2 presents a single layer autoencoder with x as input, z as the representation, and x' as the reconstructed sample. In an attempt to learn richer feature representations, Stacked Autoencoders (Vincent et al., 2010) are proposed, where multiple autoencoders are stacked on top of each other. The hidden representation learned by the first autoencoder is provided as input to the second autoencoder which further learns higher level features. Due to the large number of parameters, deep autoencoders or stacked autoencoders are often trained in a greedy layer-by-layer manner, where only a single autoencoder is trained at a time (Bengio et al., 2006).

2.2. Cosine Similarity based Autoencoder

Cosine similarity based minimization techniques have extensively been used in document modeling and retrieval (Grossman and Frieder, 2012; Lafferty and Zhai, 2001). Document retrieval techniques aim to model the underlying distribution of keyword occurrences, as opposed to the actual count of the words. A similar analogy can be drawn for structured images such as faces, where cosine based similarity measure can be used for modeling the underlying distribution of the pixel values, as opposed to calculating the distance between their actual pixel intensities. To further explain, Fig. 3 presents four face images of the same subject. We compute the correspond-

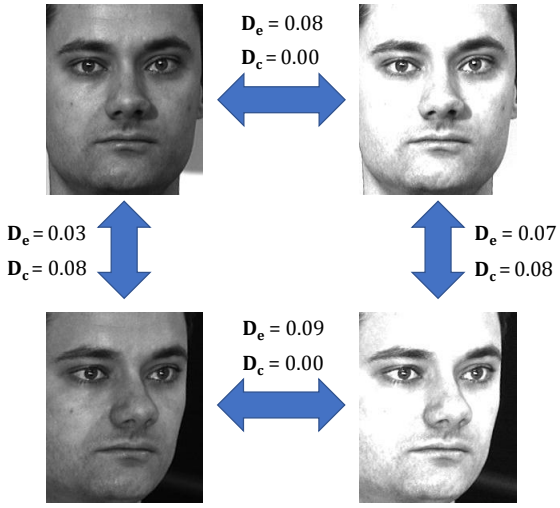

Fig. 3: Sample images illustrating the effectiveness of Euclidean distance and Cosine similarity under varying conditions. D_e corresponds to the Euclidean distance between two images, while D_c corresponds to the cosine distance. Cosine distance is computed as $1-Cosine\ Similarity$. It can be observed that for samples having slight pose variations, with constant illumination, the Euclidean distance is able to model the similarity well, however, in case of illumination variations, the Cosine similarity is able to encode the similarity better. A lower distance measure corresponds to a higher similarity.

ing Euclidean distance (D_e) and Cosine distance (D_c)[3] between the pixel values of image pairs. Since the values correspond to the distance scores, a smaller value represents more similar images. It can be observed that when the images are of the same intensity range and contain pose variations, Euclidean distance is able to model their similarity more effectively as compared to Cosine distance. On the other hand, when two images have variations in the intensity range, even with same pose, Euclidean distance is not able to encode the similarity. However, Cosine distance is able to model the similarity perfectly by producing a distance score of 0.00. Inspired by these observations, along with the properties of the Cosine similarity metric and it's applicability in high dimensional feature space, we extend the formulation of traditional autoencoder to Cosine similarity based autoencoder. Cosine similarity based loss function helps the model to learn direction based features (built on the underlying distribution of pixel values) as opposed to the magnitude (pixel intensities). The loss function of a Cosine similarity based autoencoder is defined as:

$$\ell_{Cos} = -\frac{x \cdot (\mathbf{W_d}\phi(\mathbf{W_e}x))}{\|x\|_2^2 \times \|\mathbf{W_d}\phi(\mathbf{W_e}x)\|_2^2} \tag{2}$$

where, x is the input to the autoencoder and $\mathbf{W_d}\phi(\mathbf{W_e}x)$ is the reconstructed input learned by the autoencoder. As mentioned previously, $\mathbf{W_e}$ and $\mathbf{W_d}$ represent the encoding and decoding weights of the autoencoder, respectively. It is important to note that since the Cosine metric is a *similarity* metric, a negative sign has been incorporated in the loss function in order to decrease the overall distance, or reconstruction error of the model.

[3]Cosine similarity is converted into a distance measure as $D_c = 1 - S_c$, where S_c is the Cosine similarity.

The above equation minimizes the angle between the input and it's reconstruction by minimizing the cosine distance between the two. This model is especially useful for handling images with brightness or contrast variations.

2.3. Proposed R-Codean Autoencoder

In order to learn feature representations based on both direction and magnitude, we combine Euclidean distance based loss function with Cosine similarity based loss function, and a Cosine Euclidean (Codean) Autoencoder is proposed. The loss function of the model is formulated as:

$$\ell_{Codean} = \alpha \times \ell_{Euc} + \beta \times \ell_{Cos} + \lambda R \tag{3}$$

Here, the first term corresponds to the Euclidean loss (Eq. 1), the second term refers to the Cosine loss (Eq. 2), and the third term corresponds to a regularization constraint. α, β, and λ are regularization parameters controlling the weight given to each individual term. The above equation helps the autoencoder learn a representation which minimizes both, the magnitude by the Euclidean loss and the direction by the Cosine similarity between the input and reconstructed sample. As explained previously (from Fig. 3), the Euclidean distance ensures that slight variations in pose in the reconstructed sample are handled by the autoencoder, while the Cosine distance handles illumination variations. For a single layer model, with input x, Eq. 3 with ℓ_1-norm regularization on the encoding weight matrix, can be expanded as follows:

$$\ell_{Codean} = \alpha \times \|x - (\mathbf{W_d}\phi(\mathbf{W_e}x))\|_2^2 - \beta \times \frac{x \cdot (\mathbf{W_d}\phi(\mathbf{W_e}x))}{\|x\|_2^2 \times \|\mathbf{W_d}\phi(\mathbf{W_e}x)\|_2^2} + \lambda \times \|\mathbf{W_e}\|_1 \tag{4}$$

The ℓ_1-norm regularization helps learn sparse feature representations for the given input, thereby retaining meaningful latent variables during the feature learning process. The above equation depicts a single layer Codean autoencoder, which can easily be extended for k layers as follows:

$$\ell_{Codean} = \alpha \times \|x - g \circ f(x)\|_2^2 - \beta \times \frac{x \cdot (g \circ f(x))}{\|x\|_2^2 \times \|g \circ f(x)\|_2^2} + \lambda \times \sum_{i=1}^{k} \left\|\mathbf{W_e^i}\right\|_1 \tag{5}$$

where, $f(x)$ is the encoder function, such that $f(x) = \mathbf{W_e^k}...\phi(\mathbf{W_e^2}(\phi(\mathbf{W_e^1}x)))$ and $g(x)$ is the decoder function, such that $g(x) = \mathbf{W_d^1}(...(\mathbf{W_d^{k-1}}(\mathbf{W_d^k}x)))$. Since the above loss function contains large number of parameters, training the entire model together results in the problem of vanishing gradients. Further, inspired by the analysis that adding shortcut connections facilitates learning of deeper networks, along with resulting in the network imitating the performance of multiple shallow networks (Veit et al., 2016), we next propose to incorporate shortcut connections in Codean to learn robust feature representations.

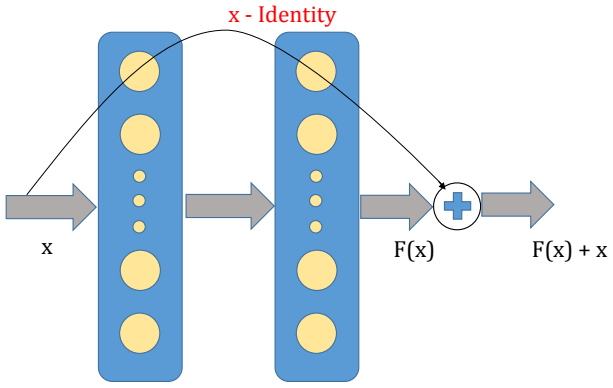

Fig. 4: Shortcut connection between two layers, representing the concept of residual learning.

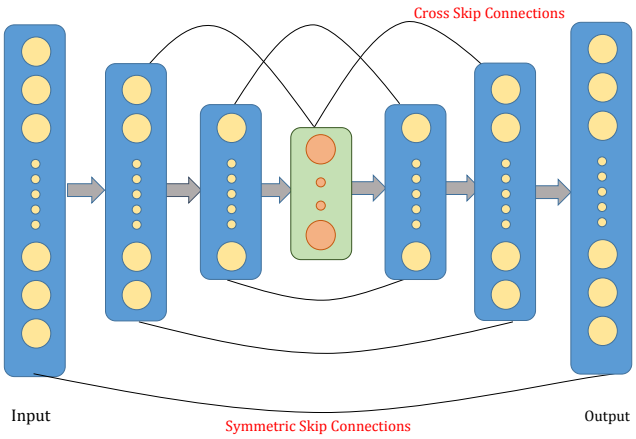

Fig. 5: Cross and symmetric shortcut (skip) connections in the proposed residual autoencoder model.

2.4. Residual Learning in Codean Autoencoder: R-Codean

Residual deep learning framework is introduced by He et al. (He et al., 2016) for Convolutional Neural Networks. The authors have observed that learning deeper models resulted in a higher error rate, as compared to their shallower counterpart. Since the aim of learning deeper models is to learn higher level features with each increasing layer, shortcut (or skip) connections are introduced in an attempt to reduce the overall error and facilitate better learning. The introduction of shortcut connections in deep networks thus results in creation of residual networks. Let $H(x)$ be the mapping to be learned by a model on an input x. Now, instead of learning $H(x)$, the residual network is made to learn the mapping $F(x)$, where $F(x)$ is given by:

$$F(x) = H(x) - x \tag{6}$$

The input (x) is then added back to $F(x)$, effectively learning $H(x)$. Residual learning as described above helps to overcome the input degradation (or vanishing gradient) problem in deep networks. In the i^{th} layer of a neural network having weight matrix as $\mathbf{W}_i$, the residual learning framework can be incorporated with a building block defined as:

$$y = F(x, \mathbf{W}_i) + x \tag{7}$$

here, x and y are the input and output vectors of the i^{th} layer. The function $F(x, \mathbf{W}_i)$ represents the mapping to be learned. Fig. 4 presents a sample residual network of two layers.

In the proposed Codean autoencoder model, two types of shortcut (or skip) connections have been introduced in order to create the proposed Residual Codean (R-Codean) Autoencoder. The concept of residual learning has been incorporated in the autoencoder model by including *cross* and *symmetric* shortcut connections. As shown in Fig. 5, cross shortcut connections are overlapping connections which are made between the alternate layers of the network. Such connections help in preventing the input degradation problem in deep networks. Symmetric skip connections are non-overlapping connections created at a larger distance between the encoder and decoder of the same autoencoder. Symmetric shortcut connections help in passing the image details forward, thereby improving the reconstruction process of the autoencoder. Both these connections help in improving the gradient flow which further enables the model to converge to the optimal parameters. Incorporating these shortcut connections in the Codean autoencoder model described above results in the proposed R-Codean Autoencoder for learning robust feature representations.

3. Proposed Facial Attribute Prediction Framework using Residual Codean Autoencoder

The proposed R-Codean autoencoder is utilized for facial attribute prediction. Fig. 6 presents the block diagram of the proposed facial attribute prediction framework. The entire pipeline can broadly be divided into the following components: (i) preprocessing, (ii) feature extraction using R-Codean autoencoder, and (iii) classification. Each step of the pipeline is explained in detail in the following subsections.

3.1. Pre-processing of Images

For a given input sample, the image is geometrically normalized and loosely cropped to obtain the face region. The cropped image is down-sampled to 64×64 and converted to grayscale. In literature, it has often been observed that facial features are encoded both locally, as well as globally while performing face recognition (Wong et al., 2011; Bharadwaj et al., 2016). Inspired by these findings, a similar approach is followed for encoding facial attributes by tessellating the input image into nine equal overlapping patches (as shown in Fig. 6). The individual patches as well as the entire image are then used for feature extraction.

3.2. Feature Extraction via Proposed R-Codean Autoencoder

The proposed R-Codean autoencoder is used for the task of feature extraction. One R-Codean autoencoder model is trained for each face image patch, and one for the entire image, thereby resulting in ten independent models (nine patches + 1 full face). Patch based feature learning enables the model to learn specific characteristic of a particular facial region. These local features can then be utilized for predicting face component-specific attributes. Coupled with the learned representation over the entire face image, the ten models provide a holistic (global), as well as local representation of the input sample.

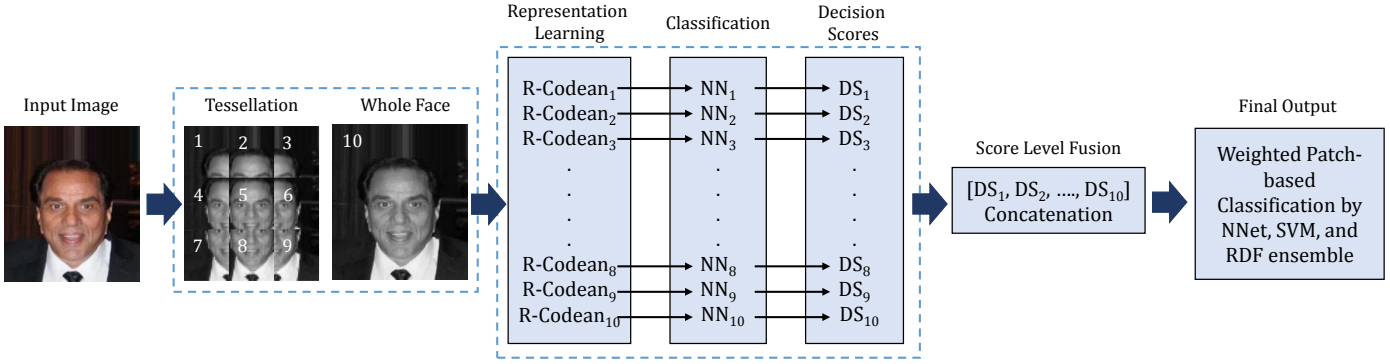

Fig. 6: Illustrating the steps involved in the proposed attribute classification pipeline built upon the proposed Residual Cosine Euclidean (R-Codean) Autoencoder.

Attributes			
5 o Clock Shadow	×	✓	×
Arched Eyebrows	✓	×	✓
Attractive	✓	✓	✓
Bags Under Eyes	×	✓	×
Bald	×	×	×
Bangs	×	×	✓
Big Lips	×	✓	✓
Big Nose	×	✓	×
Black Hair	×	✓	×
Blond Hair	×	×	×

Fig. 7: Sample images from the CelebA dataset, along with a few attributes. Each attribute is a binary attribute, having value either '1' or '0'.

3.3. Weighted Patch-based Classification

A two-step classification approach is followed in the proposed facial attribute prediction framework. In the first stage, for each R-Codean autoencoder trained for each patch in the previous step, a two layer neural network classifier is learned using *sigmoid* activation function in the output layer. Therefore, ten neural networks are trained, one corresponding to each R-Codean autoencoder, i.e., one for each facial patch (9) and one for the full face. k probability scores are obtained from the output layer of each neural network, where k corresponds to the number of facial attributes in the dataset. In the second stage, score fusion is performed by concatenating the scores obtained from the ten neural networks to create a feature vector of length $10 \times k$. This feature vector is then provided as input to an ensemble of neural network, Random Decision Forest, and Support Vector Machine for obtaining the final attribute classification. In order to emulate the human tendency of focusing on specific regions for certain attributes, a weighted patch-based mechanism is also incorporated in the classification framework. Patch-based weights are learned for each attribute, such that relevant patches are given higher weight for a given attribute. For instance, in case of *wearing necktie* attribute, patches 7, 8, and 9 (shown in Figure 6) might contribute higher in the decision making process, as opposed to the other patches. Finally, max-voting is performed on the outputs of the three classifiers in order to obtain the final decision.

3.4. Implementation Details

For all experiments, detected and normalized face images are resized to 64×64 and converted to grayscale. Images are tessellated in 3×3 overlapping patches of dimension 32×32. 10 R-Codean autoencoders having three hidden layers with dimensions $[l, l, l]$, are trained on the face patches (and full face). k corresponds to the dimension of the input vector. ℓ_1-norm regularizer with $\lambda = 0.01$ is incorporated for learning a sparse representation of the data. Each R-Codean autoencoder also incorporates cross shortcut connections between the first and third encoding layers, second encoding and first decoding layer, third encoding and second decoding layer. Symmetric shortcut connections are also introduced between the first encoding and third decoding layer, second encoding and decoding layer, third encoding and first decoding layer. R-Codean is optimized using the adam optimizer (Kingma and Ba, 2014) with a decaying learning rate. The initial learning rate was set to 0.001, which decayed by a factor of ten whenever the training loss stagnated. For each R-Codean autoencoder, two hidden layer neural network of dimension $[\frac{l}{2}, \frac{l}{4}]$ is learned in the first classification stage. The autoencoders and neural networks utilize ReLU activation function in their layers. The autoencoders are implemented in Python based Keras framework (Chollet, 2015) and the classifiers in Scikit-learn library (Pedregosa et al., 2011). The source code will be made publicly available for the research community.

4. Experimental Results and Analysis

The proposed approach is evaluated on large-scale Celeb Faces Attributes (CelebA) and Labeled Faces in the Wild Attributes (LFWA) datasets (Liu et al., 2015). The CelebA dataset contains 10,000 identities, each of which have twenty images. Therefore, the dataset contains a total of 2,00,000 images. LFWA contains 13,233 images pertaining to 5,749 subjects. Each image in both the datasets is annotated with forty binary face attributes such as Male, Young, Bangs, Gray Hair, and five facial landmark key points. Fig. 7 presents some sample images of CelebA dataset.

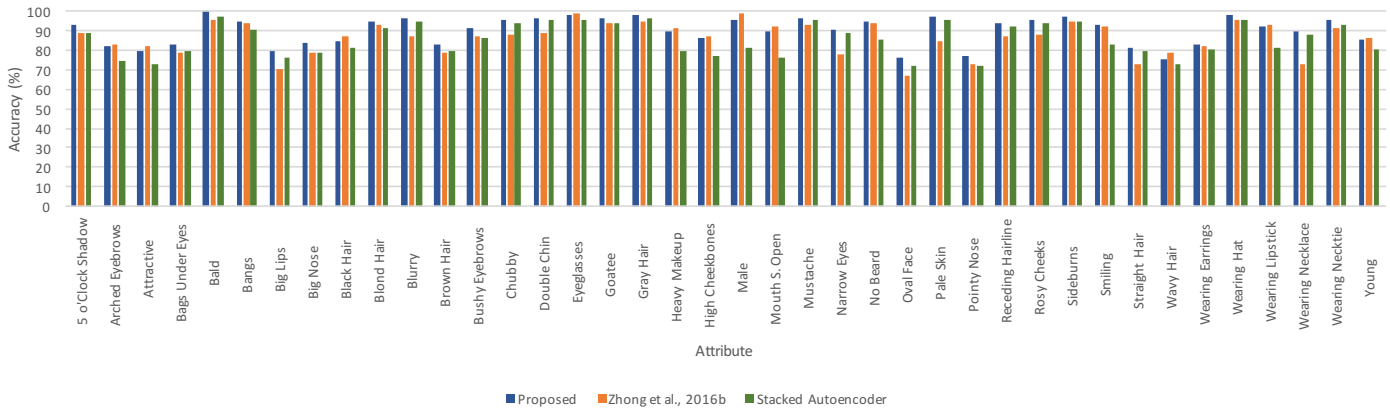

Fig. 8: Bar graph illustrating the performance of the proposed framework in comparison with current state-of-the-art technique (Zhong et al., 2016b) and stacked autoencoder. Accuracies have been reported for all 40 attributes for the CelebA dataset.

Table 2: Comparison with state-of-the-art and existing methods on CelebA and LFWA datasets. Accuracy corresponds to the mean accuracy obtained over all the attributes.

Architecture	**CelebA**	**LFWA**
Stacked Autoencoder + NNET	85.60%	76.22%
3-layer CNN + NNET	87.39%	82.37%
VGG-Face (Parkhi et al., 2015) + NNET	85.30%	81.04%
Fine-tuned VGG-Face (Parkhi et al., 2015) + NNET	87.45%	80.14%
ResNet (He et al., 2016) + NNET	84.03%	76.94%
Fine-tuned ResNet (He et al., 2016) + NNET	86.23%	78.98%
Zhong et al. (2016a)	86.80%	84.70%
Liu et al. (2015)	87.00%	84.00%
Wang et al. (2016)	88.00%	**87.00%**
Zhong et al. (2016b)	89.80%	**85.90%**
Rozsa et al. (2016)	**90.80%**	-
Rudd et al. (2016)	**90.94%**	-
Proposed	**90.14%**	**84.90%**

The efficacy of the proposed model has been evaluated on an existing commonly used experimental protocol (Liu et al., 2015; Tan et al., 2016; Günther et al., 2016; Zhong et al., 2016b). The entire dataset is partitioned into three parts, where the first 1,60,000 images of CelebA are used to train the autoencoders, and next 20,000 images are used to train the ensemble classifier. The images of the remaining 1,000 identities (with 20,000 images) are used for testing. On the other hand, LFWA dataset is divided into two, one partition is used for training, while the second forms the test set.

4.1. Comparison with Other Deep Learning Models

The proposed facial attribute framework, built upon the proposed R-Codean autoencoder yields an overall mean classification accuracy of **90.14**% and **84.90%** over the 40 attributes of CelebA and LFWA datasets, respectively. Table 2 presents the mean accuracy of the proposed framework, along with other comparative architectures. In order to compare the performance of other deep model architectures, experiments are performed using a Stacked Autoencoder, Convolutional Neural Network (CNN), and existing state-of-the-art CNN models of ResNet (He et al., 2016) and VGG-Face (Parkhi et al., 2015), while using a neural network for classification. For VGG-Face and ResNet, comparison has been performed with both pre-trained models, as well as fine-tuned models. Fine-tuning is performed on the pre-trained models using the training partition of each dataset, while the Stacked Autoencoder model is trained from scratch with the available training data. It can be observed that the proposed R-Codean autoencoder based pipeline achieves an improvement of more than 4% on the CelebA and 7% on the LFWA database, as compared to stacked autoencoders. The proposed pipeline also presents improved results as compared to several existing state-of-the-art CNN models (with and without fine-tuning). Specifically, an improvement of more than 4% is observed in comparison to the ResNet model, along with similar results for VGG-Face features.

4.2. Comparison with State-of-the-Art Techniques

Comparison has also been performed with existing state-of-the-art architectures present in the literature. Table 2 presents the accuracies of recently proposed architectures, taken directly from their publications. It can be observed that the proposed R-Codean autoencoder based approach achieves a comparable mean classification accuracy with respect to the current state-of-the-art approach (Rudd et al., 2016) on the CelebA dataset. It is important to note that while the existing architectures incorporate Convolutional Neural Networks in their pipeline, this is the first work achieving comparable classification performance using autoencoders. Training R-Codean autoencoder requires only 20 seconds per epoch on a workstation powered with NVIDIA K20 GPU and 64GB RAM. Moreover, an unseen input sample takes less than a second for the entire attribute prediction pipeline. Table 3 provides the individual accuracy of each attribute obtained on the CelebA dataset. Fig. 10 presents the bar graph, depicting comparison with Zhong et al. (2016b)

Table 3: Attribute classification accuracy (%) of all forty attributes of the CelebA dataset using the proposed facial attribute prediction framework.

Attribute	Accuracy	Attribute	Accuracy	Attribute	Accuracy
5'o' Clock Shadow	92.88	Arched Eyebrows	81.63	Attractive	79.67
Bags Under Eyes	83.15	Bald	99.52	Bangs	94.51
Big Lips	79.89	Big Nose	83.67	Black Hair	84.80
Blond Hair	94.97	Blurry	96.57	Brown Hair	82.97
Bushy Eyebrows	91.36	Chubby	95.51	Double Chin	96.45
Eyeglasses	98.18	Goatee	96.77	Gray Hair	97.92
Heavy Makeup	89.72	High Cheekbones	86.74	Male	95.87
Mouth S. Open	89.81	Mustache	96.31	Narrow Eyes	90.64
No Beard	94.64	Oval Face	76.55	Pale Skin	96.93
Pointy Nose	76.95	Receding Hairline	93.64	Rosy Cheeks	95.32
Sideburns	97.60	Smiling	92.83	Straight Hair	81.19
Wavy Hair	75.42	Wearing Earrings	82.65	Wearing Hat	97.93
Wearing Lipstick	91.96	Wearing Necklace	89.82	Wearing Necktie	95.88
Young	86.63	-	-	-	-

and stacked autoencoder features over all 40 attributes. Since the attribute-specific accuracy is not provided by Rozsa et al. (2016) or Rudd et al. (2016), attribute wise comparison is not possible.

Table 4: Evaluation of the proposed R-Codean based framework for performing facial attribute classification on CelebA dataset.

Architecture	Accuracy
Effect of Pre-Processing	
Learning a Full Face Model Only	86.86%
Learning Patch-based Models Only	88.16%
Effect of Shortcut Connections	
With Symmetric Connection only	87.90%
With Cross Connection Only	88.81%
With No Shortcut Connections	87.60%
Effect of Loss Function	
With Euclidean Distance (MSE) only	88.30%
With Cosine Distance only	85.10%
Effect of Patch-based Weighting	
Without Patch-based Weighting	89.42%
Effect of Classification Ensemble	
Support Vector Machine Only	88.81%
Neural Network Only	88.30%
Random Decision Forest Only	88.84%
Proposed Framework with R-Codean	**90.14%**

4.3. Analysis of the Proposed Facial Attribute Prediction Pipeline

In order to thoroughly evaluate the proposed R-Codean autoencoder based pipeline for facial attribute prediction, experiments have been performed on the CelebA dataset to evaluate each component of the same. Table 4 presents the mean classification accuracies for different variations of the proposed framework. To re-iterate, in the pre-processing component face tessellation is performed, and models are trained for both full face, as well as independent patches, in order to encode both local as well global features. The framework is analyzed by learning features on only the full face *or* the tessellated patches, instead of both. As can be observed from Table 4, both the techniques independently yield lower results as compared to their combination, thereby strengthening our hypothesis of utilizing both local and global features for facial attribute prediction.

Further evaluations are performed on the R-Codean autoencoder model by understanding the effect of shortcut connections, as well as the combined loss function. As can be seen from Table 4, removing residual shortcut connections from the model results in a drop of around 3%. A similar drop in accuracy can be observed upon incorporating only a single kind of shortcut connection as opposed to both. In order to evaluate the efficacy of the loss function of R-Codean autoencoder, where the loss function is a combination of Euclidean distance (MSE), as well as the Cosine distance, experiments were performed with models having only Cosine or Euclidean distance based loss function as well. An increase of at most 5% is observed upon incorporating the magnitude (Euclidean distance) as well as the direction (Cosine distance) in the feature learning process. At the classification level, the proposed framework utilizes an ensemble of Neural Network, Support Vector Machine, and Random Decision Forest. It can be observed from Table 4 that upon utilizing each of these classifiers independently, the framework does not yield optimal results. An improvement of

Fig. 9: Samples illustrating the effect of weighted patch-based attribute classification. For certain attributes, specific regions are more focused and hence more weights are assigned (learned). For example, in case of *bald*, high weight is associated with patches containing the head region of the face image, while low weight is given to other parts of the image.

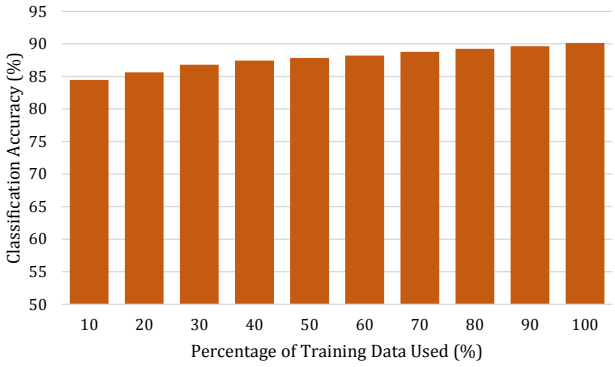

Fig. 10: Bar graph representing the accuracy of the proposed framework with varying percentage of training data, for the CelebA dataset.

close to 1.5% is observed upon using the proposed ensemble. Comparison has also been performed to understand the effect of training data size on the proposed framework. As can be observed from Fig. 10, even when only 50% of the entire training set is used, the proposed framework yields a classification accuracy of 87.83%, showcasing a drop of only 2.3%. This motivates the utility of the proposed R-Codean based framework for less training data as well.

To show the effect of learning weights in weighted patch-based classification, Fig. 9 illustrates some sample attributes, along with the associated weights. The values of weights lie between $0 - 1$ and the illustration is color coded, *red* corresponds to 1 and *blue* corresponds to 0. It can be observed that the learned weights are intuitive and the patches which relevant to the given attribute are selected. For instance, in case of *Wearing Necktie*, high weight is given to patches which contain the lower portion of the image. The opposite of this is observed for *Bald*, where high weight is observed for patches which contain the head, while very low weight is associated with the lower portion of the image.

Fig. 11 shows some sample images misclassified by the proposed facial attribute prediction pipeline. Fig. 11(a) refers to some sample false positives. These example show that large variations in facial features make the task of facial attribute prediction challenging. Variations in hairstyles across males and females also lead to mis-classifications for attributes such as *male* and *wavy hair* (Fig. 11 (a)). Fig. 11(b) also presents some false negative samples of the proposed framework. It can be observed that subjective attributes such as *attractive* and *receding hairline* are difficult to encode in the trained model.

(a) False Positives (b) False Negatives

Fig. 11: Sample mis-classifications of the proposed facial attribute prediction pipeline. (a) corresponds to the false positives, where the images in the first row are falsely predicted as attribute *male* and second row is predicted to have the attribute *wavy hair*. (b) refers to sample false negatives, where the first row corresponds to the attribute *attractive*, and second row corresponds to the attribute *receding hairline*.

5. Conclusion

This research aims to address the problem of facial attribute prediction in the wild. A novel formulation for Residual Cosine Euclidean Autoencoder, termed as R-Codean has been presented for the same. The loss function of the proposed R-Codean model consists of a Euclidean distance based term, and a Cosine similarity based term. The model aims to learn hidden representations of the input data, such that the error in direction and magnitude of the input and reconstructed sample is minimized. The Euclidean distance based component of the loss function handles slight variations across pose more efficiently, while the Cosine distance based component models the illumination variations better. Shortcut connections in the R-Codean further ensure that optimal parameters are learned during the feature learning process. An entire framework has been presented for performing facial attribute analysis, which encodes local as well as global facial representations during feature learning process, and also incorporates weighted patch-based classification of attributes. Each component of the framework has been analyzed in order to create the optimal framework for facial attribute prediction. Experimental results on the CelebA and LFWA datasets, and comparison with existing state-of-the-art techniques demonstrate the efficacy of the proposed model.

6. Acknowledgment

The authors thank the reviewers and editors for their valuable comments and feedback. This research is partially supported by MEITY (Government of India), India. M. Vatsa and R. Singh are partially supported through Infosys Center for Artificial Intelligence.

References

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2006. Greedy layer-wise training of deep networks, in: International Conference on Neural Information Processing Systems, pp. 153–160.

Berg, T., Belhumeur, P., 2013. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation, in: IEEE Conference on Computer Vision and Pattern Recognition, pp. 955–962.

Bharadwaj, S., Bhatt, H.S., Vatsa, M., Singh, R., 2016. Domain specific learning for newborn face recognition. IEEE Transactions on Information Forensics and Security 11, 1630–1641.

Chollet, F., 2015. Keras. `https://github.com/fchollet/keras`.

Chung, J., Lee, D., Seo, Y., Yoo, C.D., 2012. Deep attribute networks, in: Deep Learning and Unsupervised Feature Learning NIPS Workshop.

Ehrlich, M., Shields, T.J., Almaev, T., Amer, M.R., 2016. Facial attributes classification using multi-task representation learning, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 47–55.

Grossman, D.A., Frieder, O., 2012. Information retrieval: Algorithms and heuristics. volume 15. Springer Science & Business Media.

Günther, M., Rozsa, A., Boult, T.E., 2016. Affact-alignment free facial attribute classification technique. arXiv preprint arXiv:1611.06158 .

Hand, E., Chellappa, R., 2017. Attributes for improved attributes: A multi-task network utilizing implicit and explicit relationships for facial attribute classification, in: AAAI Conference on Artificial Intelligence, pp. 4068–4074.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Huang, C., Li, Y., Change Loy, C., Tang, X., 2016. Learning deep representation for imbalanced classification, in: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5375–5384.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. CoRR abs/1412.6980. URL: `http://arxiv.org/abs/1412.6980`.

Kumar, N., Berg, A.C., Belhumeur, P.N., Nayar, S.K., 2009. Attribute and simile classifiers for face verification, in: IEEE International Conference on Computer Vision, pp. 365–372.

Lafferty, J., Zhai, C., 2001. Document language models, query models, and risk minimization for information retrieval, in: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 111–119.

Liu, Z., Luo, P., Wang, X., Tang, X., 2015. Deep learning face attributes in the wild, in: IEEE International Conference on Computer Vision, pp. 3730–3738.

Luo, P., Wang, X., Tang, X., 2013. A deep sum-product architecture for robust facial attributes analysis, in: IEEE International Conference on Computer Vision, pp. 2864–2871.

Mittal, P., Jain, A., Goswami, G., Vatsa, M., Singh, R., 2017. Composite sketch recognition using saliency and attribute feedback. Information Fusion 33, 86 – 99.

Parkhi, O.M., Vedaldi, A., Zisserman, A., 2015. Deep face recognition., in: British Machine Vision Conference.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in python. Journal of Machine Learning Research 12, 2825–2830.

Rozsa, A., Rudd, E.M., Boult, T.E., 2016. Adversarial diversity and hard positive generation, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 25–32.

Rudd, E.M., Günther, M., Boult, T.E., 2016. Moon: A mixed objective optimization network for the recognition of facial attributes, in: European Conference on Computer Vision, pp. 19–35.

Samangouei, P., Patel, V.M., Chellappa, R., 2015. Attribute-based continuous user authentication on mobile devices, in: IEEE International Conference on Biometrics Theory, Applications and Systems, pp. 1–8.

Tan, L., Li, Z., Yu, Q., 2016. Deep face attributes recognition using spatial transformer network, in: IEEE International Conference on Information and Automation, pp. 1928–1932.

Veit, A., Wilber, M.J., Belongie, S., 2016. Residual networks behave like ensembles of relatively shallow networks, in: Advances in Neural Information Processing Systems, pp. 550–558.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research 11, 3371–3408.

Wang, J., Cheng, Y., Feris, R.S., 2016. Walk and learn: Facial attribute representation learning from egocentric video and contextual data, in: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2295–2304.

Wong, Y., Chen, S., Mau, S., Sanderson, C., Lovell, B.C., 2011. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition, in: Computer Vision and Pattern Recognition Workshops, pp. 74–81.

Zhong, Y., Sullivan, J., Li, H., 2016a. Face attribute prediction using off-the-shelf CNN features, in: IEEE International Conference on Biometrics, pp. 1–7.

Zhong, Y., Sullivan, J., Li, H., 2016b. Leveraging mid-level deep representations for predicting face attributes in the wild, in: IEEE International Conference on Image Processing, pp. 3239–3243.