Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Pranas Juknevicius

Accelerator Net

Semester project

Swiss Federal Institute of Technology (ETH) Zurich

May 22, 2023

Abstract

Particle accelerators are complex machines that are able to accelerate charged particles, therefore creating a particle beam. The key task for accelerator operators is to configure the accelerator design parameters to get the desired characteristics of the beam. One of the ways to improve the configuration procedure is to gain information about the beam at different positions throughout the beam-line. In real-life application this approach is slow and technically difficult, therefore it is unpractical. In this project we investigate, how can we use supervised machine learning to gain information about the beam throughout the beam-line. Firstly, we used a supervised deep neural network to predict the beam characteristics at the end of the beamline. Secondly, using deep neural networks we were able to reconstruct the beam profile and the particle phase space at arbitrary positions in the beam-line.

Contents

1 Introduction

Particle accelerators are a machines, that have been widely used to in explore fundamental physics (ATLAS experiment at CERN [1]), medicine (proton therapy, for an overview see [2]) and other fields. They are able to provide a beam of accelerated charged particles, which are usually directed at the object of interest (human organs, fundamental particles, etc.). By using the deflected particle trajectories and momenta, it is possible to gain information about the object of interest. One of the key requirements for these experiments is to have a precisely defined beam.
The characteristics of the beam are controlled by the physical components of the accelerator, the geometry and the tunable design variables, such as the electrical current in the magnets, the frequency of the pulse, the number of particles in a batch and others. Testing different configurations of the design variables to get the desired beam characterstics is a slow and expensive process in real-life application. Therefore there is a need for a simulation tool or a surrogate machine learning model in order to make this process more effective.
Simulation tool is a physics informed mathematical model that is able to simulate the quantities of interest of the particle beam using the design parameters of the accelerator. These tools allow to accurately evaluate the physical attributes of the beam, but they are usually slow and have a limited range of objectives, which can be calculated. A machine learning surrogate model can be used to simulate wider range of objectives and to achieve speedup in comparison to the simulation tool. The surrogate model uses data generated by the simulation tool (or could use experimental data) for training. Therefore, a surrogate model can be built, when one has an experimentally tested, accurate simulation tool.
We used an open-source Object Oriented Parallel Accelerator Library OPAL [3] as a simulation tool to model a linear accelerator. The accelerator design is based on the Argonne Wakefield Accelerator AWA [4] located at the Argonne National Laboratory. OPAL was used to generate the dataset for the machine algorithm. More importantly, it is shown that OPAL was able to accurately simulate the experiments that have been done in the AWA, therefore we trust that simulated data corresponds well to the real-life experimental data.
For the surrogate models we used supervised and deep neural networks. The rapid development in the machine learning field provides a wide range of applications to particle accelerators. In this project we chose two objectives. First is for ML algorithm to predict the characteristics of the beam at a fixed location in the beam-line. A similar research was already done by Bellotti et al. [5], where they used invertible

deep neural networks to predict the characteristics of the beam at arbitrary positions in the accelerator. Our objective is to use our dataset and a different neural network architecture to reproduce their results, therefore confirming that our dataset and techniques are valid. The second goal is to reconstruct the beam profile and the phase space at arbitrary positions in the beam-line. This would help the accelerator designing procedure, by providing insight about the beam at arbitrary positions in the beam-line.

2 Methods

2.1 Argonne Wakefield Accelerator

The techniques described in this project can be used for various particle accelerators, but we narrowed our research to a linear wake-field electron accelerator. An example of such accelerator, on which we focus our models, is the Argonne Wakefield Accelerator located in the Argonne National Laboratory. The main components of the AWA are electron gun, radio-frequency cavities for accelerating electrons and magnets to keep the beam focused.

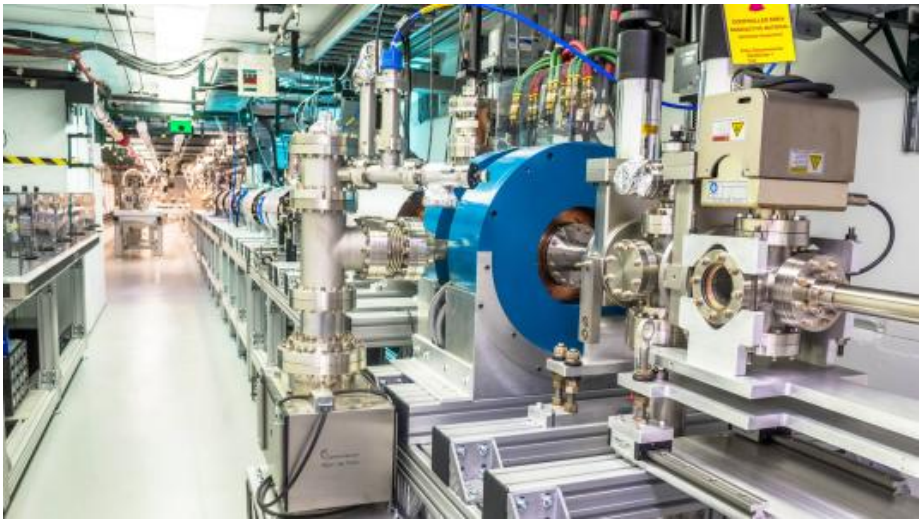

Figure 1: The Argonne Wakefield Accelerator (see [4]).

In order to control characteristics of the electron beam the accelerator engineers tune the accelerator design variables. We chose a subset of all possible design variables to restrict our search space:

1. λ – peak to peak distance between Gaussian particle bunches generated by electron gun,
2. ϕ – electron gun phase,
3. Q – electron gun charge,
4. $SIGXY$ – laser spot size,
5. IBF – electrical current in the buck focusing magnets,
6. IM – electrical current in the matching solenoids,
7. $ILS1$ – electrical current in first solenoid magnet,
8. $ILS2$ – electrical current in second solenoid magnet,
9. $ILS3$ – electrical current in third solenoid magnet.

So we have 9 design variables $X \in R^9$ defining the configuration of the accelerator.

Bound	IBF[A]	IM[A]	ϕ[°]	ILS_1[A]	ILS_2[A]	ILS_3[A]	Q[nC]	λ[ps]	SIGXY[mm]
Lower	450	100	-50	0	0	0	0.3	0.3	1.5
Upper	550	260	10	250	200	200	5	2	12.5

Table 1: The design variables and their ranges. The design variable ranges are taken from [5].

2.2 OPAL simulation

The AWA accelerator has been successfully modeled with the OPAL simulation tool. OPAL (Object Oriented Parallel Accelerator Library see [3]) is a parallel open source tool for charged-particle optics in linear accelerators and rings (see wiki for more information). Using OPAL we generated the dataset used for neural networks training and testing.
As input parameters for the simulation we used random combinations of the design variables described in section 2.1. We were interested in 2 different types of outputs from the simulation:

- Quantities of interest (QoI) describe the electron beam physical properties:
  1. transversal root mean squared beam sizes σ_x, σ_y,
  2. transversal normalized emittances ϵ_x, ϵ_y,

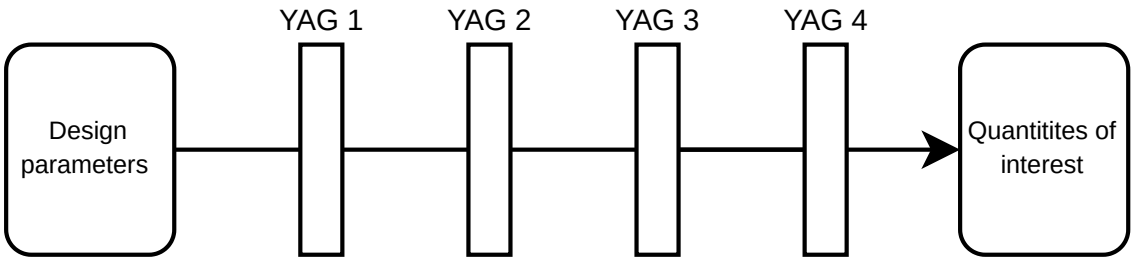

Figure 2: A schematic drawing of the AWA beamline.

3. mean bunch energy E,
4. energy spread of the beam ΔE,
5. correlations between transversal position and momentum $Corr(x, p_x)$, $Corr(x, p_x)$.

- The beam profile and the phase space of the particle bunches at different locations in the beamline captured by 4 YAG screens at different positions throughout the accelerator.

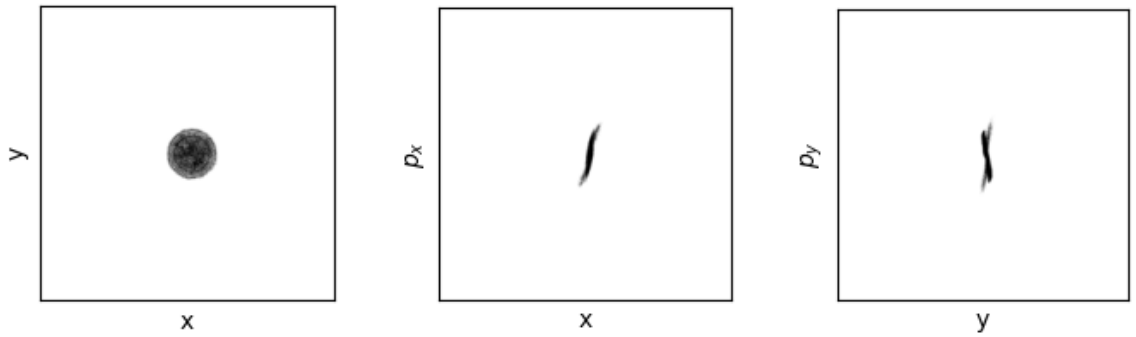

Figure 3: Example of beam profile and the phase space of the particles captured by one of the YAG screens.

Therefore using OPAL we generated the chosen outputs, which are quantities of interest $Y \in R^8$, beam profile and the $x - p_x$, $y - p_y$ phase space slices.

2.3 Data generation and processing

We generated 18.7k samples using OPAL for neural network training and testing. Each sample consisted of:

1. Design variables $X \in R^9$.
2. Quantities of interest $Y \in R^8$.

3. Phase space data and beam profile captured by 4 YAG screens at different positions throughout the beam-line.

In order to prepare the data for neural networks we normalized the design variables and quantities of interest using min-max normalization:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

In order to efficiently use the beam profile and phase space data captured by the 4 YAG screens, we transformed the data into images. This way we can take advantage of the image processing machine learning techniques. We followed the procedure:

1. We began with the data generated by OPAL, which has the size:
   (number of YAG screens, dim(*phase space*), *number of particles*) = $(4, 6, 20000)$,

2. Removed the longitudinal z and p_z components, because in real-life application only the transversal components of the beam can be measured. Therefore the data size changes: $(4, 6, 20000) \rightarrow (4, 4, 20000)$

3. Chose to use $x - y$, $x - p_x$ and $y - p_y$ phase space for training neural networks,

4. Transformed beam profile and phase space into images. Coordinates were determined by the variables (example x and y or x and p_x) and the intensity by the number of particles in a bin $(4, 4, 20000) \rightarrow (4, 3, 256, 256)$.

2.4 Quantities of interest prediction techniques

The first goal of this project was to use the design variables of the accelerator and the data captures by the YAG screens (see section 2.1) to predict the quantities of interest (see section 2.2) of the beam using supervised learning. This was already attempted by using densely connected invertible networks [5], so our goal here is to improve or reproduce the previous results. This would allow to verify the validity of the our dataset, methods, techniques, before moving towards more complicated neural network applications. Important to note that authors were able to predict the quantities of interest at any position throughout the beamline, while our approach is able to predict the quantities of interest only at a fixed positions in the beamline. We used two neural network architectures. The first one is a straightforward multi-layer percepton neural network with 4 hidden layers (see figure 4).

The second architecture combines the data captured by the YAG screens and design variables to predict the quantitites of interest. Using convolutional neural

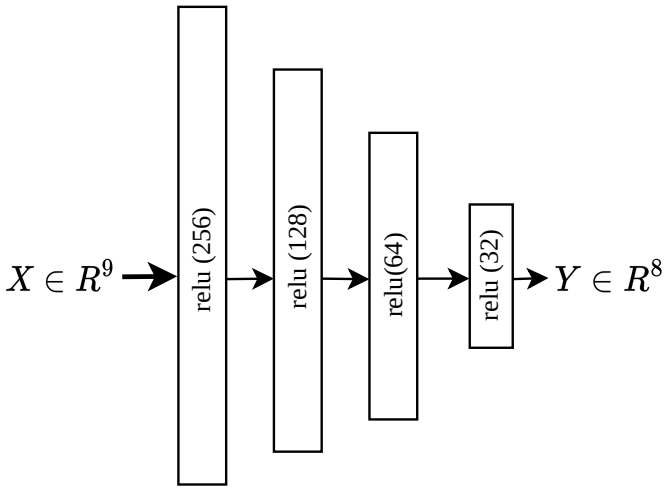

Figure 4: Fully connected multi layer percepton (MLP) neural network.

networks (CNN) [6] and a hidden layer we are able to compress the dimension of the beam profile or phase space images to a vector in the latent space, which has the same dimension size as the design variables $X \in R^9$. Then we concatenate the latent vector and the design variables into a single vector, which has the dimension R^{18}. It is important to renormalize the latent vector before the combining. Finally, we use a similar MLP (see figure 4) with the combined vector to get the quantities of interest.

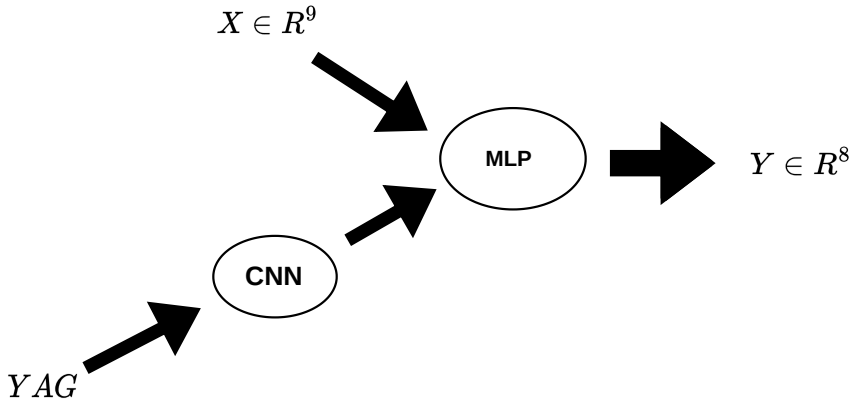

Figure 5: Deep neural network composed of CNN and MLP for quantities of interest prediction.

Finally, we can choose, which images we use to train the described neural network. We chose the data captured by the last YAG screen in the beamline. Particularly, we trained and tested the network using the beam profile (denoted MLP+xyCNN) and the $x - p_x$ phase space (denoted MLP+xpxCNN).

Both of the networks were trained for 60 epochs, used mean squared error (MSE) loss function with Adam[7] optimizer. See section 3.1 for results.

2.5 Beam profile and phase space reconstruction techniques

In order to reconstruct the beam profile or the phase space at arbitrary positions in the beamline, we use a neural network architecture based on the autoencoder architecture.
On a high level, the network encodes the beam profile captured by the last YAG screen in the beam-line into the latent space. Then it concatenates the encoded vector with the position, where we want to reconstruct the beam profile or phase space. Optionally we can also concatenate with the design variables. The concatenated vector is then decoded into a reconstructed image, which we compare to the an image simulated by OPAL.

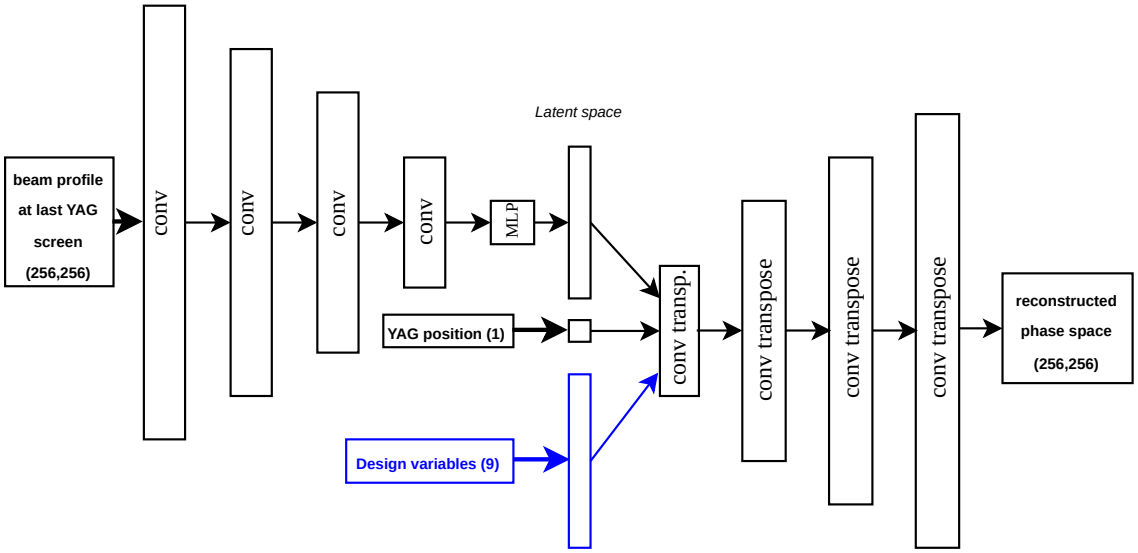

Figure 6: Architecture of the autoencoder based neural network for beam profile or phase space reconstruction.

In more detail, the training epoch follows the procedure:

1. **Input** the beam profile from the last YAG screen in the beam-line (11.48m in our case).

2. Encode using an encoder network into a latent vector of R^9.

3. Normalize latent vector between 0 and 1

4. Concatenate the latent vector with normalized position of the YAG screen, at which we want to reconstruct the beam profile or phase space image. Because we have simulated images at 4 distinct positions (see figure 2), we want to reconstruct the images at the positions of simulated YAG screens, so that we can evaluate the MSE loss. Optionally, we can also concatenate with design variables.

5. Reconstruct the image at 11.48m and calculate the loss between the simulated and reconstructed image. Take an optimization step.

6. Repeat with reconstruction at 8.76m.

7. Repeat with reconstruction at 6.38m.

8. Repeat with reconstruction at 3.1m.

9. Randomly pick a new data sample and go to 1.

2.6 Implementation and set-up

The dataset was generated using OPAL. The neural networks were implemented in *Python* using *Pytorch*[8] and *TensorFlow*[9] (Tensorflow was only used for the simple MLP model in quantities of interest prediction). The networks were trained for 60 epochs and training time per network was 6 hours or less.
The neural networks were trained on a GPUs with HPC cluster *Merlin* located at PSI. The data generation was also done with Merlin, but using CPUs.

3 Results

3.1 Quantities of interest prediction results

In order to predict the quantities of interest we used neural networks with 2 distinct architectures (see figures 4 and 5). We tried using the beam profile (MLP + xyCNN) and phase space (MLP + xpxCNN) as input for CNN and MLP combined neural network. Finally we compared the resulting $\bar{R}^2$ values with the invertible neural network[5] results.

	E	ΔE	σ_x	σ_y	ϵ_x	ϵ_y	$Corr(x, p_x)$	$Corr(y, p_y)$
MLP	0.99	0.94	0.97	0.97	0.96	0.96	0.98	0.98
MLP+xyCNN	0.997	0.965	0.997	0.997	0.993	0.993	0.990	0.989
MLP+xpxCNN	0.998	0.992	0.998	0.998	0.976	0.975	0.986	0.991
Invertible	0.97	0.97	0.99	-	0.19	-	0.93	-

Table 2: $\bar{R}^2$ values for all QoI predicted by neural networks. The - values were not presented in [5], but it was stated that they are similar to corresponding x quantities.

Note that invertible neural networks are able to predict the quantities of interest at arbitrary positions in the beamline, while the remaining models can predict only at a fixed position (we chose this position to be the end of the beamline). Therefore the absolute values should not be compared to deduce, which model is better.
Observing the high $\bar{R}^2$ values, we can state the all proposed networks can accurately predict the quantities of interest. This was expected due to the previous results of the invertible model.
Quantities of interest prediction was not the main focus of this research, but it allowed us to check the validity of the dataset and the used techniques. Most importantly, we were able to notice that adding the YAG screen data improves the performance of the neural networks. This is an important prerequisite before trying to use more complicated networks with this dataset.

3.2 Beam profile and phase space reconstruction results

Having researched a wide range of neural network architectures, we picked an autoencoder based neural network. While tuning the hyperparameters of the network we found that the most important network design parameter is the size of the latent vector to which we compress the image from the YAG screen. We found the that optimal dimension is 16.

During testing of presented autoencoder based neural networks the **input** is always the beam profile captured at the last YAG screen in the beamline (11.48m) and the position at which we want to reconstruct the image. Additionaly, we can choose whether we want to use the accelerator design variables. The **output** of the network is the desired reconstruction at the chosen position.
As a first example of the neural network performance we reconstructed the beam profile at YAG screen positions:

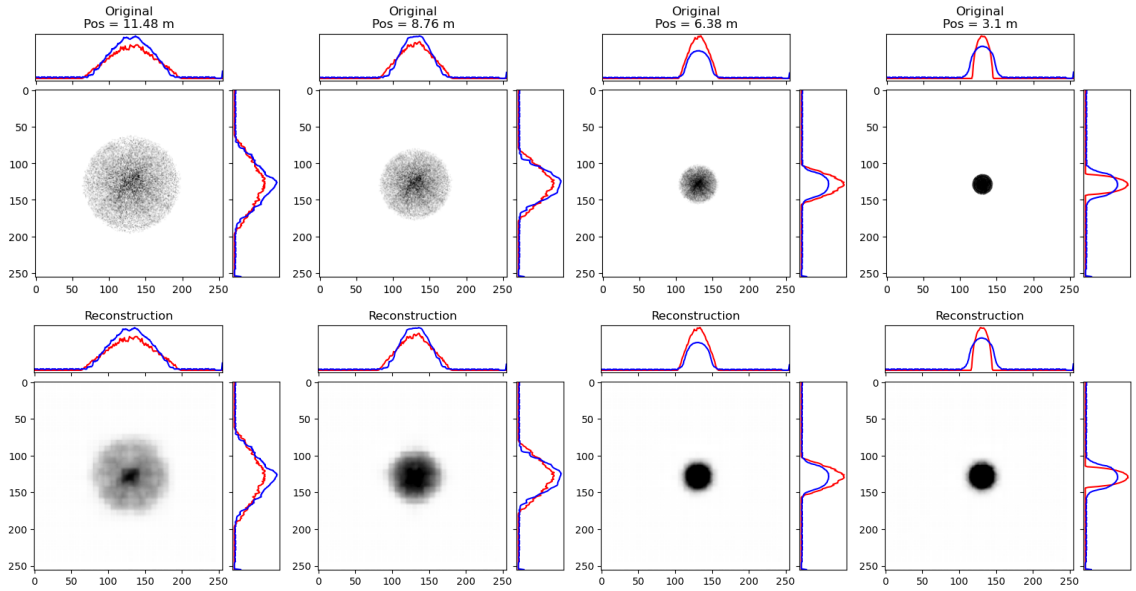

Figure 7: Beam profile reconstructions at YAG screens positions. The side plots are beam intensities. Blue line is the intensity of the simulated beam profile, and red line is the intensity of the reconstructed beam profile.

First we notice that the reconstructed beam profile is similar the original beam profile, although the internal structure of the beam profile is not reconstructed well. The reconstructed width of the beam profile matches original, except for the reconstruction at 3.1m. We also see that the intensity of at the 6.38m and 3.1m does not match the original intensity.
Next we test the same beam profile reconstruction, but this time we add the accelerator design variables. Naturally, we expect that the reconstruction accuracy would increase (see figure 8).

We noticed that adding the design variables improves the beam profile reconstruction. The beam width of the reconstructed images match the original width of the beam to a high precision. The reconstructed internal structure of the beam also matches the original better. Only the intensity of the reconstructions differs from the original.

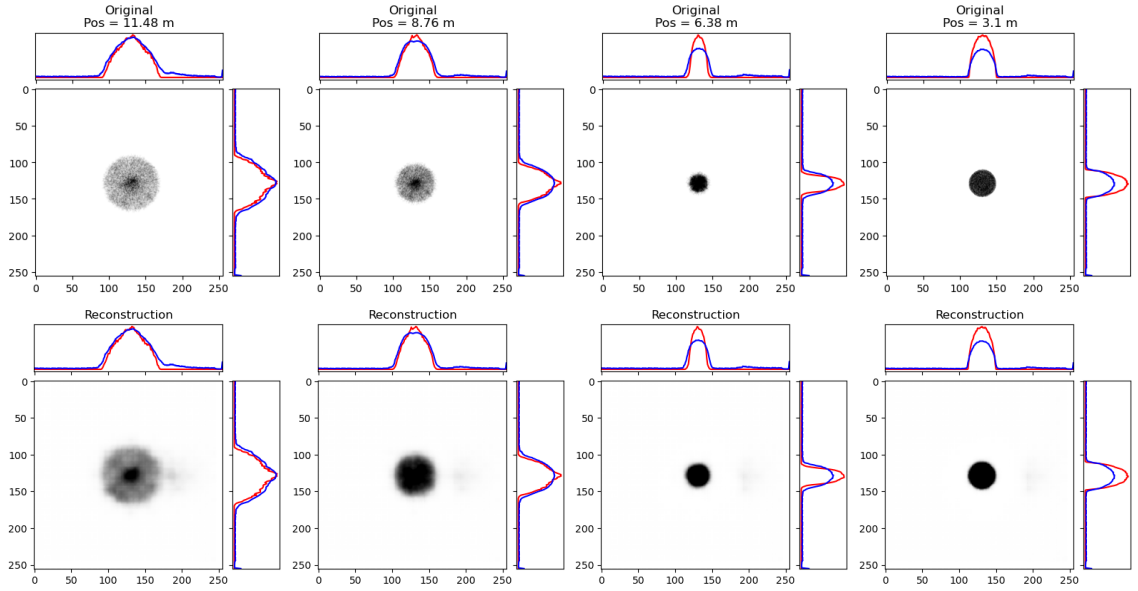

Figure 8: Beam profile reconstructions at YAG screen positions. The side plots are beam intensities. Blue line is the intensity of the simulated beam profile and red line is the intensity of the reconstructed beam profile.

Having had success with reconstructing the beam profile, we try to reconstruct the phase space at YAG screen positions using the beam profile captured at the last YAG screen and the design variables. We begin with the reconstruction of x - p_x phase space (see figure 9).

Again we see that the shape of the reconstructed x - p_x phase space closely matches the original, although we can see more fluctuations. The intensity from the reconstructed images does not match the intensity from the originals.
We repeat the process with y - p_y phase space (see figure 10). As before, the reconstruction phase space closely matches the original, but the intensity does not.

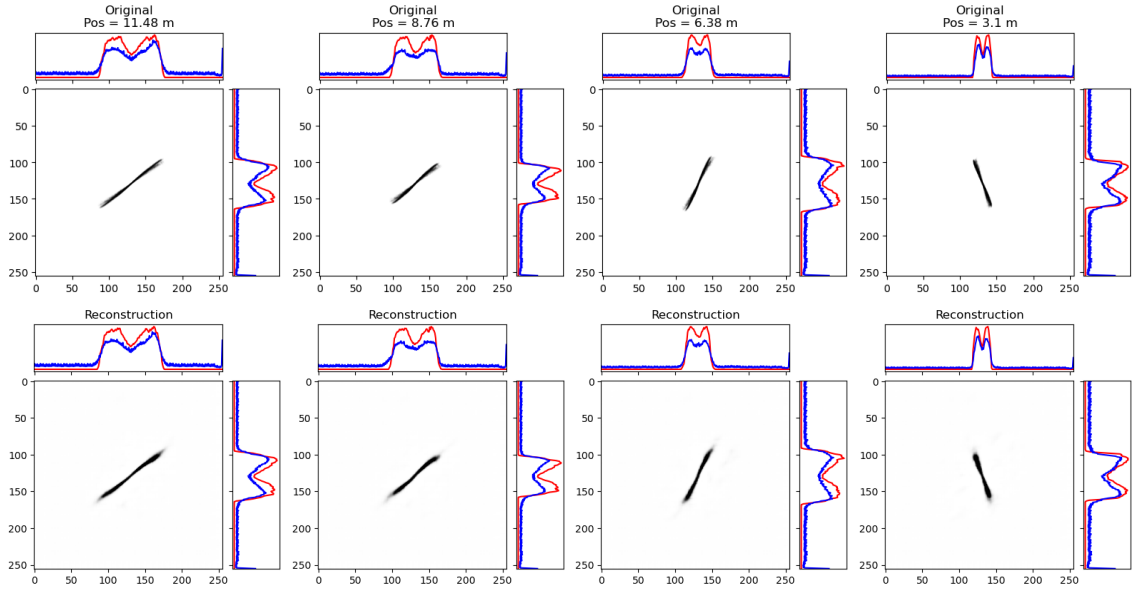

Figure 9: x - p_x phase space reconstructions at YAG screen positions. The side plots are intensities. Blue line is the intensity of the simulated phase space and red line is the intensity of the reconstructed phase space.

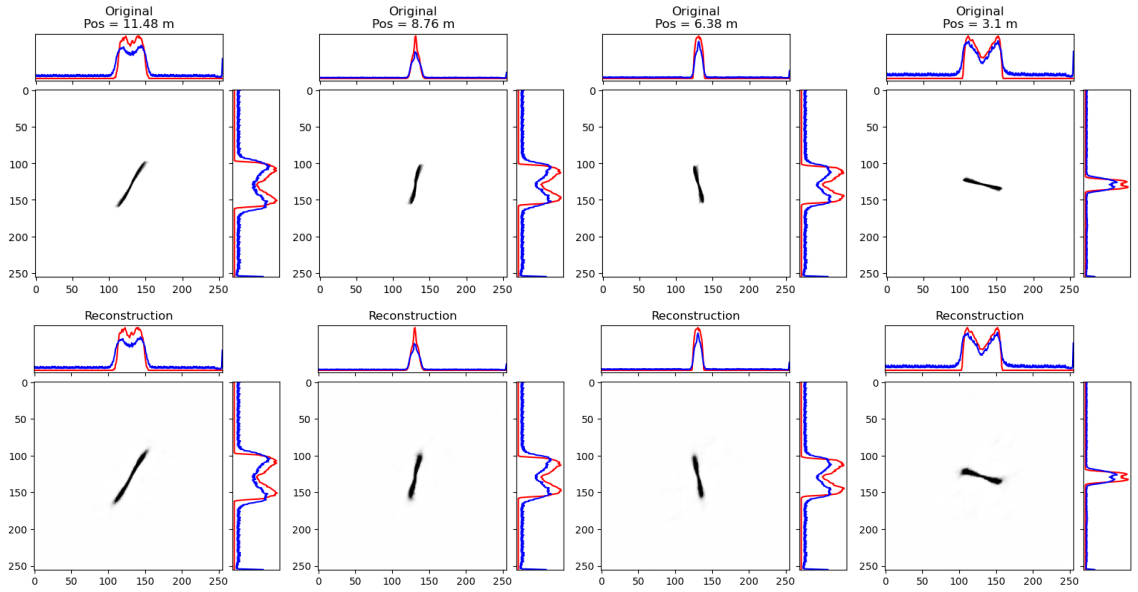

Figure 10: y - p_y phase space reconstructions at YAG screen positions. The side plots are intensities. Blue line is the intensity of the simulated phase space and red line is the intensity of the reconstructed phase space.

The final test is to see, whether the proposed autoencoder based neural network can reconstruct the beam profile at a positions, which was never used during training. For this we trained the network to reconstruct the beam profiles at 11.48m, 8.76m and 3.1m, therefore skipping the 6.38m. Again we present the reconstructions at

the positions of the YAG screen (see figure 11.

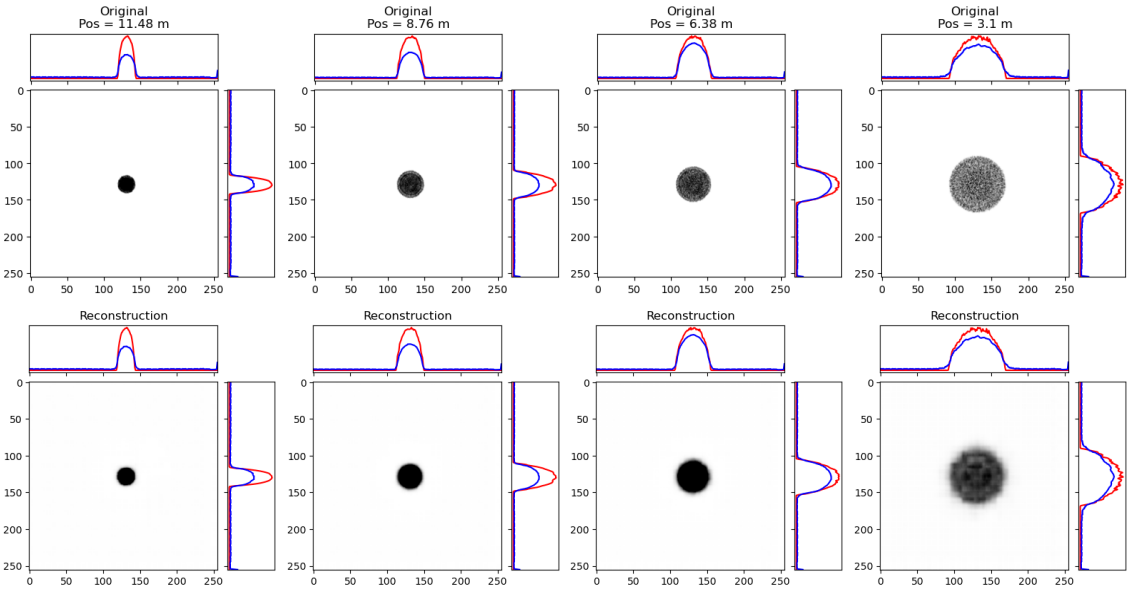

Figure 11: Testing the ability of autoencoder based neural network to reconstruct beam profile at different positions in the beamline. The side plots are intensities. Blue line is the intensity of the simulated phase space and red line is the intensity of the reconstructed phase space.

Comparing the reconstructions and the originals, we can state that the network is able to generalize for arbitrary positions in the beamline. The reconstructions have the same intensity mismatch issue, but profile shape is reconstructed extremely well.

4 Discussion

The considered neural networks for quantities of interest prediction performed well. Although important to note that during training the networks might get stuck in local loss minimum, therefore to reproduce the results it is needed to train the network more times. But because training time for these networks using the machines described in section 2.6 are around an hour, this is not a problem.
The autoencoder based neural network approach also performed well on all tested tasks, but there are drawbacks. The main issue is the intensity of the reconstructed beam profile and phase space. Note that in this project we did no post-processing of the neural network output, in other words, the reconstructions presented in section 3.2 are raw outputs of the network. Possibly if we would to renormalize the reconstructions the intensity issue would be fixed, but this requires further testing. Another remark is that the internal structure of the reconstructions is blurred compared to originals. This might be fixed with a more detailed hyperparameter tuning, but this issue also requires further investigation.
Finally, as always with image generation, reconstruction examples of the autoencoder based neural network presented in section 3.2 are subjectively picked. There are better examples and there are worse examples of the network performance, I picked the ones that captured the important aspects of network performance, but it still remains subjective (for more examples see chapter 6). This is a common issue with image generation neural networks, where no agreed upon consensus exists, on which metric to use for model performance evaluation.

5 Conclusion and Outlook

The conclusions of this project:

1. Deep neural networks can accurately predict quantities of interest of the particle beam using the design variables of the accelerator at a fixed location.

2. The autoencoder based deep neural network can accurately reconstruct the beam shape at different positions in the beamline.

3. The autoencoder based deep neural network can accurately reconstruct the phase space at arbitrary positions in the beamline.

The continuation of this project for autoencoder based neural networks should involve a thorough hyperparameter tuning, testing the reconstruction ability at locations of the beamline, which were not seen during training, fixing the intensity mismatch issue and trying to evaluate the quantities of interest using the reconstructions.
If these tasks are completed successfully, the network should be prepared to applied to real particle accelerators.

6 Annexes

Here we presented additional examples of autoencoder based neural network performance. As a reminder, for every network presented here we always used the same **input**, which is the beam profile (at 11.48m) + reconstruction position **with** design variables.

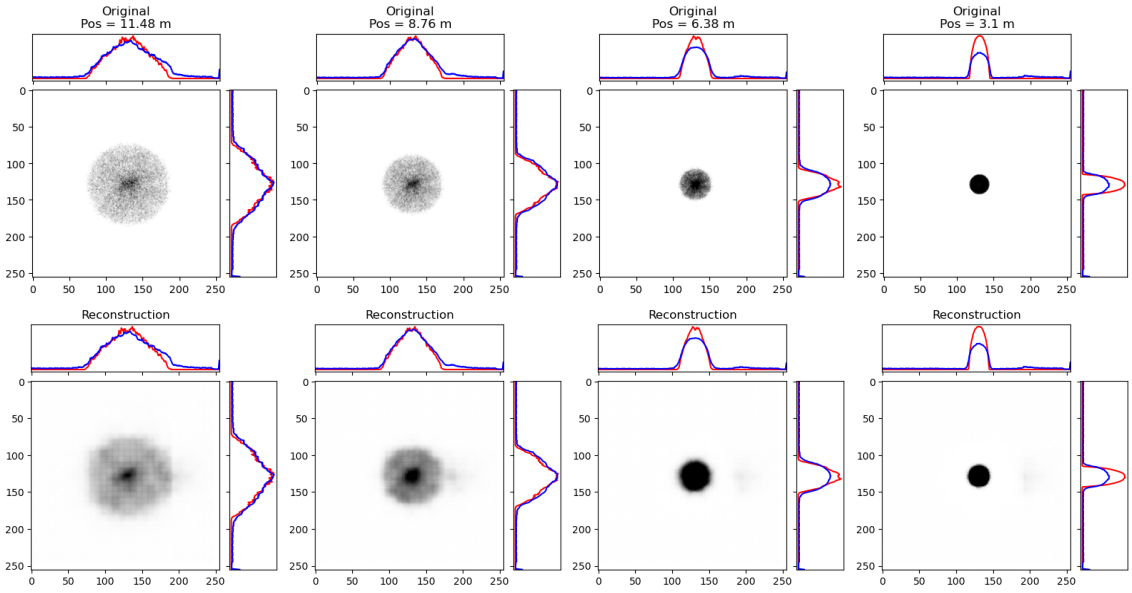

Figure 12: Another **good** example of beam profile reconstructions at YAG screen positions. The side plots are beam intensities. Blue line is the intensity of the simulated beam profile and red line is the intensity of the reconstructed beam profile.

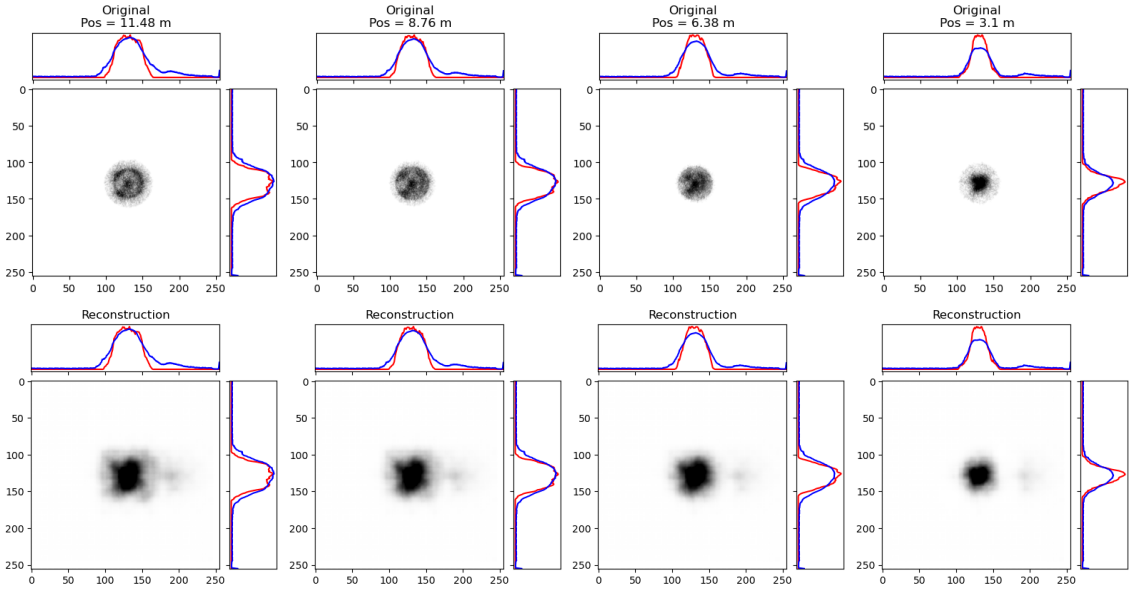

Figure 13: Bad example of beam profile reconstructions at YAG screen positions. The side plots are beam intensities. Blue line is the intensity of the simulated beam profile and red line is the intensity of the reconstructed beam profile.

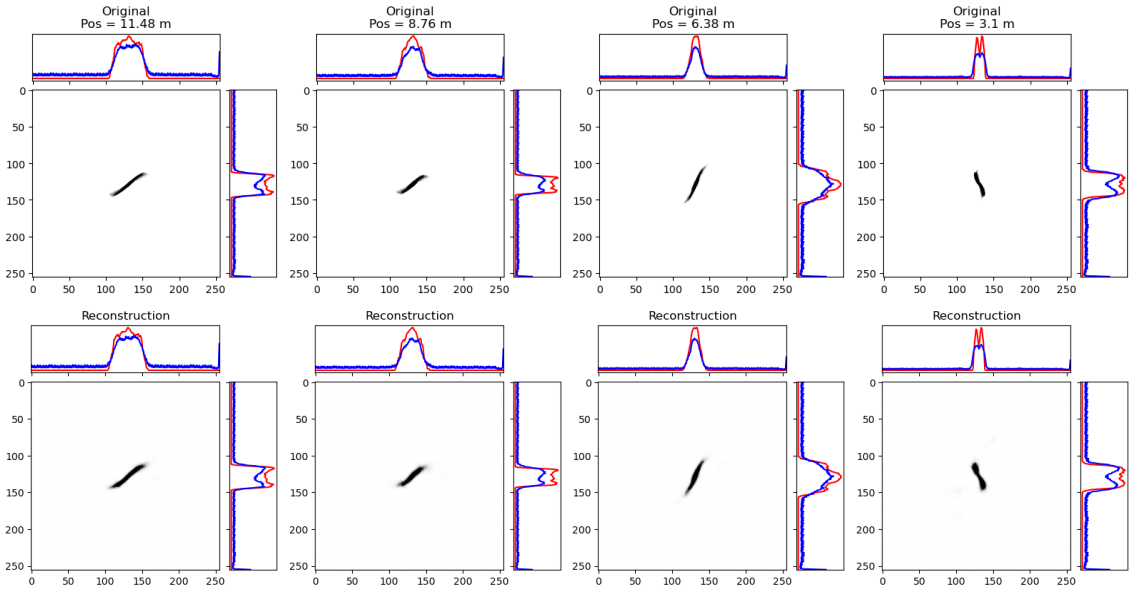

Figure 14: Another example x - p_x phase space reconstructions at YAG screen positions. The side plots are intensities. Blue line is the intensity of the simulated beam profile and red line is the intensity of the reconstructed beam profile.

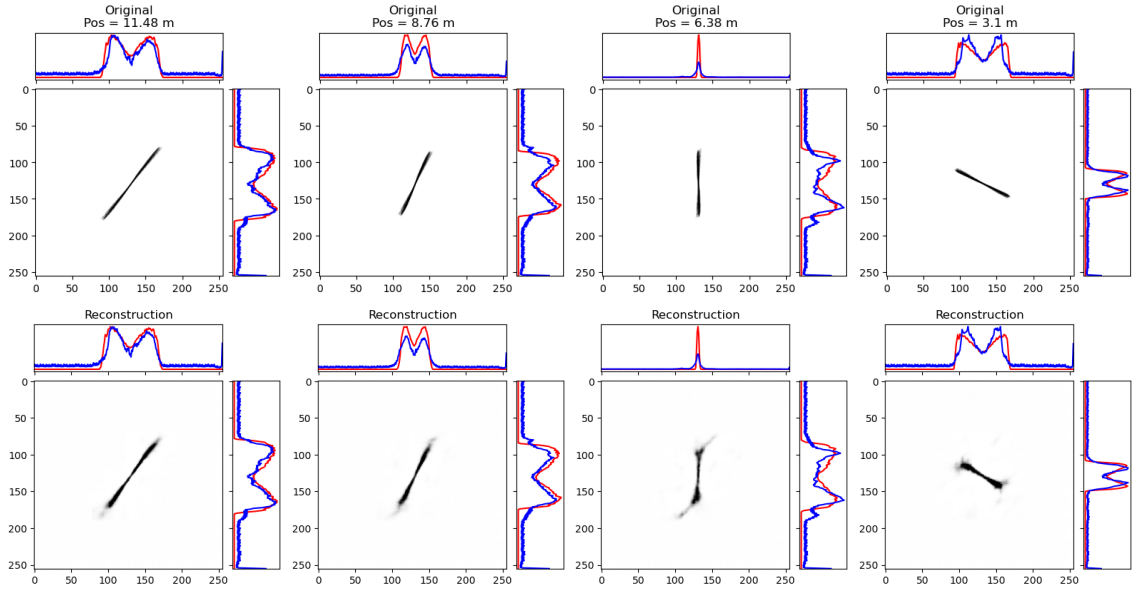

Figure 15: Another example y - p_y phase space reconstructions at YAG screen positions. The side plots are intensities. Blue line is the intensity of the simulated beam profile and red line is the intensity of the reconstructed beam profile.

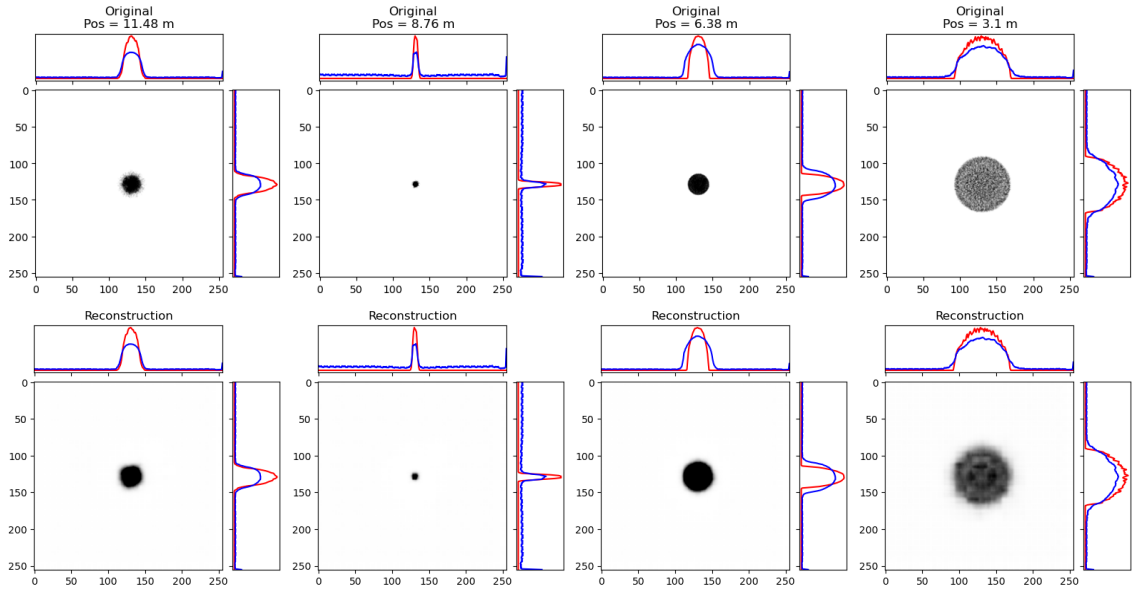

Figure 16: Another example of testing the ability of autoencoder based neural network to reconstruct beam profile at different positions in the beamline. The YAG screen beam profile at 6.38m was never seen during training. The side plots are intensities. Blue line is the intensity of the simulated beam profile and red line is the intensity of the reconstructed beam profile.

References

[1] G. Aad et al. The ATLAS Experiment at the CERN Large Hadron Collider. *JINST*, 3:S08003, 2008.

[2] Radhe Mohan. A review of proton therapy â current status and future directions. *Precision Radiation Oncology*, 6(2):164–176, 2022.

[3] Andreas Adelmann, Pedro Calvo, Matthias Frey, Achim Gsell, Uldis Locans, Christof Metzger-Kraus, Nicole Neveu, Chris Rogers, Steve Russell, Suzanne Sheehy, Jochem Snuverink, and Daniel Winklehner. OPAL a Versatile Tool for Charged Particle Accelerator Simulations. *arXiv:1905.06654 [physics]*, May 2019. arXiv: 1905.06654.

[4] P. Schoessow, E. Chojnacki, W. Gai, C. Ho, R. Konecny, J. Power, M. Rosing, and J. Simpson. The argonne wakefield accelerator-overview and status. In *Proceedings of International Conference on Particle Accelerators*, pages 2596–2598 vol.4, 1993.

[5] Renato Bellotti, Romana Boiger, and Andreas Adelmann. Fast, efficient and flexible particle accelerator optimisation using densely connected and invertible neural networks, 2021.

[6] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[9] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Title of work:

Accelerator Net

Thesis type and date:

Semester project, May 22, 2023

Supervision:

Student:

Name:	Pranas Juknevicius
E-mail:	pjuknevicius@student.ethz.ch
Legi-Nr.:	22-937-775
Semester:	2

Statement regarding plagiarism:

By signing this statement, I affirm that I have read and signed the Declaration of Originality, independently produced this paper, and adhered to the general practice of source citation in this subject-area.

Declaration of Originality:

`http://www.ethz.ch/faculty/exams/plagiarism/confirmation_en.pdf`

Zurich, 12. 7. 2023: