Neural network performance enhancement for limited nuclear fusion experiment observations supported by simulations

Marko Blatzheim$^{1,2}$, Daniel Böckenhoff$^1$, Hauke Höhbe$^1$, Thomas Sunn Pedersen$^1$, Roger Labahn$^2$ and The W7-X Team$^1$

$^1$ Max Planck Institute for Plasma Physics, Wendelsteinstraße 1, Greifswald 17491, Germany
$^2$ Institute for Mathematics, University of Rostock, Ulmenstraße 69, Rostock 18057, Germany

E-mail: marko.blatzheim@ipp.mpg.de and daniel.boeckenhoff@ipp.mpg.de

Received 5 September 2018, revised 19 October 2018
Accepted for publication 9 November 2018
Published 5 December 2018

Abstract

It has recently been shown that artificial neural networks (NNs) are able to establish nontrivial connections between the heat fluxes and the magnetic topology at the edge of Wendelstein 7-X (W7-X) (Böckenhoff et al 2018 Nucl. Fusion 58 056009), a first step in the direction of real-time control of heat fluxes in this device. We report here on progress on improving the performance of these NNs. A particular challenge here is that of generating a suitable training set for the NN. At present, experimental data are sparse, and simulated data, which are much more abundant, do not match the experimental data closely. It is found that the NNs show significantly improved performance on experimental data when experimental and simulated data are combined into a common training set, relative to training performed on only one of the two data sets. It is also found that appropriate pre-processing of the data improves performance. The architecture of the NN is also discussed. Overall a significant improvement in NN performance was seen—the normalized error reduced by more than a factor of three over the previous results. These results are important since heat flux control in a W7-X, as well as in a future fusion power plant, is likely a key issue, and must start with a very limited set of experimental training data, complemented by a larger, but not necessarily fully realistic, set of simulated data.

Keywords: neural networks, Wendelstein 7-X, plasma edge, magnetic topology

(Some figures may appear in colour only in the online journal)
Wendelstein 7-X (W7-X) [5], which recently went into operation. Experimental data are still scarce, and the interactions between the edge plasma and the plasma-facing components are multifaceted and complicated, involving at least plasma physics, atomic physics, chemistry, and solid state physics, so that state-of-the-art codes such as EMC3-Eirene [6, 7] do not yet capture all the important dynamics. The focus is on developing NNs that, trained on simulated data, supplemented with a very limited number of experimental data, generalize well onto experimental data. Our specific example is the training of a NN using simulated and experimental observations of W7-X heat load patterns to reconstruct $\iota$, a property of the magnetic field at the plasma edge that determines the heat load patterns.

The future practical application of this NN will be a part of a real-time control system, ensuring the control and safety of all W7-X plasma facing components.

In the next section we briefly explain the essentials regarding W7-X, heat load pattern retrieval from infra-red cameras and a proper formulation of $\iota$, followed by a definition of the data set composition. The two used NN architectures are described and the parametrizations introduced before the NN performance is presented and analyzed.

2. Methods

2.1. Context for the neural network application

W7-X is a world-leading, relatively large experiment of the stellarator type [5] (16 m outer diameter, 30 m$^3$ confinement volume). It has a carefully tailored magnetic field configuration designed to confine hot plasma and aims to explore if this concept can be scaled up to yield net energy production from thermonuclear fusion processes. This paper focuses on the optimization of a neural network that is to become an important piece of a real-time control system for the heat and particles exhausted by the plasma onto the components especially designed to absorb these heat and particle loads. For the present studies, these plasma-intercepting components were so-called limiters (figure 1(c)), but later divertor modules are used. Perhaps the most important parameter determining the spatial heat load distribution onto these components is the magnetic winding number $\iota$. It describes how many full poloidal turns a magnetic field line performs when performing one full toroidal turn, and is around 0.9 for these studies.
Therefore, we started investigating if a NN could determine \( k \) given, as input, measurements by infrared (IR) cameras of heat load patterns. Figure 1(a) shows an overview of W7-X and IR camera views. Our first results were recently published showing that indeed this is possible [1]. The present paper is a systematic attempt to further improve the performance of NNs for this application, by investigating how the pre-processing of the input data, the character and quantity of training sets, and the NN architecture affect the performance.

Following the previous paper, we train the NNs to estimate the current in coil set B (figure 1(b)), \( I_b \), appropriately normalized, instead of \( k \) itself, but this is a technical detail. For these studies, numerical and experimental, there is a one-to-one correspondence between the two.

For details regarding the underlying physics and data origin (experimental as well as synthetic) we refer to [1]. Figure 2 shows an example of a heat load pattern on the limiter from W7-X module 5 for vacuum field line diffusion simulation (figure 2(a)) as well as IR data (figure 2(b)) at the same magnetic configuration. The right half of the limiter is shadowed from the IR camera view. Some basic characteristics are similar, e.g. the maximum heat load is located at the third limiter tile. However, in detail, the structure of the IR observation is not reproduced.

### 2.2. Data sets

The investigated data sets result from experimental and simulated scans of W7-X. The scans were performed by varying the current in one coil set, planar coil set B (see [1, section 2.3]). The simulation set \( S \) was created by the field line diffusion approach described in [1, section 2.4.2] with \( |S| = 3993 \).

The experimental set of processed IR data \( I \) comes from six different experimental magnetic configurations that were investigated. For each magnetic configuration, two to five sets of data were taken, giving a total of 16 IR videos (see [1, section 2.4.1]).

Each IR video corresponding to one experiment contributes on average 20 frames for the same value of \( I_b \), leading to a total cardinality of \(|I| = 319|\).

A subset \( I_C \subset I \) with \(|I_C| = 190\) is defined such that \( I_C \) as well as \( I \setminus I_C \) cover all six experimental settings. Each of these two sets contain either all or none of the frames of each single experiment.

A mixed set is defined as \( M = (S \cup I) \setminus I_C \).

To determine NN quality, three disjoint subsets, namely training set, validation set and test set have to be defined.

### 2.3. NN architectures

Three NN architectures were compared. Since the data set is very limited, the number of free parameters must be kept low to avoid overfitting. The first one is a fully-connected feed-forward neural network (FFNN) with 3 layers of 64 nodes each with tanh activation function and L2-regularization with a factor of 0.001. The second one is mainly based on convolutional layers [9] as shown in figure 3. Three consecutive convolutional layers are followed by two fully-connected layers with 64 and 1 nodes, respectively. The number of filters in the convolutions are in ascending order 4, 8 and 16 with a kernel slide of 1. The third NN architecture starts with an inception module as defined in [10] but instead of a very deep structure as suggested for the large ImageNet data set [11] it is followed by pooling [12], a 1 \times 1 convolutional layer and two fully-connected layers, as shown in figure 4. The number of filters in all convolutions within the inception is 4, it is 8 for the following convolution and there are 16 and 1 nodes within the last two fully-connected layers, respectively. The pool kernel within the inception module has the dimension 3 \times 3 and the pool kernel dimension is 2 \times 2 for the following inception module. All the kernel slides are for the pool following the inception, which has a slide of 2.

In the last two architectures all activation functions except for the last layer are rectified linear units (ReLU). Because the NNs are designed to solve a regression problem, the activation function of the last layer is the identity. The analysis of the parameters presented in the following section 2.4 is mainly performed using the convolution neural network. For the other two networks only examples are shown.

Weights are initialized randomly as recommended in [13] and biases are initialized as zero. The weights and biases are iteratively improved by the adam optimizer [14] during the training process. Early-stopping is applied as soon as the validation loss-function increases. The implementation is done in TensorFlow [15]. The calculations are computed on a workstation with two Intel Xeon CPU E5 – 2650 and four NVIDIA GeForce GTX 1080 Ti.

### 2.4. Parametrization

The heat load is given on an unstructured triangular grid that represents the CAD structure of one limiter. This data is transformed affine from W7-X coordinates to an orthogonal \( \xi, \eta, \zeta \)-coordinate system, where \( \xi \) points in the direction of \( \hat{n}_m \), the normalized mean of the normals of all triangles forming the limiter. The rotation is achieved by the matrix \( R \) with

![Diagram of NN structure with three convolutional layers](image-url)

**Figure 3.** Structure of NN with three convolutional layers: the grey elements represent convolutional layers with their associated kernel sizes indicated.

\[ \text{input} \rightarrow \begin{array}{c} \text{smaller if dim} < 5 \end{array} \rightarrow \begin{array}{c} 5 \times 5 \end{array} \rightarrow \begin{array}{c} \text{smaller if dim} < 3 \end{array} \rightarrow \begin{array}{c} 3 \times 3 \end{array} \rightarrow \begin{array}{c} \text{fully connected} \end{array} \rightarrow \text{output} \]
Each axis is scaled such that the minimal value in this coordinate is 0 and the maximum value is 1. An example for such an affine transformation is shown in figure 5 for a half cylinder with remote resemblance to the W7-X limiter. Only triangles within a tight bounding box around the limiter are considered for the parametrization.

2.4.1. Partitioning. The $\xi$, $\eta$-space is then divided by various 2D grids with sizes described in table 1 and shown for simulations and experiments in figures 6(a) and (b) respectively. There are no divisions in the $\zeta$ direction.

The first three columns in table 1 show one dimensional partitionings. The inception NN is applied only for inputs of dimensionality of at least $5 \times 5$. However, for the one-dimensional inputs the convolutional kernel dimensionality is reduced from that presented in figure 3 such that it does not exceed the input dimensionality.

2.4.2. Extracted characteristics. For each element $p$ of the partitionings, characteristic values can be extracted. One is the heat-load-weighted spatial mean

$$\vec{\mu}_p = \Xi_p W_p,$$

with triangle centroids

$$\Xi_p = (\xi_{pi}, \eta_{pi}, \zeta_{pi})^T \in \mathbb{R}^{3 \times np}$$

and weight vector $W_p$ defined as

$$W_p = \left( \frac{A_{pi} q_{pi}}{\sum_{j=1}^{np} A_{pj} q_{pj}} \right) \in \mathbb{R}^{np},$$

with number of triangles per partitioning $n_p$, triangle areas $A_{pi}$, and triangle heat load $q_{pi}$ (in Wm$^{-2}$). The weighted covariance matrix is defined as

$$\text{Cov}(\Xi_p) = (\Xi_p - \vec{\mu}_p \vec{J}_n^T)(\Xi_p - \vec{\mu}_p \vec{J}_n^T) \otimes W_p \in \mathbb{R}^{3 \times 3},$$

with vector of ones $\vec{J}_n$ of dimension $n_p$ and element-wise multiplication $\otimes$. Another statistical characteristic is the spatial standard deviation $\vec{\sigma}_p$ calculated as

$$\vec{\sigma}_p = \sqrt{\text{diag}(\text{Cov}(\Xi_p))},$$

with element-wise square root and the operator $\text{diag}()$ which extracts the diagonal elements of a matrix. The direction vector $\vec{\delta}_p$, calculated as the eigenvector corresponding to the largest eigenvalue $\lambda_{\text{max}}$ of $\text{Cov}(\Xi_p)$ can be characteristic as well.

This parameter is inspired by divertor heat load patterns which show a more complicated shape [16]. The last examined parameter is the relative heat load...
Table 1. Dimensions of the used partitionings and total number of elements.

| \( n_x \) | 2 | 4 | 9 | 9 | 15 | 18 | 27 | 36 | 72 | 144 |
|---|---|---|---|---|---|---|---|---|---|---|
| \( n_y \) | 1 | 1 | 1 | 5 | 6 | 8 | 10 | 12 | 15 | 30 |
| \( n_x \cdot n_y \) | 2 | 4 | 9 | 45 | 90 | 144 | 270 | 432 | 1080 | 4320 |

Three combinations of those parameters are studied as NN input: \((\mu, \sigma), (\mu, \delta)\) and \(\rho\) exclusively. In figure 6 \(\rho\) is shown for simulation and experiment of the same physical condition. Note that the input dimension of the NNs is \(n_x \times n_y \times n_{\text{input}}\), with input channels being 6 for the cases \((\mu, \sigma)\) as well as for \((\mu, \delta)\) but 1 for \(\rho\). In the case of \(\rho\) as parameter, the 1 \times 1 convolutions within the inception module (see section 2.3) perform rather trivial operations. Whether this affects the performance was not investigated. Pooling layers are used for all partitionings to achieve comparable architectures, although they may not be important for smaller partitionings. The number of free parameters resulting from the chosen NN type, partitioning and characteristic is shown in figure 7.

$$\rho_p = \frac{\hat{q}_p}{\sum_{p'=1}^{\rho} \hat{q}_{p'}}$$  \hspace{1cm} (7)

with number of partitions \(m\) and

$$\hat{q}_p = \frac{\sum_{i=1}^{m} A_{pi} q_{pi}}{\sum_{i=1}^{m} A_{pi}}$$  \hspace{1cm} (8)

defined as the absolute heat load.

We define the following notation to describe the NN settings:

\[ f(\text{train, validate, test})_{\text{parametrization, architecture}} \]

where f can be any function.

For example \(\text{rmse}(S_{90}, S_{10}, I)_{\rho=5, \text{inception}}\) refers to the root mean square error of an inception NN trained as well as validated on samples from set \(S\) and tested with experimentally observed IR data \(I\), requiring a \(9 \times 5\) input of relative intensities \(\rho\). \(S_{90} \subset S\) and \(S_{10} \subset S\) with \(S_{90} \cap S_{10} = \emptyset\) refer to two disjunct sets for training and validation consisting of 90% and 10% of the samples within \(S\) respectively. Given a single term in the brackets implies that training, validation and test data sets are disjunct and subsets of the same super set. If one describing parameter is omitted, the entirety of all possible parameters of that kind is referred to. So \((S)_{\rho=5}\) describes all NNs parametrized by \(\rho\) and partitioned into 9 times 5 parts with both of the two considered architectures. They are trained, validated and tested on subsets of \(S\).

3. Results

The convolutional NN performance for all tested settings is shown in figure 8. These two graphics depict the \text{rmse} dependence on the partitionings defined in section 2.4.1. The NN performances measured in terms of the \text{rmse} are divided into 10 groups representing the partitioning. Within each group the parameter choices are shown. To avoid false conclusions by statistical outliers, an ensemble of 27 NNs trained with the same settings has been calculated. Variations are the randomness of the weight initialization and the mini batch sampling as well as different learning rates and batch sizes. Markers and bars indicate the mean \text{rmse} (\(\overline{\text{rmse}}\)) and the associated 95% confidence interval for the mean respectively. The confidence interval has been calculated by bootstrapping [17]. The true possible values of \(I_B\) range between \(-0.05\) and 0.18. In order to facilitate the evaluation of the NN reconstruction quality, a reference value of 10% of the total \(I_B\) range is marked by the dotted, green line. The figures 9–11 depict subsets of the
outcomes shown in figure 8 to clarify the observations, while figure 13 compares the different NN architectures as an additional hyperparameter.

3.1. Simulation trained NNs

The results of the NNs trained on $S$ are shown in figure 8(a). It can be seen that $(S)^{\rho}$ performs well with both NN architectures for partitionings up to $36 \times 12$. The $\text{rmse}$ is minimal at the partitioning $9 \times 5$ and increases with finer as well as coarser resolution. The NNs $(S)^{\mu, \sigma}$ and the $(S)^{\mu, \delta}$ perform especially well for the coarse partitionings between $2 \times 1$ and $9 \times 5$. For finer partitionings the performance decreases gradually. This can be understood contemplating the decreasing information content per section with shrinking section size.

![Figure 8](image.png)

Figure 8. Average $\text{rmse}$ with 95% confidence level for various parametrizations and limiter partitionings. In case of smaller $\text{rmse}$ than 10% of the $I_B$ range a sufficient reconstruction of $I_B$ has been achieved. (a) Logarithmic representation of $\text{rmse}(S)$ and $\text{rmse}(S_{90, S_{10}, I})$. (b) Logarithmic representation of $\text{rmse}(M_{90, M_{10}, I})$ and $\text{rmse}(I)$.}

On the basis of figure 8(a) we investigate the performance of NNs characterized by $(S)$ versus $(S_{90}, S_{10}, I)$. A good performance of the NN $(S_{90}, S_{10}, I)$ would be advantageous since it would indicate applicability to new, never conducted experiments. Although it would be preferable, the evaluation indicates that this is not the case.

We observe not only that $\text{rmse}(S) \ll \text{rmse}(S_{90}, S_{10}, I)$, but also that $\text{rmse}(S, S, I)$ significantly exceeds the 10% $I_B$ range. The NNs $(S_{90}, S_{10}, I)$ are specialized onto patterns of $S$. Those patterns are not suitable to determine $I_B$ from experimental data. The fundamentally different magnitude of the width of the $\text{rmse}$ confidence intervals $CI(S, S, I)$ as compared to $CI(S)$ points towards the same reason.

Both parametrizations including $\mu$ show a decreasing performance with growing resolution. Only a marginal difference between $\text{rmse}(S)^{\mu, \delta}$ and $\text{rmse}(S)^{\mu, \sigma}$ can be observed. The $\text{rmse}(S)$ range seems independent of the architecture.

![Figure 9](image.png)

Figure 9. The development of $\text{rmse}(S)^{\text{Conv}, \rho}$ and $\text{rmse}(M_{90, M_{10}, I})^{\text{Conv}, \rho}$ are compared for the increasingly refined partitionings. Especially for coarse and extremely fine partitionings the behavior is similar.

![Figure 10](image.png)

Figure 10. The development of $\text{rmse}(M_{90}, M_{10}, I)^{\text{Conv}, \rho}$, $\text{rmse}(M_{90}, M_{10}, I)^{\text{Conv}, \mu, \sigma}$ and $\text{rmse}(M_{90}, M_{10}, I)^{\text{Conv}, \mu, \delta}$ are compared for the increasingly refined partitionings. Over all partitionings $\mu$ & $\sigma$ and $\mu$ & $\delta$ behave similar while $\rho$ performs clearly better for fine partitionings.
The fitting 1.1 0.9 and 0.6 4.2, as seen in figure 12. The performance on the validation set (red circles) and test set $I_b$ (blue triangles) is shown. On the $x$-axis are the target values and on the $y$-axis are the NN reconstructions. An ideal result would be the identity, shown by the dashed line.

3.2. NN trained with simulation and experiment

Since $S$ based training did not lead to sufficient NN performance for application to $I$, some samples from $I$ are provided during training and validation, i.e. $M$ and $I_b$ as defined in section 2.2 are used. With this procedure, we intend to force the NNs to consider patterns present in both $S$ and $I$ during training. The performance of the NNs $(M_{90}, M_{10}, I_b)$ is compared to NNs trained, validated and tested with the small amount of available experimental data only, i.e. $(I_{37}, I_{4}, I_b)$. Note, that the performance is tested with the same set $I_b$. Figure 8(b) depicts this comparison. The upper end of the occurring $\text{rmse}$ range is reduced by two orders of magnitude in this figure as compared to 8(a). As in the case of NNs trained with $S$, the parametrizations by $\mu$ yield the best $\text{rmse}$ for partitionings between $9 \times 5$ and $36 \times 12$ while the $\mu$ based parametrizations are best for one dimensional partitionings decreasing in performance with growing resolution as visualized in figure 10.

For two-dimensional partitionings, we observe in figure 11 $\text{rmse}(M_{90}, M_{10}, I_b)_{\text{Conv}} < \text{rmse}(\|)$, as intended by training on $M$. In the case of coarse partitioning, training with $M$ has no advantages above direct training with $\|$. The most impressive results were found for resolutions between $9 \times 5$ and $36 \times 12$, for $(M_{90}, M_{10}, I_b)_{\text{Conv}}$. In order to display the good reconstruction quality, we choose a representative $\text{rmse}$, namely median $\text{rmse}$, of the NNs $(M_{90}, M_{10}, I_b)_{\text{8x8,Conv}}$, as seen in figure 12. The $\text{rmse}$ is 0.008 which is 3.5% of the $I_b$ range, so the median $\text{rmse}$ is clearly below the 10% $I_b$ range threshold. The results clearly outperform those of the NN in [1], where the best $\text{rmse}$ was 0.029.

Table 2. Mean calculation times in ms for one training cycle until the early-stopping criteria is reached, and calculation time for a single evaluation of the validation set for the best performing partitioning with $18 \times 8$ elements and the largest one with $144 \times 30$ elements.

| Partitioning | Conv | Inception | FFNN |
|--------------|------|-----------|------|
| $18 \times 8$ Training | $0.9 \cdot 10^5$ | $4.2 \cdot 10^3$ | $1.0 \cdot 10^5$ |
| $144 \times 30$ Training | $1.1 \cdot 10^3$ | $1.9 \cdot 10^5$ | $0.6 \cdot 10^5$ |
| $18 \times 8$ Validation | 8 | 10 | 5 |
| $144 \times 30$ Validation | 17 | 19 | 9 |
criteria. The complete validation set is evaluated within milliseconds, which guarantees real-time applicability.

Changing the NN architecture to an inception model seems to slightly but not significantly improve the performance for finer partitioning as seen in figure 13. This behavior is not completely unexpected, as the inception model does not take advantage of the ‘parallelization’ unless very large problem sizes occur. Such larger sizes will be relevant in the future when dealing with the more complex divertor of W7-X instead of the limiter. The FFNN is clearly outperformed by the convolutional NN as well as by the inception NN.

4. Conclusion and future work

It was shown here that it is possible to reconstruct an important property of the W7-X edge magnetic field structure from limiter heat load patterns with even better accuracy than earlier reported in [1]. The main challenge was to deal with sparse experimental data given. A naive approach to apply NNs trained and validated with synthetic data to experimental data showed good performance only in a minority of cases. After such a training process most NNs focus on patterns not present in the experimental observations.

For a more targeted training and validation, a mixture of experimental and synthetic data is formed for the training process. This approach resulted in convincing NN performance for certain NN input processing. Partitioning the limiter with resolutions between 9 × 5 and 36 × 12 and defining the NN input as the heat load in each part divided by the maximum heat load of all parts results in better performance compared to NNs trained, validated and tested with experimental data only. The low number of experimental results probably leads to overfitting in these nets but the added simulation data diminished these effects. We created NNs that extract relevant patterns from experimental as well as from synthetic data sets to reconstruct an important parameter of the magnetic field at the edge. With this systematic approach NNs were found to outperform the results found in [1].

The upgraded W7-X with installed divertors will be the next object of interest. We will start the investigation with a parametrization based on a two-dimensional partitioning of the heat load. Favoring one of the two examined NN architectures a priori and excluding the other is not possible at this stage because neither consistently outperforms the other. The reached results are satisfactory, however it remains future work to investigate other methods such as generative adversarial nets [18] to further enhance the reconstruction performance when dealing with simulated and experimental data. These could learn to generate ‘experimental data’ and provide an additional supplement for real experimental data to train NNs which reconstruct plasma properties.

Acknowledgments

This work has been carried out within the framework of the EURATOM research and training programme 2014–2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

We wish to acknowledge the helpful discussions with G. Leifert, T. Grüning, T. Strauss, M. Weidemann and J. Michael. Our gratitude goes to H. Niemann and F. Pisanò for providing the IR data geometry and heat flux mapping. We are also thankful for the support from the CTTlab (University of Rostock) and Planet-AI teams.

ORCID iDs

Thomas Sunn Pedersen https://orcid.org/0000-0002-9720-1276

References

[1] Böckenhoff D. et al 2018 Reconstruction of magnetic configurations in W7-X using artificial neural networks Nucl. Fusion 58 086009
[2] Hinton G.E. et al 2012 Improving neural networks by preventing co-adaptation of feature detectors 1–18 (arXiv:1207.0580v1)
[3] He K. et al 2016 Deep residual learning for image recognition IEEE Conf. on Computer Vision and Pattern Recognition (Las Vegas USA, 27–30 June 2016) pp 770–8
[4] Cireşan D. et al 2012 Multi-column deep neural network for traffic sign classification Neural Netw. 32 333–8
[5] Grieger G. and Milch I. 1993 Das fusionsexperiment Wendelstein 7–X Phys. J. 49 1001–5
[6] Feng Y. et al 2008 EMC3/EIRENE transport modelling of the island divertor in W7-X 35th European Physical Society Conf. on Plasma Physics, 10th Int. Workshop on Fast Ignition of Fusion Targets (Hersonissos, Crete, 9–13 June 2008) (http://epsppd.epfl.ch/Hersonissos/pdf/P2_085.pdf)
[7] Effenberg F. et al 2017 Numerical investigation of plasma edge transport and limiter heat fluxes in Wendelstein 7-X startup plasmas with EMC3-EIRENE Nucl. Fusion 57 36021
[8] Andreeva T. 2002 Vacuum magnetic configurations of Wendelstein 7-X Technical Report Max-Planck-Institut für Plasmaphysik, Greifswald
[9] LeCun Y. and Bengio Y. 1995 Convolutional networks for images, speech, and time series The Handbook of Brain Theory and Neural Networks p 3361 (http://yann.lecun.com/exdb/publis/pdf/lecun-bengio-95a.pdf)
[10] Szegedy C. et al 2015 Going deeper with convolutions Proc. IEEE conference on computer vision and pattern recognition (Boston, USA, 7–12 June 2015) pp 1–9
[11] Deng J. et al 2009 ImageNet: a large-scale hierarchical image database IEEE Conf. on Computer Vision and Pattern Recognition (Miami USA, 20–25 June 2009) pp 248–55
[12] Krizhevsky A., Sutskever I. and Hinton G.E. 2012 ImageNet classification with deep convolutional neural networks Adv. Neural Inf. Process. Syst. 25 1097–105
[13] Glorot X. and Bengio Y. 2010 Understanding the difficulty of training deep feedforward neural networks Proc. 13th Int. Conf. on Artificial Intelligence and Statistics (Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010) pp 249–56 (http://proceedings.mlr.press/v9/glorot10a.html)
[14] Kingma D.P. and Ba J.L. 2015 Adam: a method for stochastic optimization Int. Conf. on Learning Representations 2015
[15] Abadi M. et al. 2016 TensorFlow: a system for large-scale machine learning. OSDI pp 265–83 (https://ai.google/research/pubs/pub45381)

[16] Renner H. et al. 2004 Physical aspects and design of the Wendelstein 7-X divertor. Fusion Sci. Technol. 46 318–26

[17] Diciccio T.J. and Efron B. 1996 Bootstrap confidence intervals. Stat. Sci. 11 189–228

[18] Goodfellow I.J. et al. 2014 Generative adversarial nets. Advances in neural information processing systems (Montreal, Canada, 8–13 December 2014) pp 2672–80 (http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf)