PHYSICAL REVIEW ACCELERATORS AND BEAMS **28,** 074601 (2025)

Reconstructing time-of-flight detector values of angular streaking using machine learning

David Meier, Jens Viefhaus, and Gregor Hartmann
Helmholtz-Zentrum für Materialien und Energie GmbH,
Albert-Einstein-Straße 15, 12489, Berlin, Germany

Wolfram Helml
Zentrum für Synchrotronstrahlung, Technische Universität Dortmund, 44227, Dortmund, Germany

Thorsten Otto
Deutsches Elektronen-Synchrotron DESY, Notkestraße 85, 22607, Hamburg, Germany

Bernhard Sick
Intelligent Embedded Systems, University of Kassel, Wilhelmshöher Allee 73, 34121, Kassel, Germany

(Received 15 January 2025; accepted 26 June 2025; published 21 July 2025)

Angular streaking experiments enable for experimentation in the attosecond regions. However, the deployed time-of-flight (TOF) detectors are susceptible to noise and failure. These shortcomings make the outputs of the TOF detectors hard to understand for humans and further processing, such as, for example, the extraction of beam properties. In this article, we present an approach to remove high noise levels and reconstruct up to three failed TOF detectors from an arrangement of 16 TOF detectors. Due to its fast evaluation time, the presented method is applicable online during a running experiment. It is trained with simulation data, and we show the results of denoising and reconstruction of our method on real-world experiment data.

DOI: 10.1103/csvm-858f

I. INTRODUCTION

During the last years, the x-ray free electron lasers (XFELs) have been developed and established as a new generation of light source for x-ray experiments [1]. The Linear Accelerator (LINAC) Coherent Light Source (LCLS) located in Stanford, California, is such a light source [2]. XFELs can produce radiation with nearly 10 orders of magnitude higher brightness compared to previous light sources. The pulse durations can range from 500 to less than subfemtoseconds (10^{-15} s) [3]. This property is beneficial because most experiments performed at XFELs require a short, well-defined photon pulse.

At LCLS, angular streaking experiments are carried out. The main goal of these experiments is to perform measurements in the attosecond (10^{-18} s) region. These experiments allow us to gain previously inaccessible knowledge, since some atomic processes happen on this time scale. For example, tunnel ionization is an effect where a particle goes through a potential barrier. A measuring resolution in attoseconds is required to analyze this phenomenon and has successfully been conducted [4]. In angular streaking, x-rays ionize the gas introduced into the chamber via a gas needle for examination. As a result, so-called photoelectrons are emitted from this gas. A circularly polarized laser has temporal and spatial overlap with the x-ray pulses in the target gas chamber. The laser's electromagnetic field then streaks the emitted photoelectrons according to their emission time in different angular positions due to the circular polarization of the laser. In our particular experimental setup, the emitted electrons are detected by an angularly aligned array of 16 electron TOF detectors setup in the dipole plane, i.e., the plane perpendicular to the incoming x-ray pulses. The TOF detectors are located in steps of 22.5° and are thus covering the entire circle of the dipole plane. For a detailed description on the experimental setup and how the real experiment data have been recorded, please refer to [3].

The TOF detectors used in the experiment are highly sensitive, and an unknown number of TOF detectors can break before or during the experiment. They can fail completely or produce implausible results. If they produce implausible results, they are disabled manually. While automating the detection of such events using anomaly

---

detection methods could be a valuable extension, it is not covered within the scope of this study. In addition to frequent failures, the TOF detectors suffer from extremely high noise. Additionally, to avoid saturation, these detectors are restricted in the number of electrons they collect. However, this constraint results in an insufficient sample size for accurate Poisson statistics.

Therefore, we present an approach to reduce noise and reconstruct the information of failed TOF detectors using simulation knowledge and neural networks and demonstrate this approach with real-world data from the experiment at LCLS. For this approach, we train a neural network exclusively on simulated data. This neural network receives the simulated detector images augmented with artificially added noise and deactivated TOF detectors mimicking failed TOF detectors. The neural network is trained to map these noisy, incomplete images to their corresponding noise-free versions with no failed TOF detectors. Since the training data are fully simulated, the ground truth for non-noisy and complete detector images is known, enabling evaluation and refinement of the network's outputs during the training process.

The main advances provided by the method proposed in the present study are as follows:

A. Fast denoising of TOF detector data

We propose a neural network-based approach for high-noise reduction in detector images, enhancing readability and preparing the data for further automated analysis. This approach is specifically designed for *online* processing, meaning that it adds a maximum delay of 1 s per data batch. It handles data with batch sizes of at least 1024 detector images in 217.53 ms with 100 threads on an AMD EPYC 7662, allowing it to match with 1 s data batches at the LCLS experiment, which supplies data at a rate of 120 detector images per second. This capability enables near real-time feedback for the experimentator and further online processing, similar to the retrospective evaluation conducted in [3]. Additionally, extracting characteristics online from the x-ray pulse is possible, as presented in [5] for a similar experiment at European XFEL located in Hamburg, Germany.

B. Fast reconstruction of missing TOF detectors using simulation knowledge

Furthermore, we want to reconstruct missing TOF detectors. We assume that defective TOF detectors either fail completely or have all their channels manually set to zero in the event of incorrect outputs. We will train neural networks that can reconstruct up to three failed TOF detectors. We compare networks trained explicitly for a distinct number of failed TOF detectors with those trained for one up to three failed TOF detectors. This reconstruction capability is also crucial for further processing steps, with our method allowing reconstructions online, as previously defined.

C. Improving future designs of TOF arrays

Because our neural network models are trained only using simulation data, we know which information was left out for reconstruction. Based on the reconstruction quality of our TOF reconstruction approach, we can determine at which points our reconstruction algorithm has difficulties. By that, these TOF arrays presumably have less redundant information. This knowledge can be used to improve future designs of TOF arrays, for example, by having a more significant overlap of specific TOF detectors or by constructing TOF arrays entirely different. For instance, it could be beneficial to use a different geometry to avoid the loss of too much redundant information if two correlated TOF detectors fail.

II. RELATED WORK

Since the data recorded from the TOF detectors are two dimensional, we consider them as images in this discussion. In the literature, the task of reconstructing missing parts of images is referred to as *image inpainting*. Typically, parts of the image are distorted or covered, and the distorted or concealed parts should be recovered. One existing method is the simple and fast *coherence transport* [6]. These algorithms are based on nonlinear higher-order partial differential equations. Nevertheless, these methods are only based on a single image and cannot incorporate knowledge from a simulation or image database. Furthermore, they rely on containing enough information on this single image to recover the missing parts of the image.

More sophisticated approaches, which incorporate knowledge from image databases containing images from the same domain, use generative adversarial networks (GANs). The following summary on the development of GAN approaches is based on [7]. A significant advance was using convolutional neural networks (CNNs) for image inpainting [8,9]. However, these methods could not distinguish if an input pixel exists in the data or is missing. The approach of using *partial convolution* is a modification that replaces missing input pixels with zeros and normalizes the output depending on the number of valid pixels [10]. This approach, however, relies on manual hardcoded certainty propagation. There exist methods to replace these components by learning them. Nevertheless, they use about half of the network parameters to propagate the certainties through the network [11,12]. This makes the evaluation of the networks more expensive. Furthermore, GAN-based approaches are hard to train because they typically suffer from mode collapse [13]. Mode collapse means the generator repeatedly produces the same image since it is a plausible output for the discriminator. If the discriminator fails to learn to reject this sample because it is in a local minimum, the generator-discriminator loop is stuck to this image. Moreover, GANs can suffer from vanishing gradients when the discriminator is trained too well [14]. Some of these shortcomings can be mitigated using the so-called Wasserstein-GANs, which uses the Wasserstein metric to measure the distance of the latent variable distribution from a standard normal distribution [14]. However, they are slower in the training process because the Wasserstein distance calculation is computationally more expensive.

Other techniques used for inpainting tasks are convolutional autoencoders [15]. An autoencoder is a neural network consisting of two parts: The encoder is composed of several layers with decreasing size. The last layer of this encoder is called bottleneck, and its neuron count is called bottleneck size. The second part, the decoder, typically has the same amount of layers but with symmetrically increasing layer sizes. In standard autoencoders, these layers are typically fully connected, while in convolutional autoencoders (CAEs), they are replaced with convolutional layers, making them particularly well suited for image-based tasks due to their ability to capture spatial hierarchies. By spatial hierarchies, we refer to the process by which the model learns local, low-level features (such as edges or small patterns) in the early convolutional layers, which are then combined into larger-scale, more abstract features (such as shapes or structures) in the deeper layers. Usually, an autoencoder is trained to reconstruct the input data precisely. This procedure allows to learn a compressed representation of the input data using the encoder and enables the decoder to unfold the data. Autoencoders provide fast and accurate reconstructions. For autoencoders, extensions exist for taking the circular setup of the TOF detectors into account [16]. These so-called circular convolutional neural networks (CCNNs) replace the zero padding from the convolution layers in the encoder by a circular padding. This means that the left-side pixels are padded with values from the right-side edge of the input and the right-side pixels are padded with values from the left-side edge. For the transposed convolution layers in the decoder, the CCNNs add paddings to their inputs in a similar way and cut the outputs than accordingly to their expected output shapes.

Another relevant extension to the CAE that can also be used for image inpainting are U-Nets as proposed by Ronneberger *et al.* [17]. U-Nets are a type of CNN architecture designed primarily for image segmentation tasks, where the goal is to classify each pixel of an image. They have a distinctive “U” shape and are—since they are autoencoders—composed of an encoder (contracting) part that captures context and features and a decoder (expanding) part that enables precise localization and reconstruction of the segmented output. U-Nets include skip connections that transfer detailed spatial information from corresponding encoder layers to decoder layers, improving accuracy in boundary delineation. Initially developed for biomedical image segmentation, U-Nets are now widely used in various fields requiring detailed image analysis. By using an output layer with similar size as the input layer, we can adapt this approach for denoising and image inpainting.

More recently, so-called stable diffusion models, initially designed for text-to-image transformations, are used for inpainting tasks [18]. Stable diffusion methods start with a randomly generated two-dimensional noise. Then, this image is denoised iteratively with a previously trained decoder. The approach presented in [18] uses the incomplete image as input. Noise is added to both the already existing and missing parts. This image is denoised by the diffusion model. After this step, the next iteration starts, but instead of taking noise to inpaint the missing parts, the denoised output of the diffusion model is used for inpainting. Scheinker and Pokharel describe how diffusion models can be used to upscale phase-space images of electron beams at the European XFEL to high-resolution formats (i.e., larger than 1024 × 1024 pixels) [19]. In the context of free electron lasers, phase-space images visualize the distribution of electrons in terms of position and momentum at various stages along the accelerator. These images are typically used to assess beam properties when adjusting the properties of the generated light between different experiments. While diffusion models deliver high-quality results for that use case, their iterative nature—typically requiring at least 1000 evaluation steps per image—makes them computationally intensive and too slow for our intended online application.

This process can be sped up with a concept named latent diffusion as shown by Scheinker and Williams also on the example of phase-space images, where the image size that the diffusion works with is decreased to a lower dimensional embedding [20]. This dimension reduction is performed with the encoder of a variational autoencoder (VAE). A VAE is a generative model that encodes input data into a latent space, not as a fixed point as in conventional autoencoders, but as a probability distribution, typically a Gaussian distribution. This distribution is constrained to learn a continuous and latent space where neighboring points decode to similar outputs, and any sampled point from the latent space results in meaningful content, enabling the generation of new samples. This constraint is enforced by a regularization term during training—typically the Kullback-Leibler divergence—which encourages the learned latent distributions to remain close to a standard normal distribution. This design enables a balance between allowing for overlapping features in latent space and robust generation of new samples through random sampling. The approach by Scheinker and Williams uses such a VAE to learn a condensed representation of the 256 × 256 pixel phase-space images. The diffusion model then learns to iteratively denoise random noisetoward this latent representation, using a conditional vector with important properties characterizing the corresponding sample.

The final solution is then to combine that diffusion model with the VAE decoder that translates the latent representation back to the actual higher resolution image. To summarize, the approach receives any conditional vector as input, converts it with the diffusion model to a latent representation of the VAE, which then returns a 256×256 resolution phase-space image. This method could be adapted to our use case by using the original detector image as the conditional vector for the diffusion model. However, this approach has the disadvantage that it does not offer the possibility to increase the bottleneck size for more detailed reconstructions without drastically increasing the required computational time—increasing the bottleneck size also increases the size of the latent representation the diffusion model has to denoise. Since for our online use case, the execution time is crucial to keep up with the data generation rate, and we want to allow for detailed denoising and reconstructions, we develop a CAE-based solution for these purposes in our approach, which can fulfill both requirements.

As previously described, GANs exhibit significant challenges during training, and diffusion models are too slow for our intended online application. Due to this difficulties, CAEs—also with the extensions of U-Nets and circular paddings—emerge as the most promising approaches. Therefore, this study will focus on evaluating and comparing these CAEs methods.

To achieve the advances discussed in Sec. I, we need to overcome some largely unexplored challenges. First, we need to investigate ways to transfer knowledge from simulation data to real-world data. Additionally, it is important to avoid overfitting on the simulation and thus ensure that the resulting model is still general enough to be applied to the real data. Finally, we need to address how we can mitigate the disproportionately negative impact of certain failed TOF detectors on overall reconstruction quality.

III. METHOD

A. Simulation and dataset

Using the publicly available [21] partial-wave-based simulation by Hartmann *et al.*, we generate detector images similar to those recorded during the experiments. Figure 1 shows an example of such a detector image on the right. We will use these detector images later to train the neural networks, because the images generated in this way have the advantage that they are without unknown noise and complete, i.e., the complete data of all TOF detectors are shown. These detector images consist of 16 TOF angles and 60 discrete equidistant kinetic energy steps, respectively. The color shows the intensity, which means the number of electrons per energy level observed at the angularly distributed TOF detectors.

The simulation first generates random spectrograms to create these detector images. A spectrogram, as shown on the left side of Fig. 1, represents the time-energy distribution of the x-ray pulses. In Fig. 1, it contains three broad intensity peaks. Even though these spectrograms cannot be measured in real-world experiments, they visually represent the most significant properties of the x-ray pulses. These properties can only be inferred indirectly through the resulting detector images. The generated spectrograms are the sum of 1–5 two-dimensional Gaussian distributions, referred to as peaks in Table I. The variance of these distributions is randomly chosen from a uniform distribution $\mathcal{U}(0, \sigma_{\max})$, denoted as the number of phase steps, one step equals $\frac{35}{80}$ femtoseconds for the used laser.

The simulation translates these spectrograms to detector images by imitating the streaking effects of the circularly polarized laser on the photoelectrons in different angular positions due to their emission time. Since this emission is induced by the x-ray pulses, and the circularly polarized laser is synchronized with the x-ray pulses, the peaks from the spectrogram produce the waves on the detector images:

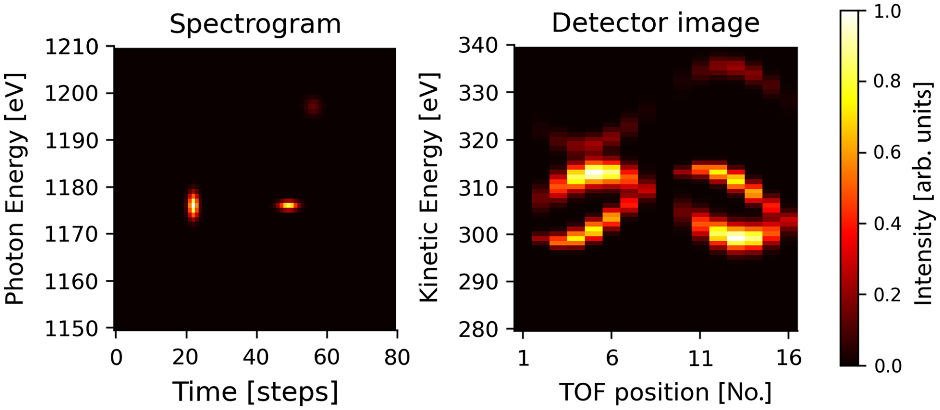

FIG. 1. The simulation generates random spectrograms (left) and translates them to the detector images (right).

TABLE I. Parameters and intervals of simulation.

Parameter	Value
Peak	$[1, \cdots, 5]$
$\sigma_{\max}$	7 steps
Kick κ	$[0, 100]$ eV
Ellipticity ε	0.73
Ellipse tilt θ	$\frac{3}{8}$ πrad
β_2	2

The top peak in the spectrogram produces the top wave in the detector image. The left intensity peak at around 20 time steps produces the broader high-intensity wave in the middle on the detector image. The right high-intensity peak produces the other wave in the middle of the detector image, showing that their produced waves can also overlap each other. It is also visible that a phase shift (displacement in time axis) results in a displacement of the TOF position of the wave and a shift in photon energy results in a displacement in the kinetic energy axis of the detector image. The remaining parameters listed in Table I are the settings of the detector setup.

The ellipticity of the polarization laser at some phase step ϕ is calculated with the following ellipticity function:

$$\mathrm{ef}(\phi) = \frac{\varepsilon^2}{[\varepsilon \cos(\phi - \theta)]^2 + [\sin(\phi - \theta)]^2}. \tag{1}$$

The ellipticity ε describes how closely the polarization of the laser approaches circularity, while the ellipse tilt angle θ represents the orientation of the polarization ellipse. These values have been measured for this experiment [3].

The translation from spectrograms $\mathbf{S} \in \mathbb{R}^{80\times 60}$ to detector images $\mathbf{D} \in \mathbb{R}^{16\times 60}$ works as follows:

$$\mathbf{D} = \sum_{i,j=0}^{i=79,j=59} \mathrm{sim}\left(j, \kappa, \frac{2\pi i}{79}\right) s_{ij}. \tag{2}$$

The values s_{ij} are the entries of the according spectrogram $\mathbf{S}$. The kick κ is the amplitude of the waves, i.e., the maximum streaking amplitude of the electrons induced by the streaking laser [5]. It is drawn from a uniform distribution $\kappa \sim \mathcal{U}(0, 100)$ for enabling unbiased training, since the kick can vary for every pulse.

The simulation of a partial wave is calculated as follows:

$$\mathrm{sim}(e_{\mathrm{kin}}, \kappa, \phi) = I_{\mathrm{electron}} \cdot g(e_{\mathrm{kin}}, \mathrm{sine}(e_{\mathrm{kin}}, \kappa, \phi), \sigma_E)^{\top}. \tag{3}$$

We set $\sigma_E = 1$ eV as the standardized width of the Gaussian curves, matching the energy step size in the detector images. This choice ensures that any realistic partial waves for the detector image can be constructed.

The angular distribution I_{electron} is calculated with the following equation [3,22]:

$$I_{\mathrm{electron}} = \frac{\beta_2}{4}[1 - \cos(2\boldsymbol{\alpha})]. \tag{4}$$

The anisotropy angular distribution parameter β_2 modulates the shape of the photoelectron angular distribution and is characteristic of the orbital of the target gas. We set $\beta_2 = 2$ because the s-orbital of neon gas is being excited.

The mathematical representation of all 16 TOF detector angles in the dipole plane is denoted with $\boldsymbol{\alpha}$.

The function g is approximating the Gaussian function with full width half maximum:

$$g(x, x_0, x_w) = \exp\left(-\frac{(x - x_0)^2}{2(\frac{x_w}{2.35})^2}\right). \tag{5}$$

While x is the independent variable for which the Gaussian function is evaluated, x_0 is the peak position of the Gaussian curve, and x_w is the width of the Gaussian curve at half of its maximum height.

The sine determines the peak position of the Gaussian curve:

$$\mathrm{sine}(e_{\mathrm{kin}}, \kappa, \phi) = e_{\mathrm{kin}} + \kappa \cos(\boldsymbol{\alpha} - \phi)\mathrm{ef}(\phi). \tag{6}$$

By running the simulation—generating random spectrograms and translating them to detector images—6 million times, the dataset for the experiments in this article is created. It is stratified by the amount of peaks, thus it contains an equal amount of samples with one to five peaks each. The number of peaks in the dataset is shuffled to prevent bias in the training process. For training, we split this dataset into 4.8 million training samples and 600 000 samples each for validation and testing. Training data are used by the neural network during training, and the validation dataset is used to assess a stopping point for training. The test dataset is not used during training and is utilized for the final evaluation of the trained models.

B. Noise and data augmentation

We incorporate different forms of artificial noise into the simulation outputs for our denoising and reconstruction method, aiming to imitate the noise of the actual TOF detectors as realistically as possible. Due to the experiment's design, noise levels are unpredictable and can vary significantly across experiments. The detector's noise can be categorized into signal-dependent and signal-independent noise.

The signal-dependent noise increases with intensity—potentially in a nonlinear manner—and can have electric or physical sources, such as secondary particles or multiphoton effects. Since the variability in noise across different experiments prevents a consistent estimation, we account for this type of noise with high-intensity peak noise in the dataset and training. We model it as a uniformly distributed random variable to avoid assumptions about the noise magnitude.

The signal-independent noise is a white detector noise that is relatively higher for weaker intensities. We address this noise with a Gaussian noise with uniformly distributed variance. Additionally, due to TOF detector calibration, saturation effects can be ruled out for our experiment. Appendix B shows that the total measured electron intensities by all TOF detectors correlate nearly linearly with the total electron intensity measured by a gas monitor detector, rather than transitioning to a flattened curve.

For training, we add noise to the detector images with disabled TOF detectors as input and the denoised complete detector image as target. Please note that the order of the following modifications is important. We choose all parameters such that the noise level of the created detector image is visually higher than in the recorded image at the real machine. (i) Dataset high-intensity peaks: During dataset creation, we simulate 1–15 random single pixels of high intensity by adding random, uniformly distributed single pixel spots to the spectrogram. The intensity of these pixels is also random, uniformly distributed. These peaks create isolated, high-intensity spots in the detector images, mimicking spikes as shown in Hartmann *et al.* [3] [Fig. 2 (c)]. (ii) Training high-intensity peaks: To create this noise, we set a probability p_{peak} to which every pixel of a TOF detector is set to the maximum value of the detector image. We choose the maximum value so that it imitates a high-intensity noise peak. In a later step, we add Gaussian noise to the entire detector image so that the Gaussian noise is also applied to the high-intensity peaks making the simulated detector images more realistic and harder to detect by the denoising mechanism. For the high-intensity peak noise, we generate a matrix $\mathbf{P}$ with equally distributed random values as shown in the following equation:

$$p_{ij} \sim \mathcal{U}(0,1). \tag{7}$$

This matrix has the same size as the detector images $i \times j$, where $i \in \{1,\cdots,16\}$ is the TOF detector position and $j \in \{1,\cdots,60\}$ is the amount of energy steps, which corresponds to the resolution of the TOF detectors. If an entry p_{ij} is less or equal to p_{peak}, it is set to the maximum value of the current detector image as follows:

$$\bar{x}_{ij} = \begin{cases} x_{ij}, & \text{if } p_{ij} > p_{\mathrm{peak}}, \\ \max\limits_{kl}(x_{kl}), & \text{else.} \end{cases} \tag{8}$$

(iii) Normalization: Next, we min-max normalize the detector images. We apply the normalization individually per image. The real-world images are also normalized per image since only the relative differences per image should be considered, given the technical functionality of TOF detectors. (iv) Gaussian noise: It is important that the previous normalization step is performed first, ensuring that the Gaussian noise is added to the normalized values, as illustrated in the following equation:

TABLE II. Different models trained with varying datasets adapted for specific purposes.

Model	Training data
1TOF, 2TOF, 3TOF	Disable 1, 2, or 3 random TOF detectors respectively
General	Disable 1–3 TOF detectors randomly
Spec	Disable the TOF detectors at positions 8 and 13

$$\tilde{x}_{ij} = \bar{x}_{ij} + p_{ij}. \tag{9}$$

The variables i and j are chosen similar to Eq. (7). The entries of the random matrix $\mathbf{P}$ are drawn from a Gaussian distribution:

$$p_{ij} \sim \mathcal{N}(0,\sigma^2). \tag{10}$$

The noise level can be chosen by setting σ which is the standard deviation of the Gaussian distribution. We choose a random noise level of $\sigma \sim \mathcal{U}(0,0.2)$ so that the noised images look as similar as possible to the real images. (v) Detector disablement: The next step is disabling the TOF detectors, with specific methods applied for training each model, as listed in Table II. The detailed detector disablement procedure is listed in Algorithm 1. In the following, “No.” denotes a TOF detector’s position, while a number before TOF detectors indicates the count of uniformly random-selected TOF detectors. We use all 4.8 million training samples for all different models and disable the TOF detectors accordingly as required for the model. Thus, all models see the same amount of data but with different TOF detectors disabled. The selection strategy of which TOF detectors to disable is shown in detail in Appendix C. The first TOF detector is selected randomly. All others are picked randomly as well, or, with probability ξ, a neighboring or opposite TOF detector is chosen for disabling if possible. (vi) Normalization: Min-max normalization is applied to each image individually again, ensuring that the values of the detector images—captured with disabled TOF detectors—are rescaled between zero and one. This normalization process can also be implemented in real-world scenarios where TOF detector failures occur. This approach remains valid since we focus on the time-energy structure rather than the beam intensity. Furthermore, as demonstrated in Appendix B, the x-ray photon energies can easily be extracted from the raw data and correlate almost linear with the corresponding values of the gas monitor detector. This correlation enables reconversion of the detector images to their original photon intensity level and thus ensuring no loss of essential information. Appendix B shows that the total measured electron intensities by all TOF detectors correlate nearly linearly with the total electron intensity measured by a gas monitor detector, rather than transitioning to a flattened curve.

As a baseline, we will use the *neighboring mean*: This approach averages over the neighbors' values to replace the values of a missing TOF detector. If a neighbor of a missing TOF detector is also missing, we take the nearest neighbor. Due to the experimental setup's circular nature, every TOF detector has two neighbors since the first TOF detector is located next to the last. Unlike the other models based on simulation data, this approach is entirely data free, meaning it relies only on information from the neighboring TOF detectors in the current detector image without requiring any training data or precomputed statistics. By checking if our proposed neural network models have a lower loss value than the neighboring mean, we verify that the neural network models do not only learn to mean over the neighbors of a missing TOF detector.

C. Models

For the neural network for denoising and TOF detector reconstruction, we assess several neural network architectures: We train a CAE architecture with five hidden layers, which we refer to as CAE-64, with filter sizes 32, 64, 128, 256, 512, and in the bottleneck 64 for the encoder, similarly for the decoder but in reverse order. The filter has dimensions 3×3, which is small enough to capture details but large enough to significantly reduce spatial dimensions. The stride is set to 2 to further reduce spatial dimensions. The padding is set to 1 to avoid underrepresentation of the border pixels. The output padding is set to 1 for a better fit to the encoder dimensions. Finally, we add a bilinear upsampling layer for getting precisely the desired output shape of 60×16, matching the simulation outputs: 16 TOF angles and 60 discrete, equidistant kinetic energy steps.

In addition, we assess alternative architectures with smaller or larger bottleneck dimensionality to provide the possibility of increasing the denoising level and decreasing reconstruction details. These architectures have filter sizes of 32, 128, 256, 512, and 1024 in the bottleneck. They replace the bottleneck layer with 64 filters from the CAE-64. This design choice reduces redundancy and inefficiency while mitigating the risk of vanishing gradients, which can occur when no significant transformations are performed. We refer to these architectures as CAE-32, CAE-128, CAE-256, CAE-512, and CAE-1024.

Due to the circular arrangement of the TOF detectors, the TOF detector at the most right in the simulation is adjacent to the most left TOF detector. We test the CCNNs described in Sec. II to model this occurrence. This extension replaces the zero padding in the convolution layers with circular padding. In circular padding, the left-side pixels are padded with values from the right-side edge of the input, and the right-side pixels are padded with values from the left-side edge.

As an additional architecture, we adapt the U-Net architecture as presented in [23] to our application scenario. An illustration of the U-Net is shown in Fig. 2. All hyperparameters and layers are similar to the CAE-512, if not explicitly stated. Thus, we refer to the model trained with this architecture as UNet-512. The encoder—highlighted in red—is completely identical to the encoder of the CAE-512. The distinctive feature of the U-Net lies in its decoder, shown in blue, which follows the bottleneck layer of size $512 \times 2 \times 1$.

The decoder begins with a transposed convolution layer of size $256 \times 4 \times 1$, which is concatenated with the corresponding encoder output of the same size, $256 \times 4 \times 1$. This concatenation returns an output of size $512 \times 4 \times 1$, which is then passed to the next decoder layer. Each subsequent decoder layer follows this pattern: it receives the output from the previous decoder layer and concatenates it with the output from the corresponding encoder layer at the same hierarchical level. This information propagation from the encoder to the decoder is illustrated by the dotted gray lines in Fig. 2.

This skip-connection strategy enables the decoder to access both high-level abstract features from the bottleneck and low-level spatial details from the encoder. As a result, the outer layers in the architecture contain more fine-grained information, while the deeper layers capture more abstract and global representations. This design is the reason for choosing a comparatively large network with a bottleneck size of $512 \times 4 \times 1$ for this architecture: U-Nets, due to their skip connections, are typically able to reconstruct more details with the same number of neurons. The architectures with larger bottlenecks (e.g., CAE-512 and CAE-1024) are specifically designed for reconstructions with a high level of detail—this is the domain where U-Nets can fully leverage their advantages by delivering detailed reconstructions through transferring information with the skip connections. In contrast, models with small to medium bottlenecks (CAE-32 to CAE-256) enforce full compression of the input into the bottleneck. This architectural constraint encourages the model to discard information that is likely attributable to noise or artifacts, resulting in a more compact and interpretable latent representation—which is particularly beneficial for downstream analysis. For this reason, we still choose the conventional CAE-64 as our reference model, despite the U-Net architecture's superior reconstruction performance.

As an activation function, the *Mish* function proposed in [24] outperformed the frequently used ReLU function. Furthermore, we used the AdamW optimizer as presented in [25]. The regularization term of the optimizer results in increased training and validation losses. However, the models denoise and reconstruct visually more accurately

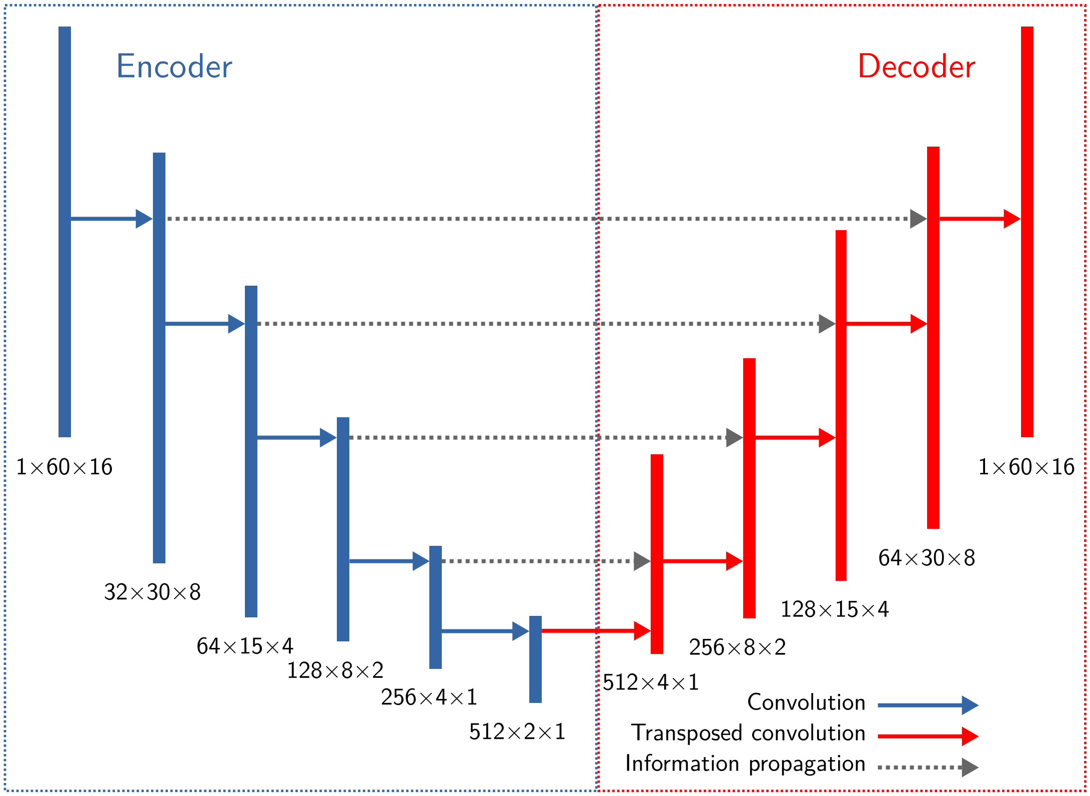

FIG. 2. U-Net architecture as adapted from [23]. The encoder (red) with the structure of the CAE-512; the decoder (blue) reconstructs the output using transposed convolutions and skip connections. Dotted gray lines indicate the flow of information from encoder to decoder.

on real-world images, as shown in Appendix D. Thus, AdamW provides a countermeasure against overfitting on the simulation. We use learning rate plateau scheduling, beginning with a learning rate $\eta = 10^4$ and decaying it exponentially every epoch by multiplying it with 0.1 if the validation loss has not improved more than $\epsilon = 10^{-8}$ for three epochs. Despite the outputs being min-max normalized and thus between 0 and 1, using no activation function after the last layer resulted in slightly more accurate reconstruction and denoising than using a sigmoid output function. We stopped the training after 50 epochs, since no significant increase in validation loss was noticeable. Using these best neural network parameters and architecture, we train the different models *1TOF*, *2TOF*, *3TOF*, *general*, and *spec* with differently augmented data as listed in Table II and compare those for different settings.

As baseline for the denoising process, we use a Wiener filter as described by Lim [26]. This Wiener filter works by minimizing the mean squared error (MSE) between the estimated output signal and the original signal. It adapts its response based on both the signal and noise characteristics. We use a 3×3 filter window size, following our choice in the CAE. The noise is estimated as the average of the local variance—which refers to the variance in the corresponding filter window—of the input. The denoising process with the Wiener filter is described in more detail in Appendix F.

D. Real-world data application

We use detector images from an experiment at LCLS as presented in [3] to show our method in a real-world setting. The time of flight t measured by the TOF detectors correlates with the energy E roughly proportional to const $+ \frac{1}{t^2}$. Measurements are taken equidistantly in time of flight and then mapped onto equally sized energy intervals. Consequently, the number of measurement points in time of flight varies per equidistant energy E interval. This mapping is used because it is physically easier to measure equal intervals in terms of time of flight rather than energy. Furthermore, the scaling factors differ for all TOF detectors, and it would be required to remeasure and calculate the time-to-energy conversion for each experiment. Thus, it is easier to map the energies to equidistant 1 eV bins rather than adapting the simulation for every experiment.

Because of this binning of the energy values in 1 eV steps in the simulation, some bins always remain empty since, in the area with lower resolution, no TOF detector measures in the specific 1 eV range. To mitigate this issue, if a bin stays empty, we distribute a third of the values of the two neighboring bins to the empty bin. If an empty bin has only one neighbor, we shift half of the value of this bin to the empty one.

Furthermore, we round the positions of the TOF detectors scientifically. For example, the 280.5 eV TOF detector value is added to the 281 eV bin. After this procedure, we shift all negative TOF detector values to 0.0 and min-max normalize every detector image.

IV. EVALUATION

In this section, we will first evaluate the denoising performance on detector images of our models. After that, we will examine the reconstruction of failed TOF detectors. For both applications, we will provide and discuss a real-world example.

A. Denoising

As shown in Table III, we compare the rooted mean squared error (RMSE) of different models. The scenario *0 failed* means that no TOF detector is disabled, thus the models perform only denoising. It is noticeable that the specifically trained models are worse in this scenario than the general model. Compared to the respectively best model, the baseline neighboring mean is worse by almost a factor of 10 in terms of the RMSE. The † next to an RMSE value indicates a significant difference from the best models' RMSE. To assess whether differences in the RMSE values are meaningful, we compare the 99% confidence intervals (CIs) of the means and consider them significantly different if the CIs of this difference does not contain zero. Given the large test sample size of 600 000, we provide the CIs in Table VI to assess whether the statistically significant differences are also practically relevant. Since the CIs are in the range of the RMSE values, we can consider the differences between the compared models and the spec model practically relevant.

In Fig. 3, we show a comparison of the original real data and the denoised results obtained with models across different-sized bottleneck architectures. All models are trained in the general setting—that is, with 1–3 TOF detectors disabled.

TABLE III. RMSE on simulated data without failed TOF detectors for all models, showing model performance on only denoising. Bold numbers indicate the lowest RMSE in each column. A dagger symbol (†) marks results that are statistically significantly different from those of the best-performing model in the same column, at the 1% significance level ($p < 0.01$).

Failed TOFs scenario	0 random
1TOF model	4.62×10^{-2} †
2TOF model	5.14×10^{-2} †
3TOF model	5.98×10^{-2} †
General model	$\mathbf{4.50 \times 10^{-2}}$
Spec model	6.87×10^{-2} †
Neighboring mean	2.78×10^{-1} †

The RMSE between the Wiener filter output and the CAE-64 output is 0.143, while the RMSE between the original image and the CAE-64 output is 0.151. Additionally, the RMSE between the original images and the Wiener filter outputs is 0.072, which is significantly smaller compared to the differences with the CAE-64 outputs.

This shows that the CAE-64 output images are closer to the Wiener-filtered images than to the original images. However, the changes introduced by the Wiener filter are much smaller than those introduced by the CAE-64.

B. Reconstruction

In Fig. 4, we show an example of a noisy image from simulation with two disabled TOF detectors in the image on the left. We used the spec model for denoising and reconstruction of the missing TOF detectors shown on the right image. The spec model reconstructs most of the details in the label (center), while providing a fitting reconstruction of the missing TOF detectors. In Table IV, we show the different architectures in terms of different sized bottlenecks. In Table VII we provide extended statistics for this comparison. There is a clear connection between the bottleneck size and the reconstruction quality on synthetic data. The comparison of architectures with varying bottleneck sizes shows that larger bottlenecks result in more minor reconstruction errors in simulation data. The CAE-1024 outperforms all other models in terms of the reached RMSE. In Fig. 5, we deleted the data for two TOF detectors at positions No. 8 and No. 13 from a real-world detector image and used the model specifically trained to reconstruct the TOF detectors No. 8 and No. 13. We decided to show the results on this combination, since it contains one easier and one harder TOF detector to reconstruct, according to Fig. 6. We can see that the image produced by the model is similar to the reconstruction of the detector image with all TOF detectors.

However, this behavior cannot be transferred to the real-world images: Larger bottlenecks exhibit weaker denoising performance when applied to real-world data, and the reconstructed TOF detector data integrate less cohesively with the rest of the detector image, as shown in Fig. 5. Additionally, larger bottlenecks tend to reproduce noise peaks in the output. Conversely, smaller bottlenecks yield detector images closer in appearance to simulation data while being less detailed than the original real-world detector images. The outputs of the CAE-32 architecture tend to be overly simplistic, while larger bottlenecks allow for too many details, including noise artifacts. The CAE-64 balances reconstruction accuracy and effective denoising, making it the focus of this study. Nevertheless, other architectures may prove beneficial for different application scenarios.

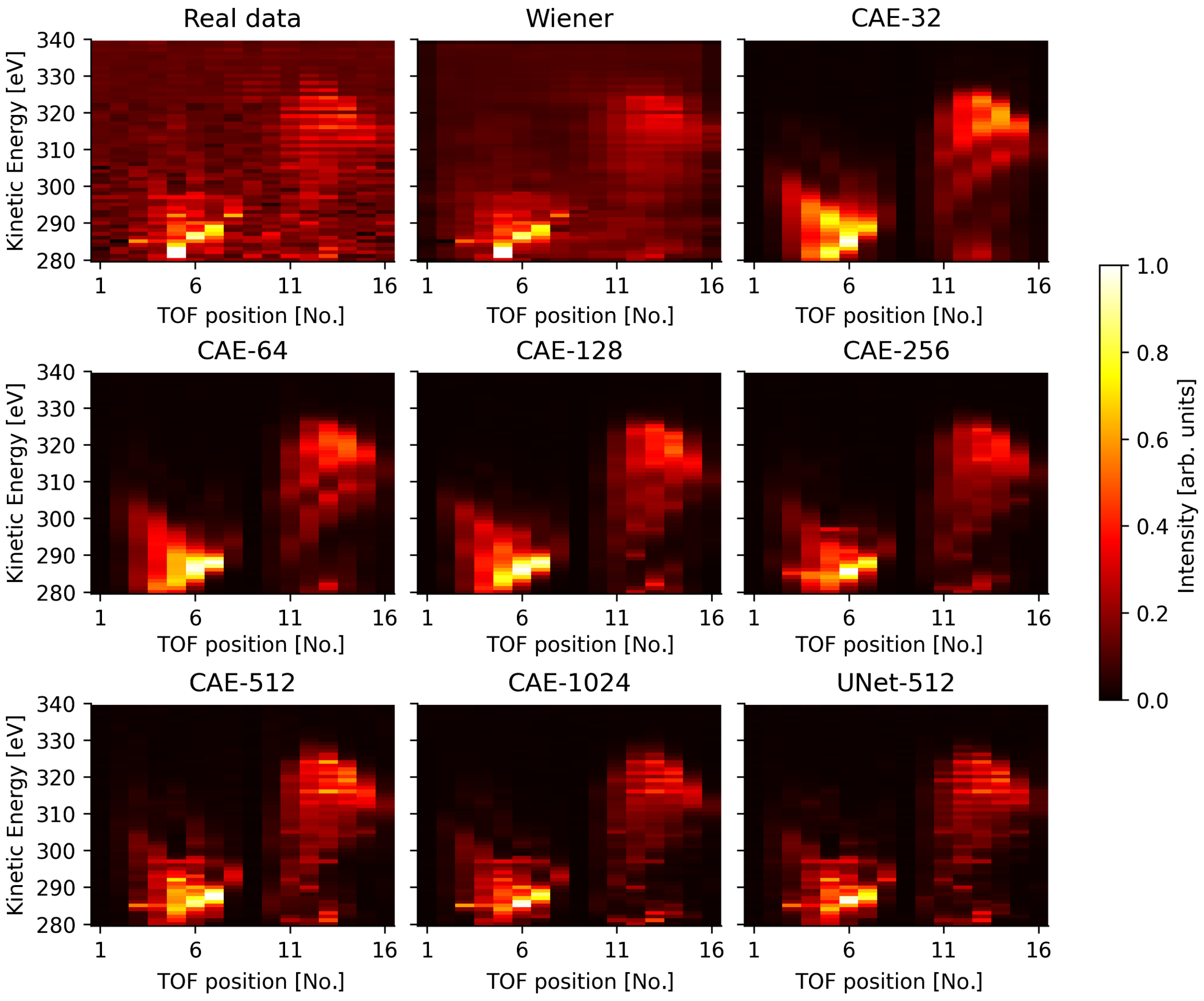

FIG. 3. The first plot shows a real-world detector image, while the second plot shows the same image denoised using Wiener filtering. The following plots show the denoised detector images produced by CAE models with varying bottleneck sizes, as well as by the model with UNet-512 architecture. All models used for creating these plots are trained in the general setting—that is, with 1–3 TOF detectors disabled.

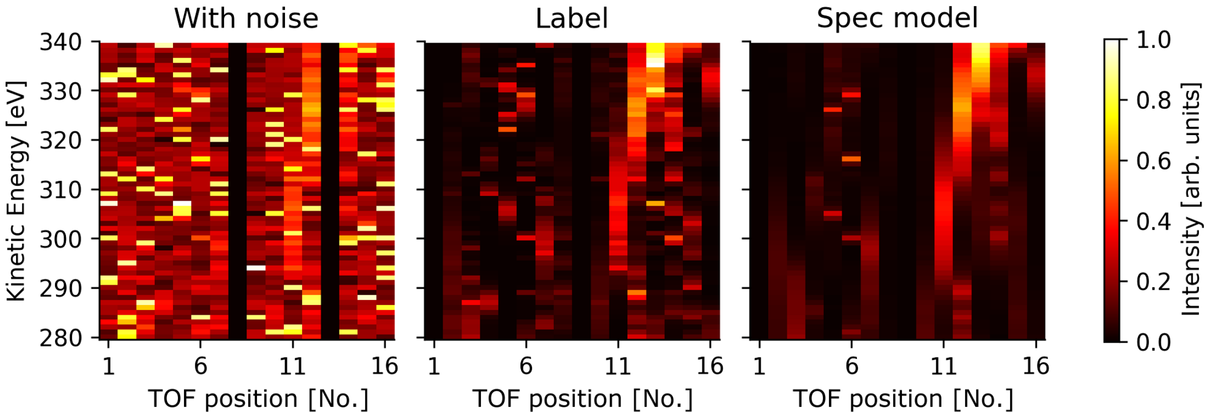

FIG. 4. The sample on the left is with two disabled TOF detectors, Gaussian noise, and hot peaks. In the middle is the same sample without disabled TOF detectors and noise. The right plot shows the reconstructed and denoised detector image.

TABLE IV. RMSE comparison across different architectures for different failure scenarios, with 1–3 TOF detectors disabled. Bold numbers indicate the lowest RMSE in each column. A dagger symbol (†) marks results that are statistically significantly different from those of the best-performing model in the same column, at the 1% significance level ($p < 0.01$).

Failed TOFs scenario	1 random	2 random	3 random	2 neighbors	2 opposite	No. 8 and No. 13 position
CAE-32	5.36×10^{-2}†	5.49×10^{-2}†	5.67×10^{-2}†	5.56×10^{-2}†	5.45×10^{-2}†	5.48×10^{-2}†
CAE-64	4.61×10^{-2}†	4.77×10^{-2}†	4.98×10^{-2}†	4.86×10^{-2}†	4.73×10^{-2}†	4.74×10^{-2}†
CAE-128	4.08×10^{-2}†	4.27×10^{-2}†	4.50×10^{-2}†	4.36×10^{-2}†	4.22×10^{-2}†	4.23×10^{-2}†
CAE-256	3.65×10^{-2}†	3.88×10^{-2}†	4.14×10^{-2}†	3.97×10^{-2}†	3.83×10^{-2}†	3.87×10^{-2}†
CAE-512	3.50×10^{-2}†	3.78×10^{-2}†	4.11×10^{-2}†	3.88×10^{-2}†	3.72×10^{-2}†	3.78×10^{-2}†
CAE-1024	$\mathbf{3.25 \times 10^{-2}}$	$\mathbf{3.51 \times 10^{-2}}$	$\mathbf{3.80 \times 10^{-2}}$	$\mathbf{3.59 \times 10^{-2}}$	$\mathbf{3.46 \times 10^{-2}}$	$\mathbf{3.52 \times 10^{-2}}$
UNet-512	3.31×10^{-2}†	3.61×10^{-2}†	3.95×10^{-2}†	3.70×10^{-2}†	3.55×10^{-2}†	3.63×10^{-2}†
Neighboring mean	2.78×10^{-1}†	2.78×10^{-1}†	2.78×10^{-1}†	2.79×10^{-1}†	2.77×10^{-1}†	2.78×10^{-1}†

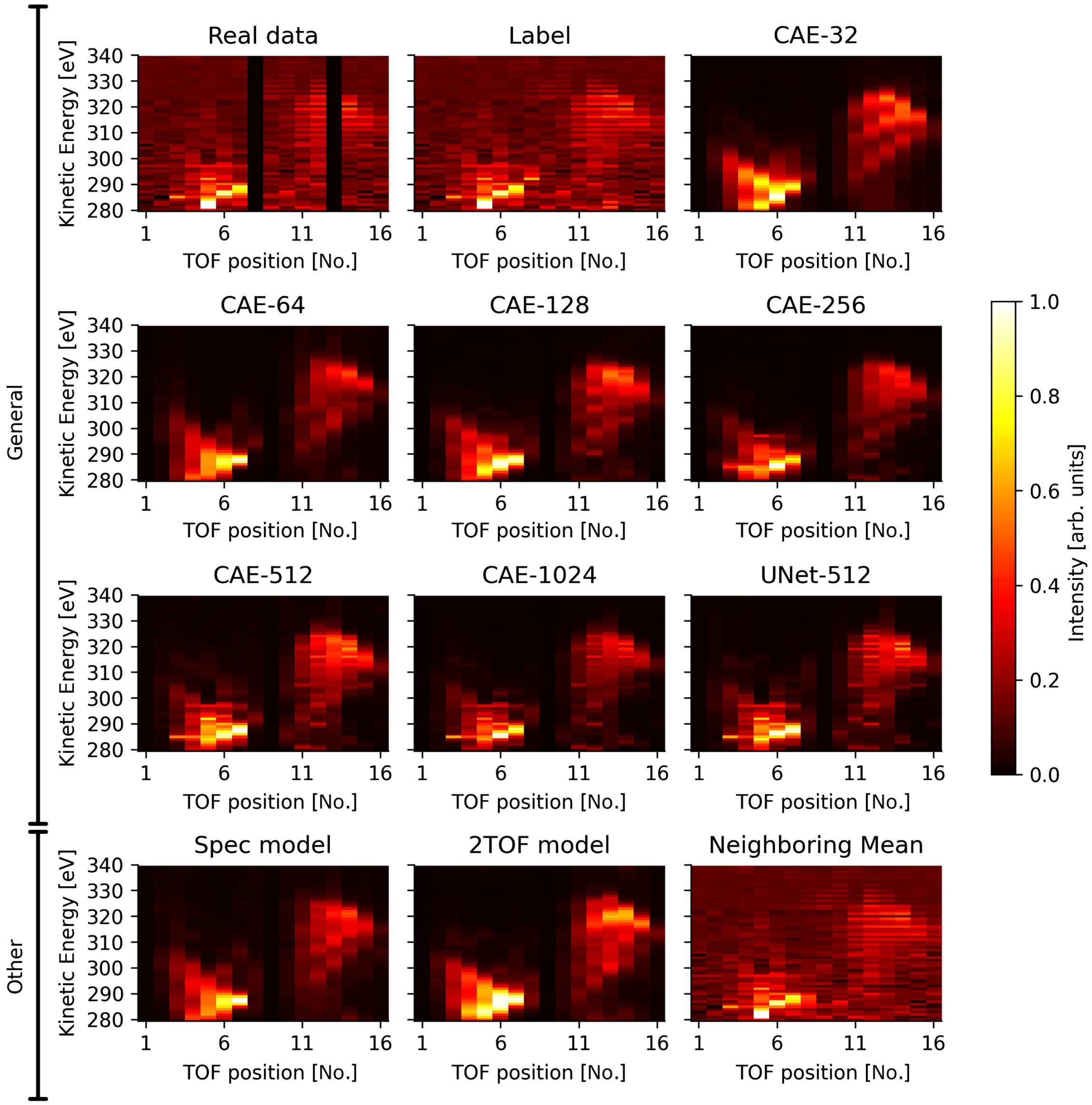

FIG. 5. The first plot shows a real-world detector image with two TOF detectors disabled, followed by the label without disabled detectors. The following plots show the denoised and reconstructed detector images across different model architectures. All models in these plots are trained in the general setting, that means with 1–3 TOF detectors disabled, except for the three models in the bottom line: The first one is the output of the CAE-64 model trained with data in spec setting (TOF positions No. 8 and No. 13 disabled, exactly the positions disabled for this reconstruction). The second one is the CAE-64 model trained with always two TOF detectors disabled. The last one is the output of the neighboring mean, which recreates the missing TOF detectors by averaging over the neighboring detectors.

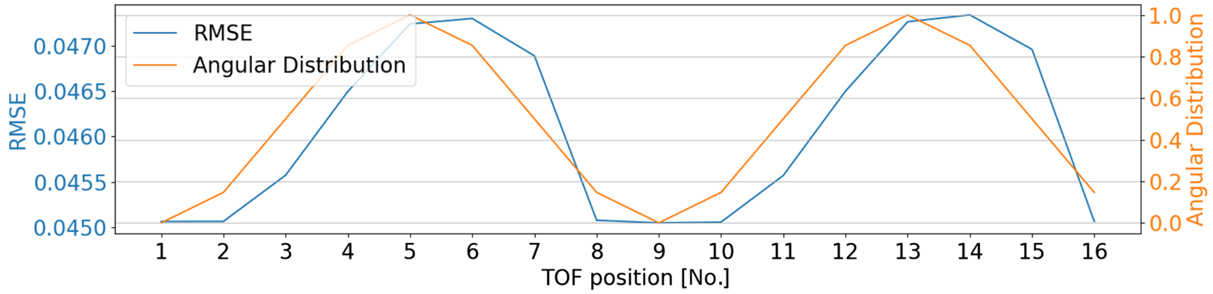

FIG. 6. The RMSE values of the reconstruction if one TOF detector has failed (blue). The y axis is truncated for emphasizing the differences. For reference, we also plotted the angular distribution (orange).

As shown in Table IV, the UNet-512 model achieves a lower reconstruction MSE with a comparable number of neurons. This improvement is primarily due to the skip connections, which allow information to bypass the bottleneck and to be transferred directly from the encoder layers to the according decoder layers. This effect is particularly relevant for models with larger bottlenecks (e.g., CAE-512 and CAE-1024), as these are designed to reconstruct more fine-grained details.

Nevertheless, for tasks requiring high-quality reconstruction and detailed denoising, the UNet-512 constitutes a valuable extension for our approach. It combines strong reconstruction performance with relatively low memory usage and fast evaluation, even when accounting for the additional overhead introduced by the skip connections.

In Table V, we quantitatively compare the models 1TOF, 2TOF, and 3TOF with the general and neighboring mean for different scenarios. In the *random* scenario, the number of disabled TOF detectors is drawn randomly from a uniform distribution. In the *neighbors* scenario, failures are introduced by disabling TOF detectors that are adjacent to each other. In the *opposite* scenario, opposing TOF detectors are disabled whenever possible. Finally, in the *position* scenario, TOF detectors at positions No. 8 and No. 13 are consistently disabled. The according general model performs significantly better than all other models, even if the model is trained for a more specialized scenario. The only exception is the 1TOF model for the scenario of one random TOF detector failure, but this difference is not statistically significant. The only exception is the scenario of the failure of TOF detectors at position No. 8 and No. 13, where the specific model that is trained for failure of these specific positions performs significantly better. In Table IX, we provide additional statistics with the CIs. The mean is in all cases almost worse by a factor of 20 in comparison to the best model. In Fig. 6, we show the evaluation of the test dataset on the general model, i.e., 600 000 simulated examples. We evaluate the model with one missing TOF detector at all 16 possible positions and calculate the mean RMSE. We can see that the RMSE values of different TOF detector positions are pretty similar but have slight peaks at position No. 6 and No. 14. This behavior arises because the angular distribution resulting from the chosen target gas leads to reduced intensity in this region, making it more challenging to predict. In Appendix E, we provide an additional, phase-separated examination of a single failed TOF detector.

In Fig. 7, we show a matrix of the evaluation of RMSE values for two failed TOF detectors. If a value is on the diagonal, we plot the RMSE value of the particular single failed TOF detector for reference. The darker the plot's color, the lower the occurring RMSE. The highest error occurs if TOF detectors No. 13 and No. 14 fail, followed by No. 5 and No. 6 in a repeating pattern shifted by 8 TOF detectors. Generally, the combinations with TOF positions No. 5, No. 6, No. 13, and No. 14 along with any other failed TOF detector have higher errors than the rest. This behavior also stems from the angular distribution of the experimental setup, similar to the single failed TOF

TABLE V. RMSE for different failure scenarios, where 1 to 3 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column. A dagger symbol (†) marks results that are statistically significantly different from those of the best-performing model in the same column, at the 1% significance level ($p < 0.01$).

Failed TOFs scenario	1 random	2 random	3 random	2 neighbors	2 opposite	No. 8 and No. 13 position
1TOF model	4.62×10^{-2}†	5.74×10^{-2}†	7.35×10^{-2}†	7.51×10^{-2}†	4.90×10^{-2}†	4.89×10^{-2}†
2TOF model	4.78×10^{-2}†	4.81×10^{-2}†	5.23×10^{-2}†	4.90×10^{-2}†	4.76×10^{-2}†	4.77×10^{-2}†
3TOF model	5.21×10^{-2}†	4.94×10^{-2}†	4.99×10^{-2}†	5.00×10^{-2}†	4.91×10^{-2}†	4.88×10^{-2}†
General model	$\mathbf{4.61 \times 10^{-2}}$	$\mathbf{4.77 \times 10^{-2}}$	$\mathbf{4.98 \times 10^{-2}}$	$\mathbf{4.86 \times 10^{-2}}$	$\mathbf{4.73 \times 10^{-2}}$	4.74×10^{-2}†
Spec model	9.38×10^{-2}†	1.10×10^{-1}†	1.24×10^{-1}†	1.10×10^{-1}†	1.09×10^{-1}†	$\mathbf{4.55 \times 10^{-2}}$
Neighboring mean	2.78×10^{-1}†	2.78×10^{-1}†	2.78×10^{-1}†	2.79×10^{-1}†	2.77×10^{-1}†	2.78×10^{-1}†

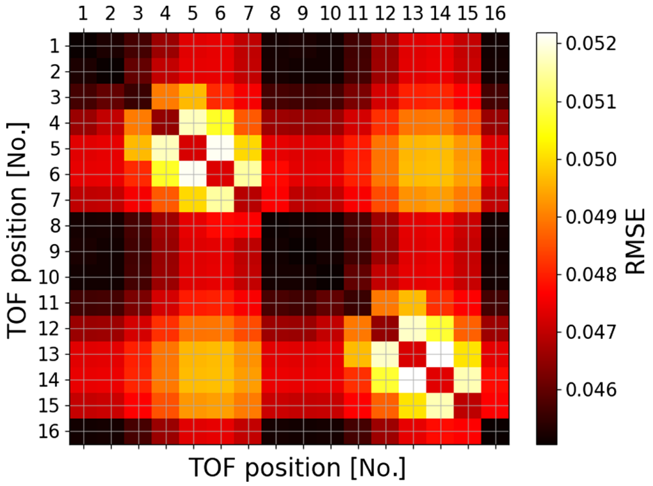

FIG. 7. The RMSE values of the reconstruction if two TOF detectors have failed.

detector examination. The best reconstructions are possible for combinations with TOF positions No. 1, No. 2, No. 8, No. 9, and No. 16 have failed. This behavior can be explained since these TOFs detectors receive no electrons or only negligible amounts. Thus, these parts of the detector image can be reconstructed with smaller errors. The occurring patterns in reconstruction errors when two TOF detectors have failed fit the patterns observed when one TOF detector has failed, as seen in the diagonal entries.

V. RESULTS AND DISCUSSION

The evaluation in the previous section has shown that the models are able to accurately denoise and reconstruct 1–3 failed TOF detectors. We also tested the reconstruction of higher TOF detector counts, however, the reconstruction quality decreases rapidly for four or five failed TOF detectors, as shown in Table X. We evaluated our models for several scenarios on an extensive test set that was not present during the training phase of the models.

The general model shows the best reconstruction quality if no TOF detectors have failed. This behavior is not surprising since no model was trained for that specific scenario, and due to its generality, this model is the most robust and thus can handle unseen cases the best. All presented neural network models reconstruct the missing TOF detectors considerably better than the baseline neighboring mean. Furthermore, we have shown qualitatively that our models work on real-world images. The produced denoised detector images look similar to the noisy real-world images. If we additionally turn off TOF detectors on the real-world images, the model produces a similar image to a complete input detector image. A numerical evaluation for real-world images is impossible since we cannot generate noiseless detector images for comparison in a real-world scenario.

The comparison of the failure of one specific TOF detector shows that the position of a TOF has a considerable impact on its reconstruction loss, as shown in Fig. 6. A higher value in the angular distribution—that is due to the choice of the target gas and the experiment setup—means these areas often have lower intensities and more possible shapes. Also, combinations of neighbors and opposite TOF detectors result in the higher intensity regions of TOF detector positions No. 4–No. 7 and positions No. 12–No. 15 a slight increase in reconstruction error. This phenomenon can be explained because neighboring or opposite TOF detectors share more information than other combinations. If both TOF detectors fail, more information is lost compared to other combinations. In the lower intensity regions of TOF detector positions No. 1–No. 3 and positions No. 8–No. 11, the differences in the reconstruction RMSE values are considerably less present. These aspects could be considered in future designs of angular streaking experiments.

Regarding neural network training, one of the essential things is regularization; we use the AdamW optimizer as described in [25]. This technique helps to keep the weights of the neural network low and thus avoids overfitting not only on training data but also on the simulation.

In Table VIII, we show the evaluation of different neighboring factors γ. The ratio of shown training samples does not significantly change the training behavior since the RMSE is almost equal for all scenarios. Since there is no adverse effect, we set $\gamma = 0.5$ for all subsequent experiments. We also tested the previously described CCNNs, which even decreased the reconstruction error. This behavior could be evoked by the cropping that is required in the transposed convolution layers in the decoder due to the additional paddings in the encoder part. Also the addition of padding to the input data decreases the reached RMSEs, this might happen because additional input comes to the cost of additional neurons that need to be learned.

The comparisons of the different models, the general model for 1–3 failed TOF detectors, and the models for the distinct number of failed TOF detectors show that the more specialized a model is, the more accurate its reconstructions are. However, we can verify this only for the simulated test data, even though the visual comparisons indicate similar results for the real-world samples. This behavior is probably caused by the more limited ranges of restored intensities in the case of more specialized scenarios. Thus, the trained model's predictions are more accurate.

VI. CONCLUSION AND OUTLOOK

The presented method provides online denoising and reconstruction of missing TOF detectors. We have shown that our presented approach for reconstructing missing TOF detectors outperforms a simple algorithm that calculates the mean over the remaining neighbors of a missing TOF detector. Validating our approach on real-world detector images is difficult because no labels exist. However, the results are visually viable, and due to its fast inference time, our presented method can be used during experimentation time for immediate monitoring and extraction of attributes for further pulse characterization. A first step toward this has been shown in [5]: Extracting the kick, Auger decay time, pulse structure, and duration is possible. The pulse structure is the intensity of a pulse over time, and the pulse duration is the period between a pulse's first and last intensity. Complete and denoised data are inevitably required to obtain correct characterization results.

Alternative layouts for the angular streaking experiment can be efficiently tested and evaluated through simulation to optimize the design of experimental setups. These setups include configurations with higher overlapping TOF detector layouts or smaller TOF detectors, which correspond to smaller solid angles. A smaller solid angle refers to a reduced portion of the detector's field of view covered by each TOF detector, this can be compensated by increasing the number of detectors. Also, to test other scenarios or setups, the other simulation parameters can be changed, such as peak numbers or kick ranges. A realistic amount of noise can be added for all these scenarios, and the expected number of failed TOF detectors can be wiped and reconstructed. The error of denoising and reconstruction indicates if there is enough redundant information encoded so the detector image can be reconstructed.

The recently emerged diffusion models could be tested to improve our approach's accuracy and robustness. However, this type of network typically requires at least 20 iterations for inference, which negatively impacts inference time, making them impracticable for an online evaluation setting. Nevertheless, a more accurate off-line evaluation could benefit from applying diffusion models.

ACKNOWLEDGMENTS

Support by the JointLab Artificial Intelligence Methods for Experiment Design (AIM-ED) between Helmholtz-Zentrum für Materialien und Energie, Berlin and the University of Kassel is gratefully acknowledged. We also acknowledge the useful feedback from the co-authors, Shang Gao and Pascal Plettenberg.

DATA AVAILABILITY

The data that support the findings of this article are openly available [27].

APPENDIX A: EVALUATION STATISTICS

In this section, we list the detailed statistics of the evaluated denoising and reconstruction scenarios for different models. The first number is the mean, followed by the standard deviation. In braces, we supply the 99% CIs of the related t test to check whether the differences of the RMSEs achieved by a model are statistically significant in comparison to the best evaluated model.

TABLE VI. RMSEs and extended statistics for denoising scenario, where no TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column.

Failed TOFs scenario	0 random
1TOF model	$4.62 \times 10^{-2} \pm 1.87 \times 10^{-2}$ $(-1.18 \times 10^{-3}, -1.17 \times 10^{-3})$
2TOF model	$5.14 \times 10^{-2} \pm 2.17 \times 10^{-2}$ $(-6.40 \times 10^{-3}, -6.38 \times 10^{-3})$
3TOF model	$5.98 \times 10^{-2} \pm 2.92 \times 10^{-2}$ $(-1.48 \times 10^{-2}, -1.48 \times 10^{-2})$
General model	$\mathbf{4.50 \times 10^{-2} \pm 1.87 \times 10^{-2}}$
Spec model	$6.87 \times 10^{-2} \pm 1.83 \times 10^{-2}$ $(-2.37 \times 10^{-2}, -2.37 \times 10^{-2})$
Neighboring mean	$2.78 \times 10^{-1} \pm 2.19 \times 10^{-2}$ $(-2.33 \times 10^{-1}, -2.33 \times 10^{-1})$

TABLE VII. RMSEs and extended statistics for reconstruction of different failure scenarios across different architectures, where 1–3 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column.

Failed TOFs scenario	1 random	2 random	3 random	2 neighbors	2 opposite	No. 8 and No. 13 position
CAE-32	5.36×10^{-2} $\pm 2.01 \times 10^{-2}$ $(-2.11 \times 10^{-2}, -2.10 \times 10^{-2})$	5.49×10^{-2} $\pm 2.03 \times 10^{-2}$ $(-1.98 \times 10^{-2}, -1.98 \times 10^{-2})$	5.67×10^{-2} $\pm 2.06 \times 10^{-2}$ $(-1.87 \times 10^{-2}, -1.87 \times 10^{-2})$	5.56×10^{-2} $\pm 2.04 \times 10^{-2}$ $(-1.97 \times 10^{-2}, -1.97 \times 10^{-2})$	5.45×10^{-2} $\pm 2.03 \times 10^{-2}$ $(-1.99 \times 10^{-2}, -1.99 \times 10^{-2})$	5.48×10^{-2} $\pm 2.02 \times 10^{-2}$ $(-1.96 \times 10^{-2}, -1.96 \times 10^{-2})$
CAE-64	4.61×10^{-2} $\pm 1.88 \times 10^{-2}$ $(-1.36 \times 10^{-2}, -1.36 \times 10^{-2})$	4.77×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-1.27 \times 10^{-2}, -1.26 \times 10^{-2})$	4.98×10^{-2} $\pm 1.94 \times 10^{-2}$ $(-1.18 \times 10^{-2}, -1.18 \times 10^{-2})$	4.86×10^{-2} $\pm 1.91 \times 10^{-2}$ $(-1.27 \times 10^{-2}, -1.26 \times 10^{-2})$	4.73×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-1.27 \times 10^{-2}, -1.27 \times 10^{-2})$	4.74×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-1.22 \times 10^{-2}, -1.22 \times 10^{-2})$
CAE-128	4.08×10^{-2} $\pm 1.62 \times 10^{-2}$ $(-8.32 \times 10^{-3}, -8.30 \times 10^{-3})$	4.27×10^{-2} $\pm 1.67 \times 10^{-2}$ $(-7.60 \times 10^{-3}, -7.58 \times 10^{-3})$	4.50×10^{-2} $\pm 1.73 \times 10^{-2}$ $(-6.99 \times 10^{-3}, -6.97 \times 10^{-3})$	4.36×10^{-2} $\pm 1.69 \times 10^{-2}$ $(-7.68 \times 10^{-3}, -7.66 \times 10^{-3})$	4.22×10^{-2} $\pm 1.67 \times 10^{-2}$ $(-7.60 \times 10^{-3}, -7.57 \times 10^{-3})$	4.23×10^{-2} $\pm 1.66 \times 10^{-2}$ $(-7.14 \times 10^{-3}, -7.12 \times 10^{-3})$
CAE-256	3.65×10^{-2} $\pm 1.42 \times 10^{-2}$ $(-4.06 \times 10^{-3}, -4.03 \times 10^{-3})$	3.88×10^{-2} $\pm 1.50 \times 10^{-2}$ $(-3.69 \times 10^{-3}, -3.67 \times 10^{-3})$	4.14×10^{-2} $\pm 1.59 \times 10^{-2}$ $(-3.42 \times 10^{-3}, -3.39 \times 10^{-3})$	3.97×10^{-2} $\pm 1.52 \times 10^{-2}$ $(-3.77 \times 10^{-3}, -3.74 \times 10^{-3})$	3.83×10^{-2} $\pm 1.49 \times 10^{-2}$ $(-3.69 \times 10^{-3}, -3.66 \times 10^{-3})$	3.87×10^{-2} $\pm 1.48 \times 10^{-2}$ $(-3.50 \times 10^{-3}, -3.48 \times 10^{-3})$
CAE-512	3.50×10^{-2} $\pm 1.34 \times 10^{-2}$ $(-2.48 \times 10^{-3}, -2.45 \times 10^{-3})$	3.78×10^{-2} $\pm 1.44 \times 10^{-2}$ $(-2.74 \times 10^{-3}, -2.72 \times 10^{-3})$	4.11×10^{-2} $\pm 1.55 \times 10^{-2}$ $(-3.08 \times 10^{-3}, -3.06 \times 10^{-3})$	3.88×10^{-2} $\pm 1.48 \times 10^{-2}$ $(-2.93 \times 10^{-3}, -2.90 \times 10^{-3})$	3.72×10^{-2} $\pm 1.44 \times 10^{-2}$ $(-2.64 \times 10^{-3}, -2.61 \times 10^{-3})$	3.78×10^{-2} $\pm 1.41 \times 10^{-2}$ $(-2.65 \times 10^{-3}, -2.63 \times 10^{-3})$
CAE-1024	$\mathbf{3.25 \times 10^{-2}}$ $\mathbf{\pm 1.27 \times 10^{-2}}$	$\mathbf{3.51 \times 10^{-2}}$ $\mathbf{\pm 1.37 \times 10^{-2}}$	$\mathbf{3.80 \times 10^{-2}}$ $\mathbf{\pm 1.48 \times 10^{-2}}$	$\mathbf{3.59 \times 10^{-2}}$ $\mathbf{\pm 1.40 \times 10^{-2}}$	$\mathbf{3.46 \times 10^{-2}}$ $\mathbf{\pm 1.37 \times 10^{-2}}$	$\mathbf{3.52 \times 10^{-2}}$ $\mathbf{\pm 1.34 \times 10^{-2}}$
UNet-512	3.31×10^{-2} $\pm 1.28 \times 10^{-2}$ $(-5.75 \times 10^{-4}, -5.51 \times 10^{-4})$	3.61×10^{-2} $\pm 1.38 \times 10^{-2}$ $(-1.04 \times 10^{-3}, -1.02 \times 10^{-3})$	3.95×10^{-2} $\pm 1.50 \times 10^{-2}$ $(-1.53 \times 10^{-3}, -1.50 \times 10^{-3})$	3.70×10^{-2} ± 1.42 $(-1.15 \times 10^{-3}, -1.12 \times 10^{-3})$	3.55×10^{-2} $\pm 1.39 \times 10^{-2}$ $(-9.62 \times 10^{-4}, -9.36 \times 10^{-4})$	3.63×10^{-2} $\pm 1.33 \times 10^{-2}$ $(-1.11 \times 10^{-3}, -1.09 \times 10^{-3})$
Neighboring mean	2.78×10^{-1} $\pm 2.19 \times 10^{-2}$ $(-2.45 \times 10^{-1}, -2.45 \times 10^{-1})$	2.78×10^{-1} $\pm 2.18 \times 10^{-2}$ $(-2.43 \times 10^{-1}, -2.43 \times 10^{-1})$	2.78×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.40 \times 10^{-1}, -2.40 \times 10^{-1})$	2.79×10^{-1} $\pm 2.15 \times 10^{-2}$ $(-2.43 \times 10^{-1}, -2.43 \times 10^{-1})$	2.77×10^{-1} $\pm 2.20 \times 10^{-2}$ $(-2.43 \times 10^{-1}, -2.43 \times 10^{-1})$	2.78×10^{-1} $\pm 2.16 \times 10^{-2}$ $(-2.43 \times 10^{-1}, -2.43 \times 10^{-1})$

TABLE VIII. RMSEs and extended statistics for reconstruction of different failure scenarios across different parameter settings, where 1–3 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column.

Failed TOFs scenario	1 random	2 random	3 random	2 neighbors	2 opposite	No. 8 and No. 13 position
$\gamma = 0.3$ CAE-64	4.64×10^{-2} $\pm 1.87 \times 10^{-2}$ $(-3.34 \times 10^{-4},$ $-3.23 \times 10^{-4})$	4.81×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-3.56 \times 10^{-4},$ $-3.42 \times 10^{-4})$	5.02×10^{-2} $\pm 1.93 \times 10^{-2}$ $((-4.17 \times 10^{-4},$ $-3.99 \times 10^{-4})$	4.90×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-4.52 \times 10^{-4},$ $-4.36 \times 10^{-4})$	-4.76×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-3.39 \times 10^{-4},$ $-3.26 \times 10^{-4})$	4.76×10^{-2} $\pm 1.88 \times 10^{-2}$ $(-2.47 \times 10^{-4},$ $-2.36e - 4)$
$\gamma = 0.7$ CAE-64	4.65×10^{-2} $\pm 1.87 \times 10^{-2}$ $(-3.68 \times 10^{-4},$ $-3.57 \times 10^{-4})$	4.81×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-3.61 \times 10^{-4},$ $-3.47 \times 10^{-4})$	5.02×10^{-2} $\pm 1.93 \times 10^{-2}$ $(-3.66 \times 10^{-4},$ $-3.48 \times 10^{-4})$	4.88×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-2.96 \times 10^{-4},$ $-2.80 \times 10^{-4})$	-4.76×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-3.58 \times 10^{-4},$ $-3.45 \times 10^{-4})$	-4.77×10^{-2} $\pm 1.88 \times 10^{-2}$ $(-3.26 \times 10^{-4},$ $-3.16 \times 10^{-4})$
$\gamma = 0.0 p = 0$ CAE-64	$\mathbf{4.61 \times 10^{-2}}$ $\mathbf{\pm 1.88 \times 10^{-2}}$	$\mathbf{4.77 \times 10^{-2}}$ $\mathbf{\pm 1.90 \times 10^{-2}}$	$\mathbf{4.98 \times 10^{-2}}$ $\mathbf{\pm 1.94 \times 10^{-2}}$	$\mathbf{-4.86 \times 10^{-2}}$ $\mathbf{\pm 1.91 \times 10^{-2}}$	$\mathbf{4.73 \times 10^{-2}}$ $\mathbf{\pm 1.90 \times 10^{-2}}$	$\mathbf{4.74 \times 10^{-2}}$ $\mathbf{\pm 1.89 \times 10^{-2}}$
$p = 1$ CAE-64	4.66×10^{-2} $\pm 1.88^{\mathbf{-2}}$ $(-4.75 \times 10^{-4},$ $-4.64 \times 10^{-4})$	-4.82×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-5.31 \times 10^{-4},$ $-5.17 \times 10^{-4})$	5.05×10^{-2} $\pm 1.94 \times 10^{-2}$ $(-6.37 \times 10^{-4},$ $-6.18 \times 10^{-4})$	-4.91×10^{-2} $\pm 1.91 \times 10^{-2}$ $(-6.04 \times 10^{-4},$ $-5.88 \times 10^{-4})$	-4.78×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-4.94 \times 10^{-4},$ $-4.81 \times 10^{-4})$	4.79×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-5.02 \times 10^{-4},$ $-4.91 \times 10^{-4})$
$p = 2$ CAE-64	4.70×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-8.62 \times 10^{-4},$ $-8.51 \times 10^{-4})$	-4.87×10^{-2} $\pm 1.92 \times 10^{-2}$ $(-9.43 \times 10^{-4},$ $-9.29 \times 10^{-4})$	-5.09×10^{-2} $\pm 1.96 \times 10^{-2}$ $(-1.10 \times 10^{-3},$ $-1.08 \times 10^{-3})$	-4.96×10^{-2} $\pm 1.93 \times 10^{-2}$ $(-1.06 \times 10^{-3},$ $-1.04 \times 10^{-3})$	-4.82×10^{-2} $\pm 1.92 \times 10^{-2}$ $(-8.95 \times 10^{-4},$ $-8.83 \times 10^{-4})$	4.82×10^{-2} $\pm 1.91 \times 10^{-2}$ $(-8.27 \times 10^{-4},$ $-8.16 \times 10^{-4})$
CCNN	4.63×10^{-2} $\pm 1.87 \times \times 10^{-2}$ $(-2.11 \times 10^{-4},$ $-2.00 \times 10^{-4})$	-4.79×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-1.81 \times 10^{-4},$ $-1.68 \times 10^{-4})$	-5.00×10^{-2} $\pm 1.93 \times 10^{-2}$ $(-1.59 \times 10^{-4},$ $-1.41 \times 10^{-4})$	4.87×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-1.76 \times 10^{-4},$ $-1.61 \times 10^{-4})$	-4.75×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-1.87 \times 10^{-4},$ $-1.74 \times 10^{-4})$	4.75×10^{-2} $\pm 1.88 \times 10^{-2}$ $(-1.23 \times 10^{-4},$ $-1.13 \times 10^{-4})$
Neighboring mean	-2.78×10^{-1} ± 2.18 $(-2.32 \times 10^{-1},$ $-2.32 \times 10^{-1})$	-2.78×10^{-1} $\pm 2.18 \times 10^{-2}$ $(-2.30 \times 10^{-1},$ $-2.30 \times 10^{-1})$	-2.78×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.28 \times 10^{-1},$ $-2.28 \times 10^{-1})$	-2.79×10^{-1} $\pm 2.15 \times 10^{-2}$ $(-2.30 \times 10^{-1},$ $-2.30 \times 10^{-1})$	-2.77×10^{-1} $\pm 2.20 \times 10^{-2}$ $(-2.30 \times 10^{-1},$ $-2.30 \times 10^{-1})$	2.78×10^{-1} $\pm 2.16 \times 10^{-2}$ $(-2.31 \times 10^{-1},$ $-2.31 \times 10^{-1})$

TABLE IX. Statistics for reconstruction of different failure scenarios, where 1 to 3 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column. RMSEs and extended statistics for reconstruction of different failure scenarios across different architectures, where 1–3 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column.

Failed TOFs scenario	1 random	2 random	3 random	2 neighbors	2 opposite	No. 8 and No. 13 position
1TOF model	4.62×10^{-2} $\pm 1.86 \times 10^{-2}$ $(-1.14 \times 10^{-4}, -1.02 \times 10^{-4})$	5.74×10^{-2} $\pm 2.68 \times 10^{-2}$ $(-9.75 \times 10^{-3}, -9.70 \times 10^{-3})$	7.35×10^{-2} $\pm 3.46 \times 10^{-2}$ $(-2.37 \times 10^{-2}, -2.36 \times 10^{-2})$	7.51×10^{-2} $\pm 3.25 \times 10^{-2}$ $(-2.66 \times 10^{-2}, -2.65 \times 10^{-2})$	4.90×10^{-2} $\pm 1.88 \times 10^{-2}$ $(-1.75 \times 10^{-3}, -1.73 \times 10^{-3})$	4.89×10^{-2} $\pm 1.87 \times 10^{-2}$ $(-3.42 \times 10^{-3}, -3.41 \times 10^{-3})$
2TOF model	4.78×10^{-2} $\pm 1.91 \times 10^{-2}$ $(-1.68 \times 10^{-3}, -1.66 \times 10^{-3})$	4.81×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-3.87 \times 10^{-4}, -3.73 \times 10^{-4})$	5.23×10^{-2} $\pm 2.04 \times 10^{-2}$ $(-2.46 \times 10^{-3}, -2.43 \times 10^{-3})$	4.90×10^{-2} $\pm 1.91 \times 10^{-2}$ $(-4.35 \times 10^{-4}, -4.19 \times 10^{-4})$	4.76×10^{-2} $\pm 1.90 \times 10^{-2}$ $(-3.45 \times 10^{-4}, -3.32 \times 10^{-4})$	-4.77×10^{-2} ± 1.88 $(-2.20 \times 10^{-3}, 2.19 \times 10^{-3})$
3TOF model	5.21×10^{-2} $\pm 2.16 \times 10^{-2}$ $(-6.05 \times 10^{-3}, -6.02 \times 10^{-3})$	4.94×10^{-2} $\pm 1.93 \times 10^{-2}$ $(-1.66 \times 10^{-3}, -1.65 \times 10^{-3})$	4.99×10^{-2} $\pm 1.93 \times 10^{-2}$ $(-1.22 \times 10^{-4}, -1.05 \times 10^{-4})$	5.00×10^{-2} $\pm 1.94 \times 10^{-2}$ $(-1.47 \times 10^{-3}, -1.45 \times 10^{-3})$	4.91×10^{-2} $\pm 1.94 \times 10^{-2}$ $(-1.86 \times 10^{-3}, -1.84 \times 10^{-3})$	4.88×10^{-2} $\pm 1.92 \times 10^{-2}$ $(-3.25 \times 10^{-3}, -3.24 \times 10^{-3})$
General model	**4.61×10^{-2} $\pm 1.88 \times 10^{-2}$**	**4.77×10^{-2} $\pm 1.90 \times 10^{-2}$**	**4.98×10^{-2} $\pm 1.94 \times 10^{-2}$**	**4.86×10^{-2} $\pm 1.91 \times 10^{-2}$**	**4.73×10^{-2} $\pm 1.90 \times 10^{-2}$**	4.74×10^{-2} $\pm 1.89 \times 10^{-2}$ $(-1.88 \times 10^{-3}, -1.87 \times 10^{-3})$
Spec model	9.38×10^{-2} $\pm 3.43 \times 10^{-2}$ $(-4.78 \times 10^{-2}, -4.77 \times 10^{-2})$	-1.10×10^{-1} $\pm 3.74 \times 10^{-2}$ $(-6.28 \times 10^{-2}, -6.27 \times 10^{-2})$	-1.24×10^{-1} $\pm 3.74 \times 10^{-2}$ $(-7.42 \times 10^{-2}, -7.41 \times 10^{-2})$	-1.10×10^{-1} $\pm 3.75 \times 10^{-2}$ $(-6.16 \times 10^{-2}, -6.15 \times 10^{-2})$	-1.09×10^{-1} $\pm 4.01 \times 10^{-2}$ $(-6.13 \times 10^{-2}, -6.12 \times 10^{-2})$	**4.55×10^{-2} $\pm 1.82 \times 10^{-2}$**
Neighboring mean	2.78×10^{-1} $\pm 2.18 \times 10^{-1}$ $(-2.32 \times 10^{-1}, -2.32 \times 10^{-1})$	-2.78×10^{-1} $\pm 2.18 \times 10^{-2}$ $(-2.30 \times 10^{-1}, -2.30 \times 10^{-1})$	-2.78×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.28 \times 10^{-1}, -2.28 \times 10^{-1})$	-2.79×10^{-1} $\pm 2.15 \times 10^{-2}$ $(-2.30 \times 10^{-1}, -2.30e - 1)$	2.77×10^{-1} $\pm 2.20 \times 10^{-2}$ $(-2.30 \times 10^{-1}, -2.30e - 1)$	2.78×10^{-1} $\pm 2.16 \times 10^{-2}$ $(-2.33 \times 10^{-1}, -2.33e - 1)$

TABLE X. Statistics for reconstruction of different failure scenarios, where up to 4 or 5 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column. RMSEs and extended statistics for reconstruction of different failure scenarios across different architectures, where 1–3 TOF detectors are disabled. Bold numbers indicate the lowest RMSE in each column.

Failed TOFs scenario	4 random	5 random	1–4 random	1–5 random
CAE-64	5.26×10^{-2} $\pm 2.01 \times 10^{-2}$ $(-3.71 \times 10^{-4}, -3.48 \times 10^{-4})$	5.64×10^{-2} $\pm 2.13 \times 10^{-2}$ $(-1.47 \times 10^{-3}, -1.44 \times 10^{-3})$	**4.96×10^{-2} $\pm 1.98 \times 10^{-2}$**	5.14×10^{-2} $\pm 2.07 \times 10^{-2}$ $(-3.06 \times 10^{-4}, -2.79 \times 10^{-4})$
4TOF	5.24×10^{-2} $\pm 1.99 \times 10^{-2}$ $(-1.69 \times 10^{-4}, -1.48 \times 10^{-4})$	5.55×10^{-2} $\pm 2.06 \times 10^{-2}$ $(-5.10 \times 10^{-4}, -4.84 \times 10^{-4})$	4.97×10^{-2} $\pm 1.96 \times 10^{-2}$ $(-1.53 \times 10^{-4}, -1.32 \times 10^{-4})$	5.11×10^{-2} $\pm 2.02 \times 10^{-2}$ $(-8.37 \times 10^{-5}, -5.94 \times 10^{-5})$
5TOF	**5.23×10^{-2} $\pm 1.98 \times 10^{-2}$**	**5.50×10^{-2} $\pm 2.04 \times 10^{-2}$**	4.98×10^{-2} $\pm 1.95 \times 10^{-2}$ $(-2.32 \times 10^{-4}, -2.11 \times 10^{-4})$	**5.11×10^{-2} $\pm 2.00 \times 10^{-2}$**
Neighboring mean	2.78×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.26 \times 10^{-1}, -2.26 \times 10^{-1})$	2.79×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.24 \times 10^{-1}, -2.24 \times 10^{-1})$	2.78×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.29 \times 10^{-1}, -2.29 \times 10^{-1})$	2.79×10^{-1} $\pm 2.17 \times 10^{-2}$ $(-2.28 \times 10^{-1}, -2.28 \times 10^{-1})$

APPENDIX B: CHECKING FOR SATURATION OF TOF DETECTORS

In this section, we compare the electron intensities measured by all TOF detectors with the outputs from the gas monitor detector. The gas monitor detector measures the calibrated ionization count, reflecting the overall ionization produced by the photons. As shown in Fig. 8, the blue data points in the plot remain linearly increasing in the higher regions of the gas monitor detector, closely following the linear approximation (orange line). This behavior suggests that the TOF detectors do not show saturation. For the calculation of the linear approximation, data points near $(0, 0)$ were excluded, as these correspond to shots recorded without lasing. For this scenario, both the gas monitor detector and the TOF detectors are uncalibrated and return values close to $(0, 0)$, making them unsuitable for reliable analysis. We corrected the baseline of the plot by shifting the linear fit to pass through the origin. Consequently, the data points were also adjusted to reflect this baseline correction.

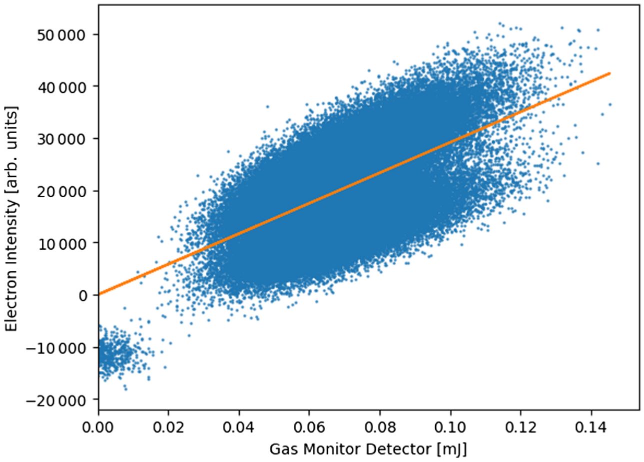

FIG. 8. Electron intensities vs gas monitor detector values. The blue dots represent the measured data points, while the orange line is the linear approximation of these points. The plot is baseline corrected.

APPENDIX C: TOF DETECTOR DISABLEMENT ALGORITHM

ALGORITHM 1. Algorithm for selecting random TOF detectors to disable, ensures to show samples with neighboring and opposite failed TOF detectors more often, as these constellations are harder to reconstruct.

```
procedure RandomTOFs($n_{\max}, \xi$) ▷$n_{\max}$ is the maximum amount of TOFs to disable, $\xi$ is the neighbor or opposite probability
    $l_{\mathrm{TOF}} \leftarrow \pi([1, \cdots, n_{\mathrm{TOF}}])$▷$n_{\mathrm{TOF}}$ is the total amount of TOFs, $\pi$ is a random permutation
    $l_{\mathrm{disabled}} \leftarrow [l_{\mathrm{TOF}}[0]]$
    $n_{\mathrm{disabled}} \sim \mathcal{U}(0, n_{\max})$
    for $i \in \{0, \cdots, n_{\mathrm{disabled}}\}$ do
        $r \sim \mathcal{U}(0, 1)$
        if $r < \xi$ then
            if $r < (\xi/2)$ then ▷ Look for neighbor
                if for one element in $l_{\mathrm{disabled}}$ the left or right neighbor $\eta$ is in $l_{\mathrm{TOF}}$ then
                    AddItem($l_{\mathrm{disabled}}$, $\eta$)
                    RemoveItem($l_{\mathrm{TOF}}$, $\eta$)
                else ▷ If no neighbor, add a random element
                    AddItem($l_{\mathrm{disabled}}$, $l_{\mathrm{TOF}}[0]$)
                end if
            else ▷ Look for opposite
                if for one element in $l_{\mathrm{disabled}}$ the opposite $\omega$ is in $l_{\mathrm{TOF}}$ then
                    AddItem($l_{\mathrm{disabled}}$, $\omega$)
                    RemoveItem($l_{\mathrm{TOF}}$, $\omega$)
                else ▷ If no opposite, add a random element
                    AddItem($l_{\mathrm{disabled}}$, $l_{\mathrm{TOF}}[0]$)
                end if
            end if
        end if
    end for
    return $l_{\mathrm{disabled}}$
end procedure
```

APPENDIX D: IMPACT OF REGULARIZATION

As shown in Fig. 9, the AdamW-trained network reconstructs the broader intensity area more accurately (denoted with 1). In the region with minimal intensity in the label, the Adam-trained network reconstructs more incorrect intensity than the AdamW-trained network (2).

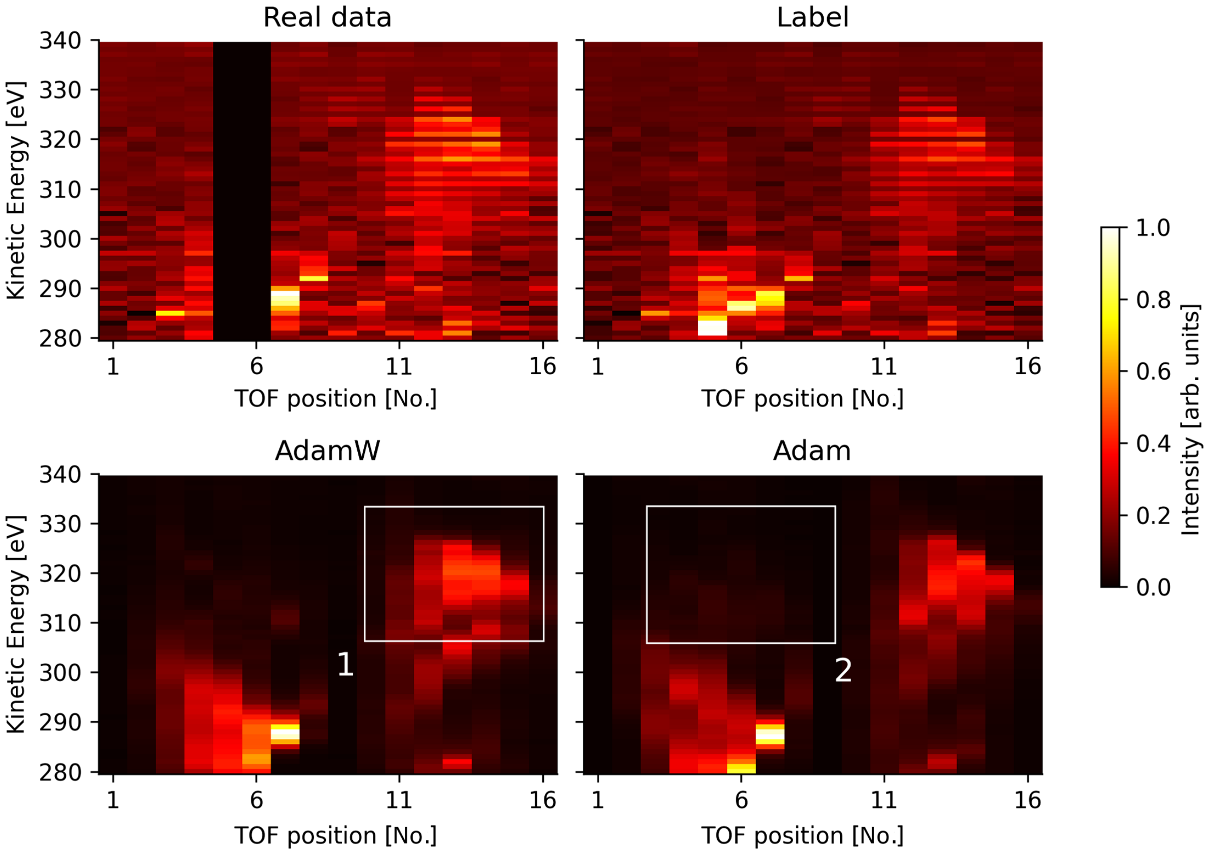

FIG. 9. Visual comparison of reconstruction performance between a network trained with AdamW (with regularization) and Adam (without regularization).

APPENDIX E: PHASE-SEPARATED EVALUATION

In this section, we analyze the reconstruction quality across different phases. To achieve this, we generate 10 000 spectrograms spanning 80 phase steps, each with a single intensity peak at the corresponding phase and a random photon energy. Noise and other parameters are set accordingly to Sec. III. These spectrograms are then transformed into detector images for evaluation.

Using the general model, we assess the detector images by systematically deactivating one TOF detector at a time and calculating the RMSE of the resulting reconstructions. The results are presented in the left plot of Fig. 10, which shows that certain phases consistently exhibit lower reconstruction errors. For instance, phases around step 40 demonstrate higher reconstruction accuracy. This improvement arises because the maxima of the corresponding partial waves align with the maxima of the angular distribution, as illustrated in Fig. 6.

Contrary, specific combinations of phases and TOF detectors result in increased reconstruction errors, especially phase steps 30 or 70, combined with a failure of TOF positions No. 5 or No. 13. These combinations lead to a significant loss of information regarding the relevant partial waves, thus degrading the reconstruction accuracy. For reference, we plotted the square root of the average intensities of all 10 000 detector images across all phases and TOF positions on the right side of Fig. 10.

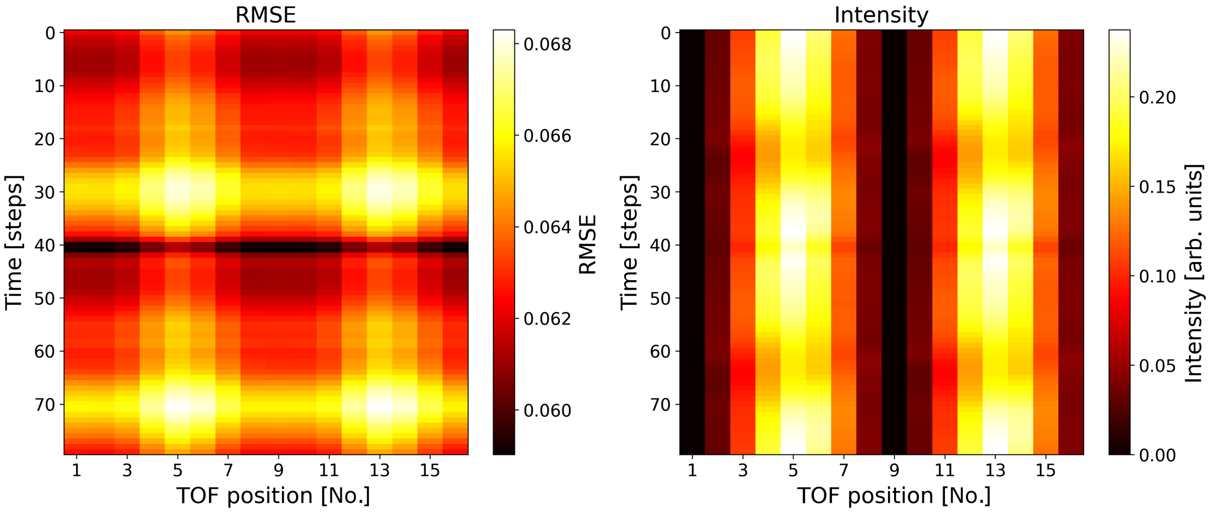

FIG. 10. Phase-resolved analysis of reconstruction quality. The left plot depicts the RMSE of the reconstruction for each phase step when a single TOF detector is disabled. The right plot shows the square root of the average intensities of 10 000 detector images across all phases and TOF positions.

APPENDIX F: WIENER FILTER

The Wiener filter in our used implementation by Virtanen *et al.* works by calculating the following steps [26,28]:

(i) Local mean $\mu_{i,j}$

$$\mu_{i,j} = \frac{1}{|N|} \sum_{(m,n)\in N} \mathbf{I}_{i+m,\,j+n},$$

where $\mathbf{I}_{i,j}$ denotes the intensity value at pixel location (i,j) in the input image $\mathbf{I}$, and N is a local neighborhood window centered at (i,j), defined as a set of relative coordinate offsets from the center. For instance, in the case of a 3×3 window: $N = \{(-1,-1),(-1,0),\cdots,(1,1)\}$. The number of pixels in the window is denoted with $|N|$. If the window extends beyond the image boundaries, these indices are excluded from the window set.

(ii) Local variance $\boldsymbol{\sigma}^2_{i,j}$

$$\boldsymbol{\sigma}^2_{i,j} = \left(\frac{1}{|N|} \sum_{(m,n)\in N} \mathbf{I}^2_{i+m,\,j+n}\right) - \mu^2_{i,j}.$$

This represents the variance of pixel intensities within the local window N centered at (i,j).

(iii) Noise power p_{noise}

$$p_{\text{noise}} = \frac{1}{|\Omega|} \sum_{(i,j)\in\Omega} \boldsymbol{\sigma}^2_{i,j},$$

where Ω is the set of all valid pixel coordinates in the image $\mathbf{I}$, and $\boldsymbol{\sigma}^2_{i,j}$ is the local variance at each pixel position (i,j).

(iv) Filtered output $\mathbf{O}_{i,j}$ The Wiener-filtered output at pixel (i,j) is computed as

$$\mathbf{O}_{i,j} = \begin{cases} \mu_{i,j}, & \text{if } \boldsymbol{\sigma}^2_{i,j} < p_{\text{noise}}, \\ \mu_{i,j} + \left(1 - \frac{p_{\text{noise}}}{\boldsymbol{\sigma}^2_{i,j}}\right)\cdot(\mathbf{I}_{i,j} - \mu_{i,j}), & \text{otherwise.} \end{cases}$$

[1] W. Decking *et al.*, A MHz-repetition-rate hard x-ray free-electron laser driven by a superconducting linear accelerator, Nat. Photonics **14**, 391 (2020).

[2] C. Bostedt, S. Boutet, D. M. Fritz, Z. Huang, H. J. Lee, H. T. Lemke, A. Robert, W. F. Schlotter, J. J. Turner, and G. J. Williams, Linac coherent light source: The first five years, Rev. Mod. Phys. **88**, 015007 (2016).

[3] N. Hartmann *et al.*, Attosecond time–energy structure of x-ray free-electron laser pulses, Nat. Photonics **12**, 215 (2018).

[4] U. S. Sainadh, H. Xu, X. Wang, A. Atia-Tul-Noor, W. C. Wallace, N. Douguet, A. Bray, I. Ivanov, K. Bartschat, A. Kheifets, R. T. Sang, and I. V. Litvinyuk, Attosecond angular streaking and tunnelling time in atomic hydrogen, Nature (London) **568**, 75 (2019).

[5] K. Dingel, T. Otto, L. Marder, L. Funke, A. Held, S. Savio, A. Hans, G. Hartmann, D. Meier, J. Viefhaus, B. Sick, A. Ehresmann, M. Ilchen, and W. Helml, Artificial intelligence for online characterization of ultrashort x-ray free-electron laser pulses, Sci. Rep. **12**, 17809 (2022).,

[6] F. Bornemann and T. März, Fast image inpainting based on coherence transport, J. Math. Imaging Vision **28**, 259 (2007).

[7] H. Hukkelås, F. Lindseth, and R. Mester, Image inpainting with learnable feature imputation, in *Pattern Recognition*, edited by Z. Akata, A. Geiger, and T. Sattler (Springer International Publishing, Cham, 2021), pp. 388–403, 10.1007/978-3-030-71278-5_280.

[8] R. Köhler, C. Schuler, B. Schölkopf, and S. Harmeling, Mask-specific inpainting with deep neural networks, in *Pattern Recognition*, edited by X. Jiang, J. Hornegger, and R. Koch (Springer International Publishing, Cham, 2014), pp. 523–534, 10.1007/978-3-319-11752-2_43.

[9] J. Xie, L. Xu, and E. Chen, Image denoising and inpainting with deep neural networks, in *Proceedings of the 25th International Conference on Neural Information Processing Systems* (Curran Associates Inc., Red Hook, NY, 2012), Vol. 1, pp. 341–349.

[10] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, Image inpainting for irregular holes using partial convolutions, in *Computer Vision–ECCV 2018*, edited by V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss (Springer International Publishing, Cham, 2018), pp. 89–105, 10.1007/978-3-030-01252-6_6.

[11] C. Xie, S. Liu, C. Li, M.-M. Cheng, W. Zuo, X. Liu, S. Wen, and E. Ding, Image inpainting with learnable bidirectional attention maps, in *Proceedings of the 2019 IEEE / CVF International Conference on Computer Vision (ICCV), Seoul, Korea* (IEEE, New York, 2019), pp. 8857–8866, 10.1109/ICCV.2019.00895.

[12] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. Huang, Free-form image inpainting with gated convolution, in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea* (IEEE, New York, 2019), pp. 4470–4479, 10.1109/ICCV.2019.00457.

[13] Y. Kossale, M. Airaj, and A. Darouichi, Mode collapse in generative adversarial networks: An overview, in *Proceedings of the 8th International Conference on Optimization and Applications (ICOA), Genoa, Italy* (IEEE, New York, 2022), pp. 1–6, 10.1109/ICOA55659.2022.9934291.

[14] M. Arjovsky, S. Chintala, and L. Bottou, Wasserstein generative adversarial networks, in *Proceedings of the 34th International Conference on Machine Learning Research*, edited by D. Precup and Y. W. Teh (PMLR, Sydney, Australia, 2017), Vol. 70, pp. 214–223.

[15] M. H. Givkashi, M. Hadipour, A. PariZanganeh, Z. Nabizadeh, N. Karimi, and S. Samavi, Image inpainting using autoencoder and guided selection of predicted pixels, in *Proceedings of the 30th International Conference on Electrical Engineering (ICEE), Tehran, Iran* (IEEE, New York, 2022), pp. 700–704, 10.1109/ICEE55646.2022.9827427.

[16] S. Schubert, P. Neubert, J. Pöschmann, and P. Protzel, Circular convolutional neural networks for panoramic images and laser data, in *Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV)* (2019), p. 653, 10.1109/IVS.2019.8813862

[17] O. Ronneberger, P. Fischer, and T. Brox, U-Net: Convolutional networks for biomedical image segmentation, Lect Notes Comput. Sci. **9351**, 234 (2015).

[18] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, Repaint: Inpainting using denoising diffusion probabilistic models, in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Antalya, Turkey* (IEEE, New York, 2022), pp. 11451–11461, 10.1109/CVPR52688.2022.01117.

[19] A. Scheinker and R. Pokharel, Enabling dynamic 3D coherent diffraction imaging via adaptive latent space tuning of generative autoencoders, npj Comput. Mater. **10**, 283 (2024).

[20] A. Scheinker and A. Williams, Adaptive conditional latent diffusion maps beam loss to 2D phase space projections, arXiv:2502.18684.

[21] https://github.com/hz-b/tof-reconstruction/blob/main/data_generation.py

[22] C. N. Yang, On the angular distribution in nuclear reactions and coincidence measurements, Phys. Rev. **74**, 764 (1948).

[23] M. I. Hosen and M. B. Islam, Masked face inpainting through residual attention UNet, in *Proceedings of the Innovations in Intelligent Systems and Applications Conference (ASYU), Antalya, Turkey* (IEEE, New York, 2022), pp. 1–5, 10.1109/ASYU56188.2022.9925541.

[24] D. Misra, Mish: A self regularized nonmonotonic activation function, arXiv:1908.08681.

[25] I. Loshchilov and F. Hutter, Decoupled weight decay regularization, arXiv:1711.05101.

[26] J. Lim, *Two-Dimensional Signal and Image Processing* (Prentice Hall, Englewood Cliffs, NJ, 1990).

[27] https://github.com/hz-b/tof-reconstruction

[28] P. Virtanen *et al.*, SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nat. Methods **17**, 261 (2020).