International Journal of Software Innovation
Volume 12 • Issue 1

Recommendation System for Hairstyle Based on Face Recognition Using AI and Machine Learning

Yogesh M. Kamble, DKTE Society's Textile and Engineering Institute, Ichalkaranji, India*
Raj B. Kulkarni, Government College of Engineering, Karad, India

ABSTRACT

Many machine learning algorithms have been introduced to solve different types of problems. Recently, many of these algorithms have been applied to deep architecture models and showed very impressive performances. In general, deep architecture models suffer from the over-fitting problem when there is a small number of training data. In this article the attempt is made to remedy this problem in deep architecture with regularization techniques including overlap pooling and flipped image augmentation and dropout; the authors also compared a deep structure model (convolutional neural network (CNN)) with shallow structure models (support vector machine and artificial neural network with one hidden layer) on a small dataset. It was statistically confirmed that the shallow models achieved better performance than the deep model that did not use a regularization technique. Faces represent complex multidimensional meaningful visual stimuli and developing a computational model for face recognition is difficult. The authors present a hybrid neural-network solution which compares favorably with other methods.

KEYWORDS

Deep Architecture, Overlap Pooling, Flipped Image

1. INTRODUCTION

The hairstyle is one of the most important aspects of people in determining their appearance and mood. People look completely different by changing their hairstyles. Hairstyle can make human appearance attractive or unattractive. If someone chooses an inappropriate hairstyle, then it gives a bad look and loses confidence. A hairstyle means styling hair on the human scalp. Hair gives various fashionable styles to a human's body. The increase of fashion's most people think of hair as their main important thing a beauty expert says that a proper hairstyle for someone depended on their face shapes. It is better to know our face shape and features well before doing hairstyles. Similar face shapes have similar hairstyles. Therefore, it is better to have a hairstyles recommendation system to know about hair styles before doing hairstyles. The major objective of the work is a proposed method to recommend hairstyles based on major face shapes with a combination of hair expert's knowledge. One of significance in the proposed methodology mainly concerns image processing techniques to

DOI: 10.4018/IJSI.309960 *Corresponding Author detect face shape rather than using other AI techniques. The proposed classification algorithm is to classify the face shapes into five shapes oval, oblong, square, round and heart.

Hairs are the most important aspect of the human body. It reflects the personality of every individual. Most of the people neglect their hairs and concentrate on their body and physics. But if you have proper hairstyle, it doesn't matter. Everyone think that any hairstyle is fine as long as it does not make them look bad. But they do not realize that they are missing opportunity to enhance their beauty. There are many reasons that tells us why hairstyle completes your entire look.

2. LITERATURE STUDY

According to Y. Zhang, and E.C. Prakash (2008), face shape is also important information for glasses design companies. In this paper, we proposed a non-contact method to classify the face shape by using Support Vector Machine (SVM) technique. This algorithm consists of three steps: head segmentation, face plane identification, and face shape classification. First, as whole 3D body data is captured and used as input of system, Eigenvector is used to define frontal side. Chin-Neck junction, Ellipsoid Fitting Technique and Mahalanobis distance are combined as a head segmentation algorithm to segment the 3D head. Second, face shape can be observed when projected on a plane. Major axes of ellipsoid are used to define a plane along the head called the face plane. Face shape on the face plane is classified into four classes in third step.Face shape is classified into four groups: ellipse, long, round, and square face shape.

Accuracy rate is 73.68%. Significant points for classification are located in 91 positions around the face.

Some research proposed face shape classification for different application. Y. Xu et al. (2010) proposed a method to measure and classify Shanghai female face shape based on 3D image feature. Young female for 201 cases were divided into eight kinds of the face shapes by cluster analysis using SPSS software. Face features used for classification are facies temporal width, bizygomatic breadth, mandibular breadth, maxilla-chin breadth, and physiognomic facial length Eight types of face shape are heart shape, roundshape, ellipse shape, long shape, pear shape, square shape, diamond shape, and melon seeds shape. This paper aims to prove that all indexes is reasonable for classification of human faces. L. Li and et al. proposed a method to classify face shape for person's expression recognition. Existing research proposed detection syndrome from face image. K. Wilamowska et al. (2009) proposed a method to classify between individuals with 22Q11.2 syndrome and general opulation based on face data. This method uses 3D face data, and finds the difference of facial features between two groups of data based on hapebased morphology. 3D snapshot, 2.5D depth image, and curved line of face are used for detection. Classification is performed by using feature vectors combining with the Principle Components Analysis. The accuracy rate ranged from 74% to 76%.

All the earlier work mentioned was trustworthy & verified by Biomedical Signal Processing Laboratory, National Electronics and Computer Technology Center, Thailand. The project is performed under the financial support from National Electronics and Computer Technology Center.

Wisuwat Sunhem et al. (2016) presented a hairstyle recommendation system for women based on hairstyle experts' knowledge and a face shape classification scheme. The system classified the user's face shape into five categories which were suitable for hairstyle recommendation using Support Vector Machine algorithms. The hairstyle e rules were based on beauty experts' suggestions. The system is based on the fine-grained similarity of face shape. However, human face shape changes over time. For example, getting fat due to higher work pressure can change the shape of a person's face. Therefore, our focus is to design and implement a coarse-grain similarity of face shape for hairstyle recommendation. Even their work was observed, verified and granted by National Natural Science Foundation of China

Based on existing research of Y. Zhang, and E.C. Prakash (2008) the face shape classification is applied for many applications. All of research proposed the method based on the assumption that frontal face direction has been known, and the head has been extracted. It is not appropriate for data in three-dimensional space without frontal face direction identification. Therefore, we propose a new method for face shape classification by using SVM techniques. This algorithm consists of three steps: head segmentation, face plane identification, and face shape classification. This method is first fully automatic and noncontact system to classify the face shape from whole 3D body data.

The things that are include in our project and improved from models built compared to previously built models are say classification of face shapes previously it was just classified into ellipse, long, round, and square but in our model round, heart, oval, square, rectangular, triangle & oblong which results in variation and improvised results in recommending hairstyles. Also algorithms used previously were using SVM giving accuracy of 74% to 76%. But using HaarCascade Classifier in OpenCV and available functions in dlib accuracy has increased even more to 89%.For hairstyle recommendation, we have seen the Naive Bayes classification algorithm being used in previous work. Given an adequate dataset with a proper amount of facial attributes and a labeled hairstyle feature, the problem of hairstyle recommendation can be reduced to classification, where a mapping function from the dependent facial attributes (as inputs) to the target hairstyle feature is established. Therefore, we use the Random Forests algorithm for hairstyle recommendation. Our Dataset, which carries about 2000 records, is shuffled and split into train/test sets .Then we fine-tune the Random Forests model concerning the number of trees (n estimators) and the maximum depth of trees (max depth). These are the parameters with which we have improved our model's efficiency & performance.

3. PROPOSED APPROACH

3.1 Algorithmic Description of Each Modules

3.1.1. Module Name: Data Preparation

Step 1: START
Step 2: Collect data i.e. collect different hairstyle photos of different face shape
Step 3: Prepare the database of different face shape like: heat, long, oval, round, square etc.
Step 4: Sorting of collected data i.e. organize or sort the collected data into respective databases according to face shape
Step 5: END

3.1.2. Module Name: Face Detection

Step 1: START
Step 2: Importing required libraries for e.g. import cv2
Step 3: Load the cascade with the function cascadeClassifier for importing harcascade_frontalface_default.xml file.
Step 4: Input the image from user for recommending hair style
Step 5: Converting user's image into grey scale image
Step 6: Detecting face in the image by the function detectMultiScale().
Step 7: Draw rectangle around faces to show the result of face detection module
Step 8: END

3.1.3. Module Name: Face Landmark Detection

Step 1: START
Step 2: Import the required libraries for e.g. numpy, dlib
Step3: importshape_predictor_68_face_landmarks.dat file into respected directory
Step 4: Input the user's image
Step 5: Convert the image into grey scale image

Step 6: Detect the face using detector() function
Step 7: Detect the landmarks using predictor() function
Step 8: Display the landmarks by drawing the circles around the key / landmark points
Step 9: END

3.1.4. Module Name: Face Shape Detection

Step 1: START
Step 2: Import required libraries like: beautifulSoup, pandas, pathlib, kmeansetc
Step 3: Apply kmeans on image who's landmark is detected to differentiate between hair and skin
Step 4: Apply following steps to get length of forehead

1. get midpoint of forehead
2. travel to right and left side
3. detect the corners of forehead which is nothing but hairs

Step 5: drawing lines on forehead with circles
Step 6: Calculating angle between the landmarks that we calculated by using arcustangens.
Step 7: Calculating similarity between the landmarks
Step 8: Face shape is detected by considering the conditions
Step 9: Display output
Step 10: END

3.1.5. Module Name: Recommend Best Suited Hairstyle

Step 1: START
Step 2: Input the hair length either long or short, input the hairstyle should be updos or not
Step 3: Input the image from user
Step 4: Processing of image. Processing includes noise removal, face detection, face landmark detection, and face shape detection
Step 5: Recommending the appropriate hairstyle. Here the detected face shape is used to fetch the images from databases
Step 6: display the recommended images along with their percentage of best suited on user's face.
Step 7: END

4. IMPLEMENTATION

4.1 Environmental Setting for Running the Project

1. Import Numpy

```
pip install numpy;
```

2. Install opencv

```
pip3 install opencv-python
```

3. Install dlib
    a. Download the Cmake installer and install it.

```
Pip install cmake.
```

   b. Add Cmake executable path to the environment variable.
   c. Restart TheCmd or PowerShell window for changes to take effect.
   d. Download the dlib source from https://pypi.org/project/dlib/#files extract it and enter into the folder.
   e. Now install dlib.

```
Pip install dlib
```

4. Install scikit-learn
   a. First you need to install numpy and scipy.

```
To install numpy: pip install numpy
To install scipy: pip install scipy
```

   b. Wheel package for scikit-learn can be installed

```
pip install scikit-learn
```

   c. Open a console and type the following to install or upgrade scikit-learn.

```
pip install -U scikit-learn
```

5. Install Pandas:

```
pip install panda
```

6. Install itertools:

```
pip install more-itertools
```

7. Install Seaborn:

```
pip install seaborn
```

8. Install glob:

```
pip install glob2
```

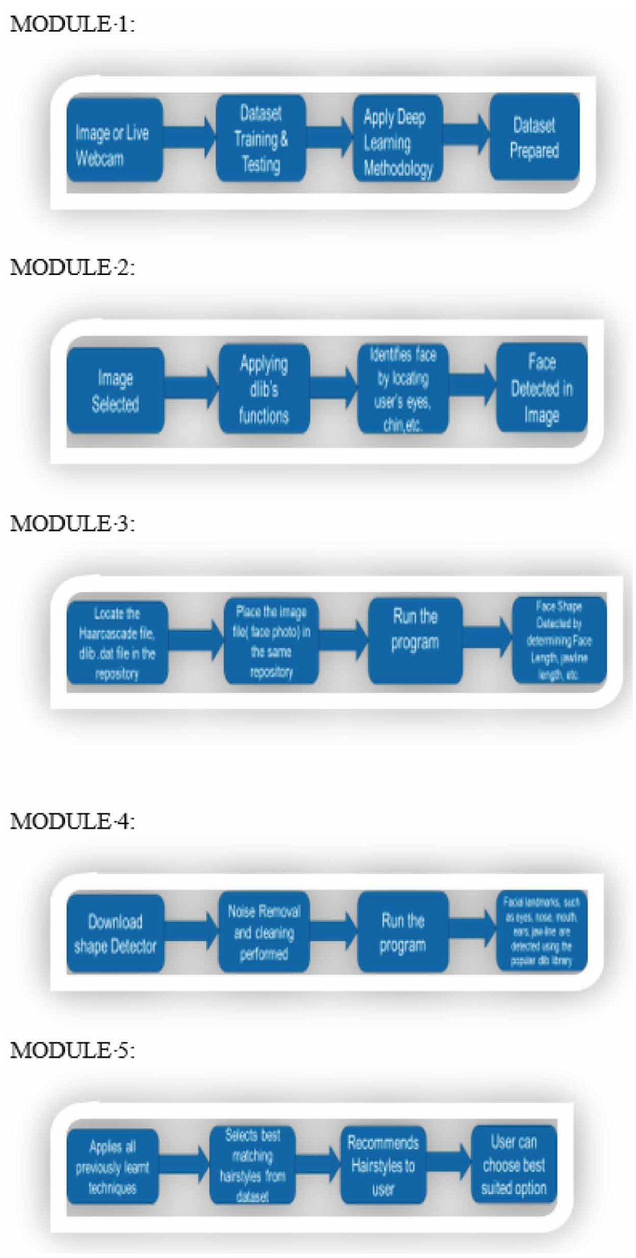
9. Install pathlib:

```
pip install pathlib
```

10. Install random:

```
pip install random
```

Figure 1. Implementation Details


MODULE-1:
Image or Live Webcam
Dataset Training & Testing
Apply Deep Learning Methodology
Dataset Prepared
MODULE-2:
Image Selected
Applying dlib's functions
Identifies face by locating user's eyes, chin,etc.
Face Detected in Image
MODULE-3:
Locate the Haarcascade file, dlib .dat file in the repository
Place the image file(face photo) in the same repository
Run the program
Face Shape Detected by determining Face Length, jawline length, etc.
MODULE-4:
Download shape Detector
Noise Removal and cleaning performed
Run the program
Facial landmarks, such as eyes, nose, mouth, ears, jaw-line are detected using the popular dlib library
MODULE-5:
Applies all previously learnt techniques
Selects best matching hairstyles from dataset
Recommends Hairstyles to user
User can choose best suited option

4.2 Detailed Description of Methods

(1)Module Name: Data Preparation
Module Input: Dataset Images
Output: Trained & Tested Dataset
(2)Module Name: Face Detection
Module Input: User Image upload
Output: Face Found in Image

(3)Module Name: Face Shape Detection
Module Input: Face identified from previous module
Output: Face Shape Identified (round, heart, oval, etc.)
(4)Module Name: Face Landmark Detection
Module Input: Face shape detected from prev. module
Output: Face Landmarks identified (such as eyes, nose, mouth, ears, jaw-line, etc. are detected
(5)Module Name: Recommend Best suited Hairstyle
Module Input: Final result after execution of all modules precisely
Output: Best Hairstyle cards for user

5. PERFORMANCE ANALYSIS

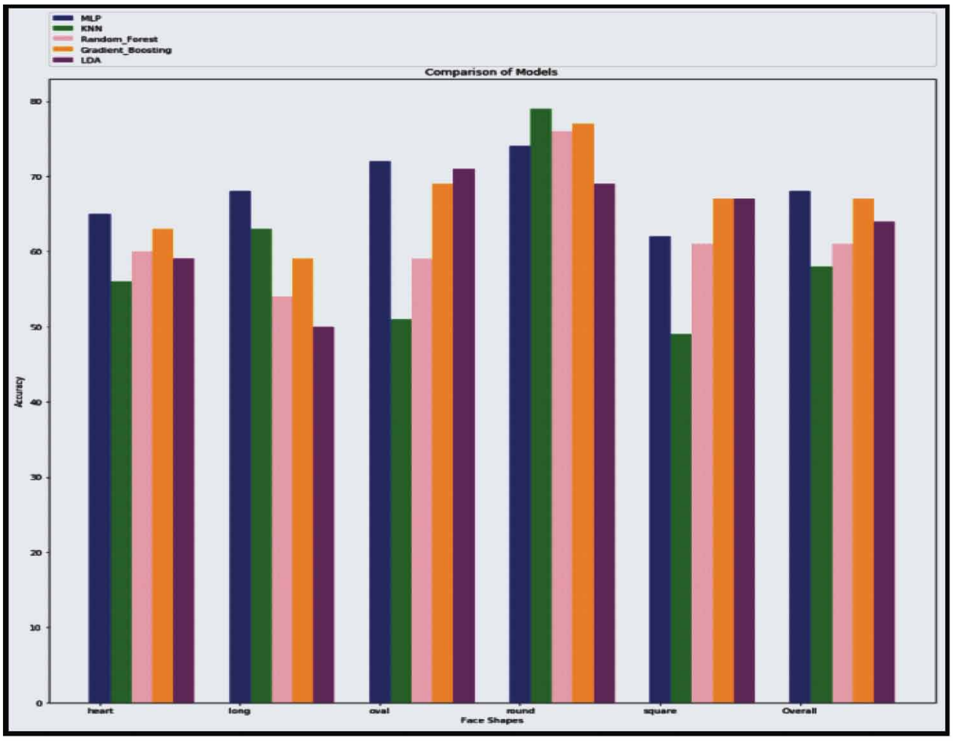
In Figure 2, we have outlined all of the models I attempted. Refer to the notebook below for the modeling implementation and validation. Below the graph, I've discussed the reasons we attempted these models, the hyperparameters we chose as well as the decision support to use my selected model.

5.1. Why Choose This Set of Models?

The reason to choose KNN because one of its most attractive features is that is simple to understand and easy to implement, especially for multi-class problems, Also KNN is also a very flexible model and does not require a certain underlying distribution of the data. After a few runs, KNN tends to perform the worst of my models.

In case of a Random Forest Classifier (Ensemble predictor), It would be useful to form a strong model using the learnings from weaker models. A disadvantage of the RF model is that it can over fit

Figure 2. Attempted models



the data, as you will observe here. I attempted to tune the hyper parameters to help the model ignore some of the noise/over-fitting in the small sample size (by changing the leaf size). The RFC performs fast (even with hundreds of iterations) which makes it a good choice as well.

In each iteration of RFC, the classifiers are trained independently. However, with Gradient Boosting, it builds on weak classifiers so that the next classifier is trained to improve the already trained ones. This may explain why GB performs better than RFC. The GB model took the longest to run with certain hyper parameters were chosen.

Linear Discriminant Analysis is also tried to compare the performance of a linear model. LDA makes predictions by estimating each probability that each input belongs to each class. The class that gets the highest probability is the output class and a prediction is made for that input.

5.1.1 LDA Limitations

- It requires normalized data with the same variance (which I provided).
- It can be impacted by outlier data, which exists in my dataset so may have impacted performance here.

Finally, a neural network is also used because they are powerful and flexible with the ability to capture nonlinear and complex underlying characteristics of a dataset with a high degree of accuracy. Neural networks boost to feed the results from one part of the model into the next part, allowing it to learn more as data flows through the layers. The disadvantage is that because NN require large datasets, the model I built has an overfitting problem.

5.2 Hyper Parameters Used for Analysis

5.2.1 MLP

Hidden_Layer_Sizes: This is the most important hyperparameter so I used an iterative approach within the solver to find the best number and size of layers. Fine tuning the layers led to better and better results:

- max_iter: 100 - I tried 100, 200, 500 and it always selected 100 so I locked that down to save time later.
- learning_rate_init: 0.01 (any smaller and the model would not converge)

5.2.2 KNN

For the KNN Model, I tuned the hyperparameters of K and the weighting. Because goal is to classify the samples correctly, I calculated the misclassification error (MSE) for each K. I determined K=4 was best through this process. But, when I changed the default weighting from uniform to weighting by distance, it led to overfitting.

5.2.3 RF

I tuned the RF model using the accuracy score and cross validation scores and mean. There are an unlimited number of combinations so I tried to focus on the hyper parameters most relevant to the dataset and the model to build.

- min_samples_leaf - way overfit because it allows leaf size to be 1; A smaller leaf makes the model more prone to capturing noise in training data. At default (1), there was significant overfitting; as I increased min_samples_leaf, the scores for both train and test decreased, but for training, there was more decline, reducing overfitting.

- n_estimators (The number of trees in the forest.) - higher # takes longer but makes predictions stronger and more stable.
- max depth - The maximum depth of the tree. As None, nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples
- min_samples_split - I set to 15 (default is 2) to try to reduce noise from small sample size. At 2, the model was significantly overfit; at 15, less so.
- max_features: max # of features a tree can use - the best model recommended sqrt (square root of the total number of features in individual run)

5.3 Gradient Boosting

GB models are similar to random forest in terms of the definition of the hyperparameters. There are more hyper parameters to choose when you use GB.

- n_estimators: higher # takes longer but makes predictions stronger and more stable. Picked 300.
- max_depth: max depth of a tree; used to control over-fitting as higher depth will allow the model to learn specifics relations for a particular sample; random search for “best” model chose 15.
- min_samples_leaf: increasing from default of 1 to 20 increased accuracy again by forcing the model not to overfit so each sample had its own leaf
- min_samples_split: worked best at 2 (default)

5.3.1 LDA

This model ran a lot more quickly so I used a grid search to find the “best” model. I had to split the grid search to solvers that allowed shrinkage (lsqr, eigen) and the svd solver which does not.

- SVD Solver:

n_components: the “best” model had 1 component
tol = 0.1

This model had the same performance as the other option with eigen or lsqr solvers (with shrinkage). I tuned the n_components (also 1) and shrinkage (auto was the best).

To evaluate each module, the MLP neural network model is used because it performed the best in 4 out of 5 shape classifications and overall. It is also believed that this model will continue to improve performance as more images are added, which will allow the application to scale efficiently. The Gradient Boosting model performed similarly to the MLP however, it is a much slower model to run.

The neural network model performed well but I know it would be a much better model with many more input images and celebrities. All of the models had over-fitting, likely because of the use of the same celebrity in many images.

REFERENCES

Wilamowska, K., Shapiro, L., & Heike, C. (2009). Classification of 3D face shape in 22Q11.2 detection syndrome. *Proc. of the 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, (pp. 534-537). IEEE.

Wisuwat, S. & Kitsuchart, P. (2016). An approach to face shape classification for hairstyle recommendation. *Conference: 2016 Eighth International Conference on Advanced Computational Intelligence (ICACI)*. IEEE. doi:10.1109/ICACI.2016.7449857

Xu, Y., Qiu, J., & Ma, L. (2010). Measurement and classification of Shanghai female face shape based on 3D image feature. *2010 3rdInt. Cong. On Image and Signal Processing (CISP2010),* (pp. 2588-2592). IEEE. doi:10.1109/CISP.2010.5648207

Zhang, Y., & Prakash, E. C. (2008). An example-based anthoropometric model for face shape synthesis. *IEEE Int. Conf. on Multimedia and Expo 2008,* (pp. 1573-1576). IEEE.

Y.M. Kamble has completed his post graduation in 2016 from Shivaji University Kolhapur. His is pursuing Ph.D. at Shivaji University Kolhapur. Currently he is working as Assistant Professor in Computer Sci. and Engg. department at DKTE Society's Textile and Engineering Institute Ichalkaranji.

R.B. Kulkarni has completed his Ph.D. in 2012 from Solapur University. His specialization of doctorate degree is Web Engineering. Currently he is working as Associate Professor in Information Technology department at Govt. College of Engineering Karad.