Challenge Data 2017: Regain

Rome Sassi, Steven Fluchaire, Carla Lasry, Master MASH

March 25, 2017

Abstract

"The qualitative assessment of image content and aesthetic impression is affected by various image attributes and relations between the attributes. Modelling of such assessments in the form of objective rankings and learning image representations based on them is not a straightforward problem."Aarushi Gaur and Krystian Mikolajczyk, University of Surrey United Kingdom in their paper "Ranking images based on aesthetic qualities" (2015)

1 Introduction

The project we chose on the Challenge Data 2017 is Regain. It has been proposed by the startup "Regaind" which offers to "mark" your portrait giving a score between 0 and 24.
The challenge was to predict the aesthetic score of portraits by combining photo analysis and facial attributes analysis.

Usually, the beauty of a face is subjective and in this challenge, the goal is to give objective aesthetic scores to portraits. That is why this project is very challenging and exciting. Furthermore, applying machine learning, deep learning methods and doing vision at the same time was interesting for us.

2 Data Description

The aesthetic score (mean of the aesthetic scores predicted by 6 people) is an integer between 0 and 24. This is the average of taggers' score which was out of 5 (they could select either 1, 2 ,3, 4 or 5 to give a score to the portrait). We printed the distribution by score of the training portraits. We observed that most of the training portraits were scored between 9 and 17.

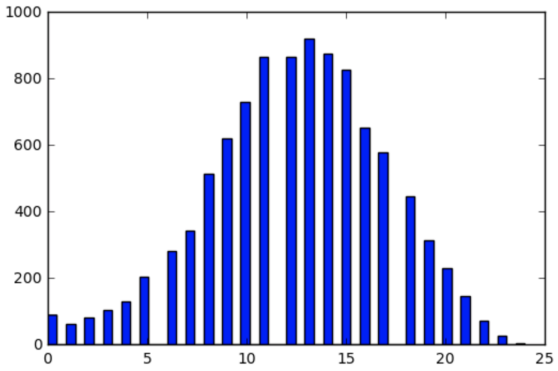

Figure 1: Different Values of Y

First of all in order to understand the problem we had to understand the training set and the test set.
The training set is composed of: 10000 images representing portraits and a dataframe of size (10000,107) where 107 is the number of features for each image. The test set is composed of: 3000 images representing portraits and a dataframe of size (3000, 95) where 95 is the number of features for each image. We observed that 95 features were in common in both sets, however, 12 features were missing in the test set. At this moment, we decided to delete them from the training set.

These 12 missing features are representing background, angle of the face (roll or pan angle), position of the face, sharpness of the face, face exposure, and face expression, for each one two values was provided in the training set, the positive (between 0 and 1) and the negative (between -1 and 0). For example if $background\_impact\_p = 0.14$, it means that 14% of the people (6 people) that annotated this picture think that the background has a positive impact on the global score.

3 Cleaning Description

Initially, we decided to work only on the features presents both in the traing set and in the test set

We have modified 8 features which were in '*object*' into float. We have also replaced '$None$' value by the mean of the feature's values.

One last variable gave us some trouble, it was '*list_others*'. We had to create new features such as '*happiness*', '*pensive*', '*smiling*'... which were coming from '*list_others*' and then we have deleted '*list_others*'.

4 New features

To improve our score we have created some new features, coming from the already existing features (in the training set) but also from the pictures themselves.

4.1 Contrast Quality

We have created '*Contrast_Quality*' from the pixel of the picture. Each picture has been imported in gray scale, then we have also computed and equalized the histogram of the grayscale image. From that, we calculated their square of differential. And to obtain only one additional feature, we calculated the mean for each image.

4.2 GLCM

The Gray Level Coocurrence Matrix (GLCM) method is a way of extracting second order statistical texture features. A number of texture features may be extracted from the GLCM. Only six second order features namely angular second moment, correlation, inverse difference moment, and entropy are computed here.

After importing pictures in gray scale, we have extracted the following texture features :

- '*contrast*': Contrast measures the local variations in the gray-level co-occurrence matrix.
- '*energy*': Energy provides the sum of squared elements in the GLCM.
- '*homogeneity*': Homogeneity measures the closeness of the distribution of elements in the GLCM to the GLCM diagonal.
- '*correlation*': Correlation measures the linear dependency of gray levels of neighboring pixels.
- '*dissimilarity*': Dissimilarity is the local homogeneity. It is high when local gray level is uniform and inverse GLCM is high.
- '*ASM*': ASM (Angular Second Moment) is a measure of homogeneity of an image. Angular Second Moment is high when image has very good homogeneity or when pixels are very similar.

4.3 Sharpness

Our data set was providing the coordinates of the left and right eyes, the nose, the mouth and the left and right eyebrows. For each image, we have extracted new images. One containing the left eye, one containing the right eye, one containing the nose and one the mouth.

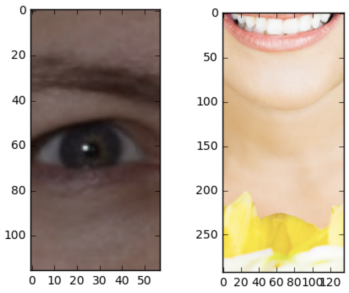

Figure 2: Example with the left eye of picture 12 and the smile of picture 2

After that, for each one of this new images, we have evaluated the sharpness of the gray image using the Tenengrad method with a threshold of zero (Ref [1]).
The Tenengrad criterion is a robust and functional accurate method for image quality measures. It is formulated as follow:
$TEN = \sum_x \sum_y S(x,y)^2$ for $S > T$ where ,here, T is the threshold zero ($T = 0$). From that, we obtained 5 new features: '$sharp\_left\_eye$','$sharp\_left\_eyebrow$','$sharp\_right\_eyebrow$','$sharp\_mouth$','$sharp\_nose$'.

4.4 Brightness and luminosity

On the one hand, a way to test if the color of the picture is dark or light was to check the brightness. We used the existing formula $Brightness = sqrt(0.241 * R^2 + 0.691 * G^2 + 0.068 * B^2)$ (Ref [2]).
However, to be more accurate, we decided to obtain 2 different features from that. We calculated the min and the max of the brightness and then from that we deduced the 2 following features:

$$Contrast1 = \frac{\max(Brightness) - \min(Brightness)}{\max(Brightness) + \min(Brightness)}$$

and

$$Contrast2 = \frac{mean(Brightness)}{\max(Brightness) + \min(Brightness)}$$

On the other hand, we extracted the luminosity of each image. The luminosity of an image is equal to the sum of the means luminance value of each pixel of the image computed with special weights. The means luminance value of one pixel is calculated by using the following formula: 0.2126*R + 0.7152*G + 0.0722*B.

4.5 Symmetry

An other idea to evaluate the aesthetic score of the portraits could be the symmetry of the picture. For this, we have written a function $symmetry\_feature$ which computes for each picture the Histogram of Oriented Gradients of the first half of the picture and the Histogram of Oriented Gradients of the second half flipped. To do this, we have preliminarily resized the portraits in a same size (120,120). Then, we have calculated the difference between the two HOG of the two halfs of the picture. This created a new feature $symmetry\_feature$.

Figure 3: From left to right: 1.1. Entire portrait, 1.2. First half of the portrait, 1.3. Flipped second half of the portrait

After trying this, we have been suggested to evaluate the symmetry of the face instead of the entire picture. Or more precisely, if the face is aligned on the y-axis of the picture. We used the two features $center\_eye\_x$ and $center\_mouth\_x$. We built a new feature $symmetry\_face$ by computing the difference between the two features mentioned above.

We have used these two features.

Having done this, we thought that we could use HOG (Histogram of Oriented Gradients) itself as a feature. However, computing HOG gives more than 2000 features more for each picture. This made our model too large to work on. Moreover we thought that would not add so much information as HOG is used in object recognition. Indeed, our objective was different to classify pictures as in the "dog/cat" problem, we wanted to rank it from an aesthetic point of view. So, we did not use this feature.

4.6 Blurry

Looking at the different portraits of the train set, we noticed that the pictures which had the lowest scores were very blurred whereas the ones with the highest scores were very sharp. That's why we tried to build an additional feature corresponding to the blurry of each picture.

To do this, reading some papers, a good and simple method was to compute the variance of the Laplacian.The method is to simply take a single channel of an image (presumably grayscaled) and convolve it with the following 3 x 3 kernel:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Figure 4: The Laplacian kernel

Then, we take the variance of the output of the last transformation. Generally, if the answer is less than 100, the picture is quite blurry, and if the answer is over 100, it is pretty sharp.

4.7 Distance between eyes

Again, looking at our train data, we tried to remark a criterion which discriminates our pictures. We noticed that when the face was the major part of the picture, i.e. when the face was relatively "big" on the picture, the score was higher. That's why, we thought that the distance between the eyes could be a good feature. We have the features "$right\_eye\_x$" and "$left\_eye\_x$" so we have computed the absolute value of the difference between the two.

5 Deep Learning

When we have tried the Lasso and the Ridge Method, we have been able to identify the most important features. Most of them were the missing features of the test set. So, we have understood that one of the only way to improve our score was to use the 12 features that were missing in the testing set.

Firstly, we tried to predict the missing features from the features available with a learning algorithm. Unfortunately, it was impossible because of the lack of dependence of the explicative features.

Hence, we used deep learning and convolutional neural networks in order to work directly on the pictures. As we only had 10000 pictures on the training set, we firstly tried to double our number of picture by flipping them. But it was not successful, we could not obtain a satisfying score even with deep learning. That is why we decided to use the weights from ImageNet. These weights are already trained on a huge amount of data. This allowed us to reach a better accuracy than with a method that would only rely on our available data.

First of all we had to adapt the weighs of the existing model to our data. As it was impossible to work on the full image (pictures are too big, it would have take too long) we have decided to resize pictures. To be sure of the needed size, we used PCA.

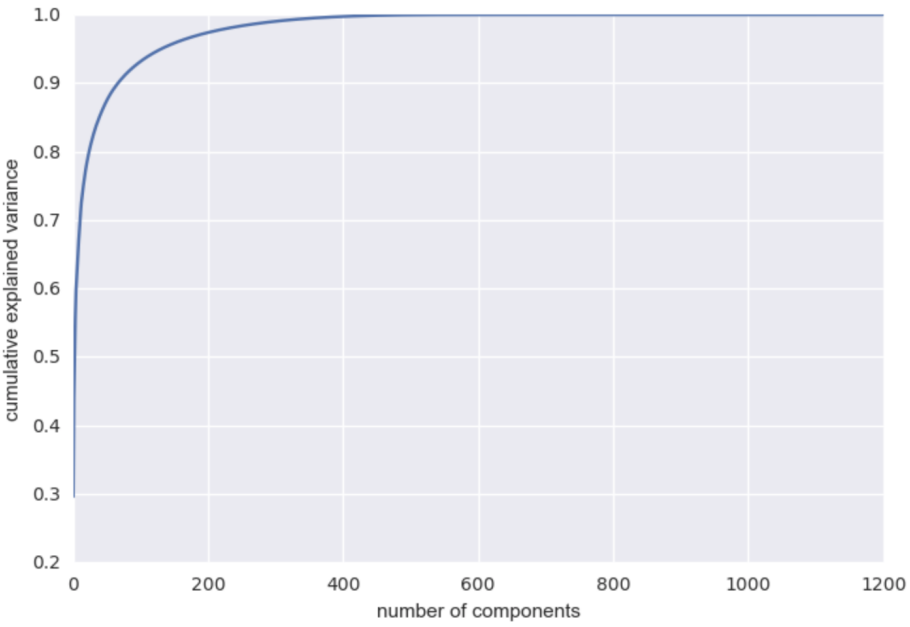

Figure 5: PCA on one picture

From that, we have seen that even if we take a picture of size 75x75, we can still reconstruct more of 90% of the picture.

Once the weights were adapted to our case. We did not need to had many layers, otherwise it would have destroyed the model. We just had to adapt it in order to predict our 12 missing features. That is why we only added the layer '$Flatten()$' because our model was in 3D dimensions and finally the layer '$Dense(12)$ to predict the 12 missing features.

Even if the score was not as high as expected we used these features, because there were quite relevant in the model.

6 Which model to choose ?

6.1 Algorithms

Once our data were cleaned and our additional features built, we decided to implement different algorithms in order to choose our model.

Firstly, we thought that we had to modelize data by a classification model because the scores given to the portraits were integers (discrete). Secondly, thanks to some tips learnt during the course, we realized that we could choose a regression model (continuous one) and round the scores. We began by a linear regression because it is the easiest way to check that our cleaned data are correct. Then, we implemented a polynomial regression which gave us a better validation score and finally we choose the random forest regressor algorithm. We can note that the support vector machine algorithm was too slow so we kept the random forest regressor algorithm.

The Random forest regressor seems to work well on our dataset, however we wanted to try others methods as *boosting* algorithms. So, we tried the Adaboost algorithm, it reduced time but also the score. An other algorithm was the Gradient boosting algorithm. However, these methods did not succeed.

We also tried to do a "VotingClassifier" on Python: we made two random forests, with different parameters, and then we applied "VotingClassifier" on both of them. This is supposed to give the best output combining the two models. However, this did not give a better score than with one random forest.

6.2 Scoring metric

To do this, we used the following evaluation metric given by the website challengedata: Spearman's rank correlation coefficient. Spearman's correlation coefficient measures the strength and direction of association between two ranked variable.

It is defined as the Pearson correlation coefficient between the ranked variables.

Let us denote our labels of our sample $Y = (Y_1, ...Y_n)$ and our predicted labels $Y' = (Y'_1, ..., Y'_n)$, the first step

is to rank it. Our ranked vector are :$Y^r = (Y_{(1)}, ..., Y_{(n)})$ and $Y'^r = (Y'_{(1)}, ..., Y'_{(n)})$. The "rho" coefficient used to predict the score is:

$$\rho_{Y^r, Y'^r} = \frac{Cov(Y^r, Y'^r)}{\sigma_{Y^r} \sigma_{Y'^r}}$$

7 Results

Our first results using simple models and the features available from the data set of the website was around 40.00%. By adding some features and especially the missing features from the neural network, we could improve our score. The best results we get is using a Random forest regressor on our built data set with 125 features for both train set and test set. We tested a GridSearch on our random forest to find the best parameters, the result was a depth of 200. The final score calculated with the Spearman's rank correlation is around 60.00% on our validation sample with all the features and with a random forest of estimators 200.

To understand our results and our model we have computed the features importances and printed the features ranking.

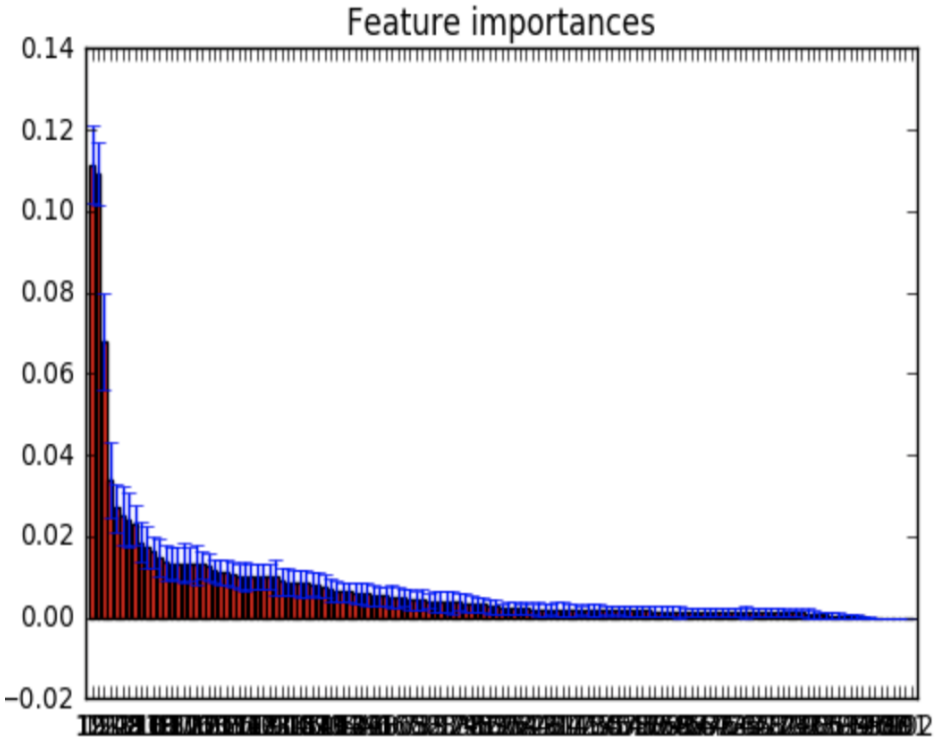

Figure 6: Importance of features in the random forest

From that we can see that the features we have created are quite important because there are appearing in our top 10 features.

1. '*blurry*' (created by us)
2. '*landmarks_confidence*'(given)
3. '*prediction*1' (deeplearning corresponging to *background_impact_n*)
4. '*prediction*0' (deeplearning corresponging *background_impact_p*)
5. '*confidence_gender*'(given)
6. '*prediction*6' (deeplearning corresponging *sharpness_impact_p*)
7. '*height*' (given)

8. '*sharp_right_eyebrow*'(given)
9. '*luminosity*'(created by us)
10. '*detection_score*'(given)

But also we have deleted the less important features in order to improve our score "*stress*""*singing*""*drinking*""*sleeping*""*despise*""*eating*""*crying*""*yawning*""*happiness*".

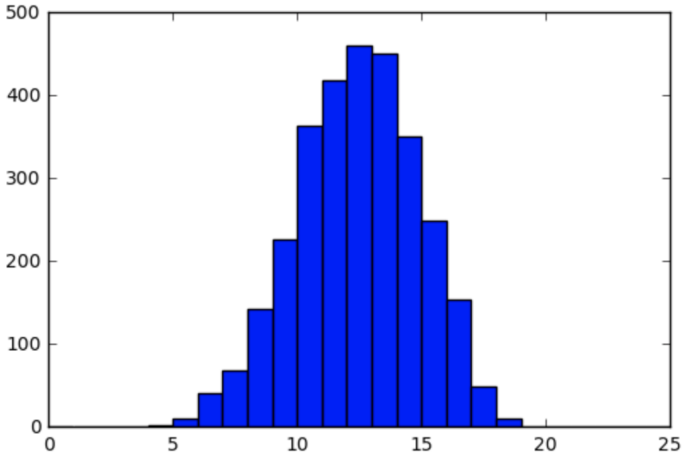

Figure 7: Histogram of predicted values of Y

8 Conclusion

To conclude, the challenge was extremely interesting. We succeeded to take the 6th place just behind the benchmark of Regaind. As we have mentioned in the abstract referencing to Aarushi Gaur and Krystian Mikolajczyk from the University of Surrey, the interaction between features is also important. Maybe there exists better combinations between our features than the one we used. Moreover, we could do different methods using SVMs algorithms and add other features as 'entropy', 'Inverse Difference Moment (IDM)'. We could also add a discriminative color feature which computes the number of each discriminative colors for each portrait.
Last but not least, we hope that during our internship we will learn new techniques (such as multi-neck convolutional network...) which could help us to climb few places on the leaderboard before the end of the challenge in june. May the "source" be with us!

References

[1] Image Analysis and Recognition: 5th International Conference, ICIAR 2008 http://books.google.fr/books?id=Phlog9JD-uQC&pg=PA71&lpg=PA71&dq=Tenegrad+criterion&source=bl&ots=fi7ZuW0KJD&sig=1AyMW-7My50a6xuQ5-jgz3egVq8&hl=fr&sa=X&ved=0ahUKEwjGvenCtvHSAhXH2xoKHVTmDRgQ6AEIIDAB#v=onepage&q=Tenegrad%20criterion&f=false

[2] Calculating the Perceived Brightness of a Color http://www.nbdtech.com/Blog/archive/2008/04/27/Calculating-the-Perceived-Brightness-of-a-Color.aspx

[3] Gray Level Coocurrence Matrix http://www.uio.no/studier/emner/matnat/ifi/INF4300/h08/undervisningsmateriale/glcm.pdf

[4] https://statistics.laerd.com/statistical-guides/spearmans-rank-order-correlation-statistical-guide.php

[5] http://www.pyimagesearch.com/2015/09/07/blur-detection-with-opencv/

[6] http://infolab.stanford.edu/~wangz/project/imsearch/Aesthetics/TMM15/lu.pdf