ON THE FUSION OF RGB AND DEPTH INFORMATION FOR HAND POSE ESTIMATION

Evangelos Kazakos, Christophoros Nikou

University of Ioannina
Dept. of Computer Science and Engineering
Ioannina, Greece

Ioannis A. Kakadiaris

University of Houston
Dept. of Computer Science
Houston, TX, USA

ABSTRACT

Recent advances in deep learning have spurred 3D hand pose estimation, as convolutional network (ConvNet) based methods outperformed random forests. However, in the state of the art, ConvNet based methods employ only depth images of the hand without leveraging color and texture information from the RGB domain. In this paper, we investigate whether ConvNets can learn more rich and discriminative embeddings, by combining RGB and depth information. To answer this question, we propose the fusion of RGB and depth information in a double-stream architecture. More specifically, RGB and depth images are fed into two separate networks by extracting features, which are subsequently fused at an intermediate layer of the ConvNet, implementing input-level fusion, feature-level fusion and score-level fusion. The double-stream scheme is coupled with a deep ConvNet, contrary to the shallow networks that are mostly proposed in the literature. Experimental results show that while the depth of the network is crucial for hand pose estimation, the double-stream nets perform very similarly with the net trained only with depth images. This may suggest that training double-stream architectures purely with supervision may be insufficient for hand pose estimation with RGB-D fusion.

Index Terms— hand pose estimation, double-stream networks, fusion, rgb-d, deep learning

1. INTRODUCTION

The problem of hand pose estimation has been studied in the computer vision literature for decades [1]. Recent methods may be classified into two different categories: the *generative* (model-based) approaches and the *discriminative* (appearance-based) approaches. In generative hand pose estimation methods [2, 3, 4, 5, 6], hypotheses are made from a 3D articulated hand model and poses are tracked by fitting the model to input image observations. Discriminative methods [7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19] learn a mapping from visual features to a target parameter space such as joint locations. Typically, a regressor or a classifier is used to infer joint locations. The popularization of RGB-D sensors has motivated the interest of the computer vision community in pose estimation as depth images have significantly improved the performance of the related methods. While a big part of the literature based on discriminative approaches used randomized decision forests (RDFs) or regression forests variants [11, 12, 13, 14, 15], here we focus on ConvNet based methods.

The widespread use of convolutional networks influenced the hand pose estimation literature as well. Tompson *et al* [7] trained a multi-resolution convolutional network to predict 2D heat-maps for each joint location. Oberweger *et al* [10] proposed an architecture which imposes a prior on the physical constraints of the hand using a "bottleneck" layer before the output layer. In [16], the same authors adopted a generative approach where a *synthesizer* network was trained to iteratively generate depth images given predicted poses. Zhou *et al* [8] introduced a model based deep learning approach, where a new layer was proposed that maps joint angles to joint locations. Oberweger and Lepetit [17] improved the performance of [10] by employing data augmentation techniques, and inserting residual connections to their original architectures inspired by [20]. In [18], the hand depth images are encoded by a volumetric representation, which is fed into a 3D ConvNet. In [19], Region Ensemble Networks are proposed, where the feature map of the last convolutional layer is divided in multiple grid regions, and each of them is fed in a separate fully-connected layer.

Depth images are useful cues for 3D pose estimation, since they encapsulate 3D information, which is directly correlated with the estimation of 3D joint positions. RGB images describe accurately the surface of objects with color and texture information, which lack from depth images resulting in a relatively less precise description of the objects. In this work, we investigate whether the combination of RGB and depth information can improve the performance of convolutional networks, since the benefits and drawbacks of each domain are complementary. To this end, we build upon the double-stream architecture paradigm [21], to fuse RGB and depth information at any layer of the network. We evaluate input-level fusion in which the input images are combined, feature-level fusion in which the streams may be fused at any convolutional or fully-connected layer, and score-level fusion

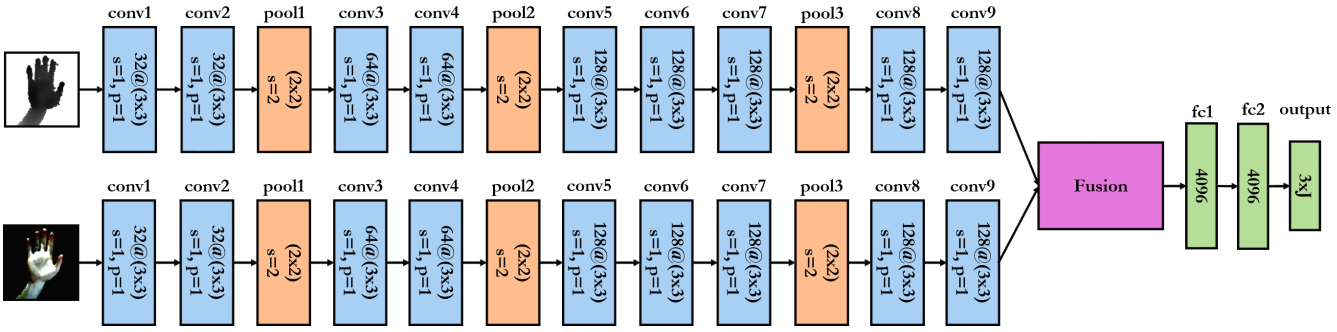

Fig. 1: FuseNet: Our architecture for double-stream architecture fusion. Two streams are trained in parallel, the depth stream and the RGB stream with depth and RGB images respectively, and at any layer the networks may be fused. After the inserted fusion each respective net is truncated and the network continues as a single ConvNet. Here, we demonstrate the fusion (purple block), in the last convolutional layer, implementing feature-level fusion; nevertheless, it can be inserted at the input layer for input level fusion, at the output layer for score level fusion, or at any intermediate feature layer (convolutional, pooling or fully-connected). We call DepthNet the single stream version with depth images as input.

where the streams' predictions are combined. We use fusion functions proposed in [21] and we propose a new trainable function for score-level fusion. In each case, we use as base network a deep ConvNet which we carefully designed.

We implemented our approach using Lasagne [22] and our code is available at: https://github.com/ekazakos/fusenet-hand-pose.

2. RGB-D FUSION WITH CONVNETS

We consider the case of estimating the 3D joint positions $\boldsymbol{J} = \{\boldsymbol{j}_i\}_{i=1}^{J}$ of a hand, where $\boldsymbol{j}_i = (x_i, y_i, z_i)$ and J is the number of joints. We also assume that we have an annotated dataset for training. While most of the modern discriminative approaches estimate the 3D hand pose from a single depth image, we employ both RGB and depth images.

For segmenting the hand from the depth images, we follow [8, 10]. RGB images are normalized to follow a normal distribution with zero mean and a standard deviation of one.

Here, we present our approach of fusing RGB and depth information with ConvNets for hand pose estimation. We adopt the double-stream architecture fusion paradigm, which was first proposed in [21, 23] for activity recognition using a spatial and a temporal stream, where in [23] the class predictions of each respective stream were fused, while in [21] the authors proposed the fusion of feature maps.

We employ two streams, the RGB stream and the depth stream, which may be fused at any layer of the architecture. Depending on the layer of fusion, someone may implement: *input-level fusion*, *feature-level fusion* and *score-level fusion*. For each fusion strategy, different fusion functions are appropriate. For input-level fusion and feature-level fusion, we investigate the performance of existing fusion functions, while for score-level fusion we propose a new trainable fusion function.

In Fig. 1, we demonstrate our double-stream fusing architecture which we call FuseNet. Two separate streams, the depth stream and the RGB stream are put in parallel and trained simultaneously with depth and RGB images respectively. Here we show the case were the fusion (purple block) takes place after the last convolutional layer, implementing feature-level fusion. By placing the fusion block at different layers, the other fusion techniques can be implemented as well. After the inserted fusion each respective net is truncated and the network continues as a single ConvNet.

Our architecture deploys in all convolutional layers 3×3 kernels with a stride of 1 and zero-padding of 1, and max-pooling layers with 3×3 receptive fields. ReLU nonlinearities are applied after all convolutional and fully-connected (fc) hidden layers and dropout [24] is inserted after each fc layer. Since the preprocessing normalises the 3D joints positions in $[-1, 1]$, we employ hyperbolic tangent activation functions for the output units which estimate the 3D joints position. Each stream has 9 convolutional layers and 3 max-pooling layers, where the number of filters starts from 32 (first convolutional group) and goes up to 128 (last convolutional group) . There are 2 hidden fully-connected layers with 4096 hidden units and the output layer is again a fully-connected layer with $3J$ output units (J being the number of joints).

To discriminate between fusion and the baseline architecture where a single stream is employed which is fed only with depth images, we call the latter DepthNet. First we designed DepthNet by varying different components of the architecture, such as the depth of the network and the size of convolutional kernels, and subsequently we used DepthNet as the basic building block for our fused architecture, namely FuseNet. Due to space limitations, we will not present here the evaluation of all the architectures we experimented with. Nevertheless, it is important to mention that the deepest architecture provided the most accurate predictions.

For a detailed description of different fusion functions, namely *max fusion*, *sum fusion*, *concatenation fusion*, and *convolutional fusion*, may refer to [21].

Input-level fusion and feature-level fusion. Input-level fusion can be achieved by placing the fusion block after the input layers of the two streams, while for feature-level fusion the fusion block can be placed after any convolutional, pooling, or fully-connected hidden layer since these layers provide intermediate representations (features) of the input images. For feature-level fusion we evaluate all fusion functions presented in [21], while for input-level fusion only concatenation fusion and convolutional fusion are considered, as max fusion and sum fusion perform element-wise operations and require same number of channels in both streams, making them not applicable in our case where the RGB stream has 4 input channels and the depth stream has 1.

Score-level fusion. Score-level fusion refers to the combination of the predictions of different models. In our case, the models that their predictions are to be fused are the depth stream and the RGB stream, and thus the fusion block is placed after the output layer of the streams. Similarly to [21, 23], the first fusion function we exploit is average fusion, which is implemented by first scaling the predictions by 0.5 and subsequently applying sum fusion.

Furthermore, we introduce a new learnable fusion function tailored for score-level fusion. Given the predictions $\widehat{\boldsymbol{y}}^a, \widehat{\boldsymbol{y}}^b \in \mathbb{R}^{3J}$ (J being the number of joints), we want to compute the final predictions as:

$$\widehat{y}_i = w_i^a \widehat{y}_i^a + w_i^b \widehat{y}_i^b + b_i, \tag{1}$$

where $1 \leq i \leq 3J$, w_i^a, w_i^b are independent weights for each element of the prediction vector, and b_i are independent biases. We model (1) using a *locally-connected* layer, which behaves as a convolutional layer with the difference that it does not share parameters across each spatial position, i.e. it computes a dot product of a kernel with the part of the input that overlaps with the kernel, using a different kernel at each shifted kernel location. We name this fusion function locally-connected fusion. Firstly $\widehat{\boldsymbol{y}}^a$, $\widehat{\boldsymbol{y}}^b$ are stacked as separate channels using concatenation fusion, resulting in $\widehat{\boldsymbol{y}}^{\text{cat}} \in \mathbb{R}^{3J\times 2}$, and afterwards an 1-dimensional locally-connected layer is incorporated which employs a filter $\boldsymbol{f} \in \mathbb{R}^{3J\times 2}$ and a bias vector $\boldsymbol{b} \in \mathbb{R}^{3J}$:

$$\widehat{\boldsymbol{y}}^{\text{local}} = \widehat{\boldsymbol{y}}^{\text{cat}} \circledast \boldsymbol{f} + \boldsymbol{b}, \tag{2}$$

where we use $\circledast$ to denote the operation that a locally-connected layer performs. Note that (2) works similarly to convolutional fusion from [21], with the difference that different parameters are used at each spatial location.

3. RESULTS

We evaluated our methods on NYU Hand pose dataset [7]. It contains 72757 training-set frames and 8252 test-set frames of RGB-D data, captured with PrimeSense, a structured-light RGB-D sensor. The training set contains samples from a single subject, while the test set contains samples from two subjects.

NYU Hand pose dataset provides very accurate ground truth annotations and as mentioned in [25], it exhibits very large pose variation, which makes it one of the most challenging datasets in the literature. Although the ground truth contains $J = 36$ annotated joints, for comparison purposes, we used a subset of $J = 14$ joints to follow the evaluation protocol of prior work [7, 8, 10, 16, 17, 18, 19].

Although there are also other popular benchmark datasets in the literature, such as [12, 13], we did not consider another dataset since the NYU Hand pose dataset is the only one that provides RGB-D data. The Dexter dataset [26] also provides RGB-D data but is very limited in pose variation, and hence evaluating in NYU is more challenging. All the other datasets provide only depth images, and thus they are inappropriate for our fusion approaches.

The evaluation metric that we used is widely used in prior works [8, 10, 12, 13, 16]. It is the **success-rate**, that is the fraction of test set frames whose max-joint-error is below a threshold. In other words, it measures the fraction of test set frames for which each predicted joint is below a maximum Euclidean distance between the ground truth and the predicted joint locations. This is a very challenging evaluation metric since with a single displaced joint prediction the whole pose can be regarded as incorrectly estimated.

The loss function that we used for training is the Euclidean loss between the ground truth and the predicted joint, summed over all joints. We trained our models using mini-batch stochastic gradient descent with Nesterov momentum [27]. We used a batch size of 128 training examples per iteration and 100 epochs. The learning rate was set to 0.009, the momentum to 0.98, and for the dropout probability we used varying values in the range $[0.02, 0.14]$. Although in a classification setting the dropout probability is usually set to values of an order of magnitude higher, we observed that very low values work well in our problem. We used early stopping such that the optimal number of training epochs was selected and training stopped when the validation error saturated. We decayed the learning rate with a factor of 0.5 every time the validation error saturated. At each epoch, the training examples were shuffled as preliminary experiments showed that this increases the performance.

In Fig. 2, we present the results of our experimental evaluation. In Fig. 2(a), the results of the different fusion techniques are demonstrated. While we performed extensive experiments by fusing using all different fusion functions mentioned above and by fusing at all possible layers for feature-

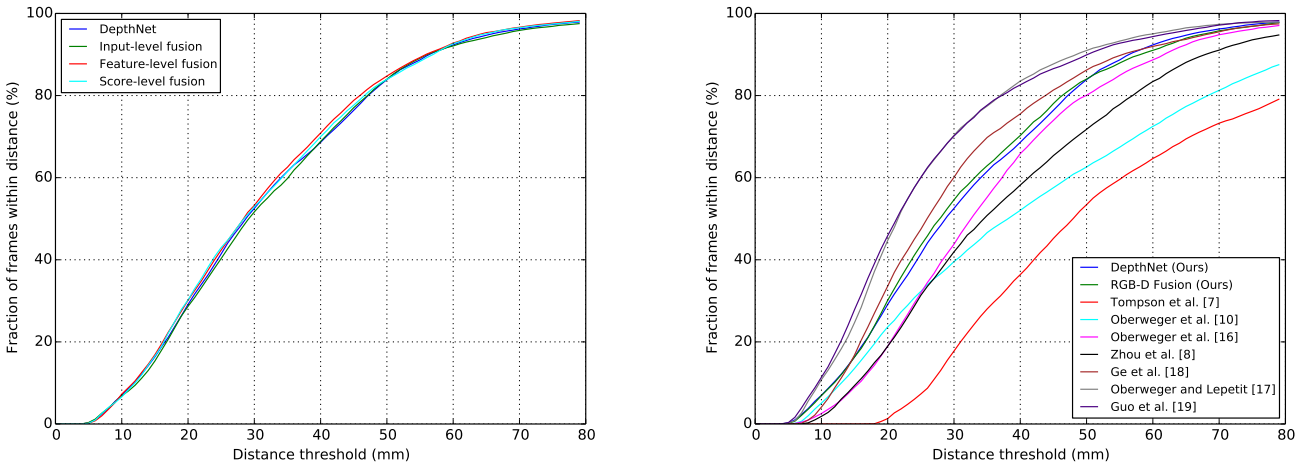

Fig. 2: Evaluation of double-stream architecture fusion and comparison with the state of the art. (a) Comparison of all fusion techniques and DepthNet. For input-level fusion the concatenation function was used, for feature-level fusion the sum function was used and the fusion was performed at conv7 layer. For score-level fusion the locally-connected function was used. (b) Comparison of double-stream architecture fusion with the state of the art. We compare using both DepthNet where no fusion is present and RGB-D fusion which refers to feature-fusion.

level fusion, here we show the best performing fusion function and layer of fusion (for feature-level fusion) for each fusion technique, due to space limitations. For input-level fusion, the concatenation function was used, feature-level fusion was performed by fusing at conv7 layer using the sum fusion function. For score-level fusion, the locally-connected fusion function was employed. The use of different fusion functions and the fusion at different layers for feature-level fusion resulted in very subtle differences in terms of performance. This result is noteworthy as it implies that the choice of the layer to perform feature-level fusion and the choice of fusion functions for all fusion techniques is not important, since the accuracy of the model will not be significantly affected.

More interestingly, in Fig. 2(a) we can see that the accuracies of all three different fusion techniques are almost identical, and even more interestingly, they perform about the same with DepthNet, where no fusion is present. In fact, their subtle differences is probable due to the random initialisation of the model parameters, and in that case, fusion has no effect in the results. From these results, we draw the conclusion that training double-stream architectures by simply using a supervision loss may be inadequate for fusing RGB-D features for hand pose estimation. A possible solution towards this direction could be the alignment of the distributions of the RGB and depth modalities.

In Fig. 2(b), we compare our approach with the state of the art. Our deep architecture, namely DepthNet, outperforms some of the compared approaches [7, 8, 10, 16] and performs similarly to [18]. This result is interesting because while the other approaches use either iterative refinement steps [10, 16] or apply inverse kinematics in the predictions of the ConvNet [7], or apply a forward kinematic function inside the network [8], our model surpass their performance without any extra refinement step. This is because we are using a deeper ConvNet than these approaches, which indicates that in many cases a network with increased depth is capable to map more accurately depth images to 3D poses than approaches with shallow nets and extra steps. Yet, while our proposed RGB-D fusion approach does not improve the results comparing to DepthNet, the performance of our model is limited comparing to [17, 18, 19].

4. CONCLUSION

In this paper, we proposed the fusion of RGB and depth images for hand pose estimation coupled with a deep convolutional network. The most important findings of our experimental evaluation are the following: i) The depth of the network is of vital importance in hand pose estimation, as our deep ConvNet (without fusion) alone outperforms some of the baseline methods which use extra refinement steps, and ii) Supervised training of double-stream architectures may be insufficient for fusing RGB-D data for hand pose estimation. The second conclusion is derived from the fact that after an extensive evaluation of fusing at any possible layer of the architecture using any of the given fusion functions, the performance of the double-stream architecture is on par with our baseline network that uses only depth images.

5. REFERENCES

[1] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *CVIU*, vol. 108, pp. 52–73, 2007.

[2] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, "Real-time and Robust Hand Tracking from Depth," in *CVPR*, 2014.

[3] I. Oikonomidis, M. Lourakis, and A. Argyros, "Evolutionary Quasi-Random Search for Hand Articulations Tracking," in *CVPR*, 2014.

[4] N. Kyriazis I. Oikonomidis and A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect," in *BMVC*, 2011.

[5] L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys, "Motion Capture of Hands in Action using Discriminative Salient Points," in *ECCV*, 2012.

[6] M. de La Gorce, D. J. Fleet, and N. Paragios, "Model-Based 3D Hand Pose Estimation from Monocular Video," *PAMI*, vol. 33, pp. 1793–1805, 2011.

[7] J. Tompson, M. Stein, Y. LeCun, and K. Perlin, "Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks," *ACM Transactions on Graphics*, vol. 33, pp. 169:1–169:10, 2014.

[8] X. Zhou, Q. Wan, W. Zhang, X. Xue, and Y. Wei, "Model-based Deep Hand Pose Estimation," in *IJCAI*, 2016.

[9] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "Robust 3D Hand Pose Estimation in Single Depth Images: From Single-View CNN to Multi-View CNNs," in *CVPR*, 2016.

[10] M. Oberweger, P. Wohlhart, and V. Lepetit, "Hands Deep in Deep Learning for Hand Pose Estimation," in *CVWW*, 2015.

[11] C. Keskin, F. Kiraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *ICCV Workshops*, 2011.

[12] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, "Cascaded Hand Pose Regression," in *CVPR*, 2015.

[13] D. Tang, H. Jin Chang, A. Tejani, and T. Kim, "Latent Regression Forest: Structured Estimation of 3D Articulated Hand Posture," in *CVPR*, 2014.

[14] C. Xu and L. Cheng, "Efficient Hand Pose Estimation from a Single Depth Image," in *ICCV*, 2013.

[15] D. Tang, T. Yu, and T. Kim, "Real-Time Articulated Hand Pose Estimation Using Semi-supervised Transductive Regression Forests," in *ICCV*, 2013.

[16] M. Oberweger, P.Wohlhart, and V. Lepetit, "Training a Feedback Loop for Hand Pose Estimation," in *ICCV*, 2015.

[17] M. Oberweger and V. Lepetit, "Deepprior++: Improving fast and accurate 3d hand pose estimation," in *ICCV*, 2017.

[18] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "3d convolutional neural networks for efficient and robust hand pose estimation from single depth images," in *CVPR*, 2017.

[19] H. Guo, G. Wang, X. Chen, and C. Zhang, "Towards good practices for deep 3d hand pose estimation," *CoRR*, vol. abs/1707.07248, 2017.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[21] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional Two-Stream Network Fusion for Video Action Recognition," in *CVPR*, 2016.

[22] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, et al., "Lasagne: First release.," Aug. 2015.

[23] K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *NIPS*, 2014.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *JMLR*, vol. 15, pp. 1929–1958, 2014.

[25] J. S. Supančič, III, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan, "Depth-Based Hand Pose Estimation: Data, Methods, and Challenges," in *ICCV*, 2015.

[26] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive Markerless Articulated Hand Motion Tracking Using RGB and Depth Data," in *ICCV*, 2013.

[27] I. Sutskever, J. Martens, G. E. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *ICML*, 2013.