Effects of Variability in Synthetic Training Data on Convolutional Neural Networks for 3D Head Reconstruction

Jan Philip Göpfert
CITEC
Bielefeld University
Bielefeld, Germany
jgoepfert@techfak.uni-bielefeld.de

Christina Göpfert
CITEC
Bielefeld University
Bielefeld, Germany
cgoepfert@techfak.uni-bielefeld.de

Mario Botsch
CITEC
Bielefeld University
Bielefeld, Germany
botsch@techfak.uni-bielefeld.de

Barbara Hammer
CITEC
Bielefeld University
Bielefeld, Germany
bhammer@techfak.uni-bielefeld.de

***Abstract*—Convolutional neural networks have recently shown great success in computer vision. They are able to automatically learn complicated mappings, often reaching human or super-human performance. However, a lack of labeled data can preclude the training of such networks. This is the case in the reconstruction of 3-dimensional human heads from 2-dimensional photographs. Approaching the problem backwards, starting from 3-dimensional heads and using photo-realistic rendering, one can create any number of training data to tackle the problem. This way, fine control over the data allows for new insights into how a convolutional neural network interprets data and how variability in the training and test data affect its performance. We perform a systematic analysis in order to determine how the presence of different types of variability in the training data affects the generalization properties of the network for 3-dimensional head reconstruction.**

***Index Terms*—convolutional, neural, network, reconstruction, interpretability**

I. Introduction

Virtual human heads that are 3-dimensional representations of actual people's heads have applications in fields as diverse as entertainment, business and security: They are used in movies to create the appearance of actors taking part in computer generated scenes. In video games – especially in online multiplayer experiences – they can heighten the sense of immersion and community. In virtual conference rooms, they enable more direct and comfortable communication. Finally, in video surveillance, they can increase robustness and performance. [1]

Creating a virtual copy of an actual, real head, is no trivial process. While it is possible for an artist to manually create such a head, this is difficult and time consuming. Modern scanning pipelines employ expensive, multi-camera setups to take a set of photographs simultaneously from different angles. These photographs are used to generate a point cloud, which is then approximated by a 3-dimensional mesh in a computationally expensive procedure. [2]

In recent years, convolutional neural networks have experienced a surge in success and popularity, mostly due to advances in hardware, and the availability of training data for classification problems. [3] They have rivaled and at times exceeded human performance in classification tasks. [4, 5, 6] If there were enough training data available, one might be able to train a neural network to re-create human heads in virtual space.

Recently, there have been advances in generating synthetic data, i.e. photo-realistic renderings, to train neural networks that solve related tasks, e.g. face scanning and garment scanning. [7, 8, 9, 10, 11] Unfortunately, theses networks are little more than black boxes and it remains unclear what effects the variabilities introduced into the training data have and whether they suffice to achieve acceptable levels of generalization. Furthermore, they depend on inputs preprocessed by other means, specifically on images where the object of interest has already been separated from the background, or on detected points of interest, e.g. the eyes, mouth, or nose; even though these are tasks that convolutional neural networks have been shown to excel at. [12, 13] Lastly, those networks that replicate human faces, are limited to exactly that – they only consider a small portion of the human head, and only so that it may, in the end, be used to re-create the input image, but not an animation or virtual avatar. We believe that considering the entire head is especially interesting, since it can never be shown in its entirety in a single image, and so a network must learn to predict correlations between visible and invisible parts of the human head.

This research/work was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

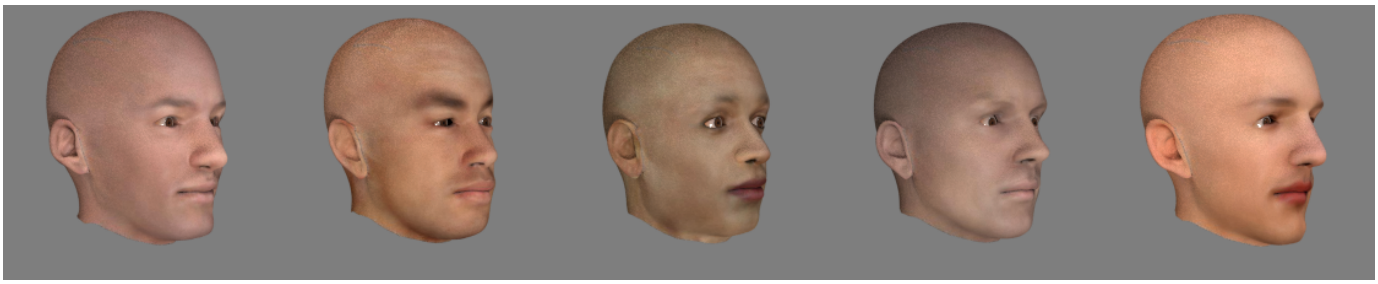

Figure 1: Samples from the training data generated generated according to the scenario *none*.

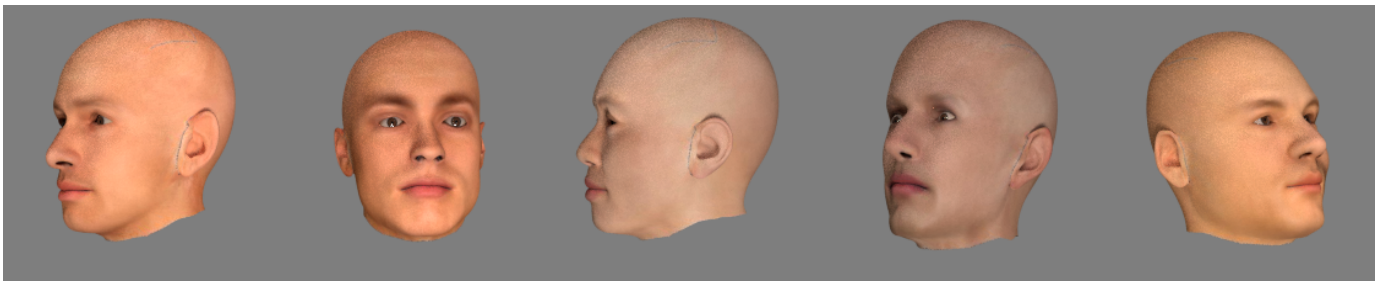

Figure 2: Samples from the training data generated generated according to the scenario *angle*.

In photo-realistic renderings, as opposed to real-world photographs, we have complete control over the images, from the high-level content to modalities such as background, lighting and lens properties. In this paper, we take advantage of the fact that neural networks can successfully be trained on synthetic data, by systematically analyzing how changes in the training data affect a network's ability to generalize, and use this to further our understanding of how invariances are learned by the network. Furthermore, we investigate how well a solution that is truly end-to-end – i.e. takes an image as input that has not been specially preprocessed and outputs a virtual head – fares with regards to changes in the variability present in its input.

The remainder of this paper is structured as follows: In Section II, we present a gentle introduction of how heads are represented in Computer Graphics, before outlining in Section III how the task of reconstructing a 3-dimensional head from a photograph can be solved by deep learning. In Section IV, we describe how photo-realistic renderings can be used to systematically analyze the generalization properties of a network with regard to different types of variability. In Section V we present our experimental results, and the paper concludes with a discussion and outlook in Section VI.

II. Representation

In order to turn a 2-dimensional image of a head into a 3-dimensional head, we first need to determine how to represent the head in 3 dimensions. For this, we choose a polygonal mesh with which we approximate the head's surface. The shape of the mesh is determined by the positions of its vertices in $\mathbb{R}^3$, represented by a vector $v \in \mathbb{R}^{3n}$ where n is the number of vertices in the mesh. We call such a vector v a *configuration* of the mesh. Figure 5 shows one possible mesh configuration. Different heads are given by different configurations of the mesh, and we require that for all configurations, the semantics of each vertex should be the same, e.g. a vertex on the tip of the nose should always remain on the tip of the nose. This ensures – among other advantages – that different heads can be mixed to obtain intermediate heads. More to the point, if $v_1, \ldots, v_k$ are configurations that represent different heads, a convex combination of the v_i typically also represents a plausible human head.

Directly modifying vertex positions to create a new mesh can be problematic due to the high dimensionality of the configuration: when the mesh has n vertices, its shape is determined by a vector $v \in \mathbb{R}^{3n}$, so a value of around $n = 25\,000$ results in around $75\,000$ degrees of freedom. On the one hand, this leads to computational difficulties. On the other hand, it makes mesh deformations extremely challenging for humans to understand. Both problems can be alleviated by a parameterized head model that determines a mesh configuration from only a couple of input parameters. A popular method to obtain such a model on the basis of sample configurations $v_1, \ldots, v_k$ is to apply Principal Component Analysis (PCA) on the configurations v_i in order to find m_v principal directions in which the meshes differ from their mean, where $m_v \leq k$. Then, a new configuration is defined by m_v parameters $p_v \in \mathbb{R}^{m_v}$:

$$v = \mu_v + X_v \cdot p_v, \tag{1}$$

where μ_v denotes the mean of $v_1, \ldots, v_k$ and X_v denotes their first m_v principal components. These principal components represent typical variations between human heads.

The mesh representation described so far does not contain color information. This information is typically given by a

Figure 3: Samples from the training data generated generated according to the scenario *angle/bg*.

Figure 4: Samples from the training data generated generated according to the scenario *all*.

texture map, which is rendered on top of the 3-dimensional mesh. Since textures are represented by images with hundreds of thousands of pixels, it is beneficial to create a texture model using the same PCA-based method as used for mesh configurations – plausible textures are given by

$$t = \mu_t + X_t \cdot p_t, \tag{2}$$

where μ_t is the mean of a set of known textures, X_t are the first m_t principal components, and $p_t \in \mathbb{R}^{m_t}$.

This method has been pioneered for human faces by Blanz and Vetter [14].

We refer to the concatenation of p_v and p_t as $p \in \mathbb{R}^m$, so $m = m_v + m_t$. Together with the model given by μ_v, X_v, μ_t and X_t, these m parameters fully define a 3-dimensional head.

III. Reconstruction by Machine Learning

Using the head model introduced in Section II, we can phrase the problem of 3-dimensional head reconstruction as a regression problem, where the input is a photograph of a human head and the output is a vector $p = (p_v, p_t)$ such that Equations (1) and (2) are good approximations of the shape and texture of the head in three dimensions, respectively.

Convolutional neural networks have been shown to excel in problems where information is extracted from images. [15, 16, 17] Unfortunately, they require large amounts of training data, which is hard to acquire in this case. Approximating ground-truth 3-dimensional representations using photogrammetry, which is the state of the art approach, requires a three-step processing pipeline: photographs of the subject are taken simultaneously from multiple angles, using an expensive multi-camera setup, where the subject must be physically present. The photos are used to produce a point cloud, which is then approximated by a mesh. Both of the latter steps are computationally expensive. [18, 2] Furthermore, privacy concerns may limit the number of people willing to supply this type of data. Together, these considerations make the collection of a sufficient amount of training data prohibitive to training a convolutional neural network on the regression problem described above.

A recent solution to problems of this type is to use artificially generated images as a training set. [7, 8, 9, 10, 11] In our head reconstruction application, this entails the following steps:

1) Randomly select a target parameter vector $p \in \mathbb{R}^m$.
2) Generate one or more photo-realistic images of the mesh and texture described by p.
3) Add the generated images to the training set with the vector p as the target.

In Figure 6, we visualize the connections between parameters, configurations and renderings.

IV. Structured Investigation of Invariances

While neural networks have shown admirable performance in diverse computer vision tasks, the resulting models are hard to interpret, and it usually remains unclear how sensitive they are to variation not present in the training data. For example, the model may eventually be deployed on pictures taken under different lighting conditions or with new cameras, leading to variability in application that the model was not trained on.

This point becomes even more pronounced when training on synthetic images: it is generally not feasible to model in the training data all types of variability that the network may encounter when deployed. Since the model should still perform well under real-world conditions, it needs to be robust to the types of variation not accounted for in the training data, but present in real-world images.

Fortunately, photo-realistic rendering also opens the door to a structured investigation of these questions. Because the images are artificially generated, we have full control over all types of variation that occur in the training data – as opposed to real-world photographs, where completely controlling conditions such as angle and lighting requires sophisticated setups. We propose to exploit this advantage by comparing the performance of neural networks trained on datasets that differ only regarding isolated types of variability. By considering several networks, trained to solve the same task, but with an exposition to different types of variability during training, we are able to investigate

- how different types and levels of variation affect a network's performance,
- how a network extrapolates what it has learned to variability unseen during training, and
- which types of variability absolutely need to be introduced into training data or controlled for in deployment.

In the following, we present our results for convolutional neural networks trained to estimate the parameters of a 3-dimensional head model from a single rendering of a human head. The types of variation introduced include lighting, angle, position, lens properties, background, and hair.

V. Experiments

Data Synthesis and Training

For our experiments, we create a model, as described above, which takes $m = 36$ parameters as input and outputs a mesh with a texture. For any sample, we draw a ground-truth vector q uniformly from $[-1, 1])^m$, which we store as the ground truth and refer to as the *unscaled* parameters. We multiply each entry in q by 1.6 times the respective principal component's standard deviation to obtain parameters p for the model. Note that the principal components, i.e. the columns in X_v and X_t, are of unit length. Thus, in order to regenerate the approximate volume spanned by the original data, multiplication with the standard deviation in necessary. Further multiplication by 1.6 slightly extends said volume, increasing the expressiveness while retaining realism.

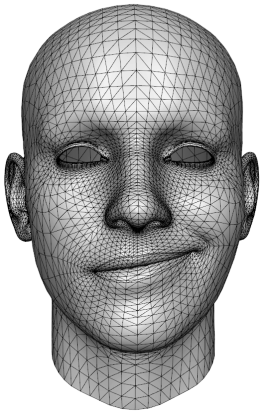

Figure 5: A mesh configuration of a human head.

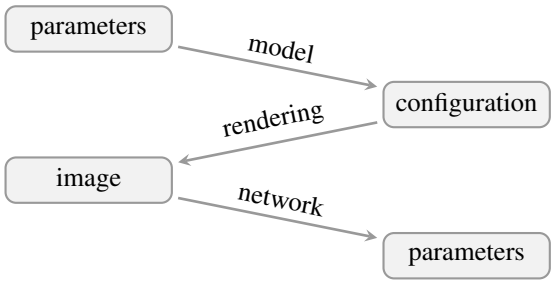

Figure 6: Overview over the pipeline. The head model is used to turn (random) parameters into configurations. These are used to render images. During the rendering, further variability is introduced. A convolutional neural network is trained on the rendered images, so that it learns to estimate parameters for a given input image.

To draw the images, we setup a scene in Blender [19] and render using Cycles. We introduce variability into the data by modifying certain characteristics according to the following scenarios, which we name by the variability they encompass:

- *none*: All heads are rendered from the same angle (roughly $45°$) in front of a a solid gray background. See Figure 1.
- *angle*: Heads are drawn from angles randomly picked between $90°$ to the left and right, in front of the same solid gray background as in the scenario *none*. See Figure 2.
- *angle/bg*: Heads are drawn like in the scenario *angle*, but overlayed on top of randomly scaled parts of randomly chosen photographs. See Figure 3.
- *all*: In addition to variability in the scenario *angle/bg*, heads are randomly drawn with or without facial hair and/or scalp hair, the light color and intensity are changed randomly, the camera lens's focal length is changed randomly, and the head is translated randomly, varying its position in the image plane and its distance from the camera. See Figure 4.

For each scenario, we create around $200\,000$ samples and then train a neural network to estimate the unscaled parameters q with a single rendered image as input. Each neural network is based on the architecture of ResNet-50 [20] with 50 parameter layers, where the last layer is replaced by a fully connected layer with m output neurons, one for each parameter. The loss is calculated as the squared euclidean distance between the estimated and the actual parameters. Training is performed for $1\,000\,000$ iterations using Caffe [21], starting with the original weights available for ResNet-50. The learning rate starts at 0.0001 and is reduced via an inverse decay with gamma 0.1 and a power of 0.75. Additionally, a momentum of 0.9 is used. We render images as 224 by 224 pixels to adhere to the original network and subtract its mean image from each input. Note that even though ResNet-50's weights were obtained solving a classification problem [3], the early layers do not change noticeably during training.

The configuration and texture are calculated from the un-

scaled parameters in a simple and direct manner, so we calculate the error directly on the unscaled parameters. This allows us consider configuration and texture as one.

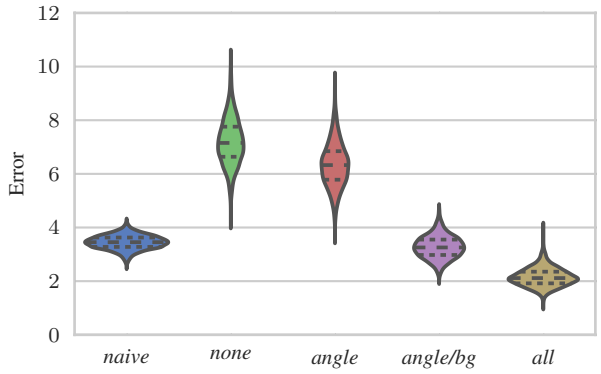

Figure 7: Distributions of errors for each net, applied to 2000 unseen images from the scenario *all*. A pseudo-network dubbed *naive* is included that always predicts zero for every parameter (recall that they are sampled uniformly between -1.0 and 1.0), to serve as a baseline. The presented violin plots are similar to more common box plots but include an approximation of the samples' histograms. The horizontal lines are the median and quartiles.

Results

We refer to each network by the name we have given the training data, i.e. the scenario the network has trained on.

General Performance: To compare the networks and judge how they generalize to variability they have not seen during training, they are presented with 2000 images from the scenario *all* and the error is logged. We present the resulting error distributions in Figure 7. The networks that have been trained on data without variability in the background, i.e. *none* and *angle*, perform worse than guessing the mean ($p = 0$). An explanation for this could be that the input is so different from what the networks have previously encountered, that they extrapolate wildly. The network that has seen varying backgrounds during training, but not all variability, i.e. *angle/bg*, manages to perform slightly better than guessing the mean, showing how gradually increasing the variability present in the training data can also gradually improve the network's performance on data with yet more variability, and so its ability to generalize to unseen variability. In particular, it appears that background variability is essential for generalization performance. The network *all* performs best. Although the error does not reach zero, the results produce heads that look indistinguishable (see Figure 9). Note that differences on the less important principal components are difficult for humans to recognize by casual observation.

Performance on one Sample: In order to gain insight into how the networks process their inputs, we present each network with a sample randomly chosen from the scenario *all* and show the results in Figure 9. We approximate the importance of each pixel in the input by calculating the gradient via backpropagation with regards to a change in the first (i.e. most important) parameter, and refer to this importance as *saliency*. Furthermore, we use the estimated parameters and the ground truth to render the corresponding heads without any added variability for a visual comparison. The networks that have been trained on data with monochromatic backgrounds, *none* and *angle*, are thrown off by the background present in the input. The network *all*, which has encountered all variability during training, gracefully handles the background and focuses on defining areas of the head, e.g. the eyes, nose, mouth, and the visible ear. Again, *angle/bg* shows similar performance to *all* but does not reach the same level of quality.

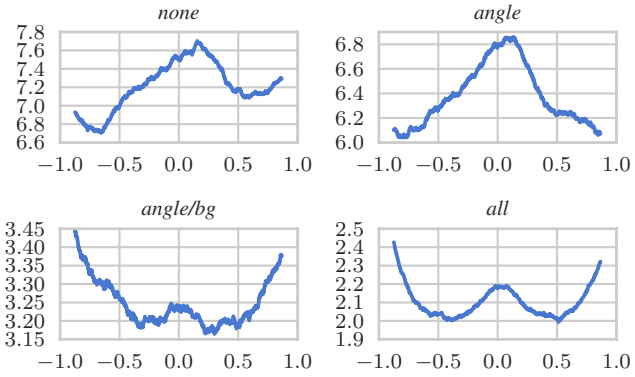

Figure 8: Each network has been applied to 2000 unseen images from the scenario *all*, where we know the exact angle from which the head has been rendered. Plotted are moving averages of the error against the angle.

Performance w.r.t. Angle: To further understand how the networks handle variability, we determine the error (as above) with respect to the angle from which the head visible in the input is rendered. See Figure 8 for the results. The network *none* has an obvious sweet spot around the angle from which its training data has been rendered, which is to be expected. It is however noteworthy that it appears to somehow generalize so that images rendered from a similar but opposite angle are handled better than frontal renderings – the network implicitly uses some knowledge about symmetry which was not explicitly contained within its training data. It might retain this property from having previously been trained on different data for classification, where invariance with regards to symmetry can also be of value. The network *all* shows the behavior one might expect from a human artist. It performs best when the head is rendered from an angle in between a frontal view and a profile, with no preference for either side.

VI. Discussion

We have demonstrated how photo-realistic renderings of artificial data can be used to analyze the effect of properties of the training data on the behavior of convolutional

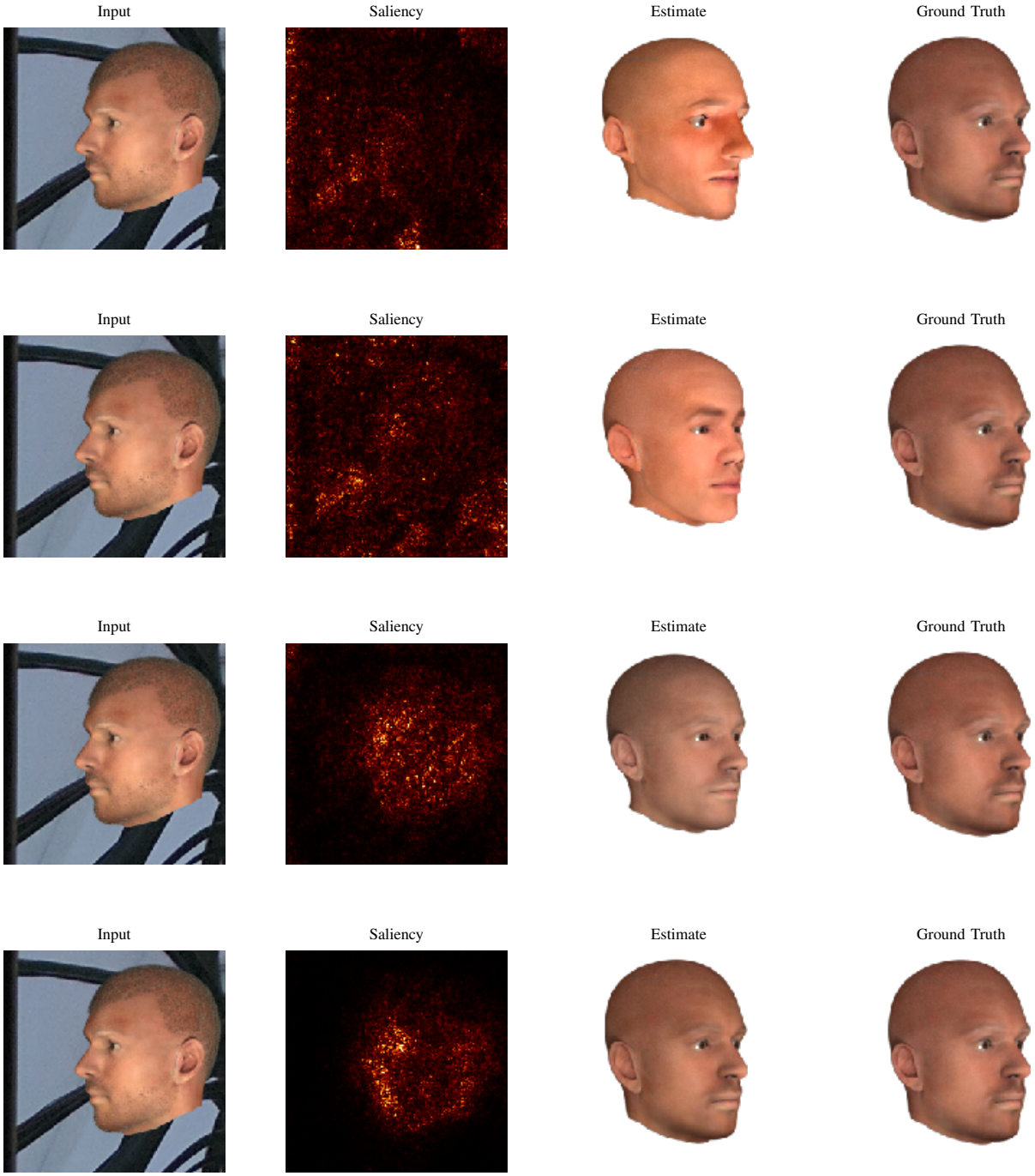

Figure 9: Each network's performance on one random sample from the scenario *all*. First column: input image. Second column: brighter pixels indicate a greater influence on the output. Third column: rendering of the output according to the scenario *none*. Fourth column: rendering of the ground truth according to the scenario *none*. Rows correspond to the networks *none*, *angle*, *angle/bg*, and *all*, respectively.

neural networks. For the application of 3-dimensional head reconstructions, we have investigated how the presence of variability in background and angle in the training set impacts the ability of a neural network to generalize to test data with high variability, and how the performance of differently trained networks depends on the level of deviation from previously seen scenarios. The results show that when confronted with unlearned variability, a network may extrapolate wildly, achieving worse results than a naive guess of the training mean, which highlights the importance of carefully considering how the distributions of training and application data may differ, especially when training a neural network on synthetic data. We have seen that the given problem can be solved in an end-to-end fashion, where the neural network learns to separate a head from the background.

We were able to observe how a neural network can make use of symmetry even if it has not explicitly seen its relevance to the given problem. This is akin to the finding of Gatys, Ecker, and Bethge [22], where a convolutional neural network trained for image classification includes knowledge that can be used to explicitly separate the content of an image from its artistic style.

This type of experiment opens the door to exploring how the architecture of a network influences its ability to learn invariances, or to generalize naturally to unseen variability. In future work, we plan to investigate this relationship, as well as the possibility of dataset augmentation by synthetic images when an invariance cannot be sufficiently represented or learned using available data. Furthermore, we would like to take advantage of how well invariances can be learned by adding variability that occludes parts of head, such as glasses or jewelery, to make head reconstruction more convenient.

Finally we would like to investigate the link between biases in the training data – e.g. an underrepresentation of certain races, or a strong correlation between hair length and gender – and the biases of models learned from this data.

References

[1] Yaniv Taigman et al. “Deepface: Closing the gap to human-level performance in face verification”. In: *CVPR*. 2014, pp. 1701–1708.

[2] Jascha Achenbach, Eduard Zell, and Mario Botsch. “Accurate face reconstruction through anisotropic fitting and eye correction”. In: *Proceedings of Vision, Modeling and Visualization* (2015).

[3] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[5] Christian Szegedy et al. “Going deeper with convolutions”. In: *CVPR*. 2015, pp. 1–9.

[6] Kaiming He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[7] Elad Richardson, Matan Sela, and Ron Kimmel. “3D Face Reconstruction by Learning from Synthetic Data”. In: *CoRR* abs/1609.04387 (2016).

[8] Elad Richardson et al. “Learning Detailed Face Reconstruction from a Single Image”. In: *CoRR* abs/1611.05053 (2016).

[9] R. Danerek et al. “DeepGarment : 3D Garment Shape Estimation from a Single Image”. In: *Comput. Graph. Forum* 36 (2017), pp. 269–280.

[10] Anh Tuan Tran et al. “Regressing Robust and Discriminative 3D Morphable Models with a very Deep Neural Network”. In: *CoRR* abs/1612.04904 (2016).

[11] Hyeongwoo Kim et al. “InverseFaceNet: Deep Single-Shot Inverse Face Rendering From A Single Image”. In: *arXiv preprint arXiv:1703.10956* (2017).

[12] Xiaoyong Shen et al. “Automatic portrait segmentation for image stylization”. In: *Computer Graphics Forum*. Vol. 35. 2. Wiley Online Library. 2016, pp. 93–102.

[13] Yi Sun, Xiaogang Wang, and Xiaoou Tang. “Deep convolutional network cascade for facial point detection”. In: *CVPR*. 2013, pp. 3476–3483.

[14] Volker Blanz and Thomas Vetter. “A morphable model for the synthesis of 3D faces”. In: *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co. 1999, pp. 187–194.

[15] Andrej Karpathy and Li Fei-Fei. “Deep visual-semantic alignments for generating image descriptions”. In: *CVPR*. 2015, pp. 3128–3137.

[16] Steve Lawrence et al. “Face recognition: A convolutional neural-network approach”. In: *IEEE transactions on neural networks* 8.1 (1997), pp. 98–113.

[17] Haoxiang Li et al. “A convolutional neural network cascade for face detection”. In: *CVPR*. 2015, pp. 5325–5334.

[18] Thabo Beeler et al. “High-Quality Single-Shot Capture of Facial Geometry”. In: *ACM Trans. on Graphics (Proc. SIGGRAPH)* 29.3 (2010), 40:1–40:9.

[19] Blender Online Community. *Blender – a 3D modelling and rendering package*. Blender Foundation. Blender Institute, Amsterdam. URL: http://www.blender.org.

[20] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *arXiv preprint arXiv:1512.03385* (2015).

[21] Yangqing Jia et al. “Caffe: Convolutional Architecture for Fast Feature Embedding”. In: *arXiv preprint arXiv:1408.5093* (2014).

[22] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. “A neural algorithm of artistic style”. In: *arXiv preprint arXiv:1508.06576* (2015).