X-ray-induced atomic transitions via machine learning: A computational investigation

Laura Budewig,[1, 2] Sang-Kil Son,[1] Zoltan Jurek,[1] Malik Muhammad Abdullah,[1] Marina Tropmann-Frick,[3] and Robin Santra[1, 2]

[1]*Center for Free-Electron Laser Science CFEL, Deutsches Elektronen-Synchrotron DESY, Notkestr. 85, 22607 Hamburg, Germany*
[2]*Department of Physics, Universität Hamburg, Notkestr. 9-11, 22607 Hamburg, Germany*
[3]*Department of Computer Science, University of Applied Sciences Hamburg, Berliner Tor 7, 20099 Hamburg, Germany*

(Dated: February 10, 2024)

Intense x-ray free-electron laser pulses can induce multiple sequences of one-photon ionization and accompanying decay processes in atoms, producing highly-charged atomic ions. Considering individual quantum states during these processes provides more precise information about the x-ray multiphoton ionization dynamics than the common configuration-based approach. However, in such a state-resolved approach, extremely huge-sized rate-equation calculations are inevitable. Here we present a strategy that embeds machine learning models into a framework for atomic state-resolved ionization dynamics calculations. Machine learning is employed for the required atomic transition parameters, whose calculations possess the computationally most expensive steps. We find for argon that both feedforward neural networks and random forest regressors can predict these parameters with acceptable, but limited accuracy. State-resolved ionization dynamics of argon, in terms of charge-state distributions and electron and photon spectra, are also presented. Comparing fully-calculated and machine-learning-based results, we demonstrate that the proposed machine-learning strategy works in principle and that the performance, in terms of charge-state distributions and electron and photon spectra, is good. Our work establishes a first step toward accelerating the calculation of atomic state-resolved ionization dynamics induced by high-intensity x rays.

I. INTRODUCTION

The enormous peak brightness of x-ray free-electron lasers (XFELs) [1–5], such as the European XFEL [6], offers exciting new opportunities for the structure determination of biomolecules with almost atomic resolution [7–13]. However, due to the high-intensity x rays the electronic structure of the investigated sample is unavoidably damaged [14–17]. As a consequence, the sample undergoes structural disintegration [18], which limits such applications.

A critical process, underlying these damages, is x-ray multiphoton ionization dynamics in atoms and molecules [19]. High-intensity x rays induce multiple sequences of one-photon ionization accompanied by Auger-Meitner decay or x-ray fluorescence. As a result, atoms or molecules often become very highly ionized during the interaction with intense XFEL pulses [20–24]. A validated approach to simulate the x-ray multiphoton ionization dynamics is by solving a coupled set of rate equations [20, 25, 26] describing the time-dependent populations of the electronic configurations visited during the ionization dynamics. Such a configuration-based rate-equation approach has been widely used and successfully applied for interpreting and designing many XFEL experiments [20–43].

However, the configuration-based approach does not include individual quantum states and individual state-to-state transitions and, thereby, cannot capture state-resolved ionization dynamics. A state-resolved approach delivers more detailed information about the x-ray multiphoton ionization dynamics, especially regarding resonant excitations and spectra. This has recently been demonstrated for neon atoms [44]. To explore state-resolved ionization dynamics based on time-dependent quantum state populations, it is necessary to include all possible electronic quantum states that may be formed by removing zero, one, or more electrons from the ground state of the neutral atom. The corresponding number of involved states is dramatically larger than the number of electronic configurations. For example, even for an isolated argon atom without considering resonant or relativistic effects the number of involved states is 262144 [44], whereas only 1323 electronic configurations are involved [34]. Thus, apart from very light atoms like neon, solving rate equations in an extremely large space of states is unavoidable. Therefore, the huge-sized rate equation calculations are performed via a more efficient Monte Carlo on-the-fly rate-equation method [21, 38]. However, even with such a Monte Carlo method, the computational effort for the state-resolved ionization dynamics calculations is inevitably large (as will be demonstrated in Sec. III C). The main bottleneck in the state-resolved ionization dynamics calculations is the first-principle calculation of all required atomic transition parameters, i.e., transition energies, cross sections and rates, which has to be performed on the fly.

With machine learning nowadays being a thriving and actively investigated field, it is natural to ask whether this critical challenge of high computational effort might be addressed by applying a suitable machine learning stategy. Machine learning, deep learning included, has already successfully supported natural science in various ways [45–47]. A prototype example is the application to protein structure predictions with atomic accuracy [48]. Other important applications of machine learning include the prediction of x-ray absorption spectra [49–52], the identification of phase transitions in condensed matter [53], the characterization and calibration of laser pulses [54–57], as well as its use in electronic-structure theory [58–63], just to name a few (for more see, e.g., Refs. [45, 46] and references therein). One high-impact role that machine learning has been playing in electronic-structure theory so far is in speeding up the construction of potential energy surfaces [64–67]. A recent review about the progress of machine learning in the context of potential energy surfaces can be found in Ref. [68]. Furthermore, another interesting approach is to reduce the high computational effort in configuration interaction calculations by preselecting only the most important configurations via machine learning models [69, 70].

In this work, we present a strategy that embeds machine learning models for predicting atomic transition parameters into a Monte Carlo on-the-fly rate-equation method for describing atomic state-resolved ionization dynamics. Recently, a state-resolved Monte Carlo implementation [44], based on a framework for performing quantum-state-resolved first-principle calculations of atomic transition parameters [71], was introduced in the *ab initio* electronic-structure and ionization dynamics toolkit XATOM [16, 72, 73]. We here combine the state-resolved Monte Carlo implementation with machine learning models for predicting atomic transition parameters. This machine-learning-based implementation can reproduce the results for neon published in Ref. [44]. However, for neon, the computational effort is too small to gain much with it. Therefore, here, we choose to focus on the much more challenging problem of state-resolved ionization dynamics of argon. To the best of the authors' knowledge, they have not been investigated before. Our work establishes a first step towards accelerating huge-sized rate-equation calculations for easily examining x-ray-induced ionization dynamics for a variety of atoms and x-ray parameters.

The paper is organized as follows. In Sec. II, a description of the methods used to obtain the results presented in Sec. III can be found. In Sec. III A, we demonstrate how to collect data of x-ray-induced atomic transitions via a Monte Carlo approach, before analyzing the performance of the machine learning models (i.e., neural networks and random forest regressors) in Sec. III B. The performance, in terms of charge-state distributions (CSDs) and spectra, is the topic of Sec. III C. The paper finishes with a conclusion and future perspectives in Sec. IV.

II. THEORETICAL DETAILS

A. State-resolved Monte Carlo calculations

We perform state-resolved x-ray multiphoton ionization dynamics calculations using the state-resolved Monte Carlo implementation [44] in the XATOM toolkit [16, 72, 73]. This implementation is based on a nonrelativistic quantum-state-resolved electronic-structure framework [71], also embedded in XATOM. It performs first-principle calculations of atomic first-order-corrected transition energies as well as state-to-state cross sections and transition rates.

To provide an overview of the accuracy of the quantum-state-resolved electronic-structure framework employed, we list selected K and L fluorescence and KLL Auger-Meitner transition energies for Ar^{1+} and Ar^{2+} (hypersatellites) in Table I. Transition energies are calculated with two different orbital optimization strategies within this framework: based on first-order-corrected energies calculated with orbitals and orbital energies optimized (i) for the initial electronic configuration only and (ii) for the initial and final electronic configurations individually. Both strategies are compared to relativistic calculations [74–77] and experimental measurements [76–79], which are in almost perfect agreement with one another (see Table I). Relativistic, quantum-electrodynamic, and finite-nuclear-size effects [77] are not included in the quantum-state-resolved electronic-structure framework employed. As a consequence, transition energies calculated with both strategies are less accurate and exhibit no spin-orbit splitting. Nonetheless, the selected transitions in Table I demonstrate an accuracy of more than 90% (initial optimization) or 97% (individual optimization), respectively. The individual optimization delivers more accurate results, however, at the expense of computational efficiency. Therefore, in what follows, we optimize for the initial electronic configuration only (as done in Refs. [44, 71]). The main focus in this work is on the usage of machine learning for atomic transition parameters, whose accuracy does not affect the general machine-learning approach.

Further, a Monte Carlo on-the-fly rate-equation method [21, 38] is employed for describing the time evolution of the atomic quantum state populations. The number of coupled rate equations (= the number of individual states involved in the x-ray multiphoton ionization dynamics) is extremely large. For instance, for argon atoms, this number is 2^{18} when all subshells are accessible for one-photon ionization, but relativistic and resonant effects are not included [44]. Therefore, the Monte Carlo method is critical since it permits us to efficiently perform huge-sized rate-equation calculations by stochastically sampling possible ionization pathways.

In the present work, we restrict the time propagation of the x-ray multiphoton ionization dynamics to a time interval of 1 ps. This implies that decay processes that occur on time scales much longer than 1 ps are assumed not to be of interest. Therefore, the rates are set to zero when they are smaller than $\Gamma_{\mathrm{thres}} = 10^{-7}$ a.u. (corresponding to the time scale of 240 ps). In what follows, these comparatively slow processes are referred to as quasi-forbidden transitions. This is reasonable since processes occurring later are actually modified by plasma

TABLE I. Accuracy of the underlying quantum-state-resolved electronic-structure framework in XATOM. Selected fluorescence and Auger-Meitner (hypersatellite) transition energies for argon are calculated with this framework [71] based on orbitals and orbital energies optimized (i) for the initial electronic configuration only and (ii) for the initial and final electronic configurations individually. Results are compared to (iii) more accurate relativistic calculations and (iv) experimental data (references are given next to values in the table), which contain energy splittings due to spin-orbit coupling.

	transition energy $E_{I^i \to I^f}$ (eV)				difference (%)	
process	(i) XATOM	(ii) XATOM–ind.	(iii) relativistic	(iv) experiment	(iv)-(i)	(iv)-(ii)
Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ (^2S) $\to 1s^2 2s^2 2p^5 3s^2 3p^6$ (^2P)	2931.5	2946.4	2957.9 [77] 2955.9 [77]	2957.7 [77] 2955.6 [77]	0.9 0.8	0.4 0.3
Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ (^2S) $\to 1s^2 2s^2 2p^6 3s^2 3p^5$ (^2P)	3140.3	3180.4	3191.5 [77] 3191.3 [77]	3190.5 [77] 3190.5 [77]	1.6 1.6	0.3 0.3
Ar^{1+}, $1s^2 2s^2 2p^5 3s^2 3p^6$ (^2P) $\to 1s^2 2s^2 2p^6 3s^1 3p^6$ (^2S)	202.4	215.3	219.5 [77] 221.5 [77]	220.2 [77] 221.8 [77]	8.8 9.6	2.3 3.0
Ar^{2+}, $1s^0 2s^2 2p^6 3s^2 3p^6$ (^2S) $\to 1s^1 2s^2 2p^5 3s^2 3p^6$ (^1P)	3105.9	3118.0	3131.5 [74]	3133.0 [78]	0.9	0.5
Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ (^2S) $\to 1s^2 2s^2 2p^4 3s^2 3p^6$ (^1D)	2646.9	2650.6	2661.8 [75]	2660.6 [79]	0.5	0.4
Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ (^2S) $\to 1s^2 2s^2 2p^4 3s^2 3p^6$ (^1S)	2634.8	2638.6	2649.9 [75]	2650.6 [79]	0.6	0.5
Ar^{2+}, $1s^0 2s^2 2p^6 3s^2 3p^6$ (^1S) $\to 1s^1 2s^2 2p^4 3s^2 3p^6$ (^2D)	2765.4	2766.4	2779.2 [76]	2779.6 [76]	0.5	0.5
Ar^{2+}, $1s^0 2s^2 2p^6 3s^2 3p^6$ (^1S) $\to 1s^1 2s^2 2p^4 3s^2 3p^6$ (^2S)	2752.5	2762.6	2769.6 [76]	2768.9 [76]	0.6	0.2

processes. Thus, the model of an isolated atom assumed in our calculations breaks down after some time. Note that at this point, the few-femtosecond ionizing pulse is long over.

Let us briefly explain what we mean by an individual quantum state and an individual state-to-state transition in the following. In our state-resolved approach, a state I is defined by the electronic configuration, $1s^{N_{1s}} 2s^{N_{2s}} 2p^{N_{2p}} \cdots$, together with additional quantum numbers (L, S, M_L, κ) (that specify a so-called zeroth-order LS eigenstate [71]). For each state, there is a corresponding (term-specific) first-order-corrected energy $E_{LS\kappa}$ that is the same for all states within a term $^{2S+1}L(\kappa)$. Note that the spin projection quantum number M_S is missing in the description of an individual quantum state. Since in our approach none of the interaction Hamiltonians couples to the spin [44, 71], transition probabilities are independent of M_S and, thus, we are not interested in spin projection quantum numbers. From now on the index i refers to the initial state (before a certain process is happening), while for the final target state we shall use the index f. Then, an individual state-to-state transition is a transition from an individual quantum state I^i to I^f with first-order-corrected transition energy $E^{(1)}_{I^i \to I^f}$, cross section $\sigma_{I^i \to I^f}$ for photoabsorption or transition rate $\Gamma_{I^i \to I^f}$ for Auger-Meitner decay or fluorescence, respectively. (The corresponding equations are given in Ref. [71].)

B. The machine learning models

a. The tasks of the machine learning models are to predict for a given transition from an individual quantum state I^i to I^f the first-order-corrected transition energy $E^{(1)}_{I^i \to I^f}$ (E model) and the cross section $\sigma_{I^i \to I^f}$ for photoabsorption (P model) or the transition rate $\Gamma_{I^i \to I^f}$ for Auger-Meitner decay (AM model) and fluorescence (F model). These four tasks are solved by separate machine learning models. All are regression problems [80] with the following inputs, i.e., features, and outputs, i.e., labels.

b. The features are given by the following quantities based on our definition of an individual quantum state in Sec. II A:

- occupation numbers n^i_{occ} of the initial electronic configuration, i.e., $(N^i_{1s}, N^i_{2s}, N^i_{2p}, \cdots)$,
- quantum numbers qn_i of the initial state, i.e., $(L_i, S_i, M_{L_i}, \kappa_i)$,
- type of process p being considered, i.e., (involved hole, first involved electron, second involved electron {for AM, otherwise zero}, kind of process {1:P, 2:AM, 3:F}),
- quantum numbers qn_f of the final state, i.e., $(L_f, S_f, M_{L_f}, \kappa_f)$.

The first case gives N_{orb} features, where N_{orb} is the number of subshells involved in the initial electronic configuration, and the rest provides 4 features per each case.

Thus, the total number of features N_{features} in a feature vector is given by

$$N_{\text{features}} = N_{\text{orb}} + 12. \tag{1}$$

Note that the type of process being considered can be interchanged with the occupation numbers n^f_{occ} of the final electronic configuration. Here, the former is employed in the feature vector to reduce the number of features, which is especially important for heavy atoms and/or the inclusion of resonant effects.

It is also worthwhile to mention that (as can be seen from Ref. [71]) cross sections and decay rates are invariant under a change of the angular momentum projection quantum numbers $M_{L_i} \to -M_{L_i}$ and simultaneously $M_{L_f} \to -M_{L_f}$—a relation the machine learning models fail to learn. Therefore, we force the machine learning models to preserve this symmetry by using $M_{L_i} \geq 0$ as a feature only. More precisely,

$$\text{if } M_{L_i} \begin{cases} > 0 : M_{L_i} \text{ and } M_{L_f} \\ = 0 : M_{L_i} = 0 \text{ and } |M_{L_f}| \\ < 0 : |M_{L_i}| \text{ and } -M_{L_f} \end{cases} \tag{2}$$

are taken as features for the projection quantum numbers.

Another important point is that the individual models are not given all features as inputs, only those that are relevant. Transition energies are independent of M_{L_i} and M_{L_f}—another relation the machine learning model fails to learn. Hence, for the energy model, they are not used and, therefore, $N^{\text{E}}_{\text{features}} = N_{\text{features}} - 2$. Similarly, for the other three models the kind of process (P, AM, or F) is an unnecessary feature as it is fixed a priori by the model used. Additionally, features for involved holes and electrons are only important if they exist for the process in question. Consequently, we have $N^{\text{P}}_{\text{features}} = N_{\text{features}} - 3$, $N^{\text{AM}}_{\text{features}} = N_{\text{features}} - 1$, and $N^{\text{F}}_{\text{features}} = N_{\text{features}} - 2$.

c. The label is always only one number since each task has its own machine learning model:

$$y^{\text{pred}} = \begin{cases} E^{(1)}_{I^i \to I^f} \text{ in eV for E model,} \\ \sigma_{I^i \to I^f} \text{ in a.u. for P model,} \\ \Gamma_{I^i \to I^f} \text{ in a.u. for AM model,} \\ \Gamma_{I^i \to I^f} \text{ in a.u. for F model.} \end{cases} \tag{3}$$

d. Data preparation depends on the type of machine learning model used.

For the neural networks, the widely recommended Z-score normalization is applied to each feature x_k [81], i.e.,

$$x'_k = \frac{x_k - \mu_k}{\sigma_k}, \tag{4}$$

with mean μ_k and standard deviation σ_k of the kth feature with respect to all training data. Consequently, the prepared input data form a distribution with zero mean and unit standard deviation. For the random forest regressors, however, no feature normalization is required since it is not distance-based.

The energy values cover a wide range from 0 eV to the energy of the incoming x rays, typically a few thousand eV. For the neural network, they are also normalized by Z-score normalization [see Eq. (4)] in order to keep their values in a smaller range. For the random forest regressor, the pure energy values are used.

Cross sections and rates cover several orders of magnitude. Thus, logarithmic scaling might be useful and is, in fact, applied for the random forest regressors. For the neural networks, however, we prefer to use pure cross section and rate values by, instead, respecting the wide range in the loss function and the output activation (see Sec. II C) without any scaling. The advantage is that this avoids back-scaling, which in combination with Z-score normalization is prone to errors, especially for very small cross sections and rates.

C. Neural network

Neural networks [80, 82, 83] are the central tool of deep learning. In general, they consist of a sequence of several sets of linear transformations followed by nonlinear activations. Each step in this sequence is called a layer [59]. The number of layers in the neural network determines its depth. Based on a chosen loss function and an optimizer, they are trained via back-propagation [84]. The basic idea of a (deep) neural network is that each layer is effectively learning a more complex representation of the raw input features and that this reduces the number of parameters needed to be fitted [85].

a. Hyperparameter tuning. Before explaining the neural network architecture and the hyperparameters employed in the present work, let us briefly explain the general way we have made these decisions. This will make some of our reasoning in the following two paragraphs clearer. The neural network architecture and hyperparameters are determined by 'trial and error'. Due to high training efforts and fluctuations in loss values from training to training, a more systematic and automated hyperparameter optimization, e.g., by a grid search [80], would not be well suited for our purpose. It is especially worthwhile to note that our models are system-specific. As will be explained in Sec. II E, for each machine-learning-based Monte Carlo calculation, the models need to be retrained. If we also had to reoptimize the hyperparameters for (almost) every machine-learning-based Monte Carlo calculation, this would be in clear contrast to our goal of speeding up the calculations. Moreover, for speeding up calculations, models should also be chosen such that training is not more time consuming than really necessary. Consequently, our aim is not to find the perfect hyperparameters and the best possible model performance for a given training set. Note that differences between different models are often anyhow only very small. Instead,

TABLE II. Neural network architecture: number of units per layer for each neural network trained in this work (see Sec. II C).

model	in	1	2	3	4	5	out
E	15	512	256	128	64	32	1
P	14	1024	512	256	128	64	1
AM	16	1024	512	256	128	64	1
F	15	1024	512	256	128	64	1

it is more critical to build models, hyperparameters included, that allow us to train them for different training sets and training set sizes with a good (but not necessarily perfect) performance within an acceptable amount of time.

b. Network architecture. For our neural networks, we employ the popular deep learning library Keras [86] of the TensorFlow machine learning platform [87]. Our neural networks are standard feedforward neural networks with seven layers, input and output layer included, and with the number of units given in Table II. This neural network architecture has sufficient model capacity to approximately fit the data, but not enough to nearly perfectly interpolate the training data (see Sec. III B). For argon, interpolating training data as suggested by the 'modern' interpolation hypothesis [88] is a tough task because it actually requires very large neural networks and, concomitantly, very long training times (see Sec. III B). Therefore, we have chosen this network architecture as a compromise between computational effort and numerical accuracy. Of course, other neural network architectures with comparable capacity would work similarly well.

For the activation function, the hyperbolic tangent (tanh) is chosen. Indeed, nowadays, in the deep learning community rectified linear units (ReLU) or variants thereof are much more recommended [80, 89]. Nonetheless, in our numerical investigations we have found that for the situations considered in this work, tanh matches or sometimes even outperforms the other available activation functions. To improve the training process with tanh activation, weights are initialized following the 'GlorotUniform' initializer explained in Ref. [90]. Energy values are normalized to zero mean and are in principle unbounded. Thus, for the energy model a linear output layer is a rational choice. On the other hand, cross sections and rates are unnormalized and cover a range between zero and unity. Hence, we chose a sigmoid output layer. Of course, in the typical ranges of output values ($\sim 10^{-3}$ to 10^{-7}) sigmoid is almost constant. But for the present models, this seems not to be a very serious problem. Overfitting to the training data is reduced by regularizing the models with dropout [91] with a soft probability of 0.01 (E) or 0.05 – 0.1 (P, AM, and F), respectively. We do not regularize our models very strongly due to the model capacity being too low for interpolating training data.

c. Optimization. The neural networks are trained on minibatches of size 2^{11} using the Adam optimization algorithm [92] with early stopping and a maximum of 1200 epochs (i.e., forward and backward passes through the neural network). We have chosen Adam as it is known to be fairly robust to the choice of hyperparameters, like the learning rate. The learning rate for Adam is set to 0.0005 (E) and 0.001 (P, AM, and F), respectively. Most important for the learning is the loss function on which the optimization is performed. For the energy model, the mean squared error (MSE) on the training set is employed. MSE on a general data set D with size N_D in one dimension is given by

$$L_D^{\mathrm{MSE}} = \frac{1}{N_D} \sum_{j=1}^{N_D} (y_j^{\mathrm{calc}} - y_j^{\mathrm{pred}})^2, \tag{5}$$

where y_j^{calc} is the label value of the jth example in D and y_j^{pred} is the prediction, respectively. However, MSE only works well if labels cover a similar range. Otherwise, errors in small label values will be overlooked. Therefore, for the other three models, we define a mean squared logarithmic error (MSLE) as

$$L_D^{\mathrm{MSLE}} = \frac{1}{N_D} \sum_{j=1}^{N_D} (\log_{10}[y_j^{\mathrm{calc}} + \epsilon] - \log_{10}[y_j^{\mathrm{pred}} + \epsilon])^2, \tag{6}$$

on which they are trained. In this expression, $\epsilon = 10^{-10}$ is used for numerical stability. Values smaller than ϵ are practically treated as zero. The value of 10^{-10} makes sense since smaller photoionization cross sections are negligible. Due to the choice of sigmoid output activation, it is guaranteed that all $y_j^{\mathrm{pred}} > 0$. (They are set to zero later if they are found to be very small). Note that MSLE basically measures by how many orders of magnitude y_j^{calc} and y_j^{pred} differ.

d. Performance measure. Finally, we need to evaluate how well the trained model behaves on a test set not seen during training. A way of measuring the performance of the model is to compute the MSE (for E) or MSLE (for P, AM, and F), respectively, on the test set. In addition to this, for each example in the test set the absolute error $y_j^{\mathrm{calc}} - y_j^{\mathrm{pred}}$ or the logarithmic error $\log_{10}[y_j^{\mathrm{calc}} + \epsilon] - \log_{10}[y_j^{\mathrm{pred}} + \epsilon]$ can be calculated and represented in a histogram. For the AM or F models, however, there is one problem with the (quasi-)forbidden transitions. If $y_j^{\mathrm{calc}} = 0$ but $y_j^{\mathrm{pred}} > 0$ or vice versa, the logarithmic error mainly depends on our choice of ϵ. As a consequence, MSLE can be quite large and (at least) for a human-based interpretation loses its usefulness. Therefore, we perform the following. We measure the accuracy on the whole test set (or any other data set we are interested in). The accuracy,

$$A_D = \frac{\text{\# of correct predictions}}{N_D}, \tag{7}$$

here is a measure of how good the machine learning model learns whether a transition is allowed or (quasi-)forbidden, i.e., '# of correct predictions' includes the cases in which $y_j^{\text{calc}} > 0$ and $y_j^{\text{pred}} > 0$ or $y_j^{\text{calc}} = 0$ and $y_j^{\text{pred}} = 0$. Knowing the accuracy, all wrong predictions, i.e., $y_j^{\text{calc}} = 0$ but $y_j^{\text{pred}} > 0$ or $y_j^{\text{calc}} > 0$ but $y_j^{\text{pred}} = 0$, can, then, be excluded from the computation of MSLE and error histograms without any loss of information. It should be stressed that during training only MSLE is evaluated, but the accuracy is not. Embedding accuracy into training by, e.g., using a preceding classification model may be pursued in a future investigation.

D. Random forest regressor

A powerful alternative to neural networks is the random forest regressor technique [93, 94]. They are known to be much faster and much easier to tune with respect to hyperparameters than neural networks. But they have a limited capacity due to a lack of the depth that deep neural networks can have [95].

Random forest regressors are a decision-tree-based ensemble method. Briefly explained, the algorithm hierarchically separates the input space into subsets with respect to a specific feature and relation operators, i.e., it creates a decision tree. For each node (i.e., subset), the most important feature and the best split are controlled by a loss function; here we use MSE (for all models). To improve generalization, several of these trees are built and the final prediction is obtained as an average over all trees.

For our random forest regressors, we made use of the scikit-learn implementation [96] with 100 trees in the forest. In accordance with the 'modern' interpolation hypothesis [88], we barely regularize our models (i.e., the whole training dataset is used for building each tree and nodes are expanded in an unrestricted manner). This enables us to perfectly fit the training data and does not bound the test performance by a poor training performance. The only regularization performed here is that only $N_{\text{features}}^{\text{Model}} - 6$ features are randomly chosen at each node. As for the neural networks, these regularizations are determined by 'trial and error' with respect to the model performance, the time efficiency, and the robustness to different training sets. Performance is measured in the same manner as for the neural networks (using MSE for the E model and MSLE for the P, AM, and F models).

E. Machine-learning-based state-resolved Monte Carlo implementation

The state-resolved Monte Carlo calculations, described in Sec. II A, are computationally very expensive for situations in which a large number of orbitals is involved (i.e., for heavier atoms or when resonant excitations are included), caused by an exponentially large number of states involved in the x-ray multiphoton ionization dynamics (see Ref. [44]). We approach this challenge by employing a machine learning strategy to support the high-level Monte Carlo calculations. For this, we implement a machine-learning-based version of the state-resolved Monte Carlo implementation within XATOM [73], which closely couples the Monte Carlo calculation (Sec. II A) with a machine learning algorithm (Sec. II C or II D). It is sketched in Fig. 1. The state-resolved Monte Carlo implementation written in Fortran is coupled to the machine learning part written in Python via FIFO (first-in-first-out) special files. This enables a hand-in-hand exchange of the most important information or data without storing them physically on the disk.

We separate the Monte Carlo part into two iterative phases: (i) an initial training and test phase and (ii) a final production phase. During the first $N_{\text{traj}}^{\text{TT}}$ Monte Carlo trajectories (training and test phase, where TT indicates training and test phase), the electronic structure as well as the atomic transition parameters, i.e., transition energies, cross sections and rates, are explicitly calculated via quantum-state-resolved first-principle calculations. The calculated atomic transition parameters are collected and redirected to a FIFO file as a combined set of training and test data. The data are split up randomly into training data (85% of the data calculated) and test data (15% of the data calculated). Based on these data, the machine learning models are trained and their performance is evaluated. The trained model parameters are redirected into another FIFO file. Based on them the machine learning algorithms are reconstructed in Fortran. This does not include any complicated training procedures. It just means evaluating the sequence of linear functions and activation functions, whose parameters are determined by the read-in parameters. Using reconstructed machine learning models eliminates the need for further calls of the machine learning part in Python. Hence, no further communication between Fortran and Python, requiring further data sharing, is necessary in the final production phase. In the production phase a lot of further Monte Carlo trajectories ($N_{\text{traj}}^{\text{Prod}}$) are run until either the maximum number of trajectories is reached or CSDs are converged. During these Monte Carlo trajectories no further quantum-state-resolved first-principle calculations are performed. On the one hand, atomic transition parameters stored in memory are employed. This is the case when the transition at hand, for which we want to know the atomic transition parameters, has already been visited during the training and test phase and atomic transition parameters are stored. On the other hand, atomic transition parameters for transitions newly visited and, thus, not stored yet are predicted on the fly via the reconstructed machine learning models.

Results of x-ray multiphoton ionization dynamics calculations in Sec. III C are obtained in a way that all Monte Carlo trajectories add up to a total number of

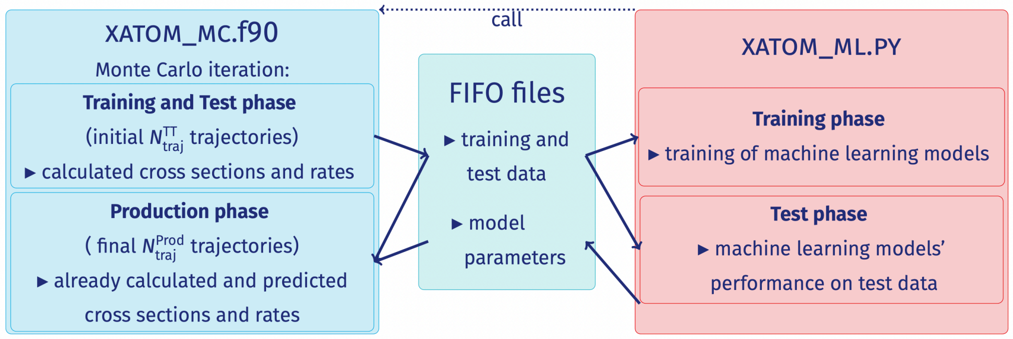
XATOM_MC.f90
Monte Carlo iteration:
Training and Test phase
(initial $N_{\text{traj}}^{\text{TT}}$ trajectories)
▸ calculated cross sections and rates
Production phase
(final $N_{\text{traj}}^{\text{Prod}}$ trajectories)
▸ already calculated and predicted cross sections and rates
call
FIFO files
▸ training and test data
▸ model parameters
XATOM_ML.PY
Training phase
▸ training of machine learning models
Test phase
▸ machine learning models' performance on test data

FIG. 1. Structure of the machine-learning-based state-resolved Monte Carlo implementation. The state-resolved Monte Carlo part in Fortran is called XATOM_MC, while XATOM_ML is the machine learning part in Python. The arrows indicate that data are streamed into (→) or are obtained from (←) the FIFO files.

TABLE III. Number of training and test Monte Carlo trajectories $N_{\text{traj}}^{\text{TT}}$ as well as the corresponding training and test data set size ($N_{\text{data}}^{\text{TT}}$) for the three different cases considered in this work for argon at 5 keV.

label	$N_{\text{traj}}^{\text{TT}}$	$N_{\text{data}}^{\text{TT}}$
(i)	4000	2 686 711
(ii)	9000	4 680 860
(iii)	28000	8 965 379

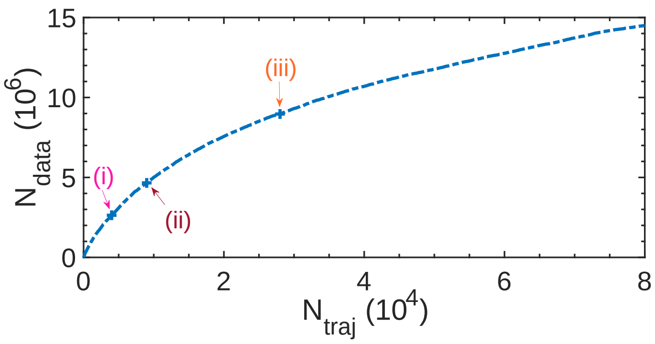

FIG. 2. Number of atomic transition parameters N_{data} collected as a function of the number of Monte Carlo trajectories N_{traj} for argon at 5 keV. The arrows indicate the three different cases considered in this work (see Table III).

80000 ($= N_{\text{traj}}^{\text{TT}} + N_{\text{traj}}^{\text{Prod}}$).

III. RESULTS AND DISCUSSION

We examine the performance of the machine-learning-based state-resolved Monte Carlo implementation for atomic argon at a photon energy of 5 keV. For such a photon energy, in principle all electrons can be ionized via x-ray sequential multiphoton ionization, i.e., a repeated sequence of one-photon ionization and inner-shell relaxation events. Therefore, no resonant excitation is involved in the x-ray multiphoton ionization dynamics. Following Ref. [44], we use a temporal Gaussian pulse envelope with 10 fs (full width at half maximum) and a fluence of 10^{12} photons/μm^2. Neither relativistic effects[34], nor nonsequential two-photon absorption, higher-order many-body processes such as shake-off, nor volume integration are included in the calculations (see Ref. [44] and references therein). Due to the lack of volume integration, a quantitative comparison with experimental results is not directly possible.

A. Data collection

Before exploring the machine-learning-based results, we start by considering how to choose the number of Monte Carlo trajectories for the training and test phase and the production phase. For the machine-learning-based state-resolved Monte Carlo implementation, the quantity that can easily be tuned in order to determine the size of the training and test data sets is the number of Monte Carlo trajectories during the training and test phase ($N_{\text{traj}}^{\text{TT}}$). Figure 2 shows the number of atomic transition parameters (N_{data}) collected as a function of the number of Monte Carlo trajectories (N_{traj}). Of course, this relation slightly varies from one Monte Carlo calculation to another, but, nonetheless, Fig. 2 gives a very good orientation on N_{data}. As can be seen, for argon at 5 keV, N_{data} is very high ($\sim 10^7$). This is related to the huge number of individual states involved in the calculations (Sec. II A). Moreover, N_{data} seems not to be saturated as N_{traj} increases, within the range of N_{traj} we used (up to N_{traj}=80000). This makes successfully training the machine learning models a tough task, as we are going to demonstrate in Sec. III B. On the other hand, this presents a situation in which a successful implementation of machine learning can be really helpful, in contrast to light atoms like neon. For our studies in Secs. III B and III C, we have chosen three different $N_{\text{traj}}^{\text{TT}}$ such that they cover a wide range of the curve in Fig. 2,

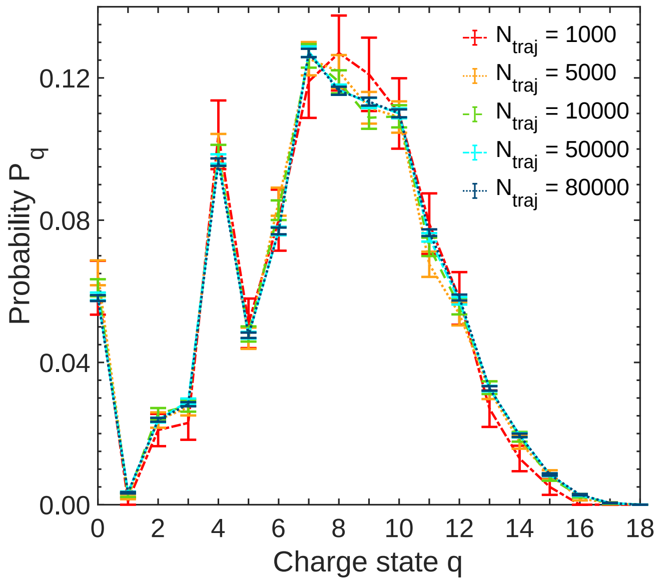

FIG. 3. Convergence behavior of argon CSDs as a function of the number of Monte Carlo trajectories N_{traj}. Results are obtained with the state-resolved Monte Carlo implementation of Ref. [44] (no machine learning). The error bars indicate the statistical error.

i.e., $N_{\mathrm{traj}}^{\mathrm{TT}}$=4000, 9000, and 28000 as listed in Table III.

To gain an overview of the remaining number of Monte Carlo trajectories in the production phase, in Fig. 3 we show argon CSDs obtained using the state-resolved Monte Carlo implementation (no machine learning) [44] with different numbers of Monte Carlo trajectories N_{traj}. The error bars represent the statistical error estimate of the Monte Carlo calculation for each charge state q, given by $\epsilon_q = \sqrt{P_q(1-P_q)/(N_{\mathrm{traj}}-1)}$, where, P_q is the population probability of the charge state q. Note the proportionality of the Monte Carlo error to $1/\sqrt{N_{\mathrm{traj}}-1}$, causing comparably large errors for small N_{traj}. Figure 3 demonstrates that for 10000 Monte Carlo trajectories the CSD is almost converged. It should be mentioned that more Monte Carlo trajectories are necessary for convergence of state-resolved quantities such as spectra. Due to this and in order to be safely sure that Monte Carlo errors are sufficiently small for our purpose, we utilize here 80000 Monte Carlo trajectories in total (i.e., training and test plus residual production trajectories). Note further that CSDs in Fig. 3 are converged when N_{data} is still far away from a saturation point (Fig. 2). This demonstrates that only a small fraction of frequently visited transitions is critical for the ionization dynamics calculations. This is also the reason why the Monte Carlo method is so powerful [38] and why the machine-learning-based state-resolved Monte Carlo implementation works (see Sec. III C).

B. Performance of machine learning models

Let us next discuss machine learning results for argon, employing both neural networks (Sec. II C) and random forest regressors (Sec. II D). We examine also how the machine learning performance depends on the training set size. In particular, the three different data set sizes (i.e., combined set of training and test data) listed in Table III are considered. According to the feature for the kind of process (p =1,2,or 3; see Sec. II B), each data set is separated into the individual subdata sets for the P model (25%−28% of $N_{\mathrm{data}}^{\mathrm{TT}}$), the AM model (57%−59% of $N_{\mathrm{data}}^{\mathrm{TT}}$), and the F model (15% − 16% of $N_{\mathrm{data}}^{\mathrm{TT}}$). For the E model, duplications with respect to the energy label are removed. (Thus, 29% − 35% of $N_{\mathrm{data}}^{\mathrm{TT}}$ are left over for the subdata sets for the E model.) Moreover, 85% of randomly selected data are employed as data for training a model, while the remaining 15% serve as test data.

To inspect the model's performance on the unseen test data set, we show three dimensional scatter plots of predicted data in Figs. 4(a)–(d) and 5(a)–(d) and the distribution of absolute or logarithmic errors in Figs. 4(e)–(h) and 5(e)-(h). For Figs. 4(a)–(d) and 5(a)–(d), the vertical axis is the predicted energy in eV [(a)] or the predicted cross section [(b)] or rate [(c)–(d)] in logarithmic scaling, while the horizontal axis is the underlying calculated value. For both, 100-eV bins [(a)] or 0.1 bins [(b)–(c)], respectively, are used. The colorbar shows the relative number of scatter points, scaled by 10^{-2} [(a)] or 10^{-3} [(b)–(c)], i.e., the number of pairs of calculation and prediction within a bin divided by the total number of test data for the model in question. The three dimensional plots supplement the common two dimensional scatter plots by futher information about the distribution of scatter points, which is sensible when the number of data is large, i.e., here, in the order of 10^5. Moreover, for brevity, only results for the case (ii) are given in the figures (for the other cases see below in the context of Table IV).

It becomes evident from Figs. 4(e) and 5(e) that transition energies are mostly predicted with better than 10-eV accuracy [i.e., the sum of error bars within the 10-eV window yields 99% (NN) or 94% (RF) for case (ii)] and with at most around 50 eV difference. The good energy model's performance is underlined by Figs. 4(a) and 5(a), looking very similar to the identity mapping of $y^{\mathrm{pred}} = y^{\mathrm{calc}}$. In contrast to the energy model, cross section and rate models perform less accurately for argon. Most predictions [85%–98% for case (ii)] deviate from the calculation by less than an order of magnitude [see 1-order windows in Figs. 4(f)–(h) and 5(f)–(h)] . However, deviations up to 4 orders of magnitude are possible. Comparably poor predictions can occur for all cross sections or rates, even though higher calculated values seem to be a little bit less inaccurate [see Figs. 4(b)–(d) and 5(b)–(d)]. Nonetheless, of course, a better accuracy is actually needed the higher the calculated cross section or rate. Using MSLE loss during training, the models

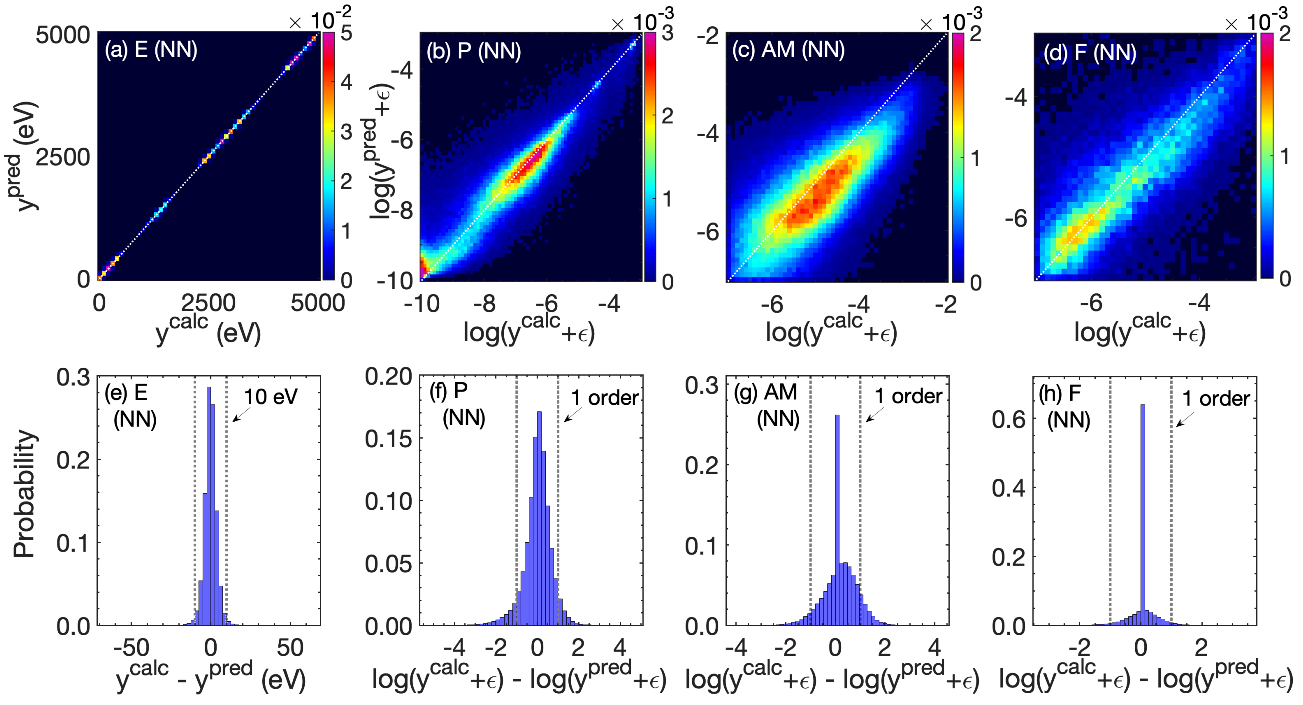

FIG. 4. Performance of the neural networks (NN) in terms of scatter plots [(a)–(d)] and of error histograms [(e)–(h)]: (a) and (e) the transition energy (E) in eV, (b) and (f) the photoionization cross section (P) in logarithmic scaling, (c) and (g) the Auger-Meitner decay rate (AM) in logarithmic scaling, and (d) and (h) the fluorescence rate (F) in logarithmic scaling, respectively. The colorbar in (a)–(d) shows the relative number of pairs $(y^{\rm calc}, y^{\rm pred})$, scaled by 10^{-2} [(a)] or 10^{-3} [(b)–(d)]. The dotted white line indicates the identity mapping. We consider the test data set of case (ii).

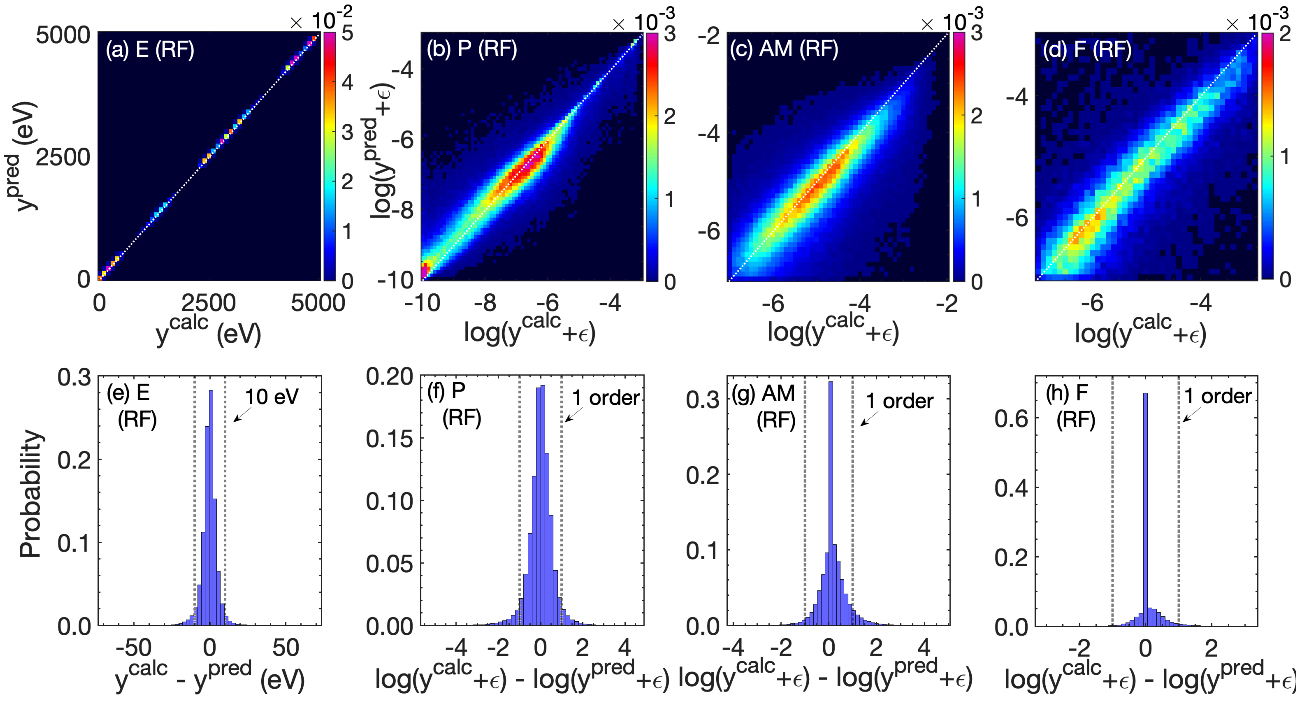

FIG. 5. Performance of the random forest regressors (RF) in terms of scatter plots [(a)–(d)] and of error histograms [(e)–(h)]: the panels show the same quantities as in Fig. 4.

TABLE IV. Test error statistics for the neural networks (NN) and random forest regressors (RF): MSE loss (L^{MSE}) [Eq. (5)] in eV^2 for the energy model (E) and MSLE loss (L^{MSLE}) [Eq. (6)] for the photoionization cross section (P), Auger-Meitner decay rate (AM), and fluorescence rate (F) models, evaluated on the test data sets of all three cases (Table III). For the AM and F models, additionally the accuracy (A) [Eq. (7)] is given, in %.

		NN			RF		
model	L or A	(i)	(ii)	(iii)	(i)	(ii)	(iii)
E	L_{Test}	13.1	12.2	9.4	30.4	27.0	24.8
P	L_{Test}	0.42	0.41	0.43	0.36	0.32	0.26
AM	L_{Test}	0.44	0.46	0.49	0.32	0.32	0.29
	A_{Test}	87.7	86.6	84.3	87.7	87.6	87.9
F	L_{Test}	0.13	0.14	0.15	0.11	0.10	0.09
	A_{Test}	90.9	91.0	89.5	89.0	89.3	89.6

cannot respect this in the model training and evaluation. Moreover, we observe that fluorescence rates are predicted the best, while Auger-Meitner decay rates possess the largest deviations. Interestingly, for photoionization and Auger-Meitner decay, there is a tendency that predictions are smaller than the calculated cross sections or rates [see Figs. 4(b)–(c) and 5(b)–(c)].

It is also worthy to note that 84%–91% of transitions for Auger-Meitner or fluorescence decay are classified as allowed or (quasi-)forbidden when also being allowed or (quasi-)forbidden, respectively. The test accuracy is listed in Table IV for both the neural network and the random forest regressor. For the other transitions that are classified incorrectly, the corresponding rates, i.e., those the transition actually has or the one the actually (quasi-)forbidden transition receives, are mostly also small ($\sim 10^{-6}$ a.u.). Some of them are, however, comparably high with up to 10^{-3} a.u. (not shown for brevity). Recall that these wrongly classified transitions are not included in any other quantity to measure the test performance, like the error distribution, to stress that this does not cause alone the deviations from a perfect performance.

Next, we examine how the machine learning models' performance depends on the training set size. Table IV reports test loss and accuracy for all three data set cases in Table III and for both the neural network and the random forest regressor. (For more details on these quantities see Sec. II C.) More training data improve the energy model, which is already quite good, whereas the cross section and rate models only marginally profit for the random forest regressor or do not profit at all for the neural networks. A possible explanation for the lack of improvement, especially for the neural networks, might be a too low model capacity that slows down the gain of better generalization (see below in the context of Table VI). In this context, it is important to emphasize that using more data for training does not have an essential impact on the machine learning model's performance, but enhances training times more than linearly. Table V compares training times for all three different data set cases (Table III). Calculations are performed on AMD EPYC 7302 16-Core processors with a maximum number of 64 virtual cores (threads).

We now contrast the neural network and the random forest regressor. As expected, both behave generally fairly similarly (compare Figs. 4 and 5). From the results in Table IV, we conclude that neural networks are better suited for predicting energies. However, for cross sections and rates, the random forest regressors outperform the neural networks. Random forest regressors for cross sec-

TABLE V. Training times, real and CPU, in minutes for the neural networks (NN) and the random forest regressors (RF) for all four models. We consider the training data sets of all three cases (Table III).

		real time (min)		CPU time (min)	
data set	model	NN	RF	NN	RF
(i)	E	92	5	2674	5
	P	112	4	4339	4
	AM	228	9	8904	9
	F	40	2	1549	2
(ii)	E	153	9	4401	9
	P	187	8	7233	8
	AM	399	22	15733	22
	F	102	4	3965	4
(iii)	E	247	16	7526	16
	P	346	15	13833	15
	AM	691	45	28627	45
	F	211	8	8523	8

TABLE VI. Training and validation performance of the neural networks (NN): MSE loss (L^{MSE}) [Eq. (5)] in eV^2 for the energy model (E) and MSLE loss (L^{MSLE}) [Eq. (6)] for the photoionization cross section (P), Auger-Meitner decay rate (AM), and fluorescence rate (F) models, evaluated on the training (Tr) and validation (V) data sets of all three cases (Table III). For the AM and F models, additionally the accuracy (A) [Eq. (7)] is given, in %.

model	L or A	(i) NN	(ii) NN	(iii) NN
E	L_{Tr}	8.2	8.0	5.9
	L_{V}	11.6	10.7	8.1
P	L_{Tr}	0.22	0.29	0.36
	L_{V}	0.42	0.42	0.43
AM	L_{Tr}	0.39	0.42	0.48
	A_{Tr}	91.4	88.7	85.3
	L_{V}	0.44	0.46	0.50
	A_{V}	87.8	86.6	84.3
F	L_{Tr}	0.09	0.11	0.13
	A_{Tr}	96.5	95.2	91.9
	L_{V}	0.14	0.14	0.15
	A_{V}	90.9	90.6	89.6

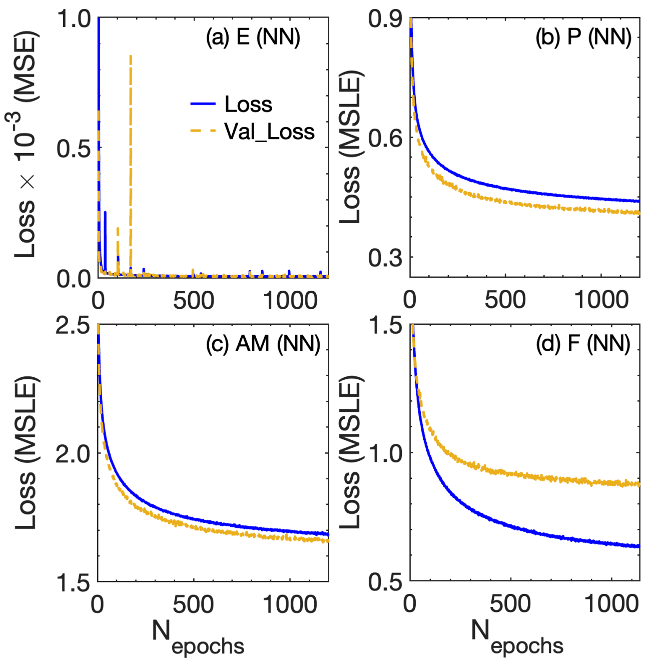

FIG. 6. Loss curves of the neural networks (NN): loss evaluated on the training data (blue) and the validation data (orange) as a function of the number of epochs trained for all four models. In (a) the MSE loss [Eq. (5)] in Z-score scaling and in (b)–(d) the unscaled MSLE loss [Eq. (6)] is shown. We consider the training data set of case (ii).

tion and rate predictions exhibit an improvement with more data, while those for neural networks decline. As a consequence, this increases the difference between neural networks and random forest regressors with increasing training set sizes. Most importantly, neural networks have a critical disadvantage. Training is very expensive (see Table V). It can cost several hours for a single neural network, though multiple cores are utilized (real time << CPU time). Unlike neural networks, random forest regressors are trained in less than an hour even for the largest training sets on a single core (real time = CPU time).

In order to complete our understanding of the neural networks' performance, we finally briefly examine the training and validation losses in Table VI. (For the random forest regressors, those investigations are not possible since training data are interpolated and, thus, always have near zero loss.) Note that per default in TensorFlow some of the data are separated from the training data and are used as validation data. Here, we use 10% of the training data for validation. The results in Table VI demonstrate that the capacities of the cross section and rate models are too low to nearly perfectly fit the training data. The task of predicting cross sections and rates is too complex. This can be seen by the large training losses, increasing with more training data. As a consequence, validation and test data cannot be predicted very accurately either. Whether a network architecture with a higher model capacity can significantly overcome this limitation remains an open question at the moment (that might be answered in the future). However, it is most likely that a higher model capacity will lead to a substantial increase in training time. Therefore, the chosen network architecture is a compromise between computational effort and numerical accuracy. In contrast, for the energy model, the capacity seems to be sufficient since predicting transition energies is an easier task. In addition, Fig. 6 shows the evolution of training and validation losses with the number of epochs trained for the case (ii) only. We remark that using dropout during training increases the training loss and, hence, it is normal that validation losses can be smaller than training losses. For the final loss, dropout is not included. Thus, the training losses in Table VI are smaller than in Fig. 6. Moreover, we note that the losses for the Auger-Meitner decay and fluorescence rate models are clearly larger in Fig. 6 than in Table VI due to wrongly classified transitions. As can be seen, loss curves are quite smooth and almost converged. Even early stopping before the maximal number of 1200 epochs due to increasing validation loss is possible [see Fig. 6(d)]. Especially for the cross section and rate models, the losses decrease only by a few percent ($< 10\%$) during the last 600 epochs. As a consequence, training longer would not have a notable effect on training, validation, and test performances. Since training times are approximately linear in the number of epochs trained, we may, on the contrary, save some training time by using fewer epochs without a noticeable reduction in performance.

C. Results for machine-learning-based state-resolved Monte Carlo calculations

Having investigated the performance of different machine learning models in the previous subsection, we now study the performance of x-ray multiphoton ionization-dynamics calculations carried out with the machine-learning-based state-resolved Monte Carlo implementation introduced in Sec. II E. For simplicity, we only employ the neural networks of Sec. III B as machine learning models in the production phase. Random forest regressors perform comparably to neural networks, as shown in Sec. III B. However, they are a bit harder to embed in the state-resolved Monte Carlo implementation due to the large number of individual trees, which all need to be redirected to the FIFO file and reconstructed in Fortran (see Sec. II E). We compare machine-learning-based state-resolved Monte Carlo calculations for argon with state-resolved Monte Carlo calculations using the implementation introduced in Ref. [44] (in which no machine learning is employed).

Figure 7 compares argon CSDs for all three previously considered data set cases, which correspond to different numbers of training and test Monte Carlo trajectories (Table III). All machine-learning-based CSDs match

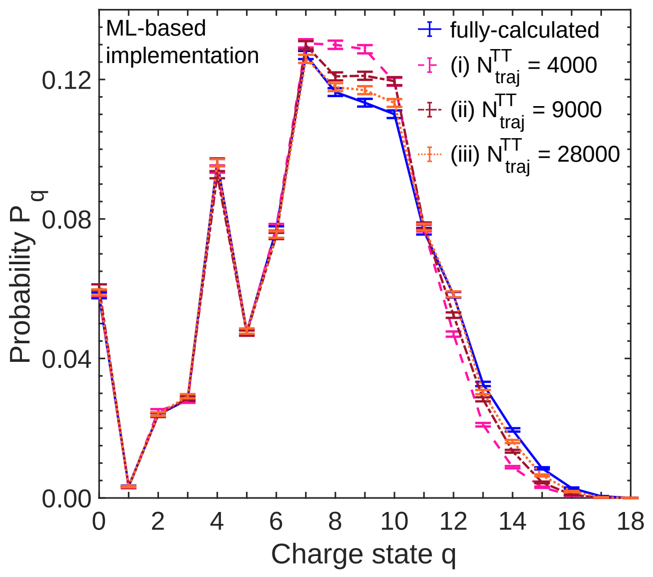

FIG. 7. Comparison of machine-learning-based CSDs with the fully-calculated CSD. Results are obtained with the state-resolved Monte Carlo implementation without machine learning [44] (fully-calculated) and with the machine-learning-based (ML-based) state-resolved Monte Carlo implementation [Sec. II E] for different numbers of training and test Monte Carlo trajectories $N_{\mathrm{traj}}^{\mathrm{TT}}$ (Table III). The error bars indicate the statistical error of the Monte Carlo calculation.

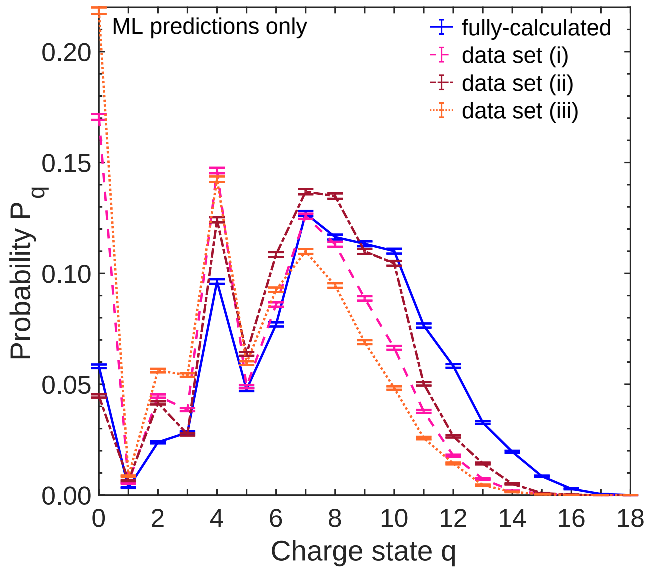

FIG. 8. Similar to Fig. 7, but all atomic transition parameters are predicted by the previously trained neural networks for the different data sets (Table III).

the overall behavior of the fully-calculated CSD (i.e., in which no machine learning is employed). Especially for low charge states (i.e., $q \leq 7$), the agreement is good for all three machine-learning cases. For larger charge states, however, deviations beyond the Monte Carlo errors can be observed, which are enhanced the smaller the number of training and test Monte Carlo trajectories [cases (i) and (ii)]. This is because of the machine-learning predictions of atomic transition parameters for the transitions newly visited in the production phase. As seen in Sec. III B, the predictions for cross sections and rates made by the neural networks are not very accurate. Since the transitions newly visited in the production phase are not directly sampled from the same distribution as the training and test data used in Sec. III B, they are generally expected to be predicted even less accurately (not shown for brevity) [49, 80]. The fact that the machine-learning-based CSDs are, nonetheless, quite good relies on the use of atomic transition parameters already calculated. For $\sim 14\%$ [case (i)], $\sim 24\%$ [case (ii)], or $\sim 40\%$ [case (iii)] of individual initial states all possible atomic transition parameters are calculated in the training and test phase, and are used in the production phase (see Sec. II E). It also explains the improvement with more training and test Monte Carlo trajectories attributed to more calculated atomic transition parameters.

To illustrate this point, in Fig. 8, we show comparisons of CSDs where the machine-learning-based CSDs are obtained by using only machine learning predictions for atomic transition parameters. In particular, we do not use the machine-learning-based implementation as described in Sec. II E, combining both calculated and predicted atomic transition parameters. Instead, for Fig. 8, only the production phase is run with all atomic transition parameters being predicted by the previously trained neural networks. As can be seen, when only predicted atomic transition parameters are used the overall behavior of the machine-learning-based CSDs still roughly matches that of the fully-calculated CSD. But the agreement is no longer close to being quantitative.

In this context, let us briefly come back to the random forest regressors (Sec. III B), which perfectly interpolate the training data. Thus, employing random forest regressors, it would barely make a difference whether atomic transition parameters already calculated are used or whether all atomic transition parameters are predicted by the random forest regressors. In particular, Figs. 7 and 8 would look very similar to each other and this would make the above investigation impossible. Moreover, having at hand the atomic transition parameters already calculated in the training and test phase (see Sec. II E), we do not consider this as an advantage of the random forest regressors.

Figure 9 shows the photoelectron [(a)–(b)], the Auger-Meitner electron [(c)–(d)], and the fluorescence [(e)–(f)] spectra with an energy resolution of 1 eV. At that energy resolution, the Auger-Meitner electron and the x-ray fluorescence spectra form a quasi-continuum over most parts of the energy ranges shown. Likewise to the CSD, fully-calculated results obtained with the implementation of Ref. [44] are compared to machine-learning-based results obtained with the implementation of Sec. II E, as well as those that are based only on machine learning predictions. (For brevity, only case (i) of Table III is

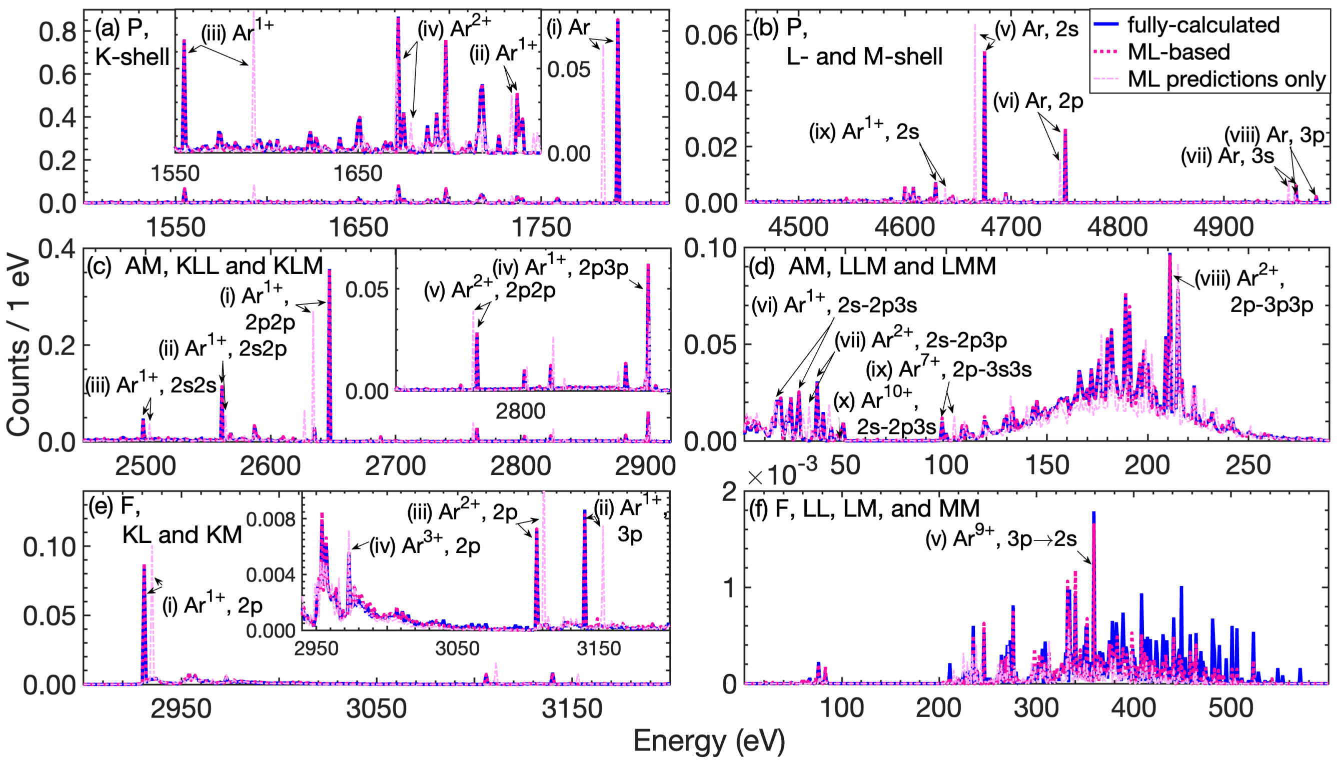

FIG. 9. Comparison of machine-learning-based (ML-based) and fully-calculated spectra for (a)–(b) photoelectron (P), (c)–(d) Auger-Meitner electron (AM), and (e)–(f) fluorescence (F). Calculations obtained with the state-resolved Monte Carlo implementation without machine learning [44] (blue) are compared to those obtained with the machine-learning-based state-resolved Monte Carlo implementation [Sec. II E] for case (i) in Table III (magenta). Additionally, results with all atomic transition parameters being predicted by the previously trained neural networks (pink) are given. The peak labels are explained in Tables VII–IX.

shown.) Some of the dominant peaks that can be assigned to at most two dominant processes are labeled with roman numbers; the corresponding transitions are specified in Table VII [for Figs. 9(a)–(b)], Table VIII [for Figs. 9(c)–(d)], and Table IX [for Figs. 9(e)–(f)].

Most importantly, we observe in Fig. 9 that the machine-learning-based spectra obtained with the implementation of Sec. II E (magenta lines) are in overall very good agreement with the fully-calculated ones, apart from small details. This is due to the fact that spectral features are dominated by peaks belonging to very low charge states (see Tables VII–IX). But for low charge states, the corresponding atomic transition parameters are mostly all already calculated in the training and test phase. Thus, they are unaltered by machine learning and the good agreement mainly relies on atomic transition parameters already calculated. The small deviations in the spectra are caused by the influence of the error between predictions and calculations on the x-ray multiphoton ionization dynamics; here mostly indirectly via the impact on the population of the underlying initial states, so that there is no energy shift [e.g., peak (ix) in Fig. 9(d) or peak(v) in Fig. 9(f)]. Shifts in energy affect peaks that are too small to be visible.

Let us also examine the machine-learning-based spectra obtained by using predicted atomic transition parameters only (pink lines in Fig. 9). Interestingly, even in this situation the spectra roughly capture the overall behavior of the fully-calculated spectra. Indeed, peak positions are shifted in energy, however, mostly within less than 10 eV (see Tables VII–IX and Sec. III B). Also the peak heights do not match very well. Nonetheless, the neural networks are good enough to recognize general tendencies in the x-ray multiphoton ionization dynamics, e.g., more Auger-Meitner decay than fluorescence decay [compare peak (i) in Fig. 9(c) to peak (i) in Fig. 9(e)]. Furthermore, it is worth mentioning that for high charge states LLM Auger-Meitner decay [i.e., $2s-2p3l$ $(l=s,p)$] is often actually forbidden due to calculated transition energies being smaller than zero. However, the neural network is unable to learn this. Consequently, actually forbidden transitions can take place in the machine-learning-based calculations [see, e.g., peak (x) in Fig. 9(d)]. In the present situation, this has only a minor impact.

Next, we investigate the time effort of the production phase in the machine-learning-based state-resolved Monte Carlo implementation (Sec. II E). Table X lists the computational times for a production phase consisting of 5000 Monte Carlo trajectories based on the three cases in Table III. For comparison, a fully-calculated

TABLE VII. Peak assignment in the photoelectron spectra [Figs. 9(a) and 9(b)]. Calculated transition energies, $E^{\mathrm{calc}}_{I^i\to I^f}$, and transition energies predicted by the neural network [case (i)], $E^{\mathrm{pred}}_{I^i\to I^f}$, are listed for the underlying process.

label	process	$E^{\mathrm{calc}}_{I^i\to I^f}$ (eV)	$E^{\mathrm{pred}}_{I^i\to I^f}$ (eV)
(i)	Ar, $1s^2 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^1 2s^2 2p^6 3s^2 3p^6$ (^2S)	1792	1784
(ii)	Ar^{1+}, $1s^2 2s^2 2p^5 3s^2 3p^6$ $(^2P) \to 1s^1 2s^2 2p^5 3s^2 3p^6$ (^3P)	1737	1734
	Ar^{3+}, $1s^2 2s^2 2p^6 3s^1 3p^4$ $(^2D) \to 1s^1 2s^2 2p^6 3s^1 3p^4$ (^1D)		1731
(iii)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^0 2s^2 2p^6 3s^2 3p^6$ (^1S)	1555	1593
(iv)	Ar^{2+}, $1s^2 2s^2 2p^4 3s^2 3p^6$ $(^1D) \to 1s^1 2s^2 2p^4 3s^2 3p^6$ (^2D)	1672	1671
	Ar^{2+}, $1s^2 2s^2 2p^4 3s^2 3p^6$ $(^1S) \to 1s^1 2s^2 2p^4 3s^2 3p^6$ (^2S)		1679
(v)	Ar, $1s^2 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^2 2s^1 2p^6 3s^2 3p^6$ (^2S)	4675	4666
(vi)	Ar, $1s^2 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^2 2s^2 2p^5 3s^2 3p^6$ (^2P)	4751	4746
(vii)	Ar, $1s^2 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^2 2s^2 2p^6 3s^1 3p^6$ (^2S)	4968	4961
(viii)	Ar, $1s^2 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^2 2s^2 2p^6 3s^2 3p^5$ (^2P)	4987	4967
(ix)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^1 2s^1 2p^6 3s^2 3p^6$ (^1S)	4629	4638

TABLE VIII. Peak assignment in the Auger-Meitner electron spectra [Figs. 9(c) and 9(d)]. Calculated transition energies, $E^{\mathrm{calc}}_{I^i\to I^f}$, and transition energies predicted by the neural network [case (i)], $E^{\mathrm{pred}}_{I^i\to I^f}$, are listed for the underlying process.

label	process	$E^{\mathrm{calc}}_{I^i\to I^f}$ (eV)	$E^{\mathrm{pred}}_{I^i\to I^f}$ (eV)
(i)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^2 2p^4 3s^2 3p^6$ (^1D)	2647	2634
(ii)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^1 2p^5 3s^2 3p^6$ (^1P)	2561	2564
(iii)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^0 2p^6 3s^2 3p^6$ (^1S)	2498	2503
(iv)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^2 2p^5 3s^2 3p^5$ (^1D)	2902	2903
(v)	Ar^{2+}, $1s^0 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^1 2s^2 2p^4 3s^2 3p^6$ (^2D)	2765	2762
(vi)	Ar^{1+}, $1s^2 2s^1 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^2 2p^5 3s^1 3p^6$ (^1P)	27	16
(vii)	Ar^{2+}, $1s^2 2s^1 2p^5 3s^2 3p^6$ $(^1P) \to 1s^2 2s^2 2p^4 3s^2 3p^5$ (^2P)	36	32
(viii)	Ar^{2+}, $1s^2 2s^2 2p^4 3s^2 3p^6$ $(^1D) \to 1s^2 2s^2 2p^5 3s^2 3p^4$ (^2D)	211	212
(ix)	Ar^{7+}, $1s^2 2s^2 2p^5 3s^2 3p^0$ $(^2P) \to 1s^2 2s^2 2p^6 3s^0 3p^0$ (^1S)	98	104
(x)	Ar^{10+}, $1s^2 2s^1 2p^4 3s^1 3p^0$ $(^3D) \to 1s^2 2s^2 2p^3 3s^0 3p^0$ (^2D)	< 0	47

state-resolved Monte Carlo calculation (i.e., using the implementation in Ref. [44]) is also included with the same number of 5000 Monte Carlo trajectories. All calculations are performed on Intel Xeon E5-2630L with a single core. A significant reduction in the computational times can be found for the production phases. Using atomic transition parameters already calculated and machine learning predictions for those not already calculated is on average 6 to 10 times faster than the full calculation. The more atomic transition parameters are already calculated, the faster is the production phase [compare cases (i)–(iii) in Table X] since fewer predictions made by the machine learning models are required. [But note that this gain is at the expense of a more expensive training and test phase and machine learning model training (see Table V).] Although machine learning models are employed in the production phase, there is still a nonnegligible time effort of about 2 hours for just 5000 Monte Carlo trajectories. Predicting a single transition via the reconstructed deep neural networks is indeed fast (< 5 ms). However, predicting a huge number of transitions (order of 10^6) is notably expensive.

To evaluate the overall saving in computational times for the machine-learning-based state-resolved Monte Carlo implementation, timings for the training and test phase, for the training of the machine learning models, and for the production phase must be compared to the full state-resolved Monte Carlo calculation. However, such timings mainly depend on the available computer architecture (i.e., parallelization of the calculations, cluster usage, number of available cores). Therefore, we do not further discuss this point here. Having at hand Tables V and X enables us to estimate for a given computer architecture whether the embedding of machine learning is more time efficient than the full calculation.

Finally, another advantage of the machine-learning-based state-resolved Monte Carlo implementation should be stressed. In the production phase it is not necessary

TABLE IX. Peak assignment in the fluorescence spectra [Figs. 9(e) and 9(f)]. Calculated transition energies, $E^{\mathrm{calc}}_{I^i \to I^f}$, and transition energies predicted by the neural network [case (i)], $E^{\mathrm{pred}}_{I^i \to I^f}$, are listed for the underlying process.

label	process	$E^{\mathrm{calc}}_{I^i \to I^f}$ (eV)	$E^{\mathrm{pred}}_{I^i \to I^f}$ (eV)
(i)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^2 2p^5 3s^2 3p^6$ (^2P)	2931	2935
(ii)	Ar^{1+}, $1s^1 2s^2 2p^6 3s^2 3p^6$ $(^2S) \to 1s^2 2s^2 2p^6 3s^2 3p^5$ (^2P)	3140	3153
(iii)	Ar^{2+}, $1s^0 2s^2 2p^6 3s^2 3p^6$ $(^1S) \to 1s^1 2s^2 2p^5 3s^2 3p^6$ (^1P)	3106	3111
(iv)	Ar^{3+}, $1s^1 2s^2 2p^4 3s^2 3p^6$ $(^2D) \to 1s^2 2s^2 2p^3 3s^2 3p^6$ (^2D)	2973	2973
(v)	Ar^{9+}, $1s^2 2s^1 2p^5 3s^0 3p^1$ $(^1D) \to 1s^2 2s^2 2p^5 3s^0 3p^0$ (^2P)	359	357

to perform electronic-structure calculations [71], which are the fundamental basis for the calculation of individual state-to-state cross sections and transition rates. In order to reduce the computational time, electronic-structure information is stored in memory. However, storing electronic-structure information for argon uses more than 100 times more memory than for storing just atomic transition parameters. In particular, the amount of memory used for storing the relevant information, i.e., electronic-structure information and atomic transition parameters, during 5000 Monte Carlo trajectories is on the order of 10^4 megabytes. As a consequence, with the machine-learning-based state-resolved Monte Carlo implementation the memory usage can be dramatically reduced. In particular, the amount of memory used for storing the relevant information, i.e., only atomic transition parameters, in the production phase is on the order of 10^2 megabytes.

IV. CONCLUSION

In this paper, we have presented a machine-learning-based state-resolved Monte Carlo implementation for computing x-ray multiphoton ionization dynamics using the XATOM toolkit. The objective of machine learning is here to accelerate the extremely time-consuming state-resolved calculations of atomic transition parameters. In particular, in an initial training and test phase of the Monte Carlo calculation, quantum-state-resolved first-principle calculations of atomic transition parameters are carried out and these data serve the training and testing of the machine learning models. The trained and tested machine learning models are then employed in a final production phase for predicting atomic transition parameters for transitions newly visited in this phase.

We have compared the performance of neural networks and random forest regressors as possible machine learning models. Both types of machine learning models exhibit a similar accuracy for the prediction of atomic transition parameters, though neural networks have the critical disadvantage of very expensive training. Subsequently, we have discussed state-resolved CSDs as well as electron and photon spectra for argon, which have not been presented before. We compare results obtained by the machine-learning-based state-resolved Monte Carlo implementation embedding the neural networks to fully-calculated results obtained with the implementation in Ref. [44]. Our work demonstrates that the proposed machine-learning-based state-resolved Monte Carlo implementation works in principle and that the performance, in terms of CSDs and spectra, is good. The achieved level of accuracy in CSDs and spectra is satisfactory in view of the fact that, for instance, higher-order many-body processes are neglected [44] and that calculated cross sections and rates are not perfect either [71]. Once the machine learning models are trained, the final production phase can be performed faster than the full calculation. However, two main shortcomings have become evident: (i) the accuracy of the machine learning predictions is limited, especially for less likely transitions, and (ii) training the neural networks is also quite time-consuming.

In summary, let us briefly answer the question whether state-resolved ionization dynamics calculations can be accelerated by the presented machine-learning-based state-resolved Monte Carlo implementation. When a computer cluster is available, running several fully-calculated Monte Carlo calculations—each with only a small number of trajectories—in parallel on several cluster nodes is indeed the more powerful method. This is attributed to large training times of the machine learning models, limited prediction accuracy, and the need for fully-calculated training and test Monte Carlo trajectories anyhow in the machine-learning-based calculations. But, if the computational resources are restricted, i.e., only a single or few computers and/or rather limited memory are available,

TABLE X. Timings for the production phase of the machine-learning-based state-resolved Monte Carlo implementation based on the cases in Table III and for $N^{\mathrm{Prod}}_{\mathrm{traj}} = 5000$. Average real times are compared to the full calculation with $N_{\mathrm{traj}} = 5000$ using the implementation in Ref. [44] [no machine learning (ML)].

	no ML	ML (i)	ML (ii)	ML (iii)
Average real time	15h 27min	2h 28min	2h 06min	1h 30min

then the machine-learning-based state-resolved Monte Carlo implementation is a promising option. After an expensive training and test phase, state-resolved ionization dynamics calculations can be performed more easily for a sufficiently large number of Monte Carlo trajectories. Another optional application of the machine-learning-based state-resolved Monte Carlo implementation would be its use in an x-ray parameter scan, i.e., performing a lot of x-ray multiphoton ionization dynamics calculations with different fluence and/or pulse duration values. In this case, the advantage is that machine learning models need to be trained only a single time and can then be reused in all other calculations. Note that for a scan of the photon energy, this would not be possible because cross sections are photon energy specific.

There are several promising perspectives for future developments of the machine-learning-based state-resolved Monte Carlo implementation. First, an important point is further feature engineering by adding more features and then sending the resulting feature vectors through, for instance, autoencoders [97] or using principal component analyses [80]. A better suited feature representation might help the machine learning models to learn and, thus, can improve the performance. Other interesting directions for improving the machine learning models would be ensemble methods, like gradient boosted trees [98], batch normalization [80], advanced random forest methods [95], inclusion of feedback, like in recurrent neural networks [80], or combining the power of neural networks and random forest regressors [94, 99]. Training times for neural networks are often reduced by using GPUs instead of CPUs [80]. A further point is that for a new Monte Carlo calculation using a different atomic species and/or a different photon energy in principle the machine learning models have to be reoptimized on the newly collected atomic transition parameters. A question here is whether information gained from previous Monte Carlo calculations can be transferred to a new Monte Carlo calculation and whether this can accelerate training and/or improve the machine learning models' performance. Lastly, another interesting aspect is that the main computational effort of the state-resolved Monte Carlo calculations is due to the extremely huge number of atomic transition parameters that need to be calculated or predicted. But are atomic transition parameters for all transitions really required? Or could a machine-learning model maybe select the most dominant transitions for a visited initial state, so that predictions (or calculations) could be restricted to this subset of dominant transitions (in spirit of proposals made in Refs. [69, 70, 100] for configuration interaction calculations)? Such developments are crucial before many practical applications can really profit from the presented machine-learning-based state-resolved Monte Carlo implementation.

An attractive application of great scientific interest is the extension of the machine-learning-based state-resolved Monte Carlo implementation to atoms as heavy as xenon. Especially for heavy atoms, relativistic, quantum-electrodynamic, and finite-nuclear-size effects play an important role [34]. It is, therefore, desirable to embed them into the quantum-state-resolved electronic-structure calculations [44, 71], though this further expands substantially the number of atomic transition parameters required and the computational effort. Consequently, accelerating huge-sized ionization dynamics calculations will be a promising perspective for the realization of more accurate calculations. It is also an important step toward the quantitative exploration of a wide variety of different atomic systems and toward the optimization of x-ray beam parameters for applications of x-ray free-electron lasers.

V. ACKNOWLEDGEMENTS

L. B., M. T.-F., and R. S. acknowledge support by DASHH (Data Science in Hamburg - HELMHOLTZ Graduate School for the Structure of Matter) with Grant-No. HIDSS-0002. L. B., S.-K. S., Z. J., M. M. A., and R. S. acknowledge support from DESY (Hamburg, Germany), a member of the Helmholtz Association HGF.

[1] P. Emma, R. Akre, J. Arthur, R. Bionta, C. Bostedt, J. Bozek, A. Brachmann, P. Bucksbaum, R. Coffee, F.-J. Decker, et al., First lasing and operation of an ångstrom-wavelength free-electron laser, *Nat. Photon.* **4**, 641–647 (2010). doi:10.1038/nphoton.2010.176.
[2] T. Ishikawa, H. Aoyagi, T. Asaka, Y. Asano, N. Azumi, T. Bizen, H. Ego, K. Fukami, T. Fukui, Y. Furukawa, et al., A compact x-ray free-electron laser emitting in the sub-ångström region, *Nat. Photon.* **6**, 540–544 (2012). doi:10.1038/nphoton.2012.141.
[3] H.-S. Kang, C.-K. Min, H. Heo, C. Kim, H. Yang, G. Kim, I. Nam, S. Y. Baek, H.-J. Choi, G. Mun, et al., Hard x-ray free-electron laser with femtosecond-scale timing jitter, *Nat. Photon.* **11**, 708–713 (2017). doi:10.1038/s41566-017-0029-8.
[4] E. Prat, R. Abela, M. Aiba, et al., A compact and cost-effective hard x-ray free-electron laser driven by a high-brightness and low-energy electron beam, *Nat. Photon.* **14**, 748–756 (2020). doi:10.1038/s41566-020-00712-8.
[5] C. Pellegrini, A. Marinelli, and S. Reiche, The physics of x-ray free-electron lasers, *Rev. Mod. Phys.* **88**, 015006 (2016). doi:10.1103/RevModPhys.88.015006.
[6] W. Decking, S. Abeghyan, P. Abramian, A. Abramsky, A. Aguirre, C. Albrecht, P. Alou, M. Altarelli, P. Altmann, K. Amyan, et al., A MHz-repetition-rate hard x-ray free-electron laser driven by a superconducting lin-

ear accelerator, *Nat. Photon.* **14**, 391–397 (2020). doi:10.1038/s41566-020-0607-z.

[7] J. C. H. Spence, XFELs for structure and dynamics in biology, *IUCrJ* **4**, 322–339 (2017). doi:10.1107/S2052252517005760.

[8] I. Schlichting and J. Miao, Emerging opportunities in structural biology with x-ray free-electron lasers, *Curr. Opin. Struct. Biol.* **22**, 613–626 (2012). doi:10.1016/j.sbi.2012.07.015.

[9] E. Sobolev, S. Zolotarev, K. Giewekemeyer, J. Bielecki, K. Okamoto, H. Reddy, J. Andreasson, K. Ayyer, I. Barák, S. Bari, et al., Megahertz single-particle imaging at the European XFEL, *Commun. Phys.* **3**, 97 (2020). doi:10.1038/s42005-020-0362-y.

[10] M. M. Seibert, T. Ekeberg, F. R. N. C. Maia, M. Svenda, J. Andreasson, O. Jönsson, D. Odić, B. Iwan, A. Rocker, D. Westphal, et al., Single mimivirus particles intercepted and imaged with an x-ray laser, *Nature (London)* **470**, 78–81 (2011). doi:10.1038/nature09748.

[11] H. N. Chapman, P. Fromme, A. Barty, T. A. White, R. A. Kirian, A. Aquila, M. S. Hunter, J. Schulz, D. P. DePonte, U. Weierstall, et al., Femtosecond x-ray protein nanocrystallography, *Nature (London)* **470**, 73–77 (2011). doi:10.1038/nature09750.

[12] J. Coe and P. Fromme, Serial femtosecond crystallography opens new avenues for structural biology, *Protein Pept Lett.* **23**, 255–272 (2016). doi:10.2174/0929866523666160120152937.

[13] D. Assalauova, Y. Y. Kim, S. Bobkov, R. Khubbutdinov, M. Rose, R. Alvarez, J. Andreasson, E. Balaur, A. Contreras, H. DeMirci, et al., An advanced workflow for single-particle imaging with the limited data at an x-ray free-electron laser, *IUCrJ* **7**, 1102–1113 (2020). doi:10.1107/S2052252520012798.

[14] U. Lorenz, N. M. Kabachnik, E. Weckert, and I. A. Vartanyants, Impact of ultrafast electronic damage in single-particle x-ray imaging experiments, *Phys. Rev. E* **86**, 051911 (2012). doi:10.1103/PhysRevE.86.051911.

[15] H. M. Quiney and K. A. Nugent, Biomolecular imaging and electronic damage using x-ray free-electron lasers, *Nat. Phys.* **7**, 142–146 (2011). doi:10.1038/nphys1859.

[16] S.-K. Son, L. Young, and R. Santra, Impact of hollow-atom formation on coherent x-ray scattering at high intensity, *Phys. Rev. A* **83**, 033402 (2011). doi:10.1103/PhysRevA.83.033402.

[17] K. Nass, Radiation damage in protein crystallography at x-ray free-electron lasers, *Acta Cryst. D* **75**, 211–218 (2019). doi:10.1107/S2059798319000317.

[18] R. Neutze, R. Wouts, D. van der Spoel, E. Weckert, and J. Hajdu, Potential for biomolecular imaging with femtosecond x-ray pulses, *Nature (London)* **406**, 752–757 (2000). doi:10.1038/35021099.

[19] R. Santra and L. Young, in *Synchrotron Light Sources and Free-Electron Lasers*, edited by E. J. Jaeschke, S. Khan, J. R. Schneider, and J. B. Hastings (Springer International Publishing, Switzerland, 2016), pp. 1233–1260.

[20] L. Young, E. P. Kanter, B. Krässig, Y. Li, A. M. March, S. T. Pratt, R. Santra, S. H. Southworth, N. Rohringer, L. F. DiMauro, et al., Femtosecond electronic response of atoms to ultra-intense x-rays, *Nature (London)* **466**, 56–61 (2010). doi:10.1038/nature09177.

[21] H. Fukuzawa, S.-K. Son, K. Motomura, S. Mondal, K. Nagaya, S. Wada, X.-J. Liu, R. Feifel, T. Tachibana, Y. Ito, et al., Deep inner-shell multiphoton ionization by intense x-ray free-electron laser pulses, *Phys. Rev. Lett.* **110**, 173005 (2013). doi:10.1103/PhysRevLett.110.173005.

[22] B. Rudek, K. Toyota, L. Foucar, B. Erk, R. Boll, C. Bomme, J. Correa, S. Carron, S. Boutet, G. J. Williams, et al., Relativistic and resonant effects in the ionization of heavy atoms by ultra-intense hard x-rays, *Nat. Commun.* **9**, 4200 (2018). doi:10.1038/s41467-018-06745-6.

[23] B. Rudek, S.-K. Son, L. Foucar, S. W. Epp, B. Erk, R. Hartmann, M. Adolph, R. Andritschke, A. Aquila, N. Berrah, et al., Ultra-efficient ionization of heavy atoms by intense x-ray free-electron laser pulses, *Nat. Photon.* **6**, 858–865 (2012). doi:10.1038/nphoton.2012.261.

[24] A. Rudenko, L. Inhester, K. Hanasaki, X. Li, S. J. Robatjazi, B. Erk, R. Boll, K. Toyota, Y. Hao, O. Vendrell, et al., Femtosecond response of polyatomic molecules to ultra-intense hard x-rays, *Nature (London)* **546**, 129–132 (2017). doi:10.1038/nature22373.

[25] N. Rohringer and R. Santra, X-ray nonlinear optical processes using a self-amplified spontaneous emission free-electron laser, *Phys. Rev. A* **76**, 033416 (2007). doi:10.1103/PhysRevA.76.033461.

[26] M. G. Makris, P. Lambropoulos, and A. Mihelič, Theory of multiphoton multielectron ionization of xenon under strong 93-eV radiation, *Phys. Rev. Lett.* **102**, 033002 (2009). doi:10.1103/PhysRevLett.102.033002.

[27] G. Doumy, C. Roedig, S.-K. Son, C. I. Blaga, A. D. DiChiara, R. Santra, N. Berrah, C. Bostedt, J. D. Bozek, P. H. Bucksbaum, et al., Nonlinear Atomic Response to Intense Ultrashort X Rays, *Phys. Rev. Lett.* **106**, 083002 (2011). doi:10.1103/PhysRevLett.106.083002.

[28] W. Xiang, C. Gao, Y. Fu, J. Zeng, and J. Yuan, Inner-shell resonant absorption effects on evolution dynamics of the charge state distribution in a neon atom interacting with ultraintense x-ray pulses, *Phys. Rev. A* **86**, 061401(R) (2012). doi:10.1103/PhysRevA.86.061401.

[29] B. Rudek, D. Rolles, S.-K. Son, L. Foucar, B. Erk, S. Epp, R. Boll, D. Anielski, C. Bostedt, S. Schorb, et al., Resonance-enhanced multiple ionization of krypton at an x-ray free-electron laser, *Phys. Rev. A* **87**, 023413 (2013). doi:10.1103/PhysRevA.87.023413.

[30] K. Motomura, H. Fukuzawa, S.-K. Son, S. Mondal, T. Tachibana, Y. Ito, M. Kimura, K. Nagaya, T. Sakai, K. Matsunami, et al., Sequential multiphoton multiple ionization of atomic argon and xenon irradiated by x-ray free-electron laser pulses from SACLA, *J. Phys. B: At. Mol. Opt. Phys.* **46**, 164024 (2013). doi:10.1088/0953-4075/46/16/164024.

[31] N. Berrah, A. Sanchez-Gonzalez, Z. Jurek, R. Obaid, H. Xiong, R. J. Squibb, T. Osipov, A. Lutman, L. Fang, T. Barillot, et al., Femtosecond-resolved observation of the fragmentation of buckminsterfullerene following x-ray multiphoton ionization, *Nat. Phys.* **15**, 1279–1283 (2019). doi:10.1038/s41567-019-0665-7.

[32] B. F. Murphy, T. Osipov, Z. Jurek, L. Fang, S.-K. Son, M. Mucke, J. H. D. Eland, V. Zhaunerchyk, R. Feifel, L. Avaldi, et al., Femtosecond x-ray-induced explosion of C_{60} at extreme intensity, *Nat. Commun.* **5**, 4281

(2014). doi:10.1038/ncomms5281.

[33] M. Hoener, L. Fang, O. Kornilov, O. Gessner, S. T. Pratt, M. Gühr, E. P. Kanter, C. Blaga, C. Bostedt, J. D. Bozek, et al., Ultraintense x-ray induced ionization, dissociation, and frustrated absorption in molecular nitrogen, *Phys. Rev. Lett.* **104**, 253002 (2010). doi: 10.1103/PhysRevLett.104.253002.

[34] K. Toyota, S.-K. Son, and R. Santra, Interplay between relativistic energy corrections and resonant excitations in x-ray multiphoton ionization dynamics of Xe atoms, *Phys. Rev. A* **95**, 043412 (2017). doi: 10.1103/PhysRevA.95.043412.

[35] S.-K. Son, R. Boll, and R. Santra, Breakdown of frustrated absorption in x-ray sequential multiphoton ionization, *Phys. Rev. Research* **2**, 023053 (2020). doi: 10.1103/PhysRevResearch.2.023053.

[36] P. J. Ho, E. P. Kanter, and L. Young, Resonance-mediated atomic ionization dynamics induced by ultraintense x-ray pulses, *Phys. Rev. A* **92**, 063430 (2015). doi:10.1103/PhysRevA.92.063430.

[37] P. J. Ho, C. Bostedt, S. Schorb, and L. Young, Theoretical tracking of resonance-enhanced multiple ionization pathways in x-ray free-electron laser pulses, *Phys. Rev. Lett.* **113**, 253001 (2014). doi: 10.1103/PhysRevLett.113.253001.

[38] S.-K. Son and R. Santra, Monte Carlo calculation of ion, electron, and photon spectra of xenon atoms in x-ray free-electron laser pulses, *Phys. Rev. A* **85**, 063415 (2012). doi:10.1103/PhysRevA.85.063415.

[39] J. M. Schäfer, L. Inhester, S.-K. Son, R. F. Fink, and R. Santra, Electron and fluorescence spectra of a water molecule irradiated by an x-ray free-electron laser pulse, *Phys. Rev. A* **97**, 053415 (2018). doi: 10.1103/PhysRevA.97.053415.

[40] C. Buth, R. Beerwerth, R. Obaid, N. Berrah, L. S. Cederbaum, and S. Fritzsche, Neon in ultrashort and intense x-rays from free electron lasers, *J. Phys. B: At. Mol. Opt. Phys.* **51**, 055602 (2018). doi:10.1088/1361-6455/aaa39a.

[41] O. Ciricosta, H.-K. Chung, R. W. Lee, and J. S. Wark, Simulations of neon irradiated by intense x-ray laser radiation, *High Energy Density Phys.* **7**, 111–116 (2011). doi:10.1016/j.hedp.2011.02.003.

[42] C. Buth, J.-C. Liu, M. H. Chen, J. P. Cryan, L. Fang, J. M. Glownia, M. Hoener, R. N. Coffee, and N. Berrah, Ultrafast absorption of intense x rays by nitrogen molecules, *J. Chem. Phys.* **136**, 214310 (2012). doi: 10.1063/1.4722756.

[43] J.-C. Liu, N. Berrah, L. S. Cederbaum, J. P. Cryan, J. M. Glownia, K. J. Schafer, and C. Buth, Rate equations for nitrogen molecules in ultrashort and intense x-ray pulses, *J. Phys. B: At. Mol. Opt. Phys.* **49**, 075602 (2016). doi:10.1088/0953-4075/49/7/075602.

[44] L. Budewig, S.-K. Son, and R. Santra, State-resolved ionization dynamics of a neon atom induced by x-ray free-electron-laser pulses, *Phys. Rev. A* **107**, 013102 (2023). doi:10.1103/PhysRevA.107.013102.

[45] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, *Rev. Mod. Phys.* **91**, 045002 (2019). doi: 10.1103/RevModPhys.91.045002.

[46] A. Karthikeyan and U. D. Priyakumar, Artificial intelligence: machine learning for chemical sciences, *J. Chem. Sci.* **134**, 2 (2021). doi:10.1007/s12039-021-01995-2.

[47] O. A. von Lilienfeld and K. Burke, Retrospective on a decade of machine learning for chemical discovery, *Nat. Commun.* **11**, 4895 (2020). doi:10.1038/s41467-020-18556-9.

[48] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al., Highly accurate protein structure prediction with AlphaFold, *Nature (London)* **596**, 583–589 (2021). doi:10.1038/s41586-021-03819-2.

[49] C. D. Rankine, M. M. M. Madkhali, and T. J. Penfold, A deep neural network for the rapid prediction of x-ray absorption spectra., *J. Phys. Chem. A* **124**, 4263–4270 (2020).

[50] M. R. Carbone, M. Topsakal, D. Lu, and S. Yoo, Machine-learning x-ray absorption spectra to quantitative accuracy, *Phys. Rev. Lett.* **124**, 156401 (2020). doi:10.1103/PhysRevLett.124.156401.

[51] D. Golze, M. Hirvensalo, P. Hernández-León, A. Aarva, J. Etula, T. Susi, P. Rinke, T. Laurila, and M. A. Caro, Accurate computational prediction of core-electron binding energies in carbon-based materials: A machine-learning model combining density-functional theory and gw, *Chem. Mater.* **34**, 6240–6254 (2022). doi: 10.1021/acs.chemmater.1c04279.

[52] K. Ghosh, A. Stuke, M. Todorović, P. B. Jørgensen, M. N. Schmidt, A. Vehtari, and P. Rinke, Deep learning spectroscopy: Neural networks for molecular excitation spectra, *Adv. Sci.* **6**, 1801367 (2019). doi: https://doi.org/10.1002/advs.201801367.

[53] J. Carrasquilla and R. G. Melko, Machine learning phases of matter, *Nat. Phys.* **13**, 431–434 (2017). doi: 10.1038/nphys4035.

[54] A. Sanchez-Gonzalez, P. Micaelli, C. Olivier, T. R. Barillot, M. Ilchen, A. A. Lutman, A. Marinelli, T. Maxwell, A. Achner, M. Agåker, et al., Accurate prediction of x-ray pulse properties from a free-electron laser using machine learning, *Nat. Commun.* **8**, 15461 (2017). doi: 10.1038/ncomms15461.

[55] O. Geffert, D. Kolbasova, A. Trabattoni, F. Calegari, and R. Santra, In situ characterization of few-femtosecond laser pulses by learning from first-principles calculations, *Opt. Lett.* **47**, 3992–3995 (2022). doi:10.1364/OL.460513.

[56] D. Kolbasova and R. Santra, Laser-pulse characterization using strong-field autocorrelation patterns and random-forest-based machine learning, *Phys. Rev. A* **107**, 013520 (2023). doi:10.1103/PhysRevA.107.013520.

[57] N. Breckwoldt, S.-K. Son, T. Mazza, A. Rörig, R. Boll, M. Meyer, A. C. LaForge, D. Mishra, N. Berrah, and R. Santra, Machine-learning calibration of intense x-ray free-electron-laser pulses using bayesian optimization, *Phys. Rev. Res.* **5**, 023114 (2023). doi: 10.1103/PhysRevResearch.5.023114.

[58] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, Machine learning of molecular electronic properties in chemical compound space, *New J. Phys.* **15**, 095003 (2013). doi:10.1088/1367-2630/15/9/095003.

[59] J. Westermayr, M. Gastegger, K. T. Schütt, and R. J. Maurer, Perspective on integrating machine learning into computational chemistry and materials science, *J. Chem. Phys.* **154**, 230903 (2021). doi:

10.1063/5.0047760.
[60] O. A. von Lilienfeld, K.-R. Müller, and A. Tkatchenko, Exploring chemical compound space with quantum-based machine learning, *Nat. Rev. Chem.* **4**, 347–358 (2020). doi:10.1038/s41570-020-0189-9.
[61] L. Fiedler, N. A. Modine, S. Schmerler, D. J. Vogel, G. A. Popoola, A. P. Thompson, S. Rajamanickam, and A. Cangi, Predicting electronic structures at any length scale with machine learning, *npj Comput. Mater.* **9**, 115 (2023). doi:10.1038/s41524-023-01070-z.
[62] L. Fiedler, K. Shah, M. Bussmann, and A. Cangi, Deep dive into machine learning density functional theory for materials science and chemistry, *Phys. Rev. Mater.* **6**, 040301 (2022). doi: 10.1103/PhysRevMaterials.6.040301.
[63] A. Stuke, M. Todorović, M. Rupp, C. Kunkel, K. Ghosh, L. Himanen, and P. Rinke, Chemical diversity in molecular orbital energy predictions with kernel ridge regression., *J. Chem. Phys.* **150**, 204121 (2019). doi: 10.1063/1.5086105.
[64] T. B. Blank, S. D. Brown, A. W. Calhoun, and D. J. Doren, Neural network models of potential energy surfaces, *J. Chem. Phys.* **103**, 4129–4137 (1995). doi: 10.1063/1.469597.
[65] J. Behler and M. Parrinello, Generalized neural-network representation of high-dimensional potential-energy surfaces, *Phys. Rev. Lett.* **98**, 146401 (2007). doi: 10.1103/PhysRevLett.98.146401.
[66] O. T. Unke, D. Koner, S. Patra, S. Käser, and M. Meuwly, High-dimensional potential energy surfaces for molecular simulations: from empiricism to machine learning, *Mach. learn.: sci. technol.* **1** (2019).
[67] G. Schmitz, I. H. Godtliebsen, and O. Christiansen, Machine learning for potential energy surfaces: An extensive database and assessment of methods., *J. Chem. Phys.* **150 24**, 244113 (2019). doi:10.1063/1.5100141.
[68] A. M. Tokita and J. Behler, How to train a neural network potential, *J. Chem. Phys.* **159**, 121501 (2023). doi: 10.1063/5.0160326.
[69] J. P. Coe, Machine learning configuration interaction., *J. Chem. Theory Comput.* **14**, 5739–5749 (2018). doi: 10.1021/acs.jctc.8b00849.
[70] P. Bilous, A. Pálffy, and F. Marquardt, Deep-learning approach for the atomic configuration interaction problem on large basis sets, *Phys. Rev. Lett.* **131**, 133002 (2023). doi:10.1103/PhysRevLett.131.133002.
[71] L. Budewig, S.-K. Son, and R. Santra, Theoretical investigation of orbital alignment of x-ray-ionized atoms in exotic electronic configurations, *Phys. Rev. A* **105**, 033111 (2022). doi:10.1103/PhysRevA.105.033111.
[72] Z. Jurek, S.-K. Son, B. Ziaja, and R. Santra, XMDYN and XATOM: Versatile simulation tools for quantitative modeling of X-ray free-electron laser induced dynamics of matter, *J. Appl. Cryst.* **49**, 1048–1056 (2016). doi: 10.1107/S1600576716006014.
[73] S.-K. Son, K. Toyota, O. Geffert, J. M. Slowik, and R. Santra, XATOM-*an integrated toolkit for x-ray and atomic physics*, CFEL, DESY, Hamburg, Germany, 2016, Rev. 3544.
[74] M. C. Martins, A. M. Costa, J. P. Santos, F. Parente, and P. Indelicato, Relativistic calculation of two-electron one-photon and hypersatellite transition energies, *J. Phys. B: At. Mol. Opt. Phys.* **37**, 3785 (2004). doi:10.1088/0953-4075/37/19/001.
[75] C. Briançon and J. P. Desclaux, Relativistic dirac-fock calculations of KLL auger transition energies in intermediate coupling, *Phys. Rev. A* **13**, 2157–2162 (1976). doi:10.1103/PhysRevA.13.2157.
[76] R. Püttner, Y. Li, J. Zeng, D. Koulentianos, T. Marchenko, R. Guillemin, L. Journel, O. Travnikova, M. Zmerli, D. Céolin, et al., Argon $1s^{-2}$ auger hypersatellites, *J. Phys. B: At. Mol. Opt. Phys.* **54**, 024001 (2020). doi:10.1088/1361-6455/abcd23.
[77] R. D. Deslattes, E. G. Kessler, P. Indelicato, L. de Billy, E. Lindroth, and J. Anton, X-ray transition energies: new approach to a comprehensive evaluation, *Rev. Mod. Phys.* **75**, 35–99 (2003). doi: 10.1103/RevModPhys.75.35.
[78] E. Mikkola and J. Ahopelto, $K\alpha^h$ hypersatellite spectrum and K shell double photoionization cross-section for Ar, *Phys. Scr.* **27**, 297 (1983). doi:10.1088/0031-8949/27/4/011.
[79] M. O. Krause, Argon KLL auger spectrum: A test of theory, *Phys. Rev. Lett.* **34**, 633–635 (1975). doi: 10.1103/PhysRevLett.34.633.
[80] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).
[81] Y. LeCun, L. Bottou, G. Orr, and K. Muller, Efficient backprop, in *Neural Networks: Tricks of the trade*, edited by G. Orr and G. B. Müller (Springer, Berlin, Heidelberg, 1998).
[82] C. M. Bishop, Neural networks for pattern recognition (Oxford University Press, New York, 1995).
[83] L. Long and X. Zeng, *Beginning Deep Learning with TensorFlow* (Apress, Berkeley, CA, 2022).
[84] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, *Nature (London)* **323**, 533–536 (1986). doi: 10.1038/323533a0.
[85] S. Liang and R. Srikant, Why deep neural networks for function approximation?, in *5th International Conference on Learning Representations ICLR*, edited by Y. Bengio and Y. LeCun (OpenReview.net, Toulon, 2017).
[86] F. Chollet et al., *Keras*, `https://keras.io` (2015).
[87] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., Tensorflow: A system for large-scale machine learning, in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation* (USENIX Association, USA, 2016), pp. 265–283.
[88] M. Belkin, D. Hsu, S. Ma, and S. Mandal, Reconciling modern machine-learning practice and the classical bias–variance trade-off, *Proc. Natl. Acad. Sci. U.S.A.* **116**, 15849–15854 (2019). doi: 10.1073/pnas.1903070116.
[89] S. Sharma, S. Sharma, and A. Athaiya, Activation functions in neural networks, *International Journal of Engineering Applied Sciences and Technology* **4**, 310–316 (2020).
[90] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, edited by Y. W. Teh and M. Titterington (PMLR, Italy, 2010), pp. 249–256.
[91] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: A simple way to pre-

vent neural networks from overfitting, *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).

[92] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, in *3rd International Conference on Learning Representations ICLR*, edited by Y. Bengio and Y. LeCun (arxiv.org, San Diego, 2015).

[93] L. Breiman, Random forests, *Mach. Learn.* **45**, 5–32 (2001). doi:10.1023/A:1010933404324.

[94] P. Roßbach, Neural networks vs. random forests – does it always have to be deep learning? (Frankfurt School of Finance and Managment, 2018).

[95] Z.-H. Zhou and J. Feng, Deep forest: Towards an alternative to deep neural networks, in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, edited by C. Sierra (International Joint Conferences on Artificial Intelligence, Melbourne, 2017).

[96] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).

[97] A. A. Patel, *Hands-On Unsupervised Learning Using Python* (O'Reilly Media, Inc., Sebastopol, 2019).

[98] J. H. Friedman, Greedy function approximation: A gradient boosting machine., *Ann. Stat.* **29**, 1189–1232 (2001). doi:10.1214/AOS/1013203451.

[99] G. Biau, E. Scornet, and J. Welbl, Neural random forests, *Sankhya A* **81**, 347–386 (2019). doi:10.1007/s13171-018-0133-y.

[100] C. Qu, P. L. Houston, Q. Yu, R. Conte, P. Pandey, A. Nandi, and J. M. Bowman, Machine learning classification can significantly reduce the cost of calculating the Hamiltonian matrix in CI calculations, *J. Chem. Phys.* **159**, 071101 (2023). doi:10.1063/5.0168590.