Data recovery in computational fluid dynamics through deep image priors

M. T. Henry de Frahan\textsuperscript{a,*}, R. W. Grout\textsuperscript{a}

\textsuperscript{a}High Performance Algorithms and Complex Fluids, Computational Science Center, National Renewable Energy Laboratory, 15013 Denver W Pkwy, ESIF301, Golden, CO 80401, USA

Abstract

One of the challenges encountered by computational simulations at exascale is the reliability of simulations in the face of hardware and software faults. These faults, expected to increase with the complexity of the computational systems, will lead to the loss of simulation data and simulation failure and are currently addressed through a checkpoint-restart paradigm. Focusing specifically on computational fluid dynamics (CFD) simulations, this work proposes a method that uses a deep convolutional neural network to recover simulation data. This data recovery method (i) is agnostic to the flow configuration and geometry, (ii) does not require extensive training data, and (iii) is accurate for very different physical flows. Results indicate that the use of deep image priors for data recovery is more accurate than standard recovery techniques, such as the Gaussian process regression (GPR), also known as Kriging. Data recovery is performed for two canonical fluid flows: laminar flow around a cylinder and homogeneous isotropic turbulence. For data recovery of the laminar flow around a cylinder, results indicate similar performance between the proposed method and GPR across a wide range of mask sizes. For homogeneous isotropic turbulence, data recovery through the deep convolutional neural network exhibits an error in relevant turbulent quantities approximately three times smaller than that for
the GPR. Forward simulations using recovered data illustrate that the enstrophy decay is captured within 10% using the deep convolutional neural network approach. Although demonstrated specifically for data recovery of fluid flows, this technique can be used in a wide range of applications, including particle image velocimetry, visualization, and computational simulations of physical processes beyond the Navier-Stokes equations.

**Keywords:** data recovery, fault tolerance, Gaussian process regression, deep convolutional neural network, computational fluid dynamics

---

1. **Introduction**

As modern computational efforts reach exascale, hardware and software faults will increasingly cause difficulty in completing simulations [1]. Current research in hardware systems and software frameworks [2, 3, 4, 5, 6, 7] continues to develop techniques to detect and anticipate system failures; however, assuming that a fault has been detected and signaled, the data loss from these failures will require data recovery processes [8]. This work addresses this challenge. Current computational codes rely on a checkpoint and restart paradigm to recover from faults, requiring either significant memory consumption for frequent checkpoints or large resimulation efforts. The ability to recover the missing data without resorting to data checkpoints has the potential to increase simulation resilience.

In the context of computational fluid dynamics (CFD), data recovery has been explored using a variety of machine learning approaches. Gappy proper orthogonal decomposition (POD) has shown particular success in reconstructing missing data [9] [10] [11]. This approach relies on combining POD with least-squares estimates [12] [13] and data from direct numerical simulations snapshots. Venturi and Karniadakis [11] expanded on the methods proposed by Everson and Sirovich [9] and used this technique to reconstruct missing data of unsteady flow past a cylinder. Other approaches rely on Gaussian process regression (GPR), often referred to as Kriging in geophysics, which is a reconstruction technique that uses the mean and covariance of the Gaussian processes prior.
The prior’s covariance is determined by a kernel whose hyperparameters are optimized using training data. Gunes et al. [14] compare POD-based and GPR-based solution reconstruction and show that GPR interpolations are particularly effective for unsteady flows (including instability regions), whereas POD-based methods are advantageous when the temporal resolution is high. In addition to these methods, Lee et al. [8] proposed a “resimulation” method in which the missing data region is resimulated using appropriate initial and boundary conditions. This new method is evaluated for lid-driven cavity flows and flows past a cylinder at low Reynolds numbers. Lee et al. [15] combined the gap-tooth algorithm, previously developed for dynamic systems [16], multiresolution information fusion, and auxiliary data to construct a general framework for fault-tolerant CFD. The method is demonstrated to work well for simulations of the heat equation and lid-driven cavity flow.

Recently, the deep learning community has been proposing methods for data recovery in the field of inverse image reconstruction problems, which include denoising, inpainting, and super-resolution, [17, 18, 19, 20, 21, 22, 23]. Deep convolutional neural networks, particularly generative adversarial networks, have been very successful at solving this class of problems [17]. Inpainting is of particular relevance to our objective of reconstructing flow solutions. The objective of inpainting is to fill in missing portions of a damaged image such that the result is indistinguishable from the original image. Various generative adversarial neural networks have been proposed for image inpainting with notable success [24, 25, 26, 27, 28]. Generally speaking, these approaches have relied on training deep neural networks with an extensive and large data set of images such that the network learns image priors that it can use in other configurations. The deep learning methodology used in this work was first developed by Ulyanov et al. [29] for various inverse image reconstruction problems, including inpainting. In contrast with previous image reconstruction solution with deep neural networks, Ulyanov et al. [29] showed that “contrary to the belief that learning is necessary for building good image priors, a great deal of image statistics are captured by the structure of a convolutional image generator independent of
learning.” Instead of training the neural networks with a large database of images, the authors use untrained neural networks to fit single degraded images, using the network weights as parameters for solving the image reconstruction problem.

In this work, we use deep convolutional neural networks, such as those proposed by Ulyanov et al. [29], for spatial reconstruction of the flow solution for simulations wherein some type of fault led to loss of data, e.g., processor failure. In contrast to gappy POD [11], we assume that the current gappy data are the only available data for the reconstruction procedure. This assumption is relevant to large simulations where it is computationally expensive to reload data residing on the file system and the reconstruction process is restricted to data already in memory. One advantage of using deep convolutional neural networks is that this approach avoids eigenmode decompositions for solution reconstruction, which could restrict the applicability or translation of the method to new configurations. As illustrated in this work, the method proposed here is not specific to the flow configuration and does not require multiple training data samples.

This paper is organized as follows. In Section 2, we present the problem formulation and define the objective function for the data recovery problem. In Section 3, we detail the architecture of the deep convolutional neural network used to perform the data recovery process for fluid flows. In Section 4, we present our results by evaluating the neural network’s ability to perform data recovery for two canonical flows: laminar flow past a cylinder, Section 4.1, and homogeneous isotropic turbulence, Section 4.2. These results are compared with data recovery performed through GPR. Finally, conclusions and future work are presented in Section 5.

2. Problem formulation

In this work, we evaluate the performance of deep convolutional neural networks for data recovery in CFD. Deep convolutional neural networks have shown
particular success for solving the image reconstruction problem \cite{17}. Image reconstruction is analogous to data recovery because they share a similar objective to provide synthetic data that closely match the missing data. The image reconstruction problem can be cast as an optimization problem:

$$\min_{x} E(x; x_0) + R(x)$$  \hspace{1cm} (1)

where $x$ is the original image that needs to be recovered, $x_0$ is the corrupted image, $E(x; x_0)$ is the task-dependent data term, and $R(x)$ is the image prior. In the case of inpainting, the task-dependent data term is:

$$E(x; x_0) = ||(x - x_0) \circ m||^2$$  \hspace{1cm} (2)

where $\circ$ is the Hadamart product, $m \in \{0,1\}^{h \times w}$ represents the binary mask, and $h$ and $w$ are the image height and width. The image prior is usually captured through the training of convolutional neural networks using a large image database. In the approach proposed by Ulyanov et al. \cite{29}, $R(x)$ is replaced by a parameterization such that the optimization problem becomes:

$$\min_{\theta} E(f_\theta(z); x_0)$$  \hspace{1cm} (3)

where $f$ represents the convolutional neural network with parameters $\theta$ that is initialized randomly, and $z$ is a fixed input. The fixed input for the neural network can take many forms but is usually chosen to be random uniform noise or smoothly varying data. Note that the neural network input is fixed. Given a deteriorated image, the neural network effectively learns, by backpropagation and network parameter tuning, the encoding necessary to map the fixed input to an output, i.e., the recovered image, which minimizes the loss function, Equation (3).

We emphasize that physical constraints are not explicitly included in the data recovery process. This has the advantage of enabling a reconstruction technique that does not depend on the physical nature of the problem. Higher fidelity can be achieved, however, by incorporating physical constraints, as suggested in \cite{30,31}. The work presented here focuses on two-dimensional reconstruction, though there is no inherent methodological limitation to reconstructing three-dimensional data directly.
3. Neural network architecture

The network chosen for this work is a convolutional neural network that exhibits an encoder-decoder architecture with approximately 2 million tunable parameters and no skip connections, Figure 1a. This architecture enables the network to encode the input in the latent space and then decode the latent space representation into the reconstructed image. The nonlinear activation function used in the network is LeakyReLU [32]. Downsampling was performed through simple striding in the convolution procedure, Figure 1b, and upsampling was done through nearest-neighbor upsampling, Figure 1c. The number of filters in the downsampling and upsampling layers, \( n_f \), was kept fixed at 128, and the kernel size, \( k \), was fixed at 3. Experiments showed that using a fixed smoothly varying input \( z \) for the neural network imposes a smoothness prior, which is beneficial for data recovery for fluid flows. The optimization process was performed using Adam [33]. The implementation was done in PyTorch [34], and the learning process was computed on a Tesla V100 graphics processing unit. The number of iterations for all the experiments was 2000, thereby reducing the loss function by three orders of magnitude.

4. Results

To demonstrate the efficacy of using deep neural networks for data recovery, we investigate two types of flows: laminar flow over a cylinder for data recovery of large flow scales and homogeneous isotropic turbulence for data recovery of flows spanning a wide range of scales. We compare the deep convolutional neural network results with GPR. The sample points used to train the GPR are located in the region surrounding the mask with a depth of 10 cells, similar to [8]. Beyond a depth of 10 cells surrounding the masked regions, training the regressor becomes computationally intractable because the GPR complexity is \( O(n^3) \), where \( n \) is the number of training points. A radial basis function is used as the GPR kernel.
Figure 1: Deep convolutional neural network used for data recovery.
4.1. Laminar flow around a cylinder

The first numerical tests of the data recovery process are performed for the laminar flow around a cylinder ($Re = 200$). The simulation is performed using Nalu-Wind, a low Mach Navier-Stokes solver leveraging the Trilinos libraries\textsuperscript{1} and the $t = 234$ s snapshot is used for the numerical tests, at which time the vortices behind the cylinder were fully developed. Masks simulating data loss because of processor failure are generated in the cylinder wake. To capture typical domain decomposition methods for structured grids, the masks are square boxes and vary in size depending on the number of processors used for the simulation. The mask box length, $L_m$, ranged from $0.5D$ to $5D$, where $D$ is the cylinder diameter, and the masks are located at 40 random locations in the cylinder wake, leading to 240 unique masks to be applied to the simulation data. For the reconstruction process, reflection padding is used for the boundary conditions.

An example reconstruction is presented in Figure\textsuperscript{2} for $L_m = 2D$, where the deep convolutional neural network presents a slightly better reconstructed velocity field than GPR. Specifically, the partially masked vortex is more accurately reconstructed using the deep convolutional neural network. The average $L_2$ error norm for the velocity fields as a function of $L_m$ is presented in Figure\textsuperscript{3}. Both reconstruction techniques, GPR and the deep convolutional neural network, present similar error profiles. At higher $L_m$ the neural network performs slightly better than GPR for the $x$-direction velocity, whereas it performs similarly for all other lengths. Given the structured nature of the flow field, it is unsurprising that GPR performs well at moderate mask sizes, given previously published results \cite{14}.

4.2. Homogeneous isotropic turbulence

For these numerical tests, we use two-dimensional slices of homogeneous isotropic turbulence with a Taylor microscale Reynolds number $Re_\lambda = \rho_0 u' \lambda / \mu = \ldots$

\textsuperscript{1}https://github.com/Exawind/nalu-wind
Figure 2: Velocity magnitude for laminar flow around a cylinder where the mask box length is twice the cylinder diameter, $L_m = 2D$.

Figure 3: Average $L_2$ error norm as a function of mask box length, $L_m$, for laminar flow past a cylinder ($Re = 200$). Red squares: deep convolutional neural network; green diamonds: GPR.
133, where \( \rho_0 \) is the reference density, \( u' = \sqrt{\frac{u_i u_i}{3}} \) is the initial mean fluctuating velocity, \( \lambda = \frac{u_i^2}{(\frac{\partial u_i}{\partial x})^2} \) is the Taylor microscale, and \( \mu \) is the dynamic viscosity; a turbulent Mach number \( M_t = u_0/c_s = 0.1 \), where \( c_s \) is the speed of sound; and a Prandtl number \( Pr = \frac{\nu c_p}{k} = 0.71 \), where \( c_p \) is the heat capacity at constant pressure, and \( k \) is the thermal conductivity. The reference temperature and pressure are 300 K and 1 atmosphere and the ideal gas equation of state is used to relate the thermodynamic quantities. The domain ranges from \([0, 2\pi]\) with periodic boundary conditions. In this work, we use PeleC\(^2\) an explicit compressible Navier-Stokes flow solver based on the AMReX library\(^3\) to demonstrate the data recovery process. For the reconstruction process, periodic, i.e., wrapped, padding was used for the boundary conditions.

Initial two-dimensional data slices are generated by slicing in each direction a numerical simulation of homogeneous isotropic turbulence at a resolution of 64 cells in each direction, leading to 192 unique slices. The velocities in each direction are assigned an input channel for the neural network. Masks simulating data loss because of processor failure are generated independently, following typical domain decomposition. We explore two different parameters associated with the mask generation process. The first is the total percentage of missing data, \( f \), ranging from 6.25% to 25%. The second is the length scale associated with each block of missing data, \( L_m \), ranging from 3.125% to 50% of the domain length, or 0.74\( \lambda \) to 11.87\( \lambda \). For each pair of parameters \( f \) and \( L_m \), we randomly generate ten different masks, resulting in 130 unique masks. These are randomly applied to 100 initial slices, resulting in 1300 slices requiring reconstruction. For each of these deteriorated slices, the neural network parameters are tuned to optimize the reconstruction loss function, Equation (3). The data from the resulting recovered slice are then used for comparison with the original slice. The data recovery process is illustrated for one slice in Figure 4.

The average error in \( u' \) and \( \lambda \) for all the reconstructions is approximately

---

\(^2\)https://github.com/AMReX-Combustion/PeleC
\(^3\)https://amrex-codes.github.io
Figure 4: Velocity magnitude in homogeneous isotropic turbulence illustrating the data recovery process, where 25% of the original data is missing and the length scale associated to each missing block is $6.25\%$ of the domain ($1.5\lambda$).

Figure 5: Energy spectrum and error as a function of wavenumber $k$. Solid red: original data; dashed green: deep convolutional neural network; dot-dashed blue: GPR.

three times larger for the GPR process compared to the deep convolutional neural network. For all the slices, individual energy spectra are calculated for the original data and the data recovered through the deep convolutional neural network and GPR. The average energy spectrum is presented in Figure 5a. The average error from the GPR reconstruction increases at high wavenumbers, indicating that it not able to accurately capture the smallest scales of turbulence, Figure 5b. This behavior is not exhibited with the deep neural network reconstruction, and the error increases slightly as a function of wavenumber.

Velocities in slices of the original and recovered data are used as initial con-
Figure 6: Normalized enstrophy as a function of time where 25% of the original data is missing. Solid black: original data; solid red: $L_m = 0.74\lambda$; dashed green: $L_m = 1.48\lambda$; dot-dashed blue: $L_m = 2.97\lambda$; dotted orange: $L_m = 5.94\lambda$; dot-dot-dashed purple: $L_m = 11.87\lambda$.

Figure 6: Normalized enstrophy as a function of time where 25% of the original data is missing. Solid black: original data; solid red: $L_m = 0.74\lambda$; dashed green: $L_m = 1.48\lambda$; dot-dashed blue: $L_m = 2.97\lambda$; dotted orange: $L_m = 5.94\lambda$; dot-dot-dashed purple: $L_m = 11.87\lambda$.

Figure 6 illustrates the decay of normalized enstrophy, $\omega = \frac{\lambda^2}{u'v'} \int_V (\nabla \times u)^2 \, dV$, for simulations where $f = 25\%$ and $L_m \in [0.74\lambda, 11.87\lambda]$, or 3.125% to 50% of the domain length. GPR reconstruction exhibits a significantly different enstrophy decay from the original data, Figure 6b. For the deep convolutional neural network, the enstrophy decay is well captured at all mask sizes, with slightly less accuracy for large $L_m$, Figure 6a. Across the range of mask sizes, the normalized error at $t = 1\tau$ in kinetic energy and enstrophy is less than 10% for the deep convolutional neural network and around 20% for the GPR reconstruction. These differences in reconstruction procedures are attributed to the deep convolutional neural network’s ability to preserve the energy spectra and accurately represent all the length scales.
5. Conclusion

This work evaluated the use of deep convolutional neural networks for data recovery of fluid flows in the context of data loss because of hardware or software faults. The method proposed here leverages an encoder-decoder deep convolutional neural network to transform a fixed input to a recovered output using only deteriorated data and eschewing a training database. Comparisons were performed with Gaussian process regression, a standard data recovery algorithm often referred to as Kriging. Data recovery was performed on two different canonical flow configurations: laminar flow around a cylinder and homogeneous isotropic turbulence. For data recovery of the laminar flow around a cylinder, results indicate similar performance between the proposed method and GPR across a wide range of mask sizes. For homogeneous isotropic turbulence, data recovery through the deep convolutional neural network exhibits an error in relevant turbulent quantities approximately three times smaller than that for the GPR. Forward simulations using recovered data illustrate that the enstrophy decay is accurately captured using the deep convolutional neural network approach.

We emphasize that the work presented here is not necessarily beholden to the specific inpainting technique used. The deep learning community has developed many different methods for image inpainting that can be used for data recovery, and it is expected that state-of-the-art methodologies would perform comparably well. Rather, the use of deep image priors as first proposed by Ulyanov et al. [29] and investigated here provides a convenient framework to perform data recovery of fluid flows because it does not require pretraining the neural network to construct image priors for different flows. It is therefore agnostic to the specific flow configuration, and the same framework can be used for very different flows. This technique, however, does necessitate the solution of an optimization problem for each data recovery task. Future work will investigate alleviating this through partial pretraining, transfer learning, and perceptual loss functions.
This work — including data sets, demonstration notebooks, analysis scripts, and figures — can be publicly accessed at the project’s GitHub page.\footnote{https://github.com/NREL/deep-image-prior-cfd} Traditional machine learning algorithms were implemented through scikit-learn \footnote{35} and the deep learning algorithms through PyTorch \footnote{34}.

Acknowledgments

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by U.S. Department of Energy Office of Science and National Nuclear Security Administration. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

This research was supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of two DOE organizations – the Office of Science and the National Nuclear Security Administration – responsible for the planning and preparation of a capable exascale ecosystem – including software, applications, hardware, advanced system engineering, and early testbed platforms – to support the nation’s exascale computing imperative.

References

References

[1] D. L. Brown, P. Messina, Scientific Grand Challenges: Crosscutting Technologies for Computing at the Exascale, Tech. Rep., Office of Advanced Scientific Computing Research, Washington, DC, 2010.
[2] W. Gropp, E. Lusk, Fault tolerance in message passing interface programs, Int. J. High Perform. Comput. Appl. 18 (3) (2004) 363–372, ISSN 10943420, doi:10.1177/1094342004046045

[3] M. Hoemmen, M. A. Heroux, Fault-tolerant iterative methods via selective reliability, Proc. 2011 Int. Conf. High Perform. Comput. Networking, Storage Anal.

[4] K. Teranishi, M. A. Heroux, Toward Local Failure Local Recovery Resilience Model using MPI-ULFM, Proc. 21st Eur. MPI Users’ Gr. Meet. - EuroMPI/ASIA ’14 doi:10.1145/2642769.2642774

[5] F. Cappello, A. Geist, W. Gropp, S. Kale, B. Kramer, M. Snir, Toward Exascale Resilience: 2014 update, Supercomput. Front. Innov. 1 (1), ISSN 23138734, doi:10.14529/jsfi140101 URL http://superfri.org/superfri/article/view/14

[6] M. Gamell, K. Teranishi, M. A. Heroux, J. Mayo, H. Kolla, J. Chen, M. Parashar, Local recovery and failure masking for stencil-based applications at extreme scales, in: Proc. Int. Conf. High Perform. Comput. Networking, Storage Anal. - SC ’15, ISBN 9781450337236, ISSN 21674337 21674329, doi:10.1145/2807591.2807672, 2015.

[7] R. Grout, H. Kolla, M. Minion, J. Bell, Achieving algorithmic resilience for temporal integration through spectral deferred corrections, Commun. Appl. Math. Comput. Sci. 12 (1) (2017) 25–50, ISSN 2157-5452, doi:10.2140/camcos.2017.12.25 URL http://msp.org/camcos/2017/12-1/p02.xhtml

[8] S. Lee, I. G. Kevrekidis, G. E. Karniadakis, Resilient algorithms for reconstructing and simulating gappy flow fields in CFD, Fluid Dyn. Res. 47 (5) (2015) 051402, ISSN 0169-5983, doi:10.1088/0169-5983/47/5/051402 URL http://stacks.iop.org/1873-7005/47/i=5/a=051402?key=crossref.e7c216f251e24b00c0b43a2e7ff5ed6f
[9] R. M. Everson, L. Sirovich, The Karhunen-Loeve transform for incomplete data, J. Opt. Soc. Am. A 12 (8) (1995) 1657–1664.

[10] B.-T. Tan, K. Willcox, M. Damodaran, Applications of Proper Orthogonal Decomposition for Inviscid Transonic Aerodynamics, Tech. Rep. 2, URL http://dspace.mit.edu/bitstream/handle/1721.1/3694/HPCS002.pdf?sequence=2 2003.

[11] D. Venturi, G. E. Karniadakis, Gappy data and reconstruction procedures for flow past a cylinder, J. Fluid Mech. 519 (2004) 315–336, ISSN 0022-1120, doi:10.1017/S0022112004001338 URL http://www.journals.cambridge.org/abstract_S0022112004001338.

[12] F. Yates, The analysis of replicated experiments when the field results are incomplete., Emp. J. Exp. Agric. 1 (2) (1933) 129–142.

[13] R. J. a. Little, D. B. Rubin, Statistical Analysis with Missing Data, ISBN 0471183865, doi:10.2307/1533221 2002.

[14] H. Gunes, S. Sirisup, G. E. Karniadakis, Gappy data: To Krig or not to Krig?, J. Comput. Phys. 212 (1) (2006) 358–382, ISSN 00219991, doi:10.1016/j.jcp.2005.06.023.

[15] S. Lee, I. G. Kevrekidis, G. Em, G. E. Karniadakis, A general CFD framework for fault-resilient simulations based on multi-resolution information fusion, J. Comput. Phys. 347 (2017) 290–304, ISSN 00219991, doi:10.1016/j.jcp.2017.06.044 URL http://dx.doi.org/10.1016/j.jcp.2017.06.044 http://linkinghub.elsevier.com/retrieve/pii/S0021999117304989.

[16] C. Gear, J. Li, I. G. Kevrekidis, The gap-tooth method in particle simulations, Phys. Lett. A 316 (3-4) (2003) 190–195, ISSN 03759601, doi:10.1016/j.physleta.2003.07.004 URL http://linkinghub.elsevier.com/retrieve/pii/S0375960103011502.
[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Nets, Adv. Neural Inf. Process. Syst. 27 (2014) 2672–2680 ISSN 10495258, doi:10.1017/CBO9781139058452 URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[18] H. C. Burger, C. J. Schuler, S. Harmeling, Image denoising: Can plain neural networks compete with BM3D?, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., ISBN 9781467312264, ISSN 10636919, 2392–2399, doi:10.1109/CVPR.2012.6247952 2012.

[19] A. Dosovitskiy, J. T. Springenberg, T. Brox, Learning to generate chairs with convolutional neural networks, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 07-12-June, ISBN 9781467369640, ISSN 10636919, 1538–1546, doi:10.1109/CVPR.2015.7298761 2015.

[20] S. Lefkimmiatis, Non-Local Color Image Denoising with Convolutional Neural Networks, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2016) 5882–5891 doi:10.1109/CVPR.2017.623 URL http://arxiv.org/abs/1611.06757

[21] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, W. Shi, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, Conf. Comput. Vis. Pattern Recognit. (2016) 1–14 ISSN 0018-5043, doi:10.1109/CVPR.2017.19 URL http://arxiv.org/abs/1609.04802

[22] Y. Tai, J. Yang, X. Liu, Image super-resolution via deep recursive residual network, in: Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, ISBN 9781538604571, 2790–2798, doi:10.1109/CVPR.2017.298 2017.

[23] W. S. Lai, J. B. Huang, N. Ahuja, M. H. Yang, Deep laplacian pyramid networks for fast and accurate super-resolution, in: Proc. - 30th IEEE Conf.
[24] R. A. Yeh, C. Chen, T. Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, M. N. Do, Semantic Image Inpainting with Deep Generative Models, ACM Trans. Graph. 36 (4) (2016) 1–14, ISSN 07300301, doi:10.1145/3072959.3073659, URL http://dl.acm.org/citation.cfm?doid=3072959.3073659http://arxiv.org/abs/1607.07539.

[25] E. Denton, S. Gross, R. Fergus, Semi-Supervised Learning with Context-Conditional Generative Adversarial Networks (2016) 1–10ISSN 1611.06430, URL http://arxiv.org/abs/1611.06430.

[26] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, A. A. Efros, Context Encoders: Feature Learning by Inpainting, in: 2016 IEEE Conf. Comput. Vis. Pattern Recognit., IEEE, ISBN 978-1-4673-8851-1, ISSN 10636919, 2536–2544, doi:10.1109/CVPR.2016.278 URL http://arxiv.org/abs/1604.07379http://ieeexplore.ieee.org/document/7780647/ 2016.

[27] Y. Li, S. Liu, J. Yang, M.-h. Yang, Generative Face Completion, in: 2017 IEEE Conf. Comput. Vis. Pattern Recognit., IEEE, ISBN 978-1-5386-0457-1, 5892–5900, doi:10.1109/CVPR.2017.624 URL http://arxiv.org/abs/1704.05838http://ieeexplore.ieee.org/document/8100107/ 2017.

[28] K. Sasaki, S. Iizuka, E. Simo-Serra, H. Ishikawa, Joint gap detection and inpainting of line drawings, in: Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, ISBN 9781538604571, doi:10.1109/CVPR.2017.611, 2017.

[29] D. Ulyanov, A. Vedaldi, V. Lempitsky, Deep Image Prior URL http://arxiv.org/abs/1711.10925

[30] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics Informed Deep Learn-
ing (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations (Part II) (2017) 1–19, URL http://arxiv.org/abs/1711.10566.

[31] J. Sirignano, K. Spiliopoulos, DGM: A deep learning algorithm for solving partial differential equations, J. Comput. Phys. 375 (2018) 1339–1364, ISSN 00219991, doi:10.1016/j.jcp.2018.08.029, URL https://linkinghub.elsevier.com/retrieve/pii/S0021999118305527

[32] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proc. IEEE Int. Conf. Comput. Vis., vol. 2015 Inter, ISBN 9781467383912, ISSN 15505499, 1026–1034, doi:10.1109/ICCV.2015.123, 2015.

[33] D. P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization URL http://arxiv.org/abs/1412.6980.

[34] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, in: NIPS-W, 2017.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine Learning in Python, J. Mach. Learn. Res. ISSN 1271-6669, doi:10.1007/s13398-014-0173-7.2