# 2 Methods

## Problem setting

We cast the current task of determining facial attractiveness as a *supervised learning problem*. Therefore, a successful supervised learning algorithm should be able to, given an input image of a face, provide a reasonable measure of attractiveness, as specified by the training data. Indeed, this is the approach taken by most previous research endeavors (Altwaijry & Belongie, 2013; Fan et al., 2012).

Since attractiveness, considered as the “ground truth” of the task, is not an inherent property of a face, but rather an external value judgement, the labeling of faces is not included *per se*, but has to be performed by a human. Moreover, while most other supervised learning problems have unambiguous assignment of labels (there is little disagreement among human observers whether a cat or a dog is present in a picture), attractiveness has to be judged on an arbitrary scale. Additionally, even though objective criteria for attractiveness or *beauty* can be found across cultures (Little et al., 2011; Jones & Hill, 1993), unique subjective tastes or preferences has to address. Thus, the concept of a *universal attractiveness learner* seems untenable.

A more flexible and useful, albeit computationally extensive, method is to come up with a general architecture of a model, which can be re-trained for each individual observer (e.g., a user in a dating site) given a reasonably large number of input-output pairs in the form of images as inputs and corresponding judgements of attractiveness as outputs. Consequently, this is the approach we will take in the current work.

## Data preparation

In order to gather a sufficiently large, and, at the same time, realistic data set of faces, we used images from two publicly available databases, namely, the 10k US Adult Faces Database (Bainbridge et al., 2013), and the Labelled Faces in The Wild Database (Huang et al., 2007). From these combined sources, we filtered out 650 male and 650 female faces of people aged between 18 – 30 years. Filtering was performed using both an automatic procedure, and visual inspection to correct for labeling errors present in the annotations. Before using the images as input to the model, pixel intensities were normalized to 0 – 1 range. After this, each of the authors rated the images of opposite sex faces on a scale from 0 to 5 (0 indexing the subjectively felt complete lack of attractiveness, and 5 representing the subjective ideal of attractiveness) with a single decimal place allowed for more subtle differentiation of judgement. 50 images were kept separate as a test set, so a total of 600 images were included in the final training set. Histograms of the ratings of the three authors are given in **Figure 1**.

## Data Augmentation

Data augmentation is a popular approach in machine learning and computer vision. It consists of a set of techniques for generating more training data by means of various random transformations applied to already available training samples. Even though this method does not produce genuinely new examples from the true distribution “out there”, it confronts the model with different perspectives on the data and thus enhances generalization performance (Chollet, 2008).

For the current task, we applied the following transformations to the data: images were rotated within a range from 0 to 40 degrees; shifted upwards or downwards by a factor ranging between 0 and .2; sheared by a factor ranging between 0 and .2; zoomed by a factor between 0 and .2; and horizontally flipped with probability .5. All transformation types were applied simultaneously to each image. This ensured that, in each epoch, the model never observed the same image twice.

## Regression versus classification

Before specifying a model to train on the prepared data, one more relevant question had to be resolved, namely, whether to treat the ratings as a continuous or a categorical variable. This decision inevitably determines whether to use a regression or a classification algorithm respectively. Technically, the present operationalization of attractiveness is merely a more fine-grained version of the proverbial Likert scale (Jamieson, 2004). As such, the controversy surrounding the *status quo* of Likert scales applies equally well here. Despite representing a set of ordered categories, data generated by Likert scales are routinely treated as being measured on an interval or an absolute scale. Thus, the most appropriate approach appears to be *ordinal regression* (Cheng, 2007).

Nevertheless, it is certainly debatable whether the underlying construct of interest, i.e. attractiveness, is not optimally represented along a continuous dimension with an arbitrary degree of precision. However, since the focus of the present work is the *practical utility* of predicting facial attractiveness, we can, provisionally, set aside the theoretical issues, and experiment with three different approaches: 1) treating the ratings as continuous real valued numbers and performing ordinary regression; 2) treating the ratings as categories by rounding them to integer numbers (0, 1, 2, 3, 4, 5) without considering any inherent ordering, and performing traditional multiclass classification; 3) treating the ratings as ordered categories (0 < 1 < 2 < 3 < 4 < 5, again, encoded as integer numbers), and performing ordinal regression.

## Using a pre-trained network

Since training an entire convolutional networks from scratch is a costly process that requires both a sufficient amount of computational power and a large enough dataset to capture the relationship between images and ratings, we opted for *transfer learning.* Transfer learning refers to the process of “borrowing” the layers of a previously trained (pre-trained) neural network, typically implemented for a similar task or in a similar domain. Since convolutional networks can be conceptualized as hierarchical feature extractors, the spatial hierarchy of features learned by the pre-trained network can be thought to represent a generic model for the visual world (Chollet, 2018; Goodfellow et al., 2016). Needless to say, this transferability is a unique advantage of deep learning models compared to shallow models.

If a pre-trained network is available for a given task, the next question that arises is whether to use the network as a fixed feature extractor or fine-tune its weights along the new application. Unsurprisingly, this decision depends largely on the peculiarities of the task at hand. As already mentioned above, facial attractiveness can be viewed as a value judgement of different facial features and combinations of features (Little et al., 2011), it is reasonable to suggest that a facial attractiveness classifier would utilize these features during training in order to capture the statistical regularities in a person’s judgement. Therefore, we opted for the option of applying a fixed feature extractor to the current task.

## VGG16-Face

As for the pre-trained convolutional network, we utilized the VGG16-Face architecture (Parkhi, Vedaldi, & Zisserman, 2015). The precise details of the network architecture and development context are described in the relevant paper, so in this subsection we merely sketch out its essential properties.

VGG16-Face war originally trained on 2.6 million images depicting more than 2.6K people in order to perform facial recognition (i.e., identifying a unique person across different images). The task was set up as an N-way classification problem. As the name suggests, the structure of VGG16-Face was inspired by the previous work of Symonyan and Zisserman (2014) who demonstrated the utility of deep convolutional networks in solving large-scale image classification problems, focusing, in particular, on the ImageNet dataset. Similarly, VGG16-Face consists of 11 blocks, each starting with a linear operator, followed by a non-linearity (ReLU), and applying a max pooling operation before passing in the activations to subsequent layers. The last layer of VGG16-Face outputs a 4,096-dimensional vector (considered as an *embedding* of an image), which is then passed through a *softmax* function in order to obtain the class probabilities.

For our current application, we removed the *softmax* layer and *froze* all previous layers, i.e., did not update their weights during backpropagation. This was done in order to prevent the algorithm from modifying the representations learned previously by these previous layers. Thus, we used the image embeddings returned by the last layer of the network as a starting point. The implicit assumption is, therefore, that the 4,096-dimensional vectors of extracted features will contain all the information needed for accurately predicting facial attractiveness. Finally, it is important to note, that using the pre-trained VGG16-Face in this manner has one more important advantage. Since the extracted facial features are considered to be invariant to head orientation, background, lighting and other low-level image properties such as brightness or contrast, the contribution of such factors to predicting attractiveness is vastly minimized. Such feature invariance is of great importance to real-world application, where the distribution of facial images exhibit high levels of natural variance.

## Current architecture

Having outlined the base of our model, we now turn to the expansion of the base model necessitated by the current task. The entire network architecture is illustrated in **Figure 2**. As evident from the graphical representation, the features extracted by VGG16-face were fed into a succession of three fully connected (FC) layers aimed at learning a user’s particular judgements of attractiveness. Additionally, we applied dropout as a regularization technique (Srivastava et al., 2014) to each FC layer. Dropout refers to the operation of multiplying each layer’s activation with a vector , thereby preventing units to overly co-adapt, which, in turn, has been shown to reduce overfitting (Srivastava et al., 2014). In all current experiments, the dropout rate was set to 0.2. We also applied batch normalization to the outputs of each FC layer (Ioffe et al., 2015). Batch normalization refers to the process of normalizing the outputs of network layer , thus essentially normalizing the inputs to layer *.* It is a simple way to prevent units from saturating and has been successfully demonstrated to accelerate training of deep networks (Ioffe et al., 2015). We now describe the output layer of the network, which differed according to which approach was taken (classification, regression, or ordinal regression).

For classification, the output of the last FC layer was passed through a softmax function. Categorical cross-entropy was used as the cost function to be minimized:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

In the above expression, *N* is the number of training examples in a given batch, *K* is the number of classes, refers to the probability of training example *i* belonging to class *k*, and refers to the predicted probability that example *i* belongs to class *k*, and **w** represents the parameters (weights) of the network.

For regression, the output layer was comprised of a single linear unit. Mean absolute error (MAE) was used as cost function:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

For ordinal regression, we used the following approach described by Cheng (2007). The idea is to frame the problem as a multi-label classification task and instead of using one-hot-encoding to represent each category *k* belonging to as a binary vector of dimension *k,* where each component . Thus, the output layer consists of *k* sigmoid units, so a natural choice of cost function is the binary cross-entropy over all sigmoid units:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Where this time, each is an individual Bernoulli variable, as specified above.

It should be noted, however, that the nature of imbalanced data (the authors tended not to give especially generous ratings) requires additional caution. There are many approaches to handle imbalanced data (He & Garcia, 2009). For the current task, we experimented separately with *oversampling* and *cost penalty*, since both appear to have found usage in deep learning applications. Whenever cost penalty was applied, the above cost function definitions simply include class weights which are given by,

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

where is the frequency of class *k*. Thus, the class weights are used to increase the “cost” of misclassifying rare examples.

## Training the network

For each of the three settings, we trained the classifier for 50 epochs using the Adam optimization algorithm (Kingma et al., 2014), with learning rate set to 0.0005. Batch size was 64 across all experiments. We monitored the loss over the epochs and applied early stopping (Yao et al., 2007) when the loss did not improve across five consecutive epochs.

**TODOS:**

INSERT DISTRIBUTION OF RATINGS

INSERT FIGURE OF OUR FINAL NETWORK STRUCTURE

# References

Altwaijry, H., & Belongie, S. (2013). *Relative ranking of facial attractiveness.* Paper presented at the Applications of Computer Vision (WACV), 2013 IEEE Workshop on.

Bainbridge, W. A., Isola, P., & Oliva, A. (2013). The intrinsic memorability of face photographs. *Journal of Experimental Psychology: General, 142*(4), 1323.

Cheng, J., Wang, Z., & Pollastri, G. (2008). *A neural network approach to ordinal regression.* Paper presented at the Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on.

Chollet, F. (2018). *Deep learning with Python*: Manning Publications.

Fan, J., Chau, K., Wan, X., Zhai, L., & Lau, E. (2012). Prediction of facial attractiveness from facial proportions. *Pattern Recognition, 45*(6), 2326-2334.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1): MIT press Cambridge.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering, 21*(9), 1263-1284.

Huang, G. B., Ramesh, M., Berg, T., & Learned-Miller, E. (2007). *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*. Retrieved from

Ioffe, S., & Szegedy, C. (2015). *Batch normalization: Accelerating deep network training by reducing internal covariate shift.* Paper presented at the International conference on machine learning.

Jamieson, S. (2004). Likert scales: how to (ab) use them. *Medical education, 38*(12), 1217-1218.

Jones, D., & Hill, K. (1993). Criteria of facial attractiveness in five populations. *Human Nature, 4*(3), 271-296.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Little, A. C., Jones, B. C., & DeBruine, L. M. (2011). Facial attractiveness: evolutionary based research. *Philosophical Transactions of the Royal Society B: Biological Sciences, 366*(1571), 1638-1659.

Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). *Deep Face Recognition.* Paper presented at the BMVC.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research, 15*(1), 1929-1958.

Yao, Y., Rosasco, L., & Caponnetto, A. (2007). On early stopping in gradient descent learning. *Constructive Approximation, 26*(2), 289-315.