13th International Symposium on Particle Image Velocimetry – ISPIV 2019
Munich, Germany, July 22-24, 2019

Deep artificial neural network architectures in PIV applications

Christian Lagemann[1]*, Kai Lagemann[2], Wolfgang Schröder[1], Michael Klaas[1]

[1] RWTH Aachen University, Institute of Aerodynamics Aachen, 52062 Aachen, Germany

[2] RWTH Aachen University, Cybernetics Lab IMA & IfU, 52068 Aachen, Germany

* c.lagemann@aia.rwth-aachen.de

Abstract

Artificial neural networks have successfully been used in a variety of tasks, e.g., image classification, object detection, or pattern recognition. Due to the combination of increased computational power and optimized network architectures tremendous breakthroughs were achieved in machine learning shifting the focus of current research of various disciplines to this promising technique. In this study, Inception-based, residual, and densely connected neural networks were trained on synthetic PIV datasets. The networks were tested on two real measurement datasets, namely zero pressure gradient turbulent boundary layer flow (TBL) and turbulent channel flow with an alternating pressure gradient, to assess the performance of end-to-end PIV image processing based on artificial neural networks (ANN). The results were compared to commercial and self-developed PIV post-processing tools. The comparison of the results from ANN based PIV analysis with standard PIV algorithms shows that very deep neural networks match the results of standard PIV algorithms in real measurement setups and can achieve comparable mean absolute errors when trained on synthetic PIV datasets.

1 Introduction

In the recent years, common tasks of computer vision, e.g., object detection and recognition, have improved dramatically due to significant advances in the field of machine learning. Low-cost high performance processing units in combination with a higher amount of reasonable training data led to outstanding breakthroughs in deep learning and convolutional neural networks (CNNs), as presented by Krizhevsky et al. (2012). Furthermore, techniques like dropout (Hinton et al. (2012)) and new activation functions, e.g., rectified linear units (Glorot et al. (2011)), helped to increase the performance of deep learning algorithms. CNNs are a special kind of multi-layer neural networks, designed to recognize visual patterns directly from pixel images with minimal preprocessing. Compared to normal artificial neural networks, CNNs are arranged in three dimensions (width, height and depth) and are sparsely connected such that the output of the current layer is only influenced by a specific number of input neurons. The basic intuition assumes that images consist of local patches of values (features) that correlate highly in space allowing CNNs to recognize patterns with extreme variability, and with robustness to distortions and simple geometric transformations. Especially in the field of object detection and pattern recognition, CNNs gave rise to significant breakthroughs (Silver et al. (2016), Redmon and Farhadi (2016), Goodfellow et al. (2014)) outperforming human designed algorithms or even human performance itself. As a consequence, first attempts were made to transfer these algorithms to different fields of fluid mechanics, e.g., turbulence modeling (Beck et al. (2018)), active flow control (Couchot et al. (2013)), and digital image processing (Lee et al. (2017)), the latter of which is a crucial part of flow analysis using image based measurement techniques like particle-image velocimetry (PIV) and particle-tracking velocimetry (PTV). Standard PIV algorithms typically compute the cross-correlation of small windows between two frames to predict the most probable displacement of the particle pattern. State-of-the-art PIV algorithms additionally use a wide range of other methods and algorithms, e.g., subpixel interpolation (Nobach and Honkanen (2005)), multigrid correlation schemes (Scarano and Riethmuller (1999)) or an automatic outlier detection (Westerweel and Scarano (2005)) to achieve a better accuracy forming very complex algorithms.

Considering the vast development and progress in deep learning, the question arises in how far deep neural networks can be trained to perform PIV processing. First conceptual studies were already reported in the literature (Rabault et al. (2017), Lee et al. (2017)) using different shallow neural network architectures stacking a maximum of two convolutional and two fully connected layers. To address more complex and diverse tasks, the recent trend is to increase the number of layers (Goodfellow et al. (2016)) and the layer size (Zeiler and Fergus (2013), Sermanet et al. (2013)), while using dropout (Hinton et al. (2012)) to address the problem of overfitting. Unfortunately, this simple solution comes with two major drawbacks. First, deeper networks typically come at the expense of a higher number of parameters, causing the enlarged network to be more prone to overfitting, especially if the number of labeled examples in the training set is limited. Second, a uniformly increased network size increases the necessary computational resources dramatically requiring carefully designed network architectures. Subsequent to the breakthrough of Krizhevsky et al. (2012), many attempts have been made to improve AlexNet's original architecture. Based on Hebbian principle, Szegedy et al. (2014) designed a sparsely connected convolutional architecture in order to deal with salient image parts varying massively in size. As a consequence of this substantial variation in information, choosing the right kernel size of convolutional operations becomes non-trivial. A larger kernel is preferred for information that is distributed more globally and a smaller kernel is preferable if information is concentrated locally. Szegedy et al. (2014) heavily used $1x1$ convolutions mainly serving as dimension reduction prior to computationally more expensive $3x3$ and $5x5$ convolutions. Thus, a significantly deeper and wider network was created at only marginally increased computational costs. The Inception-v1 network architecture contained five times less parameters compared to AlexNet and achieved even better error rates. Further improvements were achieved by He et al. (2015a) using ultra-deep CNNs. Since deeper models are harder to train and suffer from degradation (He et al. (2015a), He et al. (2015b)), the authors introduced a new residual learning framework. Having reformulated the underlying equation of the network layers, they forced the neurons to learn residual functions with reference to their preceding layer inputs rather than learning unreferenced functions. This allowed errors to be propagated directly to the preceding units. As a consequence, these networks are easier to train and optimize while being ultra-deep. Considering the success of short cuts in deep neural networks, architectures supporting further inter-layer communication are the next logical step. Huang et al. (2016) presented densely connected neural networks, connecting each layer to every other following layer in a feed-forward fashion. Compared to standard convolutional neural networks where the number of direct layer connections equals the corresponding number of layers L, DenseNets are characterized by $L(L+1)/2$ direct connections. This maximized inter-layer communication is achieved such that the feature-maps of all preceding layers are used as inputs while the output of the L-th layer is forwarded as input to all subsequent layers. Therefore, DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

Hence, the scope of the current study is to evaluate the performance of deep neural networks containing different Inception modules, residual connections, and densely connected layer blocks. For this, these network architectures are trained on different synthetic PIV image datasets. Subsequently, the trained networks were used to determine the velocity field in two real measurement setups. To assess the performance of the neural networks, the results from both test cases are compared to results achieved using standard PIV processing algorithms. Section 2 and 3 explain the datasets and the individual network architectures, while the results are presented in Section 4. Finally, in Section 5 a conclusion is drawn.

2 The Datasets

Three synthetic classes of image datasets with continuously increasing levels of complexity are used to test the performance of state-of-the-art neural network architectures. Synthetic images have two major benefits. First, it is possible to generate an infinite number of labeled training images and second, any desired flow condition can easily be created without further data augmentation. The first class of datasets is characterized by artificial label augmentation (ALA), i.e., the underlying velocity field is approximated by a second-order polynomial as reported in Rabault et al. (2017). Thus, the labels consist of the velocity in the x- and y-direction parallel to the Jacobian and Hessian of the underlying flow. This kind of artificial label enrichment provides additional flow field information to the network and thus, simplifies the learning process. The intensity distribution of the particles is approximated by the following Gaussian distribution as recommend in Raffel et al. (2018)

$$I(x,y) = I_p \; exp\left(\frac{-(x-x_p)^2-(y-y_p)^2)}{(1/8)d_\tau^2}\right), \tag{1}$$

where I_p is the particle intensity, d_τ the effective particle diameter, and (x_p, y_p) the coordinates of the particle center. Particle intensity and diameter are randomized within typical PIV thresholds ($80 < I < 255,\ 1\ px < d_\tau < 5\ px$). Note that the particle intensity is not integrated over the entire window domain. The second dataset class intends to solely predict velocity data without providing further flow information when backpropagating through the network. In this approach, no Jacobian and Hessian of the velocity field (NoJH) are considered anymore. The intensity distribution is still approximated using the Gaussian distribution described by e (1). The final synthetic dataset class (IPI) aims at reproducing highly realistic PIV flow data. Here, the particle intensity is integrated over the entire image domain using the following approach (Lecordier and Westerweel (2004))

$$I(x,y) = I_p \pi\sigma^2 \frac{\left[erf\left(\frac{(x_p-x_{Min})}{(\sqrt{2}\sigma)}\right) - erf\left(\frac{(x_p-x_{Max})}{(\sqrt{2}\sigma)}\right)\right] \cdot \left[erf\left(\frac{(y_p-y_{Min})}{(\sqrt{2}\sigma)}\right) - erf\left(\frac{(y_p-y_{Max})}{(\sqrt{2}\sigma)}\right)\right]}{2} \tag{2}$$

with the error function

$$erf(x) =) \frac{2}{\sqrt{\pi}} \int_0^x e^{t^2} dt \tag{3}$$

where σ denotes the standard deviation of the Gaussian corresponding to the particle diameter.

Every class contains four diverse types of datasets, and each dataset consists of 10^6 training and 10^5 validation image pairs. The first two datasets of each class are characterized by a constant particle density but differ in the maximum allowed particle shift. While the maximum displacement of the first dataset (D1) is limited to $s_{max} = 4\ px$ similar to Rabault et al. (2017), the maximum particle shift of the second dataset (D2) is set to $s_{max} = 10\ px$ allowing for more realistic displacements since values in this range are common for real PIV datasets. The displacement of datasets D3 and D4 equals the maximum displacement of D1 and D2. Additionally, the effects of a varying particle density, i.e., the particle density of each image varies in a range of $\pm 75\%$ compared to D1 and D2, are considered.

Each synthetically generated image pair has a size of $32x32\ px$ per image at a color depth of 8 bits grayscale. Both snapshots of each image pair are stacked on top of each other forming a 4-dimensional input matrix. Additionally, the neural networks were tested on two real measurement datasets, namely a turbulent boundary layer (TBL) with spanwise traveling surface waves (Roggenkamp et al. (2019)) and a turbulent wavy channel flow (TCF) (Rubbert et al. (2017)). The TBL experiments are conducted in a subsonic closed-loop wind tunnel containing a 1700 mm long, 1400 mm wide, and 20 mm thick flat plate mounted in the open test section. A high-speed 2D/2C PIV setup is used to capture the influence of the wave motion on the outer part of the boundary layer at $Re_\Theta = 1200$ consisting of one Photron Fastcam SA3 ($1024x1024\ px^2$ at 12 bit) equipped with an Infinity K2 long-distance microscope and a double pulsed Nd:YAG laser (Quantel Twins BSL 140). In the TCF study, two wavy channel geometries of sinusoidal shape with a wavelength of 100 mm were investigated at three Reynolds numbers $Re_\tau^* = 202,\ 350,$ and 534. The reference measurement setup comprised a sCMOS camera (PCO Edge 5.5, $2560x2160\ px^2$ at 16 bit) equipped with a Zeiss macro lenses ($f = 100,\ f_{\#,max} = 2$) and a dual cavity Nd:YAG-laser (New Wave Solo 200XT). For all details, the reader is referred to the two manuscripts.

3 The Architecture

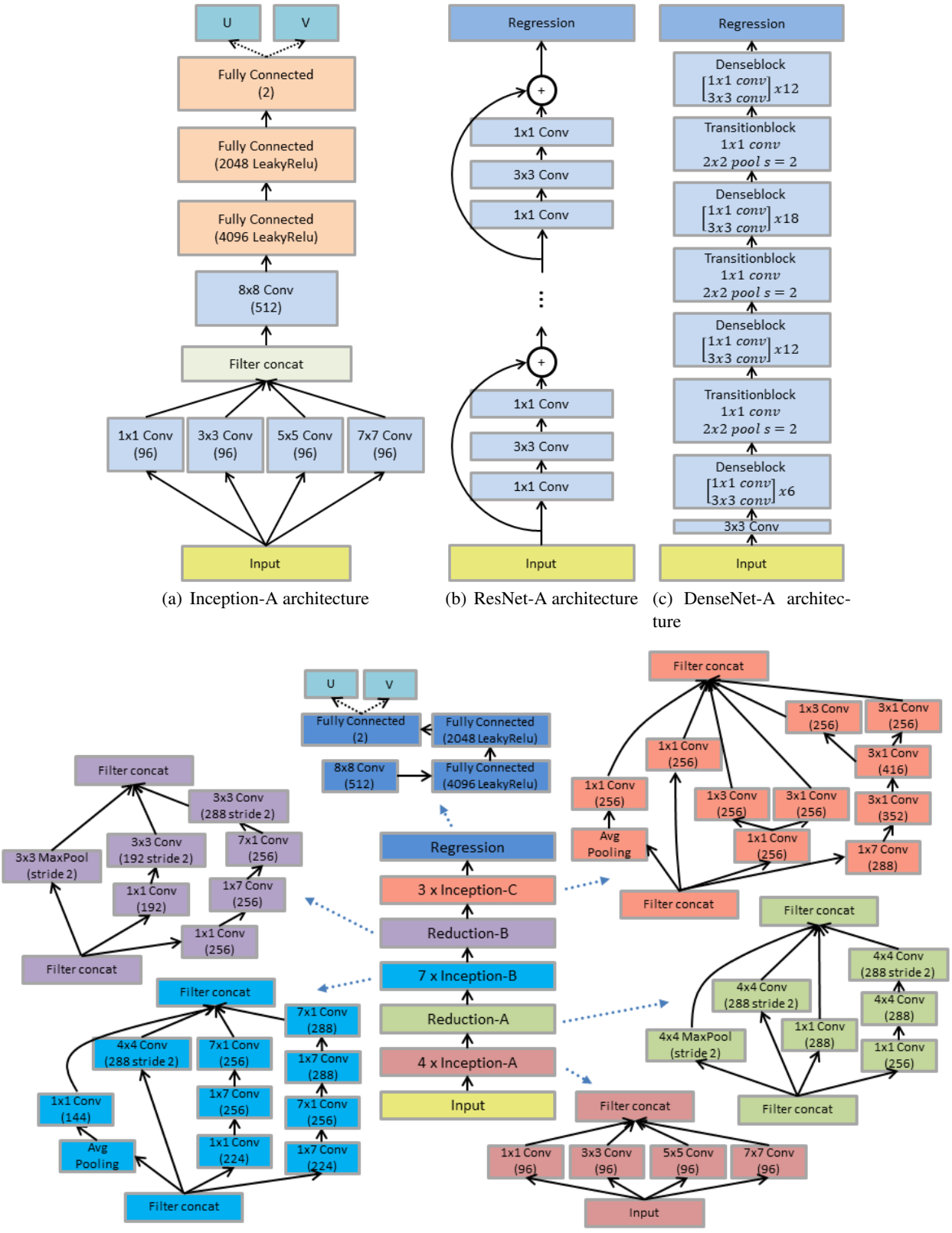
The performance of different state-of-the-art neural network structures was tested and compared to the results of a standard neural network consisting of 1 convolutional and 3 fully connected layers comparable to that reported by Rabault et al. (2017). First, the performance of Inception modules was analyzed. Based on the idea of Arora et al. (2013) that features of high correlation are clustered into groups, Szegedy et al. (2014) assumed that these units can be grouped into filter banks. This means that a lot of clusters concentrate in a single region and, hence, can be covered by a single $1x1$ convolution. Following this idea, Szegedy et al. (2014) stated that clusters which are spatially more spread can be covered by convolutions of larger kernel size, e.g., $3x3$, $5x5$, or $7x7$ and, consequently, are arranged in parallel branches next to each other resulting

in a deep and significantly wider artificial neural network. The output of these different branches are finally concatenated and are forwarded to the next module. The first Inception based network is called Inception-A and contains one module consisting of four different branches as shown in Figure 1(a). The second important idea of Inception architectures is a feature dimension reduction such that $1x1$ convolutions are added in branches prior to kernels of larger size. The idea behind this approach is that $1x1$ convolutions preserve the spatial dimensions of the input while they reduce the depths significantly by combining feature maps. This can be seen as a generalized representation of the input. As a consequence, computational bottlenecks are avoided. The second Inception based CNN (Inception-C) as shown in Figure 1(d) is mainly inspired by the latest Inception-v4 network (Szegedy et al. (2016)). This network is significantly deeper compared to Inception-A and uses factorization of computationally expensive filters into smaller and asymmetric convolutions (a $7x7$ convolution is split into a convolution of $7x1$ followed by a convolution of $1x7$) further reducing the computational effort.

The second network architecture incorporates residual connections as introduced by He et al. (2015a). In theory, stacked convolutional neural networks can asymptotically approximate any function $F(x)$, where x denotes the input. The same holds if a neural network is explicitly forced to learn the residual function of the problem, e.g., $R(x) = F(x) - x$. He et al. (2015b) found that this residual reformulation helps in fighting the degradation problem of very deep neural networks, since CNNs struggle in approximating identity mappings by multiple nonlinear layers. Explicitly reformulating learning as a residual function allows the weights of the nonlinear layers to converge against zero to create identity mappings. Thus, this approach allows significantly deeper neural networks. Our ResNet architecture is a plain 67-layer deep network as shown in Figure 1(b). ResNet architectures preserve information explicitly through identity connection, i.e., many layers contribute very little and can randomly be dropped during training. Huang et al. (2016) extended the idea of inter-layer connections and introduced densely connected neural networks (DenseNet). DenseNets explicitly differentiate between information that is added to the network and information that is preserved. In other words, a specific layer augments the feature extraction process by accessing the feature maps of all preceding layers and concatenating its own new feature maps. Therefore, each layer has direct access to the gradients from the loss function and the original input signal at the same time, leading to an improved flow of information and gradients throughout the network. No redundant feature maps are learned and fewer network parameters are necessary to achieve similar performance compared to plain network types. The specific architecture of the current DenseNet is shown in Figure 1(c). It consists of four Dense Blocks where each of the first three Dense Blocks is followed by a Transition Block to reduce the spatial feature size. No further compression was applied.

4 Results

The performance of the different network architectures is analyzed in efficiently learning challenging flow conditions. All networks were implemented in Tensorflow (Abadi et al. (2015)) using the Keras High-Level API (Chollet et al. (2015)). Due to the regression nature of the PIV problem, the loss is chosen to be the mean squared error while the evaluation metrics is defined as the mean absolute error. A standard Adam-Optimizer (Kingma and Ba (2014)) was applied to train the networks starting at an initial learning $lr = 0.00025$. Furthermore, the learning rate was reduced by a factor of 5 once the evaluation metrics stopped improving. The minimum learning rate was set to $lr_{min} = 10^{-8}$. This learning rate scheduler showed the best results in the current study. The results discussed in the following are "typical", representing neither best nor worst cases. We trained every test case of each network for 200 epochs three times independently from scratch and finally averaged the mean absolute error. All computations were run on multiple GPUs (Geforce RTX 2080Ti, Nvidia Titan RTX). The hyperparameters of the training process are summarized in Table 1. For comparison, the synthetic images are evaluated using the commercial PIV software PIVview and our self-developed inhouse PIV processing code PascalPIV. Different processing runs were performed applying single and multi pass as well as multi grid methods. The multi pass (3 passes) and grid (initial window size of $64x64px$) runs were performed on images of $96x96px$ using a third order accurate B-Spline interpolation on all passes. The absolute mean error is slightly higher compared to that in the literature and ranges from 0.04 to 0.09 px depending on the specific dataset. The in-house PIV code achieved a mean absolute error of 0.004 px. Again, images of $96x96px$ were analyzed further subdivided into windows of $32x32px$ with an overlap of 75%. In total, 6 iteration steps using a symmetric integral predictor scheme with image distortion (Lanczos4 image interpolation scheme, third order B-Spline accuracy) were performed. A normalized median test was applied to detect spurious vectors inbetween each iteration loop. However, please note that this processing



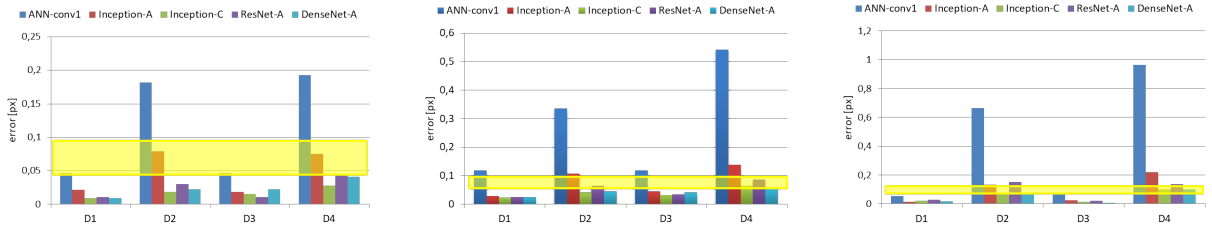
(a) Inception-A architecture (b) ResNet-A architecture (c) DenseNet-A architecture

(d) Inception-C architecture

Figure 1: Deep network architectures: (a) Inception-A: Inception based network using a single Inception module, (b) ResNet-A: ResNet architecture with 67 layers augmented by residual connections, (c) DenseNet-A: DenseNet architecture incorperating four Dense blocks and 3 Transition blocks, (d) Inception-C: Inception-v4 based network architecture combining multiple diverse Inception and reduction modules.

Parameter	epochs	batch size	initial learning rate	minimum learning rate	learning rate reducing factor	patience level
Value	200	100-2000	0.00025	1e-08	5	15

Table 1: Shared hyperparameters of the network architectures



(a) ALA (b) NoJH (c) IPI

Figure 2: Absolute mean error of the different network architectures generating predictions for the four datasets in all dataset classes: (a) artificially augmented labels (ALA), (b) reduced number of training labels, no Jacobian and Hessian during training available (NoJH), (c) integrated particle intensity (IPI); The yellow bar denotes the results of a standard PIV algorithm.

was run simultaneously on 12 cores taking approximately 100 minutes to evaluate 10^5 images. Further note that this processing run was solely intended to demonstrate the best possible result of a high-performance state-of-the-art PIV algorithm. Compared to the current neural networks, this algorithm assesses 9 times more particle information and requires minimum 130 times more time.

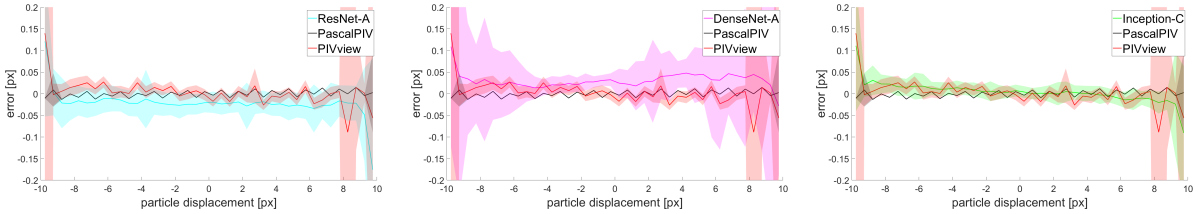
The following questions are addressed by the experimental evaluation using synthetic and real experimental data.

1. How does the artificial neural network approach perform compared to other standard PIV algorithms for processing high-dimensional input data such as synthetic PIV images?
2. Can deeper and more complex neural networks achieve higher accuracy?
3. How do deeper neural networks perform on PIV data of real measurement setups?
4. Do artificial neural networks generate predictions faster than cross-correlation based PIV algorithms?

4.1 Synthetic Flow Data

Our results are summarized in Figure 2 (a)-(c) and show in general that artificial neural networks can achieve the same level of accuracy compared to standard PIV algorithm. In contrast to the results presented in the literature (Rabault et al. (2017), Lee et al. (2017)) we were able to show for the first time that our CNN-based approach can even outperform the PIV algorithm on synthetic PIV images. Overall, two general trends can be identified. First, increasing the maximum particle shift from $s_{max} = 4\ px$ to $s_{max} = 10\ px$ increases the mean absolute error of the shallow networks (ANN, Inception-A) significantly by a factor of approximately 5. In contrast, the deeper neural networks (Inception-C, ResNet-A, DenseNet-A) are not comparably prone to an increased maximum particle displacement resulting in a doubled mean absolute error for $s_{max} = 10\ px$ compared to the results of a maximum particle shift $s_{max} = 4\ px$. Second, a largely varying particle density has only a minor influence on the performance of the neural networks. Thus, it can be assumed that the networks learn feature maps for all kind of different particles, i.e., variation in shape, intensity.

Note that the shallow ANN achieved a slightly higher mean absolute error compared to the literature (Rabault et al. (2017)). This deficit can be remedied by further optimizing the specific hyperparameters. However, additional optimization loops were intentionally refrained because first, this shallow ANN serves



(a) ResNet-A (b) DenseNet-A (c) Inception-C

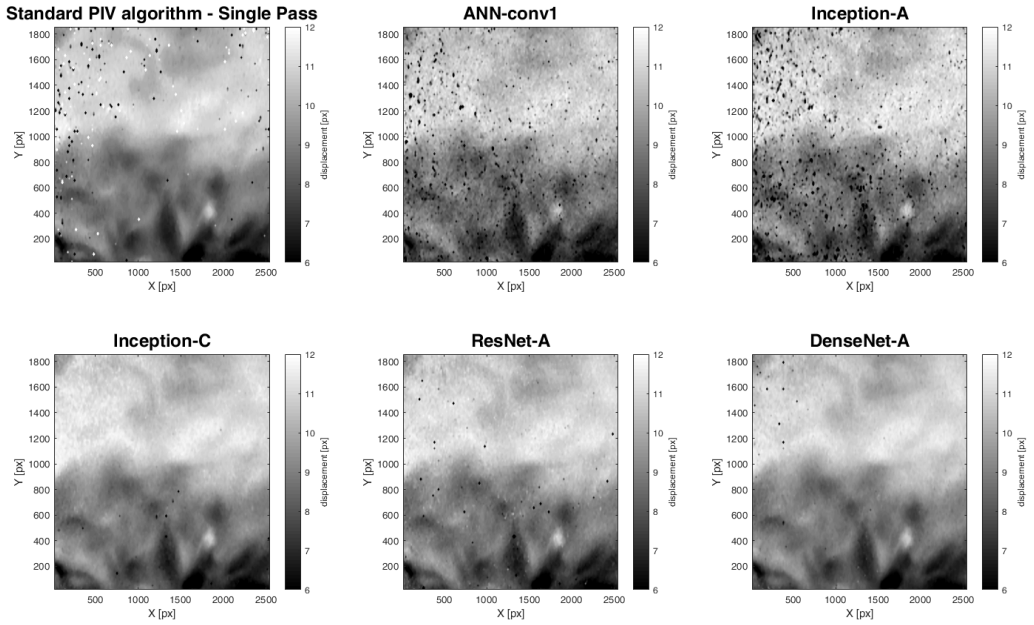
Figure 3: Comparison of the distribution of absolute mean error for IPI D4: (a) ResNet-A, (b) DenseNet-A, (c) Inception-C; the error of the different network is compared to mean and variance of the absolute mean error generated by PIVview (red) and PascalPIV (black)

as a statistical baseline for comparison with deeper architectures and second, the general performance of different ultra-deep network architectures sharing the same hyperparameters is compared. In general, the results in Figure 2 illustrate that ANN-conv1 and Inception-A can achieve the same accuracy as a standard PIV algorithm on datasets with reduced maximum particle displacement, D1 and D3, but cannot provide a sufficient accuracy in more challenging tasks. Figure 2 evidences that deeper neural networks can outperform standard PIV algorithms, but it does not become clear which architecture performs best given the same hyperparameters. On some datasets, e.g., ALA D2, ALA D4, or NoJH D3, Inception-C computes slightly smaller errors while DenseNet-A (NoJH D4, IPI D2) or ResNet-A (ALA D3) achieve the highest accuracy on others. In general, the absolute mean errors of the current deep neural network approaches (Inception-C, ResNet-A, DenseNet-A) are comparable. Figure 2(a) illustrates that all three deep neural network architectures clearly outperform the standard PIV algorithm, while Figure 2(b) shows that the same architectures compute predictions with errors in the same order of magnitude as standard PIV algorithms. Remember that the neural networks predict the displacement vectors of NoJH based on training with a reduced number of labels. Thus, it can be assumed that the additional information contained in the Jacobian and Hessian might facilitate neural networks to approximate mapping functions that connect image data to the underlying flow physics. In other words, it means that a neural network could be used to extract an approximation of the underlying flow physics data either numerically or experimentally. However, this statement requires further proof which will be part of our future work. Figure 2(c) shows that the absolute mean error of D2 and D4 of the deep architectures is slightly higher compared to standard PIV algorithms, while the predictions for D1 and D3 are equally accurate.

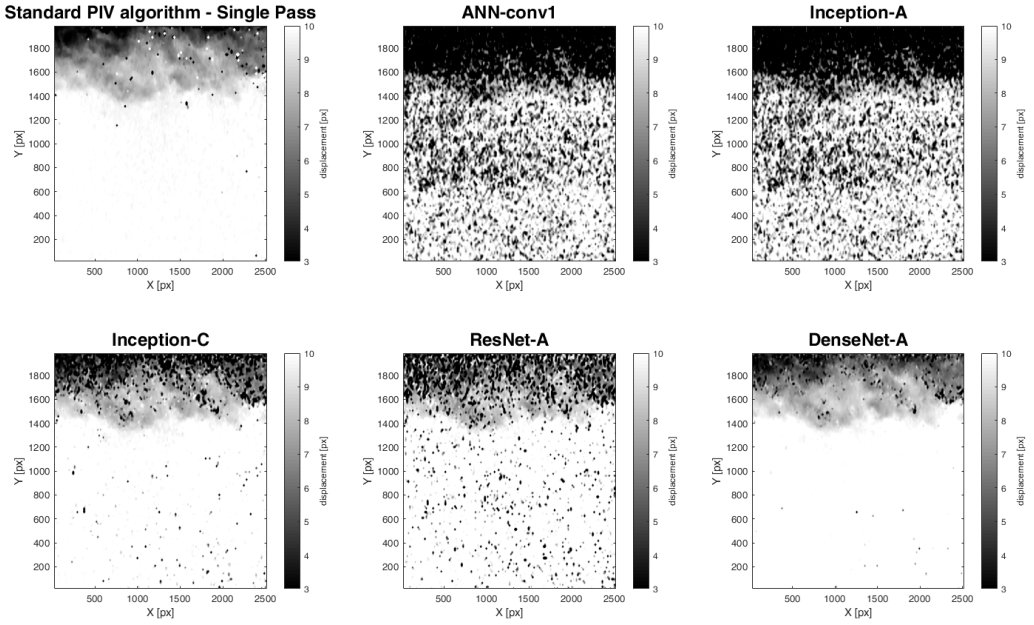
The error distributions with respect to the particle displacement, as shown in Figure 3, reveal that the absolute mean error computed by our deep neural networks are spread homogeneously. The mean of the absolute mean error of ResNet-A is slightly biased and the variance is slightly higher compared to PIVview (cf. 3(a)). Surprisingly, the bias of the absolute mean error as well as the variance of DenseNet-A are higher than the ones of ResNet-A and PIVview (cf. 3(b)). The variance of the absolute mean error of Inception-C is on the same order of magnitude compared to a commercial software tool, however, the mean correlates linearly with the particle displacements meaning that negative displacements are overpredicted while positive displacements are underpredicted. Although the deep neural networks can generate accurate predictions, they cannot reach the performance of our self-developed high-performance PIVcode PascalPIV. The variance of the absolute mean is lower by a factor of 10-100 compared to the neural networks. However, predicting particle displacements of PIV images using deep neural networks is at least 130 times faster. In comparison, the deep neural networks needed 12-47 seconds to compute 10^5 images, while the commercial tool needed for the same dataset 10 minutes and the in-house PIV processing code even 100 minutes.

4.2 Qualitative evaluations

The network structures are used to evaluate their prediction of real flow cases compared to standard PIV algorithms. Note that the networks have been trained and validated using only synthetically generated training sets, meaning that the neural networks have by no means been adapted to real flow data. Afterwards, the trained neural networks were used to generate predictions for two flow cases: The first study contains phase-locked PIV measurements in an actively drag-reduced turbulent boundary layer described in Roggenkamp et al. (2019). The second measurement setup investigated streamline segment scaling in a turbulent wavy channel flow (Rubbert et al. (2017)).



(a) Particle displacement fields (x-direction) of the turbulent wavy channel flow



(b) Particle displacement fields (x-direction) of a turbulent boundary layer

Figure 4: Network predictions for real flow cases compared to a standard single-pass PIV algorithm

The predictions of the trained neural networks are shown in Figure 4. Note that no further post-processing, e.g., outlier detection, interpolation or filtering, was applied. In general, all networks are capable of predicting the overall flow topology only varying in the number of spurious vectors. The deep neural networks show a remarkably good agreement compared to a standard single-pass PIV algorithm and even achieve less spurious vectors (cf. 4(a) Inception-C, ResNet-A, DenseNet-A). In contrast, the shallow neural networks cannot match the accuracy of an available algorithm and show a significant amount of spuriously computed velocity vectors mostly underpredicting the true value (dark spots) (cf. 4(a) ANN-conv1, Inception-A). The number of strong overpredictions (bright spots) as seen in the results of the standard PIV algorithm is negligible due to the nature of convolutional neural networks. It is well reported (Martius and Lampert (2016), Trask et al. (2018)) that CNNs struggle to extrapolate a relationship from observable numeric patterns due to the use of non-linear activation functions (ReLu or Leaky-ReLu in our case) in the hidden layers. However, such activations are a crucial part of learning abstract relations between input data and labels. Hence, CNNs prove to be good in detecting and extracting numerical patterns seen within the training data but fail to extrapolate to new conditions. This lack of extrapolation capability is in some way beneficial in PIV applications since strong overpredicted velocity vectors are simply not possible. This, however, also reduces the generalization ability of trained PIV networks meaning that experiments in which the particle shift is beyond the specific maximum displacement might not be evaluated with high accuracy.

Considering the second measurement setup, one can see that the shallow neural networks are not capable of generating accurate predictions. A large number of spuriously computed velocity vectors interfere with the predicted underlying flow leading to a very noisy flow field prediction (cf.4(b) ANN-conv1, Inception-A). The high inter-layer communication inherent to DenseNet-A (cf. 4(b) DenseNet-A) allows this network architecture to compute predictions with a low absolute mean error since learning of redundant feature maps is minimized. The generalization ability of our plain residual neural network seems to be insufficient to predict the particle displacement as accurate as the other two deep architectures. In contrast, the Inception-C network benefits from its very wide structure considering spatially spread feature clusters even in deeper layers. As a consequence, Inception-C reaches the same performance as DenseNet-A on the synthetic test sets and predicts the flow characteristics of the actively drag-reduced turbulent boundary layer with a similar accuracy compared DenseNet-A.

5 Conclusion

This manuscript presented an evaluation of deep neural network architectures that can be used for end-to-end learning in PIV applications. The current deep neural networks showed a performance comparable to standard PIV algorithms on all datasets. Furthermore, it has been shown that the current approach of deep neural networks accurately predicts particle displacements for different maximum pixel shifts. Especially wide (Inception-C) and deep networks with strong inter-layer communication (DenseNet-A) are suitable for learning mapping functions between high-dimensional input data, PIV image pairs, and the underlying flow physics. Beside the impressive results of CNNs learning to generate correct predictions on particle displacements, the presented approach is a first step. Sophisticated and highly optimized PIV algorithms achieve, at time writing, a more robust performance with lower variance. In future work, we will focus on improving the generalization ability of the trained neural networks. Improvements might be achieved using deeper residual networks (100+ layers) or aggregated residual networks applying some aggregated transformations in a grouped convolution manner as proposed by Xie et al. (2016) which enables a residual network to become not only deeper but also significantly wider. Potentially, neural arithmetic logic units composed of neural accumulators as introduced by Trask et al. (2018) might help in improving the networks extrapolation capability.

References

Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, 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, and Zheng X (2015) TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org

Arora S, Bhaskara A, Ge R, and Ma T (2013) Provable bounds for learning some deep representations. *CoRR* abs/1310.6343

Beck AD, Flad DG, and Munz C (2018) Deep neural networks for data-driven turbulence models. *CoRR* abs/1806.04482

Chollet F et al. (2015) Keras. `https://keras.io`

Couchot JF, Deschinkel K, and Salomon M (2013) Active mems-based flow control using artificial neural network. *Mechatronics* 23:898 – 905. 1. Fractional Order Modeling and Control in Mechatronics 2. Design, control, and software implementation for distributed MEMS (dMEMS)

Glorot X, Bordes A, and Bengio Y (2011) Deep sparse rectifier neural networks. in G Gordon, D Dunson, and M Dudk, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. volume 15 of *Proceedings of Machine Learning Research*. pages 315–323. PMLR, Fort Lauderdale, FL, USA

Goodfellow I, Bengio Y, and Courville A (2016) *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, and Bengio Y (2014) Generative adversarial nets. in Z Ghahramani, M Welling, C Cortes, ND Lawrence, and KQ Weinberger, editors, *Advances in Neural Information Processing Systems 27*. pages 2672–2680. Curran Associates, Inc.

He K, Zhang X, Ren S, and Sun J (2015a) Deep residual learning for image recognition. *CoRR* abs/1512.03385

He K, Zhang X, Ren S, and Sun J (2015b) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR* abs/1502.01852

Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, and Salakhutdinov R (2012) Improving neural networks by preventing co-adaptation of feature detectors. *CoRR* abs/1207.0580

Huang G, Liu Z, and Weinberger KQ (2016) Densely connected convolutional networks. *CoRR* abs/1608.06993

Kingma D and Ba J (2014) Adam: A method for stochastic optimization. *International Conference on Learning Representations*

Krizhevsky A, Sutskever I, and Hinton GE (2012) Imagenet classification with deep convolutional neural networks. in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12. pages 1097–1105. Curran Associates Inc., USA

Lecordier B and Westerweel J (2004) The europiv synthetic image generator (s.i.g.). in M Stanislas, J Westerweel, and J Kompenhans, editors, *Particle Image Velocimetry: Recent Improvements*. pages 145–161. Springer Berlin Heidelberg, Berlin, Heidelberg

Lee Y, Yang H, and Yin Z (2017) PIV-DCNN: cascaded deep convolutional neural networks for particle image velocimetry. *Experiments in Fluids* 58:171

Martius G and Lampert CH (2016) Extrapolation and learning equations. *CoRR* abs/1610.02995

Nobach H and Honkanen M (2005) Two-dimensional gaussian regression for sub-pixel displacement estimation in particle image velocimetry or particle position estimation in particle tracking velocimetry. *Experiments in Fluids* 38:511–515

Rabault J, Kolaas J, and Jensen A (2017) Performing particle image velocimetry using artificial neural networks: a proof-of-concept. *Measurement Science and Technology* 28:125301

Raffel M, Willert CE, Scarano F, Kähler C, Wereley ST, and Kompenhans J (2018) *Particle Image Velocimetry*. Springer International Publishing

Redmon J and Farhadi A (2016) YOLO9000: better, faster, stronger. *CoRR* abs/1612.08242

Roggenkamp D, Li W, Klaas M, and Schröder W (2019) Influence of spanwise transversal surface wave on coherent structures in turbulent boundary layers. *Aerospace Science and Technology* 86:387 – 400

Rubbert A, Hennig F, Klaas M, Pitsch H, Schröder W, and Peters N (2017) Streamline segment scaling behavior in a turbulent wavy channel flow. *Experiments in Fluids* 58:10

Scarano F and Riethmuller ML (1999) Iterative multigrid approach in piv image processing with discrete window offset. *Experiments in Fluids* 26:513–523

Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, and Lecun Y (2013) Overfeat: Integrated recognition, localization and detection using convolutional networks. *http://arxivorg/abs/13126229*

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, and Hassabis D (2016) Mastering the game of Go with deep neural networks and tree search. *Nature* 529:484–489

Szegedy C, Ioffe S, and Vanhoucke V (2016) Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR* abs/1602.07261

Szegedy C, Liu W, Jia Y, Sermanet P, Reed SE, Anguelov D, Erhan D, Vanhoucke V, and Rabinovich A (2014) Going deeper with convolutions. *CoRR* abs/1409.4842

Trask A, Hill F, Reed S, Rae JW, Dyer C, and Blunsom P (2018) Neural arithmetic logic units. *CoRR* abs/1808.00508

Westerweel J and Scarano F (2005) Universal outlier detection for PIV data. *Experiments in Fluids* 39:1096–1100

Xie S, Girshick RB, Dollár P, Tu Z, and He K (2016) Aggregated residual transformations for deep neural networks. *CoRR* abs/1611.05431

Zeiler MD and Fergus R (2013) Visualizing and understanding convolutional networks. *CoRR* abs/1311.2901