Optimizing training trajectories in variational autoencoders via latent Bayesian optimization approach*

IOP Publishing Mach. Learn.: Sci. Technol. 4 (2023) 015011 https://doi.org/10.1088/2632-2153/acb316

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
7 July 2022

REVISED
24 December 2022

ACCEPTED FOR PUBLICATION
13 January 2023

PUBLISHED
1 February 2023

PAPER

Optimizing training trajectories in variational autoencoders via latent Bayesian optimization approach*

Arpan Biswas[1,**], Rama Vasudevan[1], Maxim Ziatdinov[1,2] and Sergei V Kalinin[3,**]

[1] Center for Nanophase Materials Sciences, Oak Ridge National Laboratory, Oak Ridge, TN 37831, United States of America
[2] Computational Sciences and Engineering Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, United States of America
[3] Materials Science and Engineering, University of Tennessee, Knoxville, TN 37996, United States of America
**Authors to whom any correspondence should be addressed.

E-mail: biswasa@ornl.gov and sergei2@utk.edu

Keywords: high-dimensional problem, Bayesian optimization, latent space, variational auto-encoder, unsupervised learning

Supplementary material for this article is available online

Abstract

Unsupervised and semi-supervised ML methods such as variational autoencoders (VAE) have become widely adopted across multiple areas of physics, chemistry, and materials sciences due to their capability in disentangling representations and ability to find latent manifolds for classification and/or regression of complex experimental data. Like other ML problems, VAEs require hyperparameter tuning, e.g. balancing the Kullback–Leibler and reconstruction terms. However, the training process and resulting manifold topology and connectivity depend not only on hyperparameters, but also their evolution during training. Because of the inefficiency of exhaustive search in a high-dimensional hyperparameter space for the expensive-to-train models, here we have explored a latent Bayesian optimization (zBO) approach for the hyperparameter trajectory optimization for the unsupervised and semi-supervised ML and demonstrated for joint-VAE with rotational invariances. We have demonstrated an application of this method for finding joint discrete and continuous rotationally invariant representations for modified national institute of standards and technology database (MNIST) and experimental data of a plasmonic nanoparticles material system. The performance of the proposed approach has been discussed extensively, where it allows for any high dimensional hyperparameter trajectory optimization of other ML models.

1. Introduction

Unsupervised and semi-supervised ML methods have become the mainstay of multiple domain areas ranging from machine vision to physics and astronomy due to their capability to disentangle representation and find latent manifolds for classification and/or regression tasks on complex raw data [1–3]. For sufficiently simple systems, the disentangled representations can often be associated with the specific physical factors of variability in the system. In particular, unsupervised ML approaches have allowed the discovery of physics from complex and/or large microscopic images/datasets as in [4–12], where the disentangled representations provide insight into specific physical order parameters.

As is common for unsupervised ML, the training of the model is sensitively dependent on the choice of hyperparameters. Generally, a hyperparameter is a parameter which controls the learning process of the ML models, and the hyperparameter tuning (or optimization) is the problem of choosing a set of optimal values

for those hyperparameters to optimize the learning process. Extensive effort has been dedicated towards optimal tuning of ML models, with different optimization techniques such as gradient based method, genetic algorithm (GA), Bayesian optimization (BO) etc [13–19]. In the process of tuning ML models where the training cost is computationally high, any exhaustive or manual parameter space search is a highly non-desirable approach. In such cases, BO is better suited than other optimization techniques due to the inbuild adaptive sampling towards maximizing the learning of region of interest within the parameter space while minimizing the function evaluation cost or training cost of expensive ML models. This approach has been widely used in these machine learning problems [20–24]. However, the downside of standard BO is the convergence issue when the control parameter is high dimensional (dimension $\geqslant$ 15–20) [25], resulting in a sub or non-optimal hyperparameter tuning of ML models. This again, will lead to improper ML training which ultimately results in poor physical insights from data. Additionally, the increase in the dimensionality of the control parameters decreases the rate (more function evaluations) of BO convergence, thus increasing the computational cost exponentially. This decreases the general applicability of using BO in optimizing an expensive ML model. Methods have been attempted to tackle BO in high dimensional problems through a different strategy of projection with random embedding and quantile Gaussian Process [26, 27] to a reduced space, or using special kernels [28]. However, performance of the method in [26] depends on the problem and the importance of parameters in the high-dimensional space whereas the method in [27] lacks computational efficiency. The method described in [28] builds on the technique on providing special attention to avoid excessive sampling over boundary region with a cylindrical kernel.

Another popular projection strategy of high dimensional real space to a reduced latent space, is to use a variational autoencoder (VAE) model. Applying a BO incorporate the maximization of learning into the reduced latent dimension, thereby lowering computational cost of expensive function evaluations and the risk of non-convergence of BO. Previously, similar approaches were attempted to solve for high dimensional and/or discrete input space as in [29–35], in the field of chemistry [36] and medicines [37].

In different experimental analysis in the field of material science, we often encounter a large untidy dataset, which needs further analysis through ML models to interpret useful physical information. Therefore, the large-scale data labeling process becomes challenging and infeasible. Therefore, in this paper, we focused on an unsupervised ML tool, a joint rotationally invariant variational autoencoder (jrVAE) model [38]. However, depending on the complexity of the data, the performance of the training of jrVAE needs proper tuning, which also includes high-dimensional training-dependent hyperparameter trajectory functions. As per the research contribution, we adapted the standard Latent Bayesian optimization (zBO) workflow and extended the application to the high-dimensional continuous iterative dependent hyperparameter trajectory optimization of the expensive, unsupervised joint rotationally invariant jrVAE model where a trade-off between learning and function evaluation cost is critical without any prior knowledge from labeled data, allowing for multiple independent high-dimensional input (trajectory function) space in a common reduced latent space. This can be easily extended to any iteration-dependent high dimensional hyperparameter (or parameter) trajectory optimization of other expensive ML (or black box) models. The overall integrated zBO-jrVAE framework is demonstrated on Modified National Institute of Standards and Technology database (MNIST) test problem and plasmonic nanoparticles material systems containing dataset of correlated scattering spectra of gold particles and SEM images.

The outline of this paper is as follows. Section 2 describes the general jrVAE and BO methods, and finally the proposed zBO algorithm, integrated to tune jrVAE model with a demonstration of the workflow on MNIST data. Section 3 demonstrates the application of zBO-jrVAE workflow plasmonic nanoparticles experimental dataset. Section 4 concludes the paper with final thoughts and potential future directions.

2. Methodology

In this section, we start with discussing on the key components focused for this paper, namely, VAE, jrVAE and BO. Finally, we discuss the adaptation of the Latent BO (zBO) worflow implementation, and its integration with the autoencoder model.

2.1. jrVAE

A VAE [39] is a deep generative probabilistic model that belongs to the family of probabilistic graphical models and variational Bayesian methods. The VAE is comprised of the encoder and decoder, as shown in figure 1. Given the input x, the encoder transforms it into a reduced latent space as a distribution, $p(z|x)$. Then, given any sample from the latent space z, the decoder reconstructs the input as $p(\bar{x}|z)$. The overall goal is to optimize the encoding-decoding process jointly to minimize the reconstruction error and the Kullback–Leibler (KL) divergence, to ensure the best representation of the latent space and maximize the restoration of the features in the input data. Here, the reconstruction error can be chosen as mean square

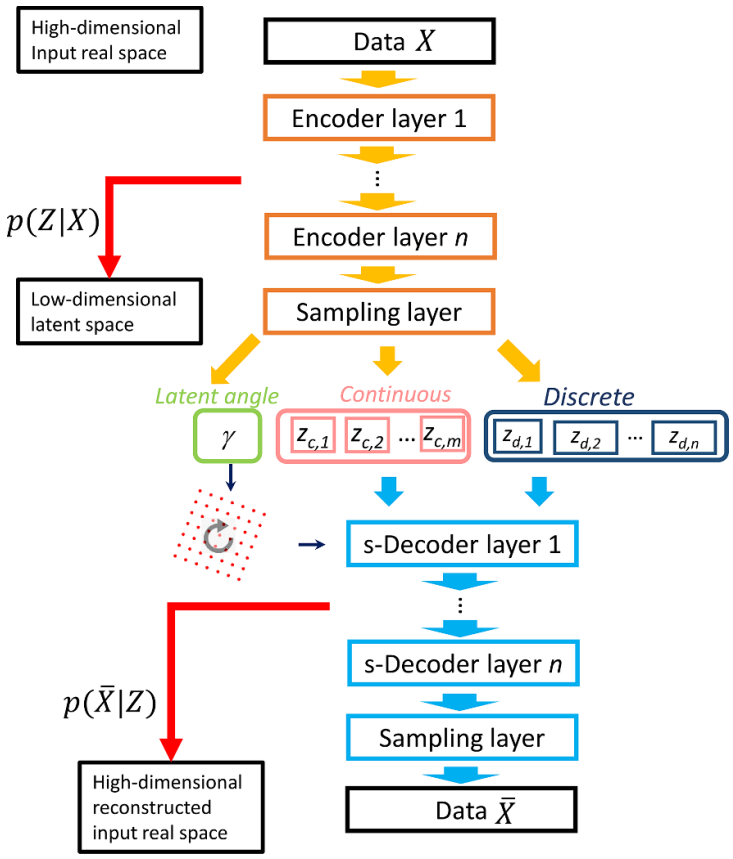

Figure 1. Joint rotational variational autoencoder (jrVAE) framework.

error, the cross-entropy error, etc. The KL divergence [40, 41] $D_{\mathrm{KL}}(p(z|x)||p(z))$ is the distance loss between the prior $p(z)$ distribution (usually chosen as standard Gaussian) and the posterior $p(z|x)$ distributions of the latent representation from data. Different VAE models have been applied to materials systems, in attempt to learn from complex image data [11, 38, 42–44].

Here, we considered a specific example of jrVAE, as demonstrated in figure 1, as implemented in pyroVED package in Python [45]. Here, the goal is to divide and to learn the latent space into both continuous $p(z_c|x)$ and discrete $p(z_d|x)$ latent representation from the data, while enforcing rotational invariances. The loss function, $\mathcal{L}$, can be mathematically represented as follows:

$$\mathcal{L} = \varphi + \boldsymbol{\beta_c}(i)\, D_{KL}(p(z_C|x)||p(z_C)) + \beta_d(i)\, D_{\mathrm{KL}}(p(z_d|x)||p(z_d)) \tag{1}$$

where φ is the reconstruction error, $D_{\mathrm{KL}}(p(.|x)||p(.))$ is the KL divergence, $\boldsymbol{\beta_c}(i)$ and $\beta_d(i)$ are the continuous and discrete scale factors of KL divergence respectively at i^{th} training cycle of jrVAE. The scale factors encourage a better disentanglement of the data, thus provides better learning [46]. However, as from equation (1), these are N-dimensional hyperparameters, where $d = e$, i.e. the dimension increases with the increase of training iteration e. Our objective in this paper is to tackle these high dimensional scale factors and build an optimization framework, balancing the accuracy in learning the optimal tuning and the computational cost of exploration.

2.2. BO

BO [47], has been originally developed as a low computationally cost global optimization tool for design problems having expensive black-box objective functions. Here, as shown in figure 2, the BO replicates the expensive functions with a cheap surrogate model and then utilizes an adaptive sampling technique through maximizing an acquisition function to learn or update the knowledge of the parameter space towards finding the optimal region.

Though the application of BO is focused on problems with continuous response functions, attempts have been made when the response is discontinuous [48] or discrete such as in consumer modeling problems where the responses are in terms of user preference [47, 49]. Here, the user preference discrete response function is transformed into continuous latent functions using Binomial-Probit model for binary choices

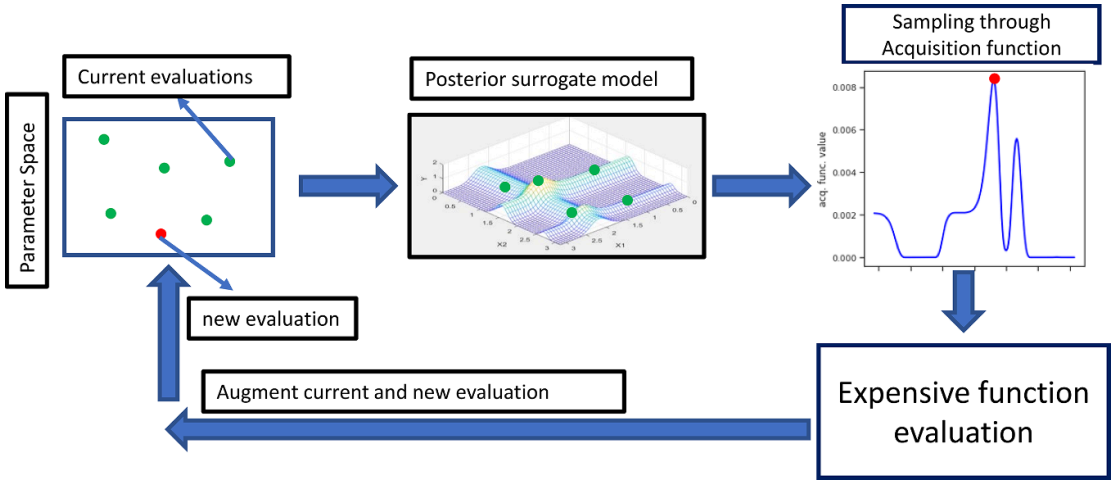

Figure 2. Bayesian optimization (BO) framework.

[50, 51] and polychotomous regression model is used for more than two choices where the user can state no preference [52]. However, the vast majority of these applications address for low-dimensional parameter optimization, especially when the function evaluations are expensive. Here, we aim to introduce a direction for BO application towards high-dimensional optimization problems with expensive evaluations.

A Gaussian Process Model (GPM) is generally integrated in BO as the cheap surrogate model. However, random forest regression has been proposed as an expressive and flexible surrogate model in the context of sequential model-based algorithm configuration [53]. Although random forests are good interpolators in the sense that they output good predictions in the neighborhood of training data, they are very poor extrapolators [54]. This can lead to selecting redundant exploration (more experiments) in the non-interesting region as suggested by the acquisition function in the early iterations of the optimization, due to having additional prediction error of the region far away from the training data. This motivates us to consider the GPM in a Bayesian framework while extending the application to high dimensional optimization problems.

Figures 3(a)–(d) shows a simple 1D Gaussian Process Model with one control parameter x and one objective function variable $z = f(x)$, based on the evaluated locations (green dots). GPM then fits the function with the posterior mean and variance as shown by grey dotted line and the area within the black lines where the variance near the evaluated region is small and increases as the design samples are farther away from the data. Much research has been ongoing regarding incorporating and quantifying uncertainty of the experimental or training data by using a nugget term in the predictor GPM. It has been found that the nugget provides better solution and computational stability framework [56, 57]. Furthermore, GPM has also been attempted in high dimensional design space exploration [58] and big data problems [59], as an attempt to increase computational efficiency. A survey of implementation of different GP packages has been provided in different coding languages such as MATLAB, R, and Python [60]. The detailed mathematical representation of GPM is provided in supplementary material (appendix A).

Once a cheap surrogate model is fitted in a BO iteration with the sampled data, the next task is to find the best locations for sampling in the next iteration through defining and maximizing the acquisition function (AF). Figures 3(a)–(d) shows a simple example of BO exploration with one control parameter x and one objective function variable $z = f(x)$ of the sequential selection of samples by maximizing the acquisition function, given posterior GP model in iterations 1, 2, 4 and 50. Here for each iteration, the red dot (top figures) is the new suggested location, as per maximizing the acquisition function (bottom figures). Thus, with iterative learning, the search space is explored towards finding the optimum. We can see from the figure that the acquisition function value is highest where the samples have high prediction mean and/or high variance and the lowest where the samples have low prediction, low variance or both. The acquisition function can be defined with different measure of trade-off between exploration and exploitation of the search space. One such method is the Probability of Improvement, PI [55] which is improvement-based acquisition function. However, Jones [61] highlighted that the limitations of the performance of PI(·) acquisition function towards efficient balance between exploration and exploitation. As alternative, the Expected Improvement (EI) acquisition function, EI [47, 62], is widely used over PI which generally provides a good measure of trade-off between exploration and exploitation. Another acquisition function is the Confidence Bound criteria, CB, introduced by Cox and John [63], where the selection of points is based on

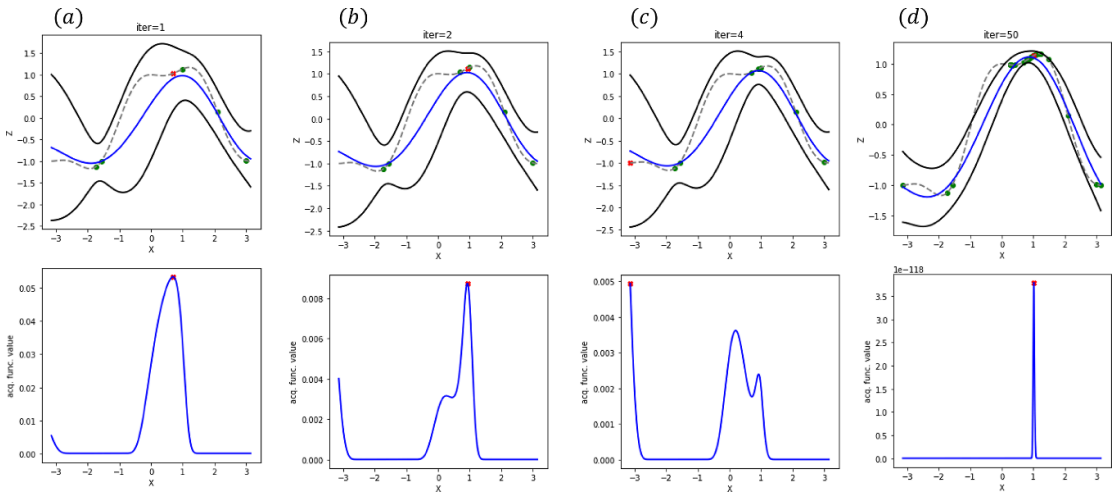

Figure 3. 1D Gaussian process and search space exploration by maximizing acquisition function. The top images of (a)–(d) are the Gaussian process illustration. Gray dotted lines are the true function value, blue solid lines are the GP predicted mean, black solid lines are the GP mean $\pm$ 2$\times$ standard dev. Green dots are the current evaluated locations. Red dots are the new locations for next evaluations. The bottom images of (a)–(d) are the acquisition function illustration. Blue solid lines are acquisition function. For this theoretical example, we use Probability of Improvement (PI) [55] acquisition function. Red dots are the new locations (at max. acq. func. value) for next evaluations.

the upper or lower confidence level of the predicted design surface for maximization or minimization problem respectively.

2.3. Latent Bayesian optimization (zBO): integrated to jrVAE

As our objective mentioned in section 2.1, to optimize the high dimensional KL factors or KL trajectory, we considered BO technique due to adaptive sampling (from acquisition function) for fast learning to find the region of interest in the parameter space. However, the standard BO (section 2.2) is not well suited to tackle high-dimensional parameter optimization. Hence, here we illustrate the modification to build a Latent Bayesian Optimization (zBO). Figure 4 shows the overall workflow of zBO, integrated to jrVAE model. Table 1 shows the detailed algorithm of the workflow. However, the zBO framework is a standalone application, which can be easily coupled with other ML or mathematical models, which requires high dimensional (hyper)parameter optimization.

We further elaborate Step 6 of the stated Algorithm (table 1) and describe the workflow of the objective function considered in this paper. However, to generalize, the proposed approach can be easily coupled with any objective functions as required for a given problem. In this paper, our general task is to undergo different multi-label classification problems through unsupervised learning techniques of jrVAE model, and the attempt for better quality of solution from optimal tuning of scale ($\boldsymbol{\beta}$) trajectories through zBO framework. In the experimental data, often we do not possess any prior knowledge of discrete classes for supervised or semi-supervised learning, thus, we have focused on the unsupervised ML learning as the problem domain in this paper. Table 2 provides the workflow of the objective function evaluation, which aids the task of better separation of different classes from the data.

3. Results: benchmark problem

To demonstrate the proposed workflow, we first considered a test MNIST dataset [64], which is a large database of handwritten digits, commonly used for training various image processing systems. The database contains 60 000 training datasets, where each digit can be considered as a label for our multi-class classification problem. To note, the digits contain rotational variability, which is a target to tackle by jrVAE model, and with the hyperparameter tuning.

In this case study, we considered gaussian decoder sampler with sigma $= 0.3$, learning rate $= 1 \times 10^{-4}$ and training cycles $= 1000$ to initialize and train the VAE model for 2D latent representation of N-dimensional $\boldsymbol{\beta}$ trajectories (table 1, Step 2). For initializing the jrVAE model, we considered Bernoulli decoder sampler, learning rate $= 1 \times 10^{-3}$ and training cycle, $N = 120$. For the zBO, we started with 20 randomly selected samples with maximum of 120 BO iteration, thus a total of 140 function evaluations. We choose EI acquisition function. To simplify the problem, we considered to optimize the continuous scale factor, $\boldsymbol{\beta}_c$ trajectory only and set the discrete scale factor at constant setting as $\boldsymbol{\beta}_d(i) = 3; i = 1,\ 2, \ldots,\ N$

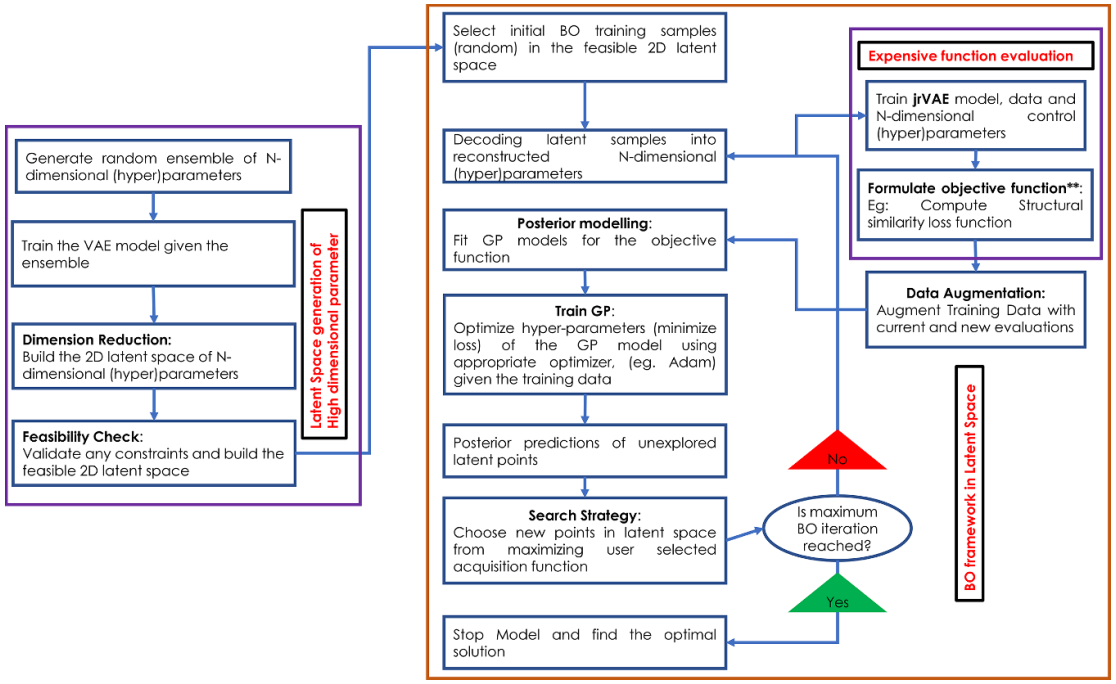

Figure 4. Latent Bayesian optimization (zBO) architecture: application to high-dimensional hyperparameter tuning of jrVAE model for performance enhancement.
** *Note:* The objective function formulation is based on the analysis on this paper and can be easily updated in the framework with different formulation as per the problem requirement.

Table 1. Algorithm: latent Bayesian optimization: to optimize high-dimensional hyperparameter tuning of jrVAE.

1. **Initialization of high-dimensional parameters**: Generate, either random or with some prior knowledge, a set of N-dimensional scale (β) trajectories. N represents the number of training cycles of jrVAE model.
2. **Training VAE:** Define and train a standard VAE model on the set of β trajectories. We assume here the VAE model is tuned properly for maximized learning.
3. **Dimension reduction of high-dimensional parameters:** With the trained VAE model (step 2), we build a 2D latent space of N-dimensional β trajectories.
4. **Feasibility Check:** Formulate all the physical constraints (if any). We validate the 2D latent space for any constraint's violations. In this case, $\beta(i) > 0; i = 1,\ 2, ..,\ N$. Finally, build the feasible 2D latent space.
5. **Initialization for BO:** State maximum BO iteration, M. Randomly select j samples from the feasible 2D latent space, $\boldsymbol{Z} = \{\boldsymbol{Z}_1,\ \boldsymbol{Z}_2\}$. Assuming f is the expensive objective function. Set $k = 1$.

For $k \leqslant M$

6. **Decoding into reconstructed high-dimensional parameters for expensive function evaluations**: Given the trained VAE model (Step 2), decode each latent sample $z = \{z_1,\ z_2\}; \boldsymbol{z} \in \boldsymbol{Z_k}$ into reconstructed N-dimensional sampled β trajectories, as $\bar{\boldsymbol{x}}|z = \{\bar{x}_1,\ \bar{x}_2, \ldots .\bar{x}_N|z_1, z_2\}; \bar{x} \in \overline{\boldsymbol{X}}_{\boldsymbol{k}}$. Evaluate j samples for objective as, $\boldsymbol{y_j}(\bar{\boldsymbol{x}}|\boldsymbol{z}); \boldsymbol{y} \in \boldsymbol{Y_k}$. The detailed formulation of the objective function in this case is provided later in the section. Build training data matrices (in 2D latent space), $\boldsymbol{D_k} = \{\boldsymbol{Z_k},\ \boldsymbol{Y_k}\}$.
7. **Surrogate Modeling**: Develop or update GPM models, given the training data, as $\boldsymbol{\Delta}(\boldsymbol{D_k})$.
   a. Optimize the hyper-parameters of GPM by minimizing the loss (negative marginal log-likelihood) function using Adam optimizer algorithm. Here, we consider learning rate 1×10^{-4}.
8. **Posterior Predictions**: Given the surrogate model, compute posterior means and variances for the unexplored locations, $\overline{\overline{\boldsymbol{Z_k}}}$, over the 2D latent space as $\boldsymbol{\pi}\left(\boldsymbol{Y}\left(\overline{\overline{\boldsymbol{Z_k}}}\right)|\boldsymbol{\Delta}\right.$ and $\boldsymbol{\sigma^2}\left(\boldsymbol{Y}\left(\overline{\overline{\boldsymbol{Z_k}}}\right)|\boldsymbol{\Delta}\right.$ respectively. It is to be noted that we directly compute the posterior predictions from the input 2D latent samples, given the GPM, without the need to decode to high dimensional parameter.
9. **Acquisition function:** Compute and maximize acquisition function, $\max_{z} U(f|\Delta)$ to select next best location in the 2D latent space, $\boldsymbol{z_{best,k}}$ for evaluations.
10. **Augmentation:** Following step 6, decode $z_{best,k}$ into $\bar{x}_{best,k}|z_{best,k}$, and evaluate the same as $y(\bar{x}_{best,k}|z_{best,k})$. Augment data, $D_{k+1} = [D_k; \{z_{best,k},\ y\}$.

where $N = 120$ in this case. As per pre-optimization analysis, we set the maximum and minimum bounds of $\beta_c(i)$, as $1 \leqslant \beta_c(i) \leqslant 50$. However, the proposed workflow can be easily extended to optimize both scale factors jointly, which is considered in future scope.

Table 2. Workflow: objective function evaluation (for problem of multi-label classification) as in step 6 and 10 of table 1.

1. **Train jrVAE model**: Initialize number of discrete classes, D_c. Given the data (simulated or experimental), and the decoded $\boldsymbol{\beta}$ trajectories, as $\bar{\boldsymbol{x}}|\boldsymbol{z}$, train jrVAE model for N training cycles. Compute the 2D trained manifold for each discrete classes, as in matrix $\boldsymbol{\Phi}$.
2. **Evaluate objective 1:** Choose learned manifolds $\boldsymbol{\Phi_i}$, $\boldsymbol{\Phi_j}$ for two discrete classes, $i,\ j; j > i; i, j = 1, 2, \ldots . D_C$. Calculate structural similarity (SSIM) loss function between them. Do the same between every discrete class. Compute the total loss as $\ell_1(\boldsymbol{\beta}|data) = \sum_{i,j;j>i}^{D_c} \ell_{i,j}$.
3. **Evaluate objective 2:** Choose learned manifold $\boldsymbol{\Phi_i}$ for a discrete class $j;\ j = 1, 2, \ldots . D_C$. Randomly choose unique I pairs, as $i_1, i_2; i_1 \neq i_2;\ i_1, i_2 = 1, 2, \ldots . I$, of image grid location within the learned manifold. Calculate structural similarity (SSIM) loss function between each pair. Compute the mean loss for learned manifolds of each discrete class as $\ell_j = \frac{\sum_{i_1, i_2; i_1 \neq i_2}^{I} \ell_{i_1, i_2}}{k}$ where k is the number of unique location pairs. Do the same for every discrete class. Compute the total loss as $\ell_2(\boldsymbol{\beta}|\text{data}) = \sum_{D_c}^{j} \ell_j$. We avoid computing for every possible combination of grid location for significant (exponential) increase in computational cost as the number of grid locations and the discrete classes increases.
4. **Formulate final objective function:** Here our goal is to maximize the SSIM loss of learned manifolds among different discrete classes, thus maximize objective 1, to minimize the SSIM loss within the images of the grid location of a learned manifold of a discrete class, thus minimize objective 2. Finally, to modify into a maximization problem (as per default setting of zBO), the objective function is stated as

$$\max_{\boldsymbol{\beta}} \ell_1(\boldsymbol{\beta}|\text{data}) - \ell_2(\boldsymbol{\beta}|\text{data}) \tag{2}$$

During the function evaluation process within BO, we considered a subset of data (randomly selected) to avoid redundant computational cost from training with a large dataset (as we need function evaluations multiple times). Since the dataset is balanced, it is easy to work with the subset of the data and the objective function should be able to still capture the necessary information from data to find the optimal region. Also, we avoided the loss function computation of table 2, Step 3 as we observe the other loss function is sufficient to achieve the optimal region for this case study, and therefore further reducing the computational expense. Thus, in this case study, we only consider the first part of equation (2). Once the zBO model is converged, we train the jrVAE model with full (large) dataset, stated settings, and optimal tuning of $\boldsymbol{\beta_c}$.

Figure 5 shows an example of the projection of N-dimensional search space to the feasible 2D latent search space, following the algorithm in table 1, Step 1–4. To increase the complexity, we defined the training trajectories from two different functions, as per domain expert knowledge: (a) linear cooldown and (b) randomly segmented with random noise (see figure 5(a)), assuming the optimal trajectory exists. We modified the training data at the same scale. However, it is to be noted, our zBO workflow can accompany any trajectory functions (as per the problem and domain expert knowledge) which is suitable for application to hyperparameter trajectory optimization to other ML models. With VAE training and constraint validation, we defined the feasible 2D latent space (figure 5(b)) where each latent samples can be reconstructed to a $\boldsymbol{\beta_c}$ trajectory (figures 5(c)–(e)). For the constraint validation, we simply eliminated any points priorly in the 2D latent space which decoded to an infeasible trajectory (in this case, as per table 1 step 4). Thus, we build a dataset of latent points with only feasible decoded trajectories as the set of potential design solutions for BO. The blue dots over the 2D latent space are the training data. We can clearly see the trained VAE model build clusters of two different patterns of trajectories (defined from separate functions) and the decoding provides the pattern of reconstructed trajectories (red dots) with a weighted information, depending on the distance from these clusters (comparing figures 5(c)–(e)). This shows the VAE model is well trained which not only restores sufficient knowledge of each defined trajectories separately (storing original patterns) but also provides reconstruction with mixing of both knowledge (introducing new hybrid patterns). This increases the possible set of new solutions for zBO without losing pre-considered solutions.

Figure 6 shows the result of zBO convergence, and the optimal $\boldsymbol{\beta_c}$ trajectory, whereas figure 7 shows the comparison among different VAE models, with and without the high-dimensional continuous scale factor tuning. It is clear that the Vanilla VAE model provides the worst result (see figure 7(b)), where it could not extract all the labels from data and has rotational variability in the learned manifolds. With the jrVAE model, but with default constant setting of $\boldsymbol{\beta_c}(i) = 1; i = 1,\ 2, \ldots ,\ 120$, though the model improved much and could be able to separate the labels with rotational invariances, there are still some mixings of labels at some locations of the trained manifolds (see figure 7(c)). We see once the jrVAE trained with optimal tuning of $\boldsymbol{\beta_c}$, we get the best solution among other scenarios, where all discrete classes separated efficiently with negligible mixing of labels at any locations of the trained manifolds (see figure 7 (d)).

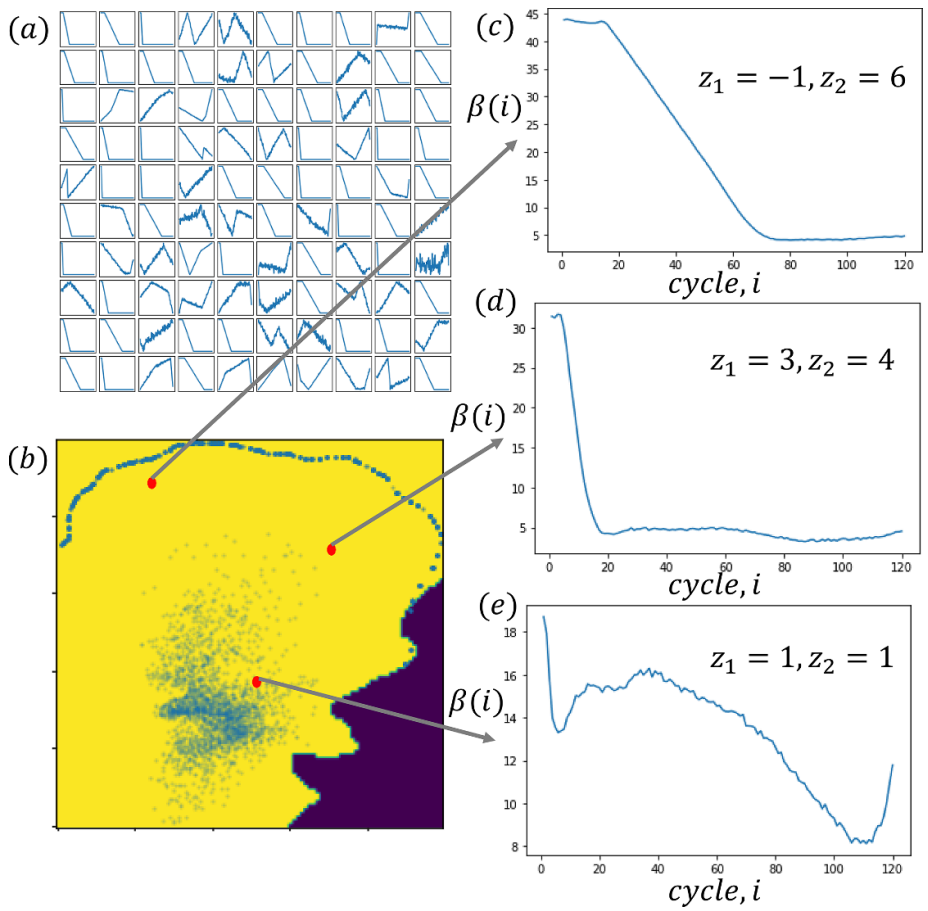

Figure 5. Workflow to generate a feasible 2D latent search space from N-dimensional (hyper)parameter space. (a) Samples of different $\boldsymbol{\beta_c}$ trajectories (here defined from two different functions) as input for training VAE model (ref table 1, step 2). (b) 2D latent representation of the $\boldsymbol{\beta_c}$ trajectories, given the trained VAE model. The blue dots are the training data as shown in (a). The light area and dark area are the feasible and infeasible region. The feasible region is only considered for the search space during optimization. (c)–(e) Are the reconstructed $\boldsymbol{\beta_c}$ trajectories from the trained VAE decoder at locations (red dots): $(z_1,\ z_2) = (-1,\ 6)$, $(z_1,\ z_2) = (3,\ 4)$ and $(z_1,\ z_2) = (1,\ 1)$ respectively.

4. Results: experimental analysis

In this section, we showcased the proposed zBO-jrVAE workflow to an experimental data of plasmonic nanoparticles. The datasets contain correlated scattering spectra of gold particles and scanning electron microscope images (in supplementary material, figure B1). Here, our task is to conduct the multi-label classification, where we attempt to extract and separate the discrete labels as particle counts in the images through the unsupervised learning of jrVAE model, integrated with zBO.

In this case study, we considered gaussian decoder sampler with sigma $= 0.3$, learning rate $= 1 \times 10^{-4}$ and training cycles $= 2000$ to initialize and train the VAE model for 2D latent representation of N-dimensional β trajectories (table 1, Step 2). For initializing the jrVAE model, we considered gaussian decoder sampler with sigma $= 0.01$, learning rate $= 1 \times 10^{-4}$ and training cycle, $N = 200$. It is to be noted that here we have different input scale factor dimension ($N = 200$) unlike the MNIST analysis ($N = 120$), depending on the number of training cycles of jrVAE model. However, the dimension of the problem in zBO (during optimization) can be still represented as 2D (in latent space) and does not increase with the increase of dimension of the input in real space. For the zBO, we started with 20 randomly selected samples with maximum of 100 BO iteration, thus a total of 120 function evaluations. We choose EI acquisition function. Here to find the suitable scaling of $\boldsymbol{\beta}$, we considered different scale factors and attempted multiple BOs in coarse grid search. Finally, with domain expert knowledge, we set the maximum and minimum bounds of $\beta_c(i)$, as $0.001 \leqslant \beta_c(i) \leqslant 0.05$, assuming the optimal solution exists within the bound. Here also, we optimize the continuous scale factor, $\boldsymbol{\beta_c}$ trajectory only and set the discrete scale factor at constant setting as $\beta_d(i) = 0.01; i = 1,\ 2, \ldots,\ N$ where $N = 200$ in this case. We first normalized and balanced (through weighted resampling technique) the raw experimental data set. During the function evaluation process within BO, as previous analysis, we considered a subset of data (randomly selected). We executed the zBO on

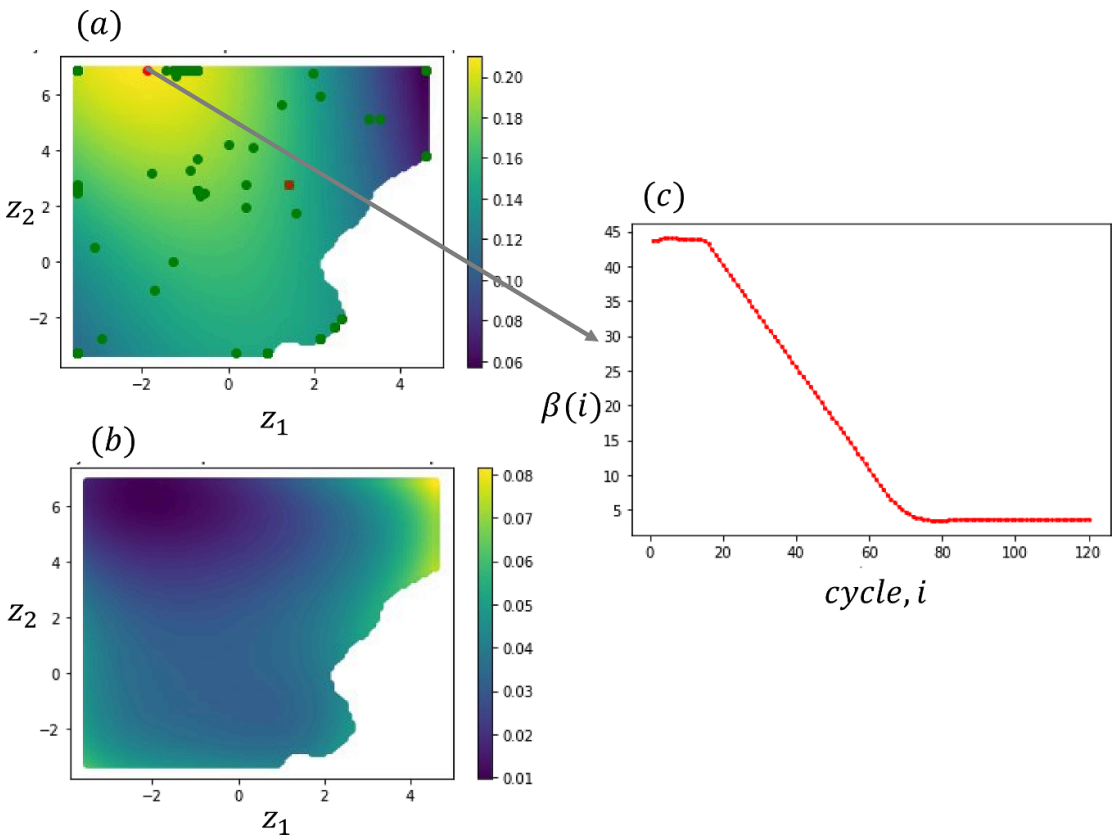

Figure 6. Results of zBO to find optimal region of β_c trajectories. (a), (b) GP estimated objective function (SSIM loss) map and its uncertainty map in the latent space respectively. The red dot is the maximum estimated objective value, and the red cross is the maximum among evaluated samples. We consider the GP estimated maximum value as the optimal solution and (c) is the respective reconstructed β_c trajectory at that optimal location (red dot) in the feasible 2D latent space.

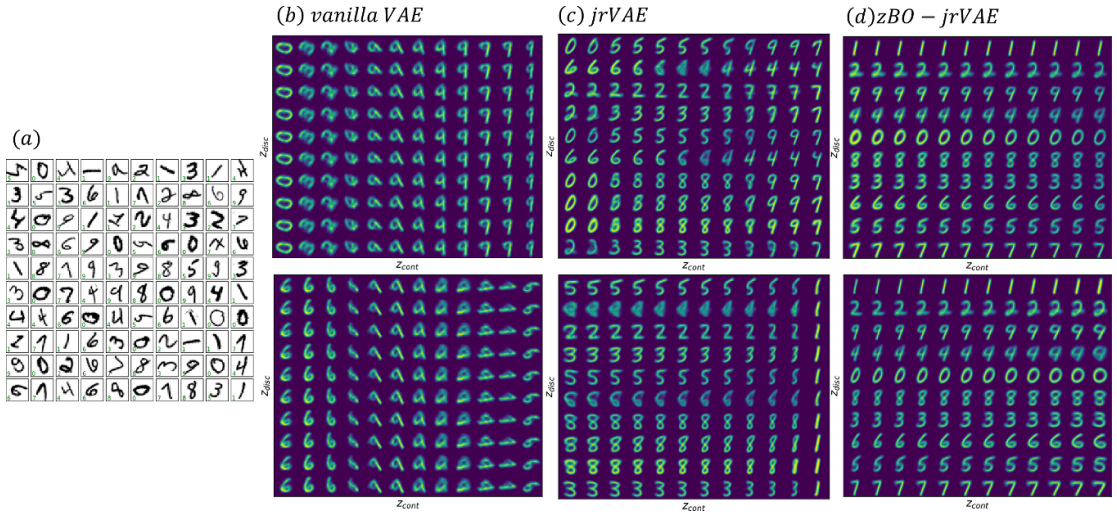

Figure 7. Comparison of different VAE models, with and without β_c trajectories tuning. (a) Few training samples of MNIST dataset as the task of multi-label (digits) classification problem. We have trained different VAE model for 120 cycles, given the full MNIST data (60 000 training data). The learned manifolds traversal for the 2D latent space (two latent variables) as shown in rows after training of respective (b) Vanilla VAE model, (c) jrVAE model, with default setting of β_c trajectory (no zBO implementation) and (d) jrVAE model, with optimal tuning (as shown in figure 6(c)) of β_c trajectory (zBO implementation).

NVIDIA's DGX-2 server. Once the zBO model is converged, we train the jrVAE model with large dataset, stated settings, and optimal tuning of β_c. We use the Google Colab Pro to execute the final training. However, unlike the previous analysis, here we considered the full loss function (Step 2, 3 of table 2) as per equation (2).

We started with the classification of three labels as images with one, two and three particles. Figure 8 shows the projection of N-dimensional search space to the feasible 2D latent search space. In this case, we

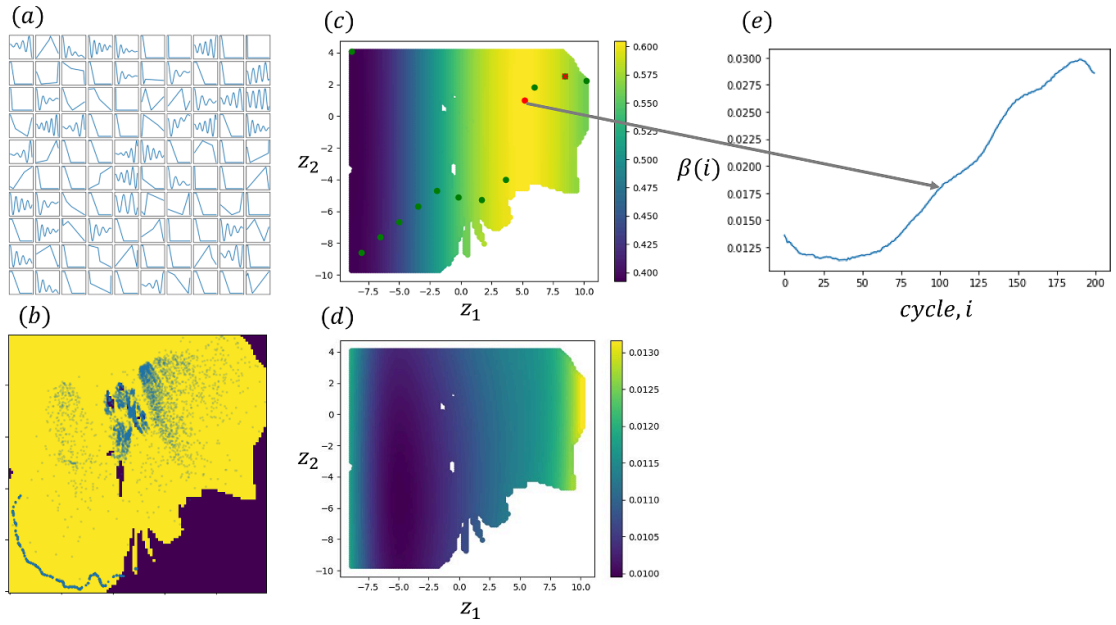

Figure 8. Classification problem of plasmonic nanoparticles data (3 discrete labels). Results of zBO to find optimal region of $\boldsymbol{\beta_c}$ trajectories. (a) Samples of different $\boldsymbol{\beta_c}$ trajectories (here defined from three different functions) as input for training VAE model. (b) 2D latent representation of the $\boldsymbol{\beta_c}$ trajectories where the blue dots are the training data (c), (d) GP estimated objective function (SSIM loss) map and its uncertainty map in the latent space respectively. The red dot is the maximum estimated objective value, and the red cross is the maximum among evaluated samples. We consider the GP estimated maximum value as the optimal solution and (e) is the respective reconstructed $\boldsymbol{\beta_c}$ trajectory at that optimal location (red dot) in the feasible 2D latent space.

defined the training trajectories, with same scale, from three different functions: (a) linear cooldown, (b) randomly segmented and (c) periodic (see figure 8(a)). Figure 8(b) shows the respective feasible 2D latent representation. Figures 8(c)–(e) shows the result of zBO convergence, and the optimal $\boldsymbol{\beta_c}$ trajectory. The model converged earlier than 100 function evaluations as the acquisition function value goes to negligible, meaning the optimal region is already found with negligible uncertainty and no further meaningful learning is possible as trade-off with expensive function evaluations at any other locations. We can see the optimal trajectory is very different than what we found in solving MNIST problem (see figure 6(c)).

Figure 9 shows the comparison among different VAE models, with and without the high-dimensional continuous scale factor tuning. We can clearly see the Vanilla VAE model provides the worst result again (see figure 9(a)). Even with tweaking other hyperparameters, though we are able to get the images with distinct features (see figure 9(b)), however for both cases the models could not separate the labels and thus provide no physical insight from the data. With the jrVAE model, but with unscaled default constant setting of $\boldsymbol{\beta_c}(i) = 1; i = 1,\ 2, \ldots,\ 200$, though the model shows some separation of classes (class of two particles), there are still some mixings of labels specially at top rows of the trained manifolds (see figure 9(c)), which results in some infeasible physical behavior. As we scaled $\boldsymbol{\beta_c}(i) = 0.01$ within the bounds for optimization (see figure 9(d)), we see the performance improved but still we find the separation of two classes (class of 1 and 3 particles). Finally, we see once the jrVAE trained with optimal tuning of $\boldsymbol{\beta_c}$, we see further enhancement and get the best solution among other scenarios. Unlike in figure 9(d) where the top and bottom rows have the same class of 2 particles, in figure 9(e), we see the top and the bottom rows of the learned manifolds now have distinct classes of 2 and 1 particles respectively. The middle row contains classes of 2 and 3 particles at different locations. We have also compared the results (in supplementary material, figure B2.) with optimal tuning of $\boldsymbol{\beta_c}$ found in solving MNIST problem (figure 6(c)) to understand the sensitivity of $\boldsymbol{\beta_c}$ for a given problem. We see the optimal setting found for the nanoparticles problem provides better training of jrVAE, in better separating into discrete class (rows) of the manifolds.

Similar analysis to materials and methods is done utilizing the GPU server for fast training, with more complex nanoparticles dataset where we considered 7 labels (particles counts). A sample of the dataset is provided in supplementary material as figure B2. Workflow on 2D latent representation and optimizing $\boldsymbol{\beta_c}$ through zBO is illustrated in figure 10. Here also, the model converged earlier than 100 function evaluations as the acquisition function value goes to negligible. Figures 11 and 12 show the similar comparative analysis among different VAE models. Here also, the Vanilla VAE models (figures 11(a) and (b)) are not able to

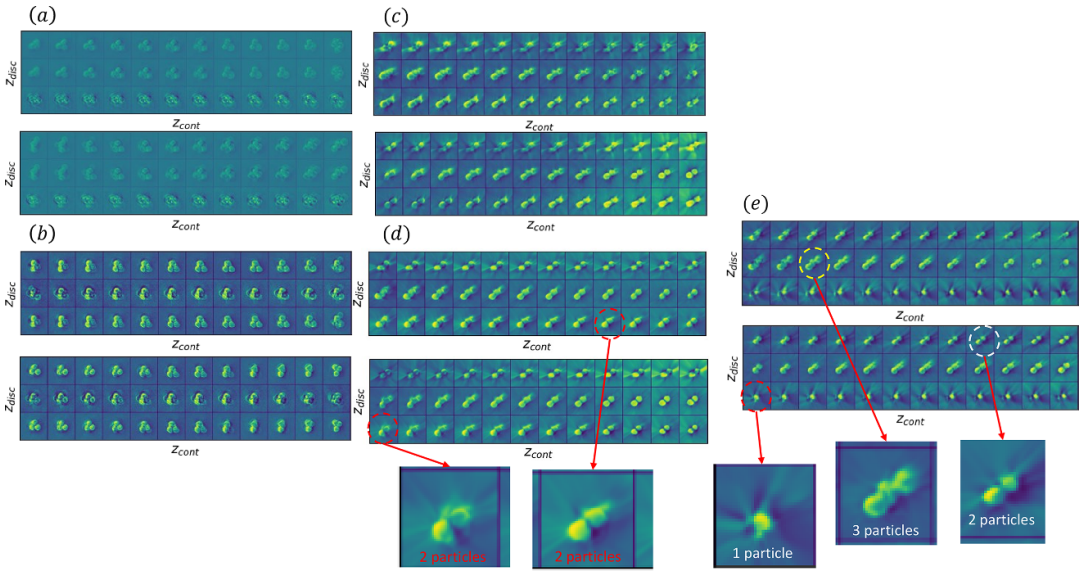

Figure 9. Comparison of different VAE models, considering plasmonic nanoparticles dataset, with 3 discrete labels (particle counts). We have trained different VAE model for 200 cycles. The learned manifolds traversal for the 2D latent space (two latent variables) as shown in rows after training of respective (a) Vanilla VAE model (all parameter in default settings as in [45]), (b) VAE model with other hyperparameter setting as stated for this case study, (c) jrVAE model, with constant unscaled default setting of $\boldsymbol{\beta_c}$ trajectory (no zBO implementation), (d) jrVAE model, with constant scaled $\boldsymbol{\beta_c}$ trajectory (no zBO implementation) and (e) jrVAE model, with optimal tuning (as shown in figure 8(c)) of $\boldsymbol{\beta_c}$ trajectory (zBO implementation).

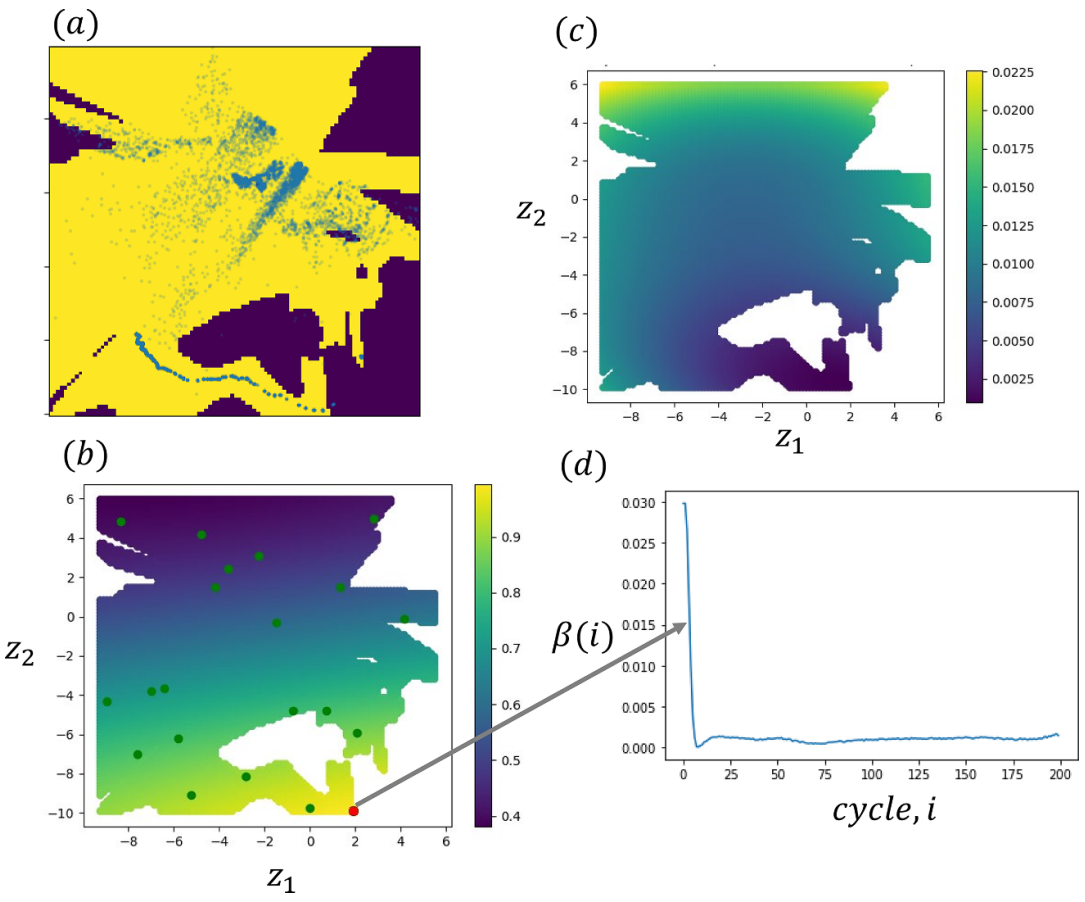

Figure 10. Classification problem of plasmonic nanoparticles data (7 discrete labels). Results of zBO to find optimal region of $\boldsymbol{\beta_c}$ trajectories. (a) 2D latent representation (through VAE) of the $\boldsymbol{\beta_c}$ trajectories where the blue dots are the training data (b), (c) GP estimated objective function (SSIM loss) map and its uncertainty map respectively. The red dot is the maximum (optimal) estimated objective value and (e) is the respective reconstructed $\boldsymbol{\beta_c}$ trajectory at that optimal location (red dot) in the feasible 2D latent space.

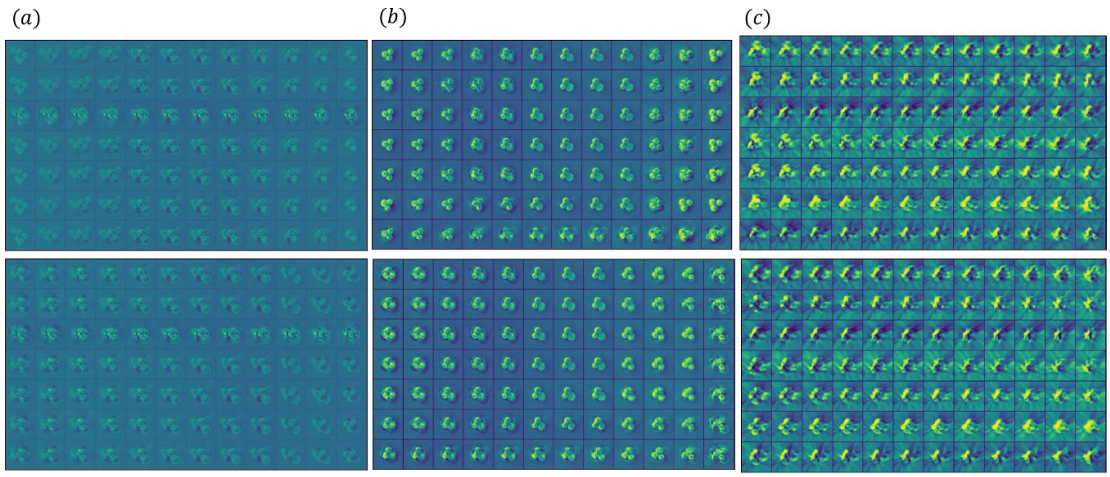

Figure 11. Comparison of different VAE models, for multilabel classification of plasmonic nanoparticles dataset, considering 7 discrete labels (particle counts). We have trained all VAE models for 200 cycles. The learned manifolds traversal for the 2D latent space (two latent variables) as shown in rows after training of respective (a) Vanilla VAE model (all parameter in default settings as in [45]), (b) VAE model with other hyperparameter setting as stated for this case study and (c) jrVAE model, with constant unscaled default setting of $\boldsymbol{\beta_c}$ trajectory (no zBO implementation).

separate any labels and therefore fail to extract any valuable physics from data. Surprisingly, the jrVAE model with unscaled default constant setting of $\boldsymbol{\beta_c}(i) = 1; i = 1,\ 2, \ldots,\ 200$ gives very poor solution with too distorted images to identify any labels properly, which results in an unrealistic extraction of physical information (see figure 11(c)). Similarly with proper scaling the trajectory within the stated bounds for optimization as $\boldsymbol{\beta_c}(i) = 0.01$, we get better solution but could only extract 2 meaningful discrete labels (particle counts 3 and 5) out of 7 (see figure 12(a)). Training the model with optimal tuned $\boldsymbol{\beta_c}$, shows the best disentanglement of data among all other scenarios (see figure 12(b)). Here, we could be able to extract 5 discrete labels (particle counts 1–5) out of 7. This shows a much better improvement than rest of the scenarios, where we have extracted >70 % physical insights from the data, with the optimization approach through zBO. With the increase in complexity of the data, we see higher value to optimize the scale trajectory for optimal jrVAE training process, instead of considering any unscaled or scaled constant values. However, we believe for further improvement may need the optimization of β_d, which is a task for future investigation. However, it is evident that the extraction of knowledge from experimental data is much harder than test data (like in MNIST), however, we still get a very good improvement from optimal tuning with zBO in learning plasmonic nanoparticle system, where we can get an appropriate physical insight of all the classes from the data (considering 3 labels) and above 70% physical information from the data (considering 7 labels). The purpose of this work is the zBO framework to guide towards finding the maximum (not necessarily always 100%) learning of physical insights from optimizing high dimensional parameters or hyperparameters of other mathematical or ML models for a given application, given the degree of complexity (variability) of the problem (data) and the fixed settings of other parameters or hyperparameters.

5. Conclusion

To summarize, here, we extend Latent Bayesian optimization (zBO) to tackle any iteration- dependent high-dimensional hyperparameter optimization (or hyperparameter trajectory) problem for jrVAEs. In high dimensional parameter optimization, considering expensive function evaluations, the guidance of where to invest more on the search space is very critical in terms of reducing overall computational cost, thus manual or exhaustive search is very tedious or infeasible. Due to the curse of dimensionality, the standard BO is also not entirely reliable or computationally efficient. In the zBO framework, the high dimensional parameter space is compressed into a two-dimensional latent space, where we capture sufficient variability of parameters through training a VAE model. We see the latent space preserves the pattern of original training samples (as per domain expert knowledge) while introducing some variability (new hybrid patterns) as well in the feasible input set of solutions. Thus, the prior expert knowledge is still preserved as we project into the reduced latent space. Then, the reduced latent space is considered as the proxy search space for optimization where the optimal latent solution can be easily decoded into the high dimensional trajectory.

In this analysis, we choose to optimize the high dimensional continuous scale parameters of jrVAE model and apply to multi-label classification problems of MNIST and plasmonic nanoparticles dataset. We see the

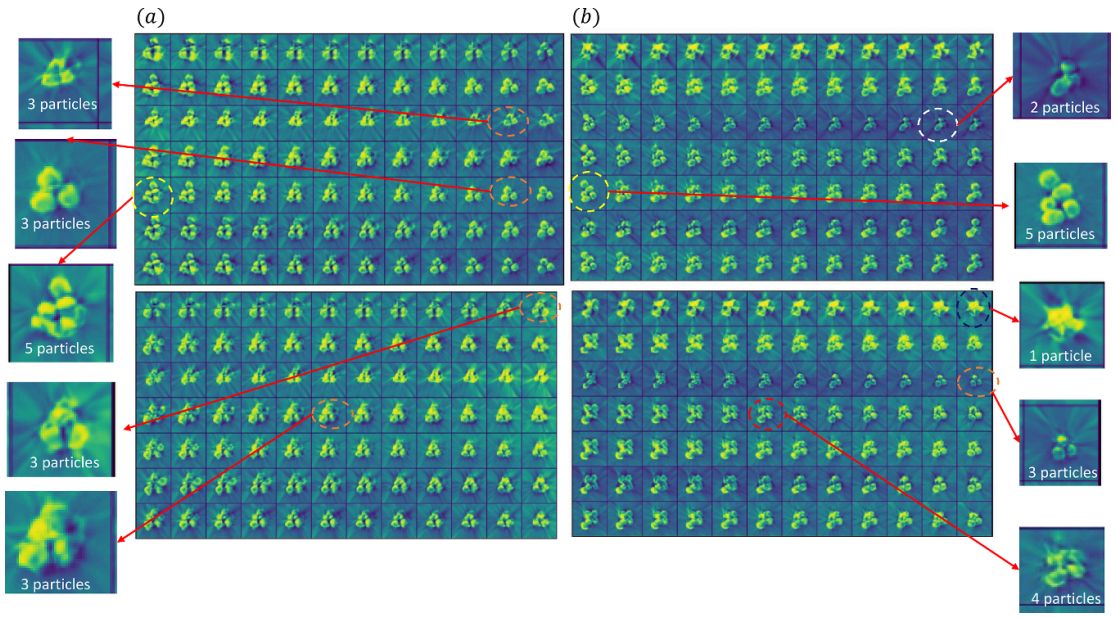

Figure 12. Comparison of different VAE models, for multilabel classification of plasmonic nanoparticles dataset, considering 7 discrete labels (particle counts). We have trained all VAE models for 200 cycles. The learned manifolds traversal for the 2D latent space (two latent variables) as shown in rows after training of respective (a) jrVAE model, with constant scaled $\boldsymbol{\beta_c}$ trajectory (no zBO implementation) and (b) jrVAE model, with optimal tuning of $\boldsymbol{\beta_c}$ trajectory (zBO implementation). For the optimal tuned $\boldsymbol{\beta_c}$, we find 5 discrete labels (as particle counts from 1 to 5) out of 7 labels from the learned manifolds as marked by black, white, orange, red and yellow circles respectively, whereas for the constant (but scaled) $\boldsymbol{\beta_c}$, we find only 2 discrete labels (as particle counts 3 and 5) out of 7 labels as marked in the respective locations of the learned manifolds.

performance of jrVAE model, with optimal tuning of the scale parameters through zBO is promising. Interestingly, the optimal tuning of scale parameters varies even in patterns for given problems, where an optimal trajectory pattern (e.g. Cooldown trajectory) in one problem (MNIST dataset) seemed not ideal for the other problem (plasmonic nanoparticles), and thus cannot be generalized to guarantee best performance of the same model to extract information from all type of dataset. This observation further values the need of zBO framework in enhancing a ML model, without the assumption of generalizing optimal tuning, for different problems separately. As per the results in figure 11 showcased the room for improvement, one way to do is to extend the framework to optimize both the high dimensional discrete and continuous scale factors jointly, which is a scope for future. Another future task would be to investigate the trade-off between overall computational cost and solution improvement with higher dimension of latent space. However, the overall approach is flexible to incorporate various pattern of trajectories (from different functionals) in the same latent space, to handle any dimension of correlated input parameters without increasing the dimension of the reduced latent space, to consider different problem objectives (other than classifications) as set by user defined objective functions in the zBO.

Data availability statement

The analysis reported here is summarized in Colab Notebook for the purpose of tutorial and application to other models [65].

Acknowledgments

This work was supported by the US Department of Energy, Office of Science, Office of Basic Energy Sciences, as part of the Energy Frontier Research Centers program: CSSAS—The Center for the Science of Synthesis Across Scales—under Award No. DE-SC0019288, located at University of Washington, DC. The autoencoder research was supported by the Center for Nanophase Materials Sciences (CNMS), which is a US Department of Energy, Office of Science User Facility at Oak Ridge National Laboratory. The experimental dataset used in analysis was supported from Ginger Lab, University of Washington. We also thank José Miguel Hernández-Lobato for valuable feedback.

Conflict of interest

The authors declare no conflict of interest.

ORCID iDs

Arpan Biswas https://orcid.org/0000-0002-4054-7700
Rama Vasudevan https://orcid.org/0000-0003-4692-8579
Maxim Ziatdinov https://orcid.org/0000-0003-2570-4592

References

[1] Sarker I H 2021 Machine learning: algorithms, real-world applications and research directions *SN Comput. Sci.* **2** 160
[2] Ge M, Su F, Zhao Z and Su D 2020 Deep learning analysis on microscopic imaging in materials science *Mater. Today Nano* **11** 100087
[3] Kalinin S V *et al* 2022 Machine learning in scanning transmission electron microscopy *Nat. Rev. Methods Primer* **2** 1
[4] Kalinin S V, Steffes J J, Liu Y, Huey B D and Ziatdinov M 2021 Disentangling ferroelectric domain wall geometries and pathways in dynamic piezoresponse force microscopy via unsupervised machine learning *Nanotechnology* **33** 055707
[5] Liu Y, Proksch R, Wong C Y, Ziatdinov M and Kalinin S V 2021 Disentangling ferroelectric wall dynamics and identification of pinning mechanisms via deep learning *Adv. Mater.* **33** 2103680
[6] Jesse S and Kalinin S V 2009 Principal component and spatial correlation analysis of spectroscopic-imaging data in scanning probe microscopy *Nanotechnology* **20** 085714
[7] Blum T, Graves J, Zachman M J, Polo-Garzon F, Wu Z, Kannan R, Pan X and Chi M 2021 Machine learning method reveals hidden strong metal-support interaction in microscopy datasets *Small Methods* **5** 2100035
[8] Li L, Yang Y, Zhang D, Ye Z-G, Jesse S, Kalinin S V and Vasudevan R K 2018 Machine learning–enabled identification of material phase transitions based on experimental data: exploring collective dynamics in ferroelectric relaxors *Sci. Adv.* **4** eaap8672
[9] Taranto W *et al* 2022 Unsupervised learning of two-component nematicity from STM data on magic angle bilayer graphene (arXiv:2203.04449)
[10] Venderley J *et al* 2021 Harnessing interpretable and unsupervised machine learning to address big data from modern x-ray diffraction (arXiv:2008.03275)
[11] Kalinin S V, Dyck O, Jesse S and Ziatdinov M 2021 Exploring order parameters and dynamic processes in disordered systems via variational autoencoders *Sci. Adv.* **7** eabd5084
[12] Kalinin S V *et al* 2021 Unsupervised machine learning discovery of chemical and physical transformation pathways from imaging data (arXiv:2010.09196)
[13] Yang L and Shami A 2020 On hyperparameter optimization of machine learning algorithms: theory and practice *Neurocomputing* **415** 295–316
[14] Luketina J, Berglund M, Greff K and Raiko T 2016 Scalable gradient-based tuning of continuous regularization hyperparameters *Proc. 33rd Int. Conf. on Machine Learning* pp 2952–60 (available at: https://proceedings.mlr.press/v48/luketina16.html) (Accessed 7 June 2022)
[15] Sinha A, Khandait T and Mohanty R 2020 A gradient-based bilevel optimization approach for tuning hyperparameters in machine learning (arXiv:2007.11022)
[16] Xiao X, Yan M, Basodi S, Ji C and Pan Y 2020 Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm (arXiv:2006.12703)
[17] Young S R, Rose D C, Karnowski T P, Lim S-H and Patton R M 2015 Optimizing deep learning hyper-parameters through an evolutionary algorithm *Proc. Workshop on Machine Learning in High-Performance Computing Environments* (*New York, USA*) pp 1–5
[18] Wu J, Chen X-Y, Zhang H, Xiong L-D, Lei H and Deng S-H 2019 Hyperparameter optimization for machine learning models based on bayesian optimization *J. Electron. Sci. Technol.* **17** 26–40
[19] Park S M, Yoon H G, Lee D B, Choi J W, Kwon H Y and Won C 2022 Optimization of physical quantities in the autoencoder latent space *Sci. Rep.* **12** 1
[20] Lizotte D, Wang T, Bowling M and Schuurmans D 2007 Automatic gait optimization with Gaussian process regression *Proc. 20th Int. Joint Conf. on Artificial Intelligence* (*Hyderabad, India, 6–12 January 2007*) pp 944–9 (available at: www.ijcai.org/proceedings/2007/)
[21] Lizotte D 2008 Practical Bayesian optimization *PhD Thesis* University of Alberta (https://doi.org/doi/10.5555/1626686)
[22] Cora V M 2008 Model-based active learning in hierarchical policies *PhD Dissertation* University of British Columbia Library, Vancouver (https://doi.org/10.14288/1.0051276)
[23] Frean M and Boyle P 2008 Using Gaussian processes to optimize expensive functions *AI 2008: Advances in Artificial Intelligence* (Berlin: Springer) pp 258–67
[24] Martinez-Cantin R, de Freitas N, Brochu E, Castellanos J and Doucet A 2009 A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot *Auton. Robots* **27** 93–103
[25] Greenhill S, Rana S, Gupta S, Vellanki P and Venkatesh S 2020 Bayesian optimization for adaptive experimental design: a review *IEEE Access* **8** 13937–48
[26] Wang Z, Hutter F, Zoghi M, Matheson D and De Freitas N 2016 Bayesian optimization in a billion dimensions via random embeddings *J. Artif. Intell. Res.* **55** 361–87
[27] Moriconi R, Kumar K S S and Deisenroth M P 2020 High-dimensional Bayesian optimization with projections using quantile Gaussian processes *Optim. Lett.* **14** 51–64
[28] Oh C, Gavves E and Welling M 2019 BOCK: Bayesian optimization with cylindrical Kernels (arXiv:1806.01619)
[29] Valleti M, Vasudevan R K, Ziatdinov M A and Kalinin S V 2022 Bayesian optimization in continuous spaces via virtual process embeddings (arXiv:2206.12435)
[30] Siivola E, Paleyes A, González J and Vehtari A 2021 Good practices for Bayesian optimization of high dimensional structured spaces *Appl. Lett.* **2** e24

[31] Kusner M J, Paige B and Hernández-Lobato J M 2017 Grammar variational autoencoder *Proc. 34th Int. Conf. on Machine Learning* pp 1945–54 (available at: https://proceedings.mlr.press/v70/kusner17a.html)
[32] Gómez-Bombarelli R, Wei J N, Duvenaud D, Hernández-Lobato J M, Sánchez-Lengeling B, Sheberla D, Aguilera-Iparraguirre J, Hirzel T D, Adams R P and Aspuru-Guzik A 2018 Automatic chemical design using a data-driven continuous representation of molecules *ACS Cent. Sci.* **4** 268–76
[33] Grosnit A *et al* 2021 High-dimensional Bayesian optimisation with variational autoencoders and deep metric learning (arXiv:2106.03609)
[34] Notin P, Hernández-Lobato J M and Gal Y 2021 Improving black-box optimization in VAE latent space using decoder uncertainty (arXiv:2107.00096)
[35] Tripp A, Daxberger E and Hernández-Lobato J M 2020 Sample-efficient optimization in the latent space of deep generative models via weighted retraining (arXiv:2006.09191)
[36] Griffiths R-R and Hernández-Lobato J M 2019 Constrained Bayesian optimization for automatic chemical design (arXiv:1709.05501)
[37] Dhamala J, Bajracharya P, Arevalo H J, Sapp J, Horácek B M, Wu K C, Trayanova N A and Wang L 2020 Embedding high-dimensional Bayesian optimization via generative modeling: parameter personalization of cardiac electrophysiological models *Med. Image Anal.* **62** 101670
[38] Ziatdinov M and Kalinin S 2021 Robust feature disentanglement in imaging data via joint invariant variational autoencoders: from cards to atoms (arXiv:2104.10180)
[39] Kingma D P and Welling M 2019 An introduction to variational autoencoders *Found. Trends Mach. Learn.* **12** 307–92
[40] Asperti A and Trentin M 2020 Balancing reconstruction error and Kullback-Leibler divergence in variational autoencoders *IEEE Access* **8** 199440–8
[41] Prokhorov V, Shareghi E, Li Y, Pilehvar M T and Collier N 2019 On the importance of the Kullback-Leibler divergence term in variational autoencoders for text generation (arXiv:1909.13668)
[42] Ziatdinov M, Wong C Y and Kalinin S V 2021 Finding simplicity: unsupervised discovery of features, patterns, and order parameters via shift-invariant variational autoencoders (arXiv:2106.12472)
[43] Ziatdinov M, Ghosh A, Wong T and Kalinin S V 2021 AtomAI: a deep learning framework for analysis of image and spectroscopy data in (scanning) transmission electron microscopy and beyond (arXiv:2105.07485)
[44] Creange N, Dyck O, Vasudevan R K, Ziatdinov M and Kalinin S V 2022 Towards automating structural discovery in scanning transmission electron microscopy *Mach. Learn. Sci. Technol.* **3** 015024
[45] Ziatdinov M 2022 *pyroVED* (available at: https://github.com/ziatdinovmax/pyroVED)
[46] Burgess C P *et al* 2018 Understanding disentangling in β-VAE (arXiv:1804.03599)
[47] Brochu E, Cora V M and de Freitas N 2010 A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning (arXiv:1012.2599 [cs.LG])
[48] Biswas A and Hoyle C 2021 An approach to bayesian optimization for design feasibility check on discontinuous black-box functions *ASME. J. Mech. Des.* **143** 3
[49] Chu W and Ghahramani Z 2005 Extensions of Gaussian processes for ranking: semisupervised and active learning *NIPS Workshop on Large Scale Kernel Machines, Whistler 2005* (available at: www.merlot.org/merlot/viewMaterial.htm?id=975278)
[50] Thurstone L L 1927 A law of comparative judgment *Psychol. Rev.* **34** 273–86
[51] Mosteller F 2006 Remarks on the method of paired comparisons: I. The least squares solution assuming equal standard deviations and equal correlations *Selected Papers of Frederick Mosteller* ed S E Fienberg and D C Hoaglin (New York: Springer) pp 157–62
[52] Holmes C C and Held L 2006 Bayesian auxiliary variable models for binary and multinomial regression *Bayesian Anal.* **1** 145–68
[53] Hutter F, Hoos H H and Leyton-Brown K 2011 Sequential model-based optimization for general algorithm configuration *Learning and Intelligent Optimization* (Berlin: Springer) pp 507–23
[54] Shahriari B, Swersky K, Wang Z, Adams R P and de Freitas N 2016 Taking the human out of the loop: a review of Bayesian optimization *Proc. IEEE* **104** 148–75
[55] Kushner H J 1964 A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise *J. Basic Eng.* **86** 97–106
[56] Andrianakis I and Challenor P 2012 The effect of the nugget on Gaussian process emulators of computer models *Comput. Stat. Data Anal.* **56** 4215–28
[57] Pepelyshev A 2010 The role of the nugget term in the Gaussian process method *mODa 9—Advances in Model-Oriented Design and Analysis* (Heidelberg: Physica-Verlag HD) pp 149–56
[58] Xing W, Elhabian S Y, Keshavarzzadeh V and Kirby R M 2020 Shared-Gaussian process: learning interpretable shared hidden structure across data spaces for design space analysis and exploration *J. Mech. Des.* **142** 12
[59] Bostanabad R, Chan Y-C, Wang L, Zhu P and Chen W 2019 Globally approximate Gaussian processes for big data with application to data-driven metamaterials design *J. Mech. Des.* **141** 11
[60] Erickson C B, Ankenman B E and Sanchez S M 2018 Comparison of Gaussian process modeling software *Eur. J. Oper. Res.* **266** 179–92
[61] Jones D R 2001 A taxonomy of global optimization methods based on response surfaces *J. Glob. Optim.* **21** 345–83
[62] Jones D R, Schonlau M and Welch W J 1998 Efficient global optimization of expensive black-box functions *J. Glob. Optim.* **13** 455–92
[63] Cox D D and John S 1992 A statistical method for global optimization *Proc. 1992 IEEE Int. Conf. on Systems, Man, and Cybernetics* vol 2 pp 1241–6
[64] Deng L 2012 The MNIST database of handwritten digit images for machine learning research [best of the web] *IEEE Signal Process. Mag.* **29** 141–2
[65] Biswas A 2022 *Notebook for LatentBO-jrVAE* (available at: https://github.com/arpanbiswas52/PaperNotebooks)