Prediction of Attributes

Anna Canal Garcia, Marine Collery, Sandra Pico

KTH Royal Institute of Technology

Abstract. This project attempts to perform a comparison and evaluation between one pre-trained model with two different fine-tuning and two implemented CNN architectures (Cascade CNN and TCDCN architecture) for face-image classification. The studied attributes are gender (male/female) and smiling (y/n) from the Multi-Task Facial Landmark (MTFL) dataset. The metrics used are accuracy, precision and recall.

Keywords: Deep Learning, Faces attributes, pre-trained models,CNN

1 Introduction

Deep learning methods, and specially deep Convolutional Neural Networks (CNN), have achieved remarkable successes in several Computer Vision tasks during the recent years. The purpose of this project is to compare different Convolutional Neural Networks approaches to predict some of the Multi-Task Facial Landmark attributes.

The Multi-Task Facial Landmark (MTFL) dataset [1] is composed by several labelled face-images. Explicitly, each image is annotated with: *1)* 5 facial landmarks, *2)* Gender attribute (male/female), *3)* Smiling attribute (yes/no), *4)* Wearing glasses (yes/no), and *5)* Head pose (left profile/left/frontal/right/right profile).

The dataset is divided into a training and a testing set. The training dataset is composed of 5590 face-images in the wild (LFW) and 7876 other images downloaded from the web (Net), while the testing set gathers 2995 images of the Annotated Facial Landmarks in Wild (AFWL) database.

In this project, only gender and smiling attributes will be predicted. Pre-trained models and own deep CNN architectures based on results shown in [2] will be used in order to obtain these predictions. Accuracy, precision and recall will be the main metrics to extract conclusions and evaluate the achieved results.

2 Background

Most face recognition problems consist in finding solutions for the following two tasks: extracting features and using these features to classify faces as presented in [3] for example. Facial landmarks are fundamental components in many face recognition tasks and provide crucial information for classification or regression

problems. Nevertheless, as pointed out in [1], nowadays it still remains a challenge when partial occlusion and large head pose variations occur in images .

Traditionally, facial landmark detection was addressed as a single and independent problem, but recently, deep models have also been applied. In Sun et al. [4] a coarse-to-fine regression using a cascade of deep CNN is presented in order to estimate the position of facial key points. The method used showed superior accuracy compared to previous methods but it required a complex cascade architecture of a deep model.

Later, in 2014, Zhang et al. [1] proposed a Tasks-Constrained Deep Convolutional Network (TCDCN), the first attempt to investigate how facial landmark detection can be optimized together with heterogeneous but subtly correlated tasks. The method aims to optimize facial landmark detection with a set of auxiliary/related tasks, that include head pose estimation, gender classification, age estimation, facial expression recognition or facial attribute inference. Since different tasks are inherently different in learning difficulties, they implemented a task-wise early stopping criterion to facilitate learning convergence by using different learning and convergence rates. The updated version of this model [5] was taken into consideration in order to build one of the CNN approaches in this project. It takes a 60x60 image as input. The feature extraction stage contains four convolutional layers, three pooling layers and one fully connected layer; where a different set of filters is applied at every location in the input map. The fully connected layer produces a feature vector which is shared by the multiple tasks in the estimation stage. Then the output is obtained by regression.

Coming back to cascade of deep CNN, in 2014 Sun et al. [6] showed that effective feature representations can be solved with deep learning by using both face identification and verification signals as supervision. The Deep IDentification-verification features (*DeepID2*) are learned with carefully designed deep convolutional networks. The CNNs have four convolutional layers (with 20, 40, 60 and 80 feature maps), followed by max-pooling, a 60-dimensional fully-connected layer and a softmax layer for the identities. The higher convolutional layers have locally shared weights. The final accuracy for faces identities recognition is of 97.45 $\pm$ 0.26%. Later, in *DeepID2+* version [7] they increased the number of feature maps to 128 in the four convolutional layers, the size of fully connected layer to 512 dimensions and expanded their training set. This resulted in a higher accuracy: 99.47 $\pm$ 0.12%. Finally, they presented *DeepID3* [8], a deeper network with ten to fifteen non-linear extraction layers with Inception layers and successive convolutional layers without any pooling layer in between on a similar overall pipeline to *DeepID2+*. This third version achieved an accuracy of 99.53 $\pm$ 0.10%.

On the other hand, publicly available pre-trained deep learning models optimized on huge dataset (for example ImageNet [9]) granted recently anyone with a normal computer the power to classify any small dataset. It has even be proven in [10] that fine-tuning a pre-trained model provides better generalized results than a complete training on a small dataset (with weights randomization).

3 Approach

3.1 Method

All the code can be found in our git repository `https://github.com/MarineCollery/DeepLearningProject`.

Pre-Processing Images are cropped in order to have the same dimensions in all the used datasets (LFW, Net and AFWL). This is achieved by using the nose landmark as the center of the image and cropping from its position. Then, the input size of the following implemented networks is a 150x150 pixels.

Evaluation The models are evaluated with the following metrics: accuracy, precision and recall. Those three combined provide us information on the global success of the model, how much selected items are relevant and how much relevant items are selected.

Technical Tools

- **Google Cloud Platform**: The main platform used to compute and train the implemented networks is a Google Cloud Virtual machine. Our server is a n1-standard-16 machine, based on 16 CPU's and 60GB of memory.
- **Parallel Data Center(PDC) - Center for High Performance Computing**: TEGNER computer is used to fine-tune pre-trained models. We use K80 GPUs.

All the code can be found in our git repository `https://github.com/MarineCollery/DeepLearningProject`.

3.2 Models

Here we compare different CNN approaches to predict some of the facial Landmark attributes (smiling and gender attributes). The first approach is to use a pre-trained model and adapt it to our dataset. The second one is to implement our own CNN and train it with our dataset.

The smiling and gender attributes are tested independently.

3.3 Pre-trained Model

The pre-trained model InceptionV3 [11] is tested on both attributes (smiling and gender) independently with two different methods. We also train on the original dataset and the cropped one presented in 3.1 Inception V3 [11] is a model trained on the ImageNet [9] Dataset composed of 1000 different classes. The input model format of the model is 299x299, some image processing is applied to get a plausible image sized. However, images are not distorted during training (or testing) for both approaches due to computational complexity.

The structure of the InceptionV3 network is shown in Fig. 1.

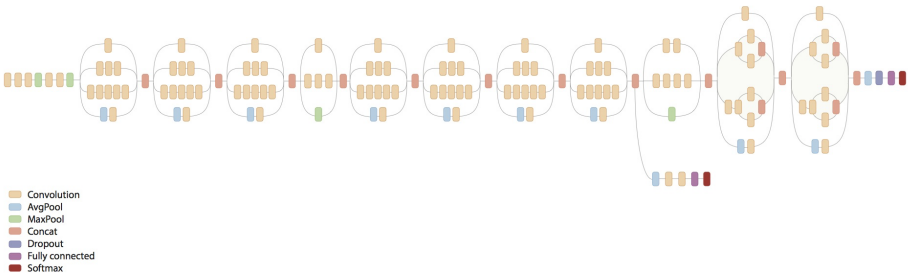

Fig. 1: Visualization of the Inception V3 model architecture. Source `https://github.com/tensorflow/models/tree/master/research/inception`

Bottlenecks computations approach The first approach is to retrain the model with reasonable parameters and evaluate its efficiency on the test set by computing for each image of our dataset the layer just before the final output layer (called 'bottlenecks') and then train the softmax top layer with the new classes. This means that the weights are not recalculated and therefore the computational complexity is reasonable. This approach is inspired by the Tensorflow [12] tutorial, that you can found here. The parameters used are described in Fig.1

Table 1: Parameters for the bottlenecks computation and retraining

Parameters	Values
Learning rate	0.01
Train batch size	100
Number of training steps	4000
Training steps evaluate every	10
Validation percentage	10%

Complete fine-tuning approach A second approach was to fine-tune the model by retraining completely the last two layers of the Pretrained InceptionV3 model (those layers are called "InceptionV3/Logits" and "InceptionV3/AuxLogits"). This means that the weights of the model are recalculated for those layers, the rest will remain frozen. Then after evaluating the model, all the layers are fine tuned for an other 500 steps. Evalutations on the validation set (10%) are not used to evaluate the model but only to supervise the learning. The steps are described in Fig.2.

This approach was based on the tutorial presented here.

Table 2: Description of the complete fine-tuning approach

Step	Parameters	Values	Description
1	Initial learning rate Batch size Optimizer Weight Decay	0.01 100 RMSprop 0.00004	Fine-tune only the new layers for 1000 steps.
2	Batch size	100	Evaluation on validation set
3	Batch size	100	Evaluation on test set
4	Initial learning rate Batch size Optimizer Weight Decay	0.001 100 RMSprop 0.00004	Fine-tune all layers for 500 steps.
5	Batch size	100	Evaluation on validation set
6	Batch size	100	Evaluation on test set

3.4 CNN implementation

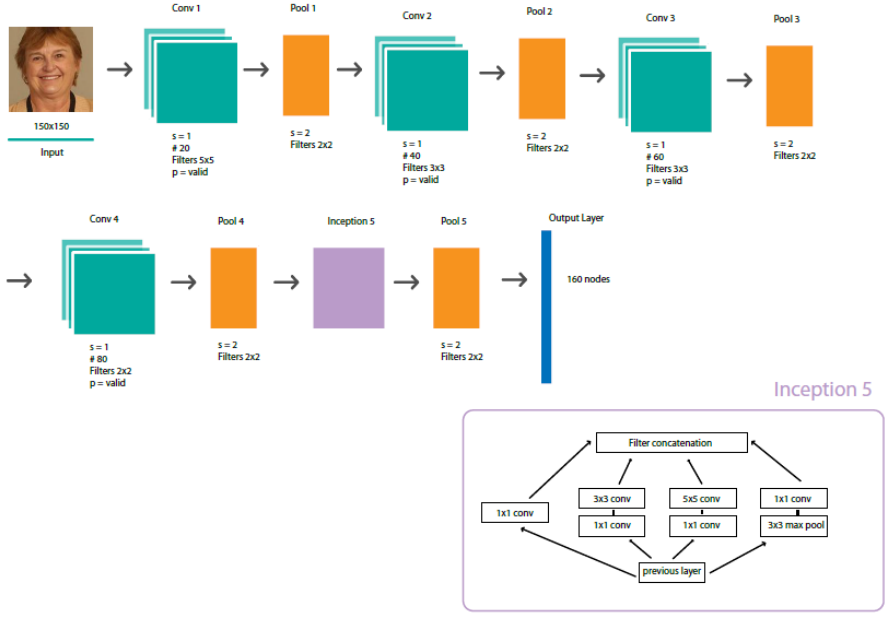

Fig. 2: Visualization of the Cascade CNN model architecture.

Two different ConvNet models are built in order to train and test on our datasets. The training set contains images from LFW and Net datasets. We split this data into training set (80%) and validation set (20%) in order to check the performance of the model. Both approaches were coded in python using Tensorflow [12] and following a Tensorflow tutorial, which can be found here.

Cascade CNN implementation Cascade of CNN with four convolutional layers, four max-pooling layers, one inception layer and one fully connected layer. The commonly used rectified linear unit (RLU) is selected as the activation function. This architecture has the same number of convolutional layers with the same characteristics (size and number of filters, stride, etc) of *DeepID2+* [7] with the addition of an inception layer, type of layer used in *DeepID3*. It does not implement local convolutional layers as in *DeepID2+* or *DeepID3* since is not possible in TensorFlow. The network architecture is shown in Fig.2.

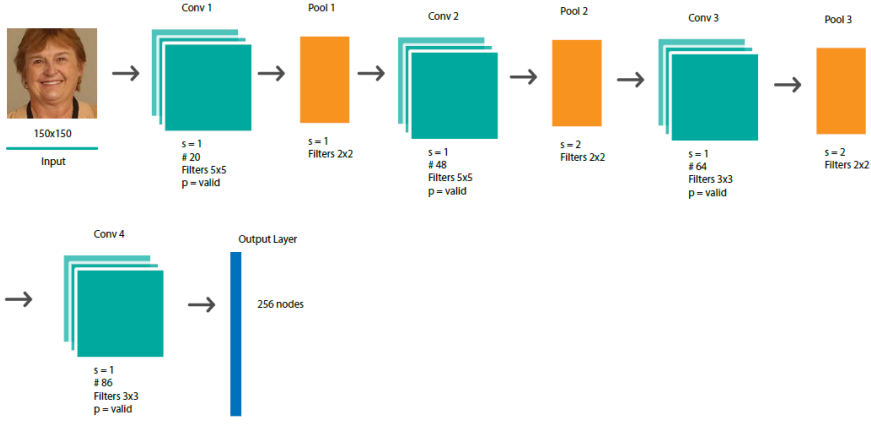

Fig. 3: Visualization of the TCDCN model architecture.

TCDCN implementation Based on the implementation of Zhang et al. [5], the feature extraction stage contains four convolutional layers, three max-pooling layers and one fully connected layer. It also uses RLU as the activation function. The fully connected layer produces a feature vector of 256 dimensions. The network architecture is shown in Fig.3.

4 Experiments/Results/Conclusions

4.1 Pre-trained models

Both approaches presented in 3.3 resulted in promising results.

As shown in Table.3 and Table.4, accuracies for the gender attribute are definitely higher than for the smiling attribute. This was expected as smiling is more subjective than the gender attribute. This is shown via examples of misclassified test images in Fig.4.

Fig. 4: Example of misclassified images for the smile attribute

Regarding the smiling attribute, Table.3 shows that training the model with cropped images centered on the nose does not necessarily give better results than with the original files for training (the files are still cropped but not centered on the nose). The precision of the smiling attributes actually gets up to 97.6% with the second approach and the original sized pictures as input. The second approach seems to be providing better results for this attribute even though the Non-Smiling (NS) state is neglected.

Table 3: Results comparison on the test set for both approaches for the smile attribute (* stands for training on cropped images as described in 3.1)

Attribute	Metrics	First approach					Second approach				
		Values		Confusion Matrix			Values		Confusion Matrix		
		S	NS				S	NS			
Smile	Accuracy	68.5%			S	NS	87.5%			S	NS
	Precision	64.1%	73.8%	S	64%	36%	97.6%	75.4%	S	98%	2%
	Recall	75.6%	63.2%	NS	26%	74%	82.6%	96.3%	NS	25%	75%
Smile*	Accuracy	70.1%			S	NS	72.5%			S	NS
	Precision	63.4%	78.4%	S	63%	37%	86.7%	55.4%	S	87%	13%
	Recall	77.9%	64.1%	NS	22%	78%	69.9%	77.7%	NS	45%	55%

Regarding the gender attribute, as shown in Table.4 the nose-centered cropping does not have the a lot of influence on the results for each model. This could actually be explained by the fact that the gender attribute is definitely

not only defined by the mouth but by the whole face and head. The cropping might therefore be less significant to model this attribute. Here we can also see that the results differences are not accuracy related, however first and second approaches come with radically different interpretations. The first approach is distinguishing Male (M) and Female (F) with comparable precision and recall whereas the second approach tends to predict with a lot more precision Male attribute than Female one. The first approach with the original sized input pictures is therefore the best model for this attribute.

Table 4: Results comparison on the test set for both approaches for the gender attribute (* stands for training on cropped images as described in 3.1)

		First approach					**Second approach**				
Attribute	Metrics	Values		Confusion Matrix			Values		Confusion Matrix		
		F	M				F	M			
Gender	Accuracy	86.3%			F	M	83.5%			F	M
	Precision	87.1%	85.2%	F	87%	13%	76.9%	94.3%	F	77%	23%
	Recall	90.1%	81.0%	M	15%	85%	95.4%	72.4%	M	6%	94%
Gender*	Accuracy	86.1%			F	M	85.1%			F	M
	Precision	88.5%	82.4%	F	88%	12%	80.8%	91.7%	F	80%	19%
	Recall	88.6%	82.1%	M	18%	82%	93.8%	75.5%	M	8%	92%

Both approaches provided relatively high accuracies but are differentiable by their precision and recall. The first approach tends to provide balanced results between between possible issues in opposition to the second approach that tends to lean towards a certain state.

4.2 Implemented models

The performances achieved with different learning rates and number of epochs for both approaches explained in 3.4 are evaluated and compared.

Cascade-CNN implementation Different hyper-parameters are tested to train the model using gender attribute dataset. The validation accuracy is used to select the best hyper-parameters for the following computations.

– **Hyper-parameters selection**
  - *Learning rate*
    From results achieved with the learning rates: $1e^{-3}$,$1e^{-4}$ and $1e^{-5}$ all for 63 epochs, only the learning rate $1e^{-3}$ and $1e^{-4}$ are kept for the following model trainings.

- *Number of epochs*
  Different number of epochs for the prediction of the gender attribute are also tested. With learning rate fixed to $1e^{-3}$, results for 25,63,125 and 250 number of epochs only 63 and 125 number of epochs are kept for the following model trainings.

– **Evaluation on the test dataset**
With the previous mentioned configurations, the results achieved for the *Smiling* and *Gender* attributes on the test set are described in Table.5

Table 5: Results on the test dataset of the Cascade-CNN architecture with different configurations.

Attribute	Learning Rate	Epochs	Accuracy	Precision	Recall
Gender	$1e^{-3}$	63	71.4%	79.4%	60.2%
		125	74.6%	73.8%	65.7%
	$1e^{-4}$	63	73%	77%	62.7%
		125	75%	75.4%	65.8%
Smile	$1e^{-3}$	63	67.2%	71.5%	69.3%
		125	67.6%	68.3%	71.1%
	$1e^{-4}$	63	71.4%	72.4%	74.4%
		125	71.5%	71.4%	75.1%

A quick observation of the results exposed in Table5 tells us that:

- Learning rate $1e^{-4}$ gives as expected better results than learning rate $1e^{-3}$.
- Better accuracy is obtained for the gender attribute than for the smiling attribute.
- More accurate models are built with 125 epochs than with 63 but the precision decrease at the same time.

Based on the previous conclusions, it is quite expected to achieve more accuracy in the gender attribute as it is less subjective as shown in Fig.4. At the same time, taking into account the results obtained more effort should be invested to continue analyzing more configurations around a learning rate $1e^{-4}$ and 125 number of epochs.

TCDCN implementation Regarding the second CNN architecture, although it seemed to be an easier model to train and test, due to the amount of nodes that the model have in the last layer, it takes a lot of time to train and achieve results.

For all the mentioned reasons, we could not test a lot of models for the TCDCN architecture. Only one model with learning rate $1e^{-4}$ and 45 epochs was built for the Smiling attribute.

The model, that achieved 72.5% of accuracy in test and 86% in the validation set, demonstrate a better performance than the Cascade-CNN architecture. Definitely, as a future work, more tests and models should be tested with this second implementation.

5 Conclusion and future work

This project has shown us that deep Convolutional Neural Networks provide good performance for facial attributes classification. This assignment was an interesting way to realize how deep learning methods are able to approach a real-life problem as this work was all about.

At the same time, even though the pre-trained models seems to be more accurate, the implemented architectures have also achieved promising and proper results. We believe that with more tests and variations for the hyper-parameters configurations we could improve the results obtained. TCDCN architecture seems to be a good way to achieve considerable improvement.

Furthermore, investigation regarding the difficulty per each analyzed attribute should be done. It has been proven that smiling attribute implies another level of complication: subjectivity.

Both for the pre-trained and implemented model, a simple but computationally demanding improvement that could provide even better models would be it add random distortions at each training step. This method as prove its efficiency in different studies and would be an interesting path to follow.

To finalize, even though good results were obtained, it is important to state that a lot of future work could be done. Analyzing the other possible attributes labeled in the MTFL dataset could also give us more intuition that probably each attribute prediction would need a specific model configuration. Following the same path, a model analyzing all the attributes together could also be another good approach.

References

1. Zhang, Z., Luo, P., Loy, C.C., Tang, X.: Facial Landmark Detection by Deep Multi-task Learning. In Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., eds.: Computer Vision ECCV 2014. Volume 8694. Springer International Publishing, Cham (2014) 94–108
2. Learned-Miller, E., Huang, G.B., RoyChowdhury, A., Li, H., Hua, G.: Labeled Faces in the Wild: A Survey. In Kawulok, M., Celebi, M.E., Smolka, B., eds.: Advances in Face Detection and Facial Image Analysis. Springer International Publishing, Cham (2016) 189–248
3. Leng, B., Liu, Y., Yu, K., Xu, S., Yuan, Z., Qin, J.: Cascade shallow CNN structure for face verification and identification. Neurocomputing **215** (November 2016) 232–240
4. Sun, Y., Wang, X., Tang, X.: Deep Convolutional Network Cascade for Facial Point Detection, IEEE (June 2013) 3476–3483
5. Zhang, Z., Luo, P., Loy, C.C., Tang, X.: Learning Deep Representation for Face Alignment with Auxiliary Attributes. IEEE Transactions on Pattern Analysis and Machine Intelligence **38**(5) (May 2016) 918–930 arXiv: 1408.3967.
6. Sun, Y., Wang, X., Tang, X.: Deep Learning Face Representation by Joint Identification-Verification. arXiv:1406.4773 [cs] (June 2014) arXiv: 1406.4773.
7. Sun, Y., Wang, X., Tang, X.: Deeply learned face representations are sparse, selective, and robust. arXiv:1412.1265 [cs] (December 2014) arXiv: 1412.1265.
8. Sun, Y., Liang, D., Wang, X., Tang, X.: DeepID3: Face Recognition with Very Deep Neural Networks. arXiv:1502.00873 [cs] (February 2015) arXiv: 1502.00873.
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. (June 2009) 248–255
10. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., eds.: Advances in Neural Information Processing Systems 27. Curran Associates, Inc. (2014) 3320–3328
11. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567 [cs] (December 2015) arXiv: 1512.00567.
12. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015) Software available from tensorflow.org.