Structured Variationally Auto-encoded Optimization

Xiaoyu Lu [1] **Javier González** [2] **Zhenwen Dai** [2] **Neil D. Lawrence** [2 3]

Abstract

We tackle the problem of optimizing a black-box objective function defined over a highly-structured input space. This problem is ubiquitous in machine learning. Inferring the structure of a neural network or the Automatic Statistician (AS), where the kernel combination for a Gaussian process is optimized, are two of many possible examples. We use the AS as a case study to describe our approach, that can be easily generalized to other domains. We propose an *Structure Generating Variational Auto-encoder* (SG-VAE) to embed the original space of kernel combinations into some low-dimensional continuous manifold where Bayesian optimization (BO) ideas are used. This is possible when structural knowledge of the problem is available, which can be given via a simulator or any other form of generating potentially good solutions. The right *exploration-exploitation* balance is imposed by propagating into the search the uncertainty of the latent space of the SG-VAE, that is computed using variational inference. The key aspect of our approach is that the SG-VAE can be used to bias the search towards relevant regions, making it suitable for transfer learning tasks. Several experiments in various application domains are used to illustrate the utility and generality of the approach described in this work.

1. Introduction

In Science and Engineering, optimization problems with very structured input domains are ubiquitous. In machine learning we have a few examples. In deep learning, the architecture of neural networks is defined by a large number of parameters, like the number of layers, activation functions, etc. These parameters show conditional dependencies, which makes the problem specially hard to treat as different network configurations cannot be represented in an unique Euclidean space (Jenatton et al., 2017; Swersky et al., 2013). In kernel-based methods, the Automatic statistician (AS) is another example. With a Gaussian process (GP) as the class of models of choice, the goal is to automatically select the best kernel combination to explain a data set, which is chosen by enumerating a countably infinite set of arbitrarily complex kernels composed via additions and multiplication of simple ones (Duvenaud et al., 2013).

These examples, among others that we will discuss later, can be formalized as optimization problems in which an expensive black-box objective function, typically assumed 'well behaved', is optimized in a highly structured input space. By 'highly structured' we mean that the solutions cannot be trivially represented in a Euclidean input space, making standard optimization approaches impractical. Instead, they might belong to a tree or to other non-Euclidean input domain. In this paper, we are interested on these problems. More specifically, we study how knowledge about the problem structure can be leveraged to solve them successfully. The methods and ideas proposed here are general and can easily be extended to other domains. As a way of illustrating them we use the AS as a case-study but the experimental section of this work contains further experiments in other domains such is textual images description.

Before we get into further details, we start by formalizing the AS problem, around which we will provide the details of our approach.

1.1. Case Study: the Automatic Statistician

In the AS the goal is to solve a supervised learning problem given a dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$ where $\mathbf{x}_i \in \mathbb{R}^p$ are the inputs and $y_i \in \mathbb{R}$ are the outputs. Let $\mathbb{M}$ be the class of models that we will use to explain the data. In particular, we consider $\mathbb{M}$ to be the family of GPs with different kernel combinations (Rasmussen & Williams, 2005). We denote by $\mathcal{M}$ each model of this set and we denote by $\mathcal{P}_{\mathcal{M}}$ its associated parameter space. The models in $\mathbb{M}$ represent different structural assumptions about the data such as trends or periodicity. The values of $\beta \in \mathcal{P}$, the model hyper-parameters, differentiate models within the same family.

The goal of the AS is to select one single model from $\mathbb{M}$

[*]Equal contribution [1]Department of Statistics, University of Oxford, United Kingdom [2]Amazon, Cambridge, United Kingdom [3]University of Sheffield,United Kingdom. Correspondence to: Javier Gonzalez <gojav@amazon.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

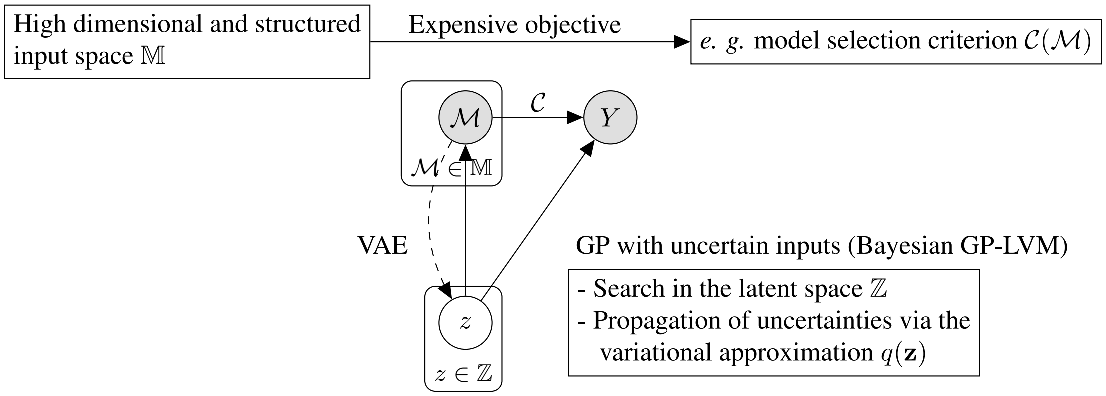
High dimensional and structured input space $\mathbb{M}$
Expensive objective
e. g. model selection criterion $\mathcal{C}(\mathcal{M})$
$\mathcal{M}$
$\mathcal{C}$
Y
$\mathcal{M}' \in \mathbb{M}$
VAE
GP with uncertain inputs (Bayesian GP-LVM)
- Search in the latent space $\mathbb{Z}$
- Propagation of uncertainties via the variational approximation $q(\mathbf{z})$
z
$z \in \mathbb{Z}$

Figure 1. Main elements of SVO: An expensive objective function needs to be optimized in a structured input space. A Variational auto-encoder is trained to learn a latent space $\mathbb{Z}$ using data produced by a context-free grammar. BO is applied over the latent space via a GP model with uncertain inputs to find the optimal kernel combination. The uncertainty of the latent space, computed using variational inference techniques, is used in the search to balance exploration and exploitation.

that explains the data $\mathcal{D}$ the best. We denote by $\mathcal{C} : \mathbb{M} \rightarrow \mathbb{R}$ some 'goodness of fit measure' that quantifies the quality of the fit. We keep $\mathcal{C}(\mathcal{M})$ generic for the moment, and we just assume that it is expensive enough such that it is only feasible to evaluate in a few models in $\mathbb{M}$. The problem reduces to find

$$\mathcal{M}_{\beta}^{opt} := \arg \max_{\mathcal{M} \in \mathbb{M}} \mathcal{C}(\mathcal{M}). \qquad (1)$$

Problem (1) is not suitable to be solved directly using Bayesian optimization (BO) (Shahriari et al., 2016). The reason is that the input space of Problem (1) is very structured, highly dimensional, and non-continuous so most BO methods will fail. The key idea of the approach presented in this paper is to transform Problem (1) into a problem that can be handled with standard BO methods.

1.2. Related work

The idea of using BO in high dimensional and structured spaces has already been explored in the literature. Random projections (Wang et al., 2016), other generative models like deep Gaussian processes (Dai et al., 2016) and combinations of optimization and sampling strategies (Abbati et al., 2017) have been developed. Bayesian optimization methods able to deal with hierarchical dependencies have also been proposed (Jenatton et al., 2017; Swersky et al., 2013). Some interesting applications are the design of genes (Gonzalez et al., 2014) and molecules (Kusner et al., 2017). Although both works carry out optimization in the latent space produced by a VAE, the goal and approach of our work are different, where we use a likelihood that directly encodes the knowledge of the grammar in the decoder to map any point in the latent space into a valid structure representation. The works also differ in the application domains and the propagation of the uncertainty in the latent space into the search.

Regarding the AS, a few approaches have followed the original compositional approach (Grosse et al., 2012; Duvenaud et al., 2013). Kim & Teh (2016) scales this method to big data scenarios by using sparse GPS. Hwang & Choi (2015) developed relational kernel learning methods. Malkomes et al. (2016) proposed, to the best of our knowledge, the first approach that uses BO in this context. A parametric measure of kernels similarity, the *Hellinger distance* is used to guide the search over a selected sets of kernel combinations.

1.3. Contributions and organization

We use the idea that kernel combinations can be expressed as operations of a context-free grammar (Hopcroft et al., 2006). This is used to simulate combinations with certain properties or structure, similarly to the work of (Kusner et al., 2017) where a grammar is used to generate molecules. Gómez-Bombarelli et al. (2018) also used a context-free grammar to optimize molecules. They used a RNN that allows it to learn the structure of the problem directly from data.

The idea we follow here is (i) to use a mechanism to generate data that represent well the set of feasible solutions of the problem and (ii) use it to learn a low-dimensional manifold in which the search for the optimal solution takes place. To this end, we learn a latent variable model. We used a Variational auto-encoder (VAE) (Kingma & Welling, 2013) for practicality, although any other continuous latent variable model could be considered. With the problem mapped into a low-dimensional space BO can be used to find the best combination, circumventing the issues described in Section 1.1. Because there is uncertainty associated to the learned latent space, we use variational inference to incorporate it into the search. This has a positive effect, balancing exploration and exploitation. See Figure 1 for a graphical description of

Algorithm 1 Context-free grammar for kernel expressions generation.

```
Input: $N_{max}, p_{\mathcal{B}}, \mathcal{B}, p_{\mathcal{O}}, \mathcal{O}, \mathcal{S} = \emptyset$.
for k < $N_{max}$ do
    $\mathcal{S} \leftarrow \mathcal{S} + B$, select kernel with probability $p_{\mathcal{B}}$.
    $\mathcal{S} \leftarrow \mathcal{S} + O$: select operation with probability $p_{\mathcal{O}}$.
end for $O$ is Stop or $k = N_{max}$.
```

the main elements of the approach described in this work. The main contributions are:

- A new Variational auto-encoder, called 'Structure Generating Variational auto-encoder' (SG-VAE). We describe and used it in the context of mapping kernel combinations produced by a context-free grammar into a continuous and low-dimensional latent space. Although here we use it in the AS context, it is broadly applicable. The main novelty of this approach is that all information about the problem is directly encoded in the likelihood of the model, in contrast to previous approaches in which is learned from the data (Gómez-Bombarelli et al., 2018).

- A variational approximation of the distribution of the latent space of the SG-VAE. This distribution is used to propagate the uncertainty of the SG-VAE into the BO search. A GP with uncertain inputs is used to make this step practical.

- A series of experiments that illustrate the utility of this work in model selection and in a problem where the goal is to find the best textual description of a Minecraft image.

- An implementation of the proposed approach that can be found at the GPyOpt library `https://github.com/SheffieldML/GPyOpt` together with the scripts to reproduce the results of this work.

Section 2 describes the SG-VAE model and in Section 2.4 we detail how the SG-VAE can be used in optimization problems. In Section 3, we illustrate its performance with a series of experimental results. In Section 4 we include some conclusions and further lines of research derived from this work.

2. Variational Auto-encoders for structured spaces representation

In this section we present a new VAE to map structured input spaces into low-dimensional latent manifold. We describe it in the the context of the AS, so the goal is to use it to find good representations of kernel combinations produced by a context-free grammar. We describe the encoder, decoder and a variational approximation of the distribution of the latent variables.

2.1. Grammar-based Kernel Representation

It is possible to generate a countably infinite kernel space through closure of kernels via a context-free grammar. Given a set of base kernels $\mathcal{B}$ we can generate an expression (kernel combination) $\mathcal{S}$ by subsequently adding kernels and operations $\mathcal{O}$ (additions, multiplications, replacements or Stop) (Duvenaud et al., 2013; Grosse et al., 2012). Both the kernels and the operations are chosen according to prespecified probabilities $p_{\mathcal{B}}$ and $p_{\mathcal{O}}$. See Algorithm 1.

We use 1-hot encoding vectors for both, kernels and operations, to represent each expression $\mathcal{S}$. Suppose that $\mathcal{B} = \{K_1, K_2, K_3, K_4\}$ is the set of four base kernels and $\mathcal{O} = \{+, \times, Stop\}$ is the set of operations. Any expression $\mathcal{S}$ is transformed into a binary vector by recurrently attaching the 1-hot vectors of each kernel and operation. When the operation is $Stop$ the vector is completed with zeros. For instance, in the following example, four kernels are combined using a number of N_{max} operations. Before termination we have:

$$\underbrace{K_2}_{0100}\underbrace{+}_{100}\underbrace{K_1}_{1000}\underbrace{*}_{010}\underbrace{K_3}_{0010}\underbrace{*}_{010}\underbrace{K_1}_{1000}\underbrace{Stop}_{001}\underbrace{\ldots}_{Add\ 000}$$

This *grammar-based* representation, that we denote by $\mathbf{x}_g$, captures the complexity of the combination [1].

The grammar-based representation accounts for kernel complexity but it does not take into account the differences in the combinations due to the dataset $\mathcal{D}$. For this, a *data-based* representation is proposed in the next section.

2.2. Data-based Kernel Representation

We use the vector of distances between the kernel matrices of the base kernels evaluated in the data and the kernel matrices of the combinations. Details about how to deal with the hyper-parameters of the kernels are in the experimental section. As a measure of distance, the *Hellinger distance* could be used (Malkomes et al., 2016). However its $\mathcal{O}(n^4)$ complexity makes it prohibitively slow. We found that the Frobenius distance works well and it is also very quick to compute. We denote this data-based representation by $\mathbf{x}_d$ which we normalise before the VAE. The global representation for each combination is therefore $\mathbf{x} = [\mathbf{x}_g, \mathbf{x}_d]$, which has dimension $(N_{max} + 1)|\mathcal{B}| + (N_{max} - 1)|\mathcal{O}|$ for N_{max} the maximum number of allowed operations (added kernels).

[1] Note that $\mathbf{x}$ here and in the definition of $\mathcal{D}$ represent different vectors.

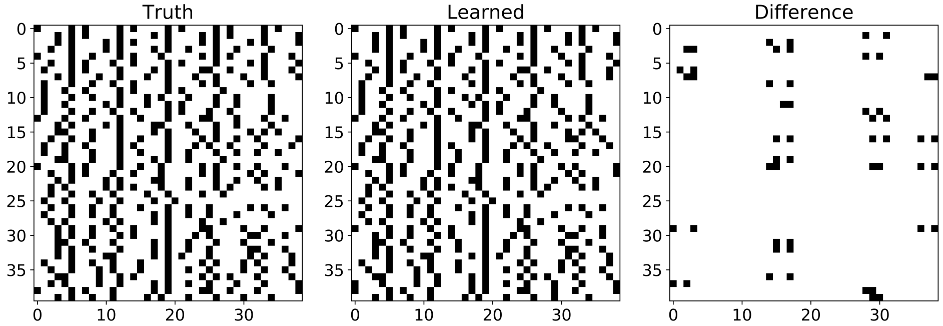

Figure 2. Recovery of a various kernel combinations after encoding and later decoding them with a SG-VAE for (see experimental section for further details). Black dots represent ones and white dots zeros. *Left*: original one-hot encoding vectors (one per row) representing kernel combinations. *Centre*: vectors produced by the SG-VAE after encoding and decoding (using the mode). *Left*: difference between the original and the mapped vectors.

```
Algorithm 2 Precomputation of the SG-VAE.
    Input: Dataset $\mathcal{D}$, Context-free grammar ($\mathcal{B}$, $p_{\mathcal{B}}$, $\mathcal{O}$, $p_{\mathcal{O}}$), $N_{max}$.
    1. Generate a dataset with Algorithm 1.
    2. Computer the representations $\mathbf{x} = [\mathbf{x}_g, \mathbf{x}_d]$.
    3. Optimize the ELBO of the SG-VAE.
    Returns: Pre-computed SG-VAE.
```

2.3. Structure Generating Variational Auto-encoder (SG-VAE)

This section describes a bespoke VAE, the Structure Generating Variational Auto-encoder (SG-VAE), for learning low dimensional representations of the kernel combinations represented by $\mathbf{x} = [\mathbf{x}_g, \mathbf{x}_d]$. The original VAE proposed by (Kingma & Welling, 2013) interprets points $\mathbf{z}$ as elements in a latent space of probabilistic generative model with two main components. A *decoder* given by a likelihood parametrized by θ, $p_\theta(\mathbf{x}|\mathbf{z})$ and a probabilistic *encoder* $q_\phi(\mathbf{z}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$ where $p(\mathbf{z})$ is the prior over the latent variables.

The parameters of $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ are optimized jointly by maximizing the the evidence lower bound (ELBO)

$$\mathcal{L}(\phi, \theta; \mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})\right],$$

which is done by using gradient descent as long as $p_\theta(\mathbf{x}|\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{x})$ are differentiable. SG-VAE extends VAE by proposing a type of encoder and decoder that are specialized in representing the structured search space.

2.3.1. SG-VAE Encoder

The *encoder* of the SG-VAE is a Gaussian distribution where the mean is the output of a *Multilayer Perceptron (MLP)* . In particular we define

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z} : \mu_\phi, \sigma^2_{en}\mathbf{I})$$
$$\mu_\phi = MLP(x; \phi)$$

where *MLP* represents a feed-forward network with two hidden layers and *tanh* activation function parametrized by weights ϕ and σ^2_{en} denotes the variance of the encoder Gaussian distribution.

2.3.2. SG-VAE Decoder

Although the encoder is somehow standard, the *decoder* is specific to the model representation problem that we are addressing. The challenge is to learn a decoder that always provides a valid kernel representation. To this end, we encode this feature in the model likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ as we next detail. First of all, define the output from the neural network

$$l = MLP(\mathbf{z}; \theta)$$

with the same MLP defined in the encoder and θ the corresponding weights that need to be optimized jointly with ϕ.

The likelihood factorizes in two independent components

$$p_\theta(\mathbf{x}|\mathbf{z}) = p_\theta(\mathbf{x}_d|\mathbf{z})p_\theta(\mathbf{x}_g|\mathbf{z})$$

that corresponds to the grammar and data representations $p_\theta(\mathbf{x}_d|\mathbf{z})$ and $p_\theta(\mathbf{x}_g|\mathbf{z})$ respectively. The data representation is modelled with a Gaussian likelihood

$$p_\theta(\mathbf{x}_d|\mathbf{z}) \sim \mathcal{N}(\mathbf{x}_d : \mu_\theta, \sigma^2_{de}\mathbf{I}).$$

where μ_θ is the part of the decoder parameters $\boldsymbol{l}$ associated to the data-based representation. In particular if we partition $\boldsymbol{l} = [\boldsymbol{l}_g, \boldsymbol{l}_d]$ we have that $\boldsymbol{l}_d = \mu_\theta$.

Algorithm 3 Structured Variational auto-encoded optimization, the SVO algorithm.

```
**Input:** Dataset $\mathcal{D}$, precomputed SG-VAE, model selection criterion $\mathcal{C}(\mathcal{M})$ and number of iterations $N_{iter}$.
1. Select initial point $\mathbf{z}_1$. Compute $q_\gamma(\mathbf{z}_1)$, the mode of the decoder $\tilde{\mathbf{x}}|\mathbf{z}_1$ and $\mathcal{C}(\mathcal{M}_{\tilde{\mathbf{x}}|\mathbf{z}_1})$.
2. Update the dataset $\tilde{\mathcal{D}}_1 = \{(q_\gamma(\mathbf{z}_1), \mathcal{C}(\mathcal{M}_{\tilde{\mathbf{x}}|\mathbf{z}_1}))\}$.
for $j = 1$ to $N_{iter}$ do
    3.1 Fit a GP with uncertain inputs to $\tilde{\mathcal{D}}_j$.
    3.2 Maximize the Expected Improvement to obtain $\mathbf{z}_{j+1}$.
    3.3 Evaluate the objective at the mode of the decoder $\mathcal{C}(\mathcal{M}_{\tilde{\mathbf{x}}|\mathbf{z}_{j+1}})$.
    3.4 Compute the variational approximation of the latent space $q_\gamma(\mathbf{z}_{j+1}|\tilde{\mathbf{x}})$.
    3.5 Augment data set: $\tilde{\mathcal{D}}_{j+1} = \{\tilde{\mathcal{D}}_j \cup (q_\gamma(\mathbf{z}_{j+1}), \mathcal{C}(\mathcal{M}_{\tilde{\mathbf{x}}|\mathbf{z}_{j+1}}))\}$.
end for
Returns: Report best obtained latent point $q(\mathbf{z}^\star)$ and the associated model $\mathcal{M}_{\mathbf{x}|\mathbf{z}^\star}$.
```

The part of the likelihood that corresponds to the grammar-based representation factorizes across all the kernels and operations, which are modelled via Multinomial distributions $Multi(n, \boldsymbol{\pi})$ where $\boldsymbol{\pi}$ are some normalised version of *softplus*(l_g) which is defined as $softplus(a) := \log(1+e^a)$ that maps to positive values. In particular, denote by $\mathbf{x}_{k_j}$ and $\mathbf{x}_{o_j}$ the one-hot representation of the j-th added kernel and operation. We write the grammar-based likelihood of the decoder as

$$
\begin{aligned}
& p_\theta(\mathbf{x}_g|\mathbf{z}) \\
& = p_\theta(\mathbf{x}_{k_1}|\mathbf{z}) \prod_{j=1}^{N_{max}-1} p_\theta(\mathbf{x}_{o_j}|\bar{\mathbf{x}}_{o_j}, \mathbf{z}) p_\theta(\mathbf{x}_{k_{j+1}}|\bar{\mathbf{x}}_{o_{j+1}}, \mathbf{z})
\end{aligned}
$$

where $\bar{\mathbf{x}}_{o_j} = \{\mathbf{x}_{o_1}, ..., \mathbf{x}_{o_{j-1}}\}$,

$$
\mathbf{x}_{o_j}|\bar{\mathbf{x}}_{o_j}, \mathbf{z} \sim \begin{cases} \textit{Multi}(1, \boldsymbol{\pi}_{o_j}) & \text{if none in } \bar{\mathbf{x}}_{o_j} \text{ is Stop} \\ 1 & \text{otherwise,} \end{cases}
$$

for $j = 1, \ldots, N_{max}$ and

$$
\mathbf{x}_{k_j}|\bar{\mathbf{x}}_{o_j}, \mathbf{z} \sim \begin{cases} \textit{Multi}(1, \boldsymbol{\pi}_{k_j}) & \text{if none in } \bar{\mathbf{x}}_{o_j} \text{ is Stop} \\ 1 & \text{otherwise,} \end{cases}
$$

for $j = 2, \ldots, N_{max}$. The vector parameters $\boldsymbol{\pi}_{o_j}$ and $\boldsymbol{\pi}_{k_j}$ of the multinomial are computed as the normalized versions of the corresponding blocks of $softplus(\boldsymbol{l}_g)$. Note that once the $Stop$ operation has been observed, the remaining $p_\theta(\mathbf{x}_{k_j}|\mathbf{z})$ and $p_\theta(\mathbf{x}_{o_j}|\mathbf{z})$ are set to one. In this way, there exists a conditional dependency of each kernel/operation on the previous factor, which refers to the appearance of $Stop$ operation. This procedure is summarized in algorithm 2.

The SG-VAE always learn vectors that directly map to feasible kernel combinations produced by the grammar (see Figure 2). Note also that this model is applicable in any scenario where a context-free grammar is available. The data-based representation can also be easily dropped if not needed in other domains.

2.3.3. Variational posterior for test points

To quantify the uncertainty of the latent variable $\mathbf{z}^\star$ of test points, we propose a separate mean-field variational inference, where the posterior distribution is parameterized as

$$
q_\gamma(\mathbf{z}^\star) = \mathcal{N}(\mathbf{z}^\star : \mu_\gamma, \sigma_\gamma^2 \mathbf{I}).
$$

This estimation of $q_\gamma(\mathbf{z}^\star)$ is done by minimizing the Kullback-Leibler divergence $\mathrm{KL}(q_\gamma(\mathbf{z}^\star) \| q_\phi(\mathbf{z}|\mathbf{x}))$. There are mainly two reasons why this is necessary to quantify the uncertainty. First, σ_{en}^2 is constant across all the latent space and one would expect a different level of uncertainty for different test points. Second, if a test point $\mathbf{x}^\star$ is dissimilar to any of the training points, there is no guarantee of good predictions by the encoder.

2.4. SG-VAE based search

Denote by $\mathcal{M}_\mathbf{x}$ the model configuration associated to the representation $\mathbf{x}$. For simplicity in the notation, in this section we will drop the dependence of the model on the parameters β. We denote by $\tilde{\mathbf{x}}|\mathbf{z}^\star$ the mode of the decoder p_θ at $\mathbf{z}^\star$. Also, let us denote by $\mathbb{Z}$ the latent space learned by the pre-trained SG-VAE. We reformulate Problem (1) as finding

$$
\mathbf{z}^{opt} = \arg\max_{\mathbf{z}^\star \in \mathbb{Z}} \ \mathcal{C}(\mathcal{M}_{\tilde{\mathbf{x}}|\mathbf{z}^\star}). \tag{2}
$$

The hypothesis is that points $\mathbf{z}^\star$ in the latent space, and models represented by the corresponding modes $\tilde{\mathbf{x}}|\mathbf{z}^\star$, are well mapped to each other, as it is shown in Figure 2 in Section 2.3. Therefore it is possible to search for the best model in $\mathbb{Z}$ rather than in the original space.

To find $\mathbf{z}^{opt}$ we use Bayesian optimization (Shahriari et al., 2016). We select a series of locations $\mathbf{z}_1, \ldots, \mathbf{z}_{N_{iter}}$ such that the maximum of $\mathcal{C}$ is evaluated as quickly as possible. Following the standard practice in BO, we use a Gaussian process $p(f) = \mathcal{GP}(\mu; k)$ with mean function μ and positive-definite kernel k to model the underlying objective function, now defined between $\mathbb{Z}$ and the domain of the

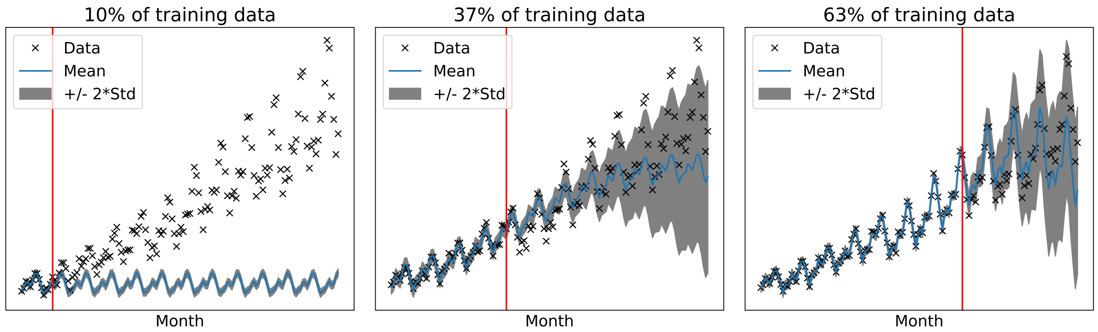

Figure 3. Results obtained by the SVO method in the Airline dataset using four base kernels. We search for the optimal combination with different sample sizes. The vertical red lines of the plots represent the separation between training data (on the left of the line) and test data (on the right).

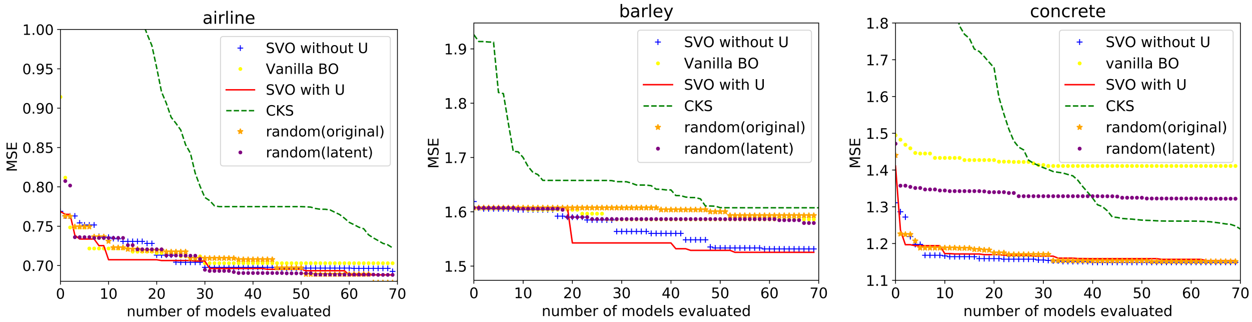

Figure 4. Comparison of the SVO algorithm and the compositional kernel search CKS. SVO is able to optimize the model selection criterion faster in all datasets. Interestingly, SVO does no need to start the seach on the base kernels which allows to consider more complex combinations faster that the CKS.

model selection criterion. In standard cases, the inputs of the GP are also the inputs of the objective function ($\mathcal{C}(\mathcal{M})$ in AS). In SVO, the inputs of the GP are the latent representations $\mathbf{z}$, which is different from the inputs of the objective function. This gives rise to a problem: a model configuration $\mathbf{x}$ may correspond to multiple points $\mathbf{z}$ in latent space. This uncertainty can be captured by the posterior distribution $p(\mathbf{z}|\mathbf{x})$, which is intractable in SG-VAE. Instead, we estimate a mean-field variational approximation $q_\gamma(\mathbf{z}|\mathbf{x})$ of the posterior detailed in Section 2.3.3. To factor in this uncertainty in the inference of GP, we use Gaussian process with uncertainty inputs (Damianou et al., 2016), in which we fit a Gaussian process between considering inputs following the distribution $q_\gamma(\mathbf{z}|\mathbf{x})$.

The posterior of the GP is used to create an acquisition function that is used to select the next points to evaluate. In this work we used the Expected Improvement (Mockus, 1977) (EI) but other acquisition functions are also possible. Note that, as we use mean and variance from the GP with uncertain inputs, the distribution $q_\gamma(\mathbf{z}^\star)$ is automatically pushed into the acquisition. The next evaluation is placed at the global maximum of the EI function (Shahriari et al., 2016). See Algorithm 3 for a full description of the algorithm that we call *Structured Variationally auto-encoded optimization* (SVO).

3. Experiments

We show three experiments. The first one explains the behaviour of the method in a time series. The goal is to interpret how new kernels are selected as soon as we provide more data to the system so more structure in the kernel can be inferred. The second experiment compares SVO with the *compositional kernel search* (CKS) (Duvenaud et al., 2013) method. The third one formulates natural scene understanding as a searching problem in structured space and apply SVO to infer the content in a natural scene. Inspired by (Wu et al., 2017), we use “Minecraft" as a nature scene generation engine and show SVO successfully produces a good interpretation of an image with a few attempts.

3.1. The Airline dataset

We apply the SVO algorithm to fit the Airline passenger data (Box & Jenkins, 1990), a time series that consists of

144 monthly totals of airline passengers from January 1949 to December 1960. We consider 4 base kernels: squared exponential (SE), linear (LIN), periodic (PER) and rational quadratic (RQD) and we optimize the mean squared error (MSE) of the prediction. We use uniform prior probabilities in both the grammar operations and the base kernels. We allow a maximum of 5 operations in the grammar generating process. We train a SG-VAE with 2 hidden layers with 400 hidden units on $20,000$ simulated kernel combinations (from which 4697 where unique) and 5 dimensions in the latent space. We apply Algorithm 3 using the first 10%, 37% and 63% of the data. The rest is used for testing. In Figure 3 we show the fit of the GP models with the combinations proposed by SVO. Interestingly, with 10% of the data the seasonal structure is captured. The solution found is $K = SE \times PER + RQ + PER$. With 37% of the data the solution found is $PER \times PER + PER + RQ + LIN$, which is capturing the linear trend. With 63% of the data the best solution found $PER \times SE \times RQ \times LIN + SE$. In this case, the method is capturing not only the linear trend but also the interactions with the seasonal components. SVO is able to extract coherent structure in the data.

3.2. Comparison across different optimization approaches

We do optimal model selection in GPs in three different datasets: the Airline dataset described in previous section and the Barley and Concrete[2]. The concrete dataset has dimension 8, while the rest are unidimensional. The experimental set up is the same as we used in section 3.1. We compare: SVO with different configurations. We learn the SG-VAE using both the grammar data and the distance representations described in Section 2.2. To generate the distances, we take as parameters of the base kernels those obtained after maximizing the marginal log-likelihood of the models. The parameters of the kernel combinations are sampled from hyper-priors that were specified as in (Malkomes et al., 2016). In the search, we include the case in which the uncertainty of the latent space of the SG-VAE is used using Section 2.3.3 and the case in which no uncertainty is propagated (just $\mathbf{z}^\star$ is used in the search). We compare different versions of the SVO (with uncertainty and without uncertainty) with (i) CKS, (ii) a random search carried out in the original kernel configurations space, (iii) a random search carried out in the latent space of the SG-VAE learned in each scenario and (iv) standard BO. A detailed description of each of the baselines can be found in the supplementary material.

In all cases we run the optimization for 70 iterations. In Figure 4 we show the best current MSE found in each iteration, which has been averaged over 5 runs for different initial points.

The CKS needs to compose complex kernels from single ones, which makes its convergence slow (it always starts with one kernel and sequentially composes more complex ones). In SVO, this is not needed as complex combinations can be selected from the beginning of the optimization. This is why the SVO convergence curves are flatter, which also converges faster and to a better optimum that CKS in the three datasets. Propagating the uncertainty of the SG-VAE has a positive effect in the search in the Barley dataset.

3.3. Natural Scene Understanding

Understanding natural scene is a challenging task because of the almost infinite number of possible combinations of objects and large variation of their appearance on image. On the other hand, we have sophisticated graphics rendering engines. We use computer graphics engine to improve scene understanding by formulating natural scene understanding as a search problem, i.e., given a natural image, whether we can find a configuration of a graphics rendering engine to reproduce the query.

Following the experiment setting of (Wu et al., 2017), we take “Minecraft” as the rendering engine and use XML as the description of configuration. We consider 12 Minecraft objects such as pig, cow, sheep and chicken, etc., and consider two attributes of each object, i.e., the location of an object on the image parametrized by the distance to the camera and the 1D angle from right to left. Both attributes are discretized into five different values. The task is to search for the XML configuration of which the generated image matches the target image as close as possible. For simplicity, we use L2 norm to measure the dissimilarity between a generated image and the target image, but more sophisticated dissimilarity measure can be used here.

The target images are shown in Figure 5a, 5b, 5c. We apply SVO to this problem by defining a structure generation process like the one in Section 2.3. We consider two types of operations: adding a new object and stop. When adding a new object, we choose its object type and the values of two attributes. With this generation process, we generated 20,000 configurations for training the SG-VAE. We use the grammar-based representation for simplicity. Then, we apply the search algorithm described in Section 2.4 to look for the best configuration. After 100 iterations, the image generated by the best configuration is shown in Figure 5e, 5f, 5g. The generated images closely match with the target ones with small disparity. Note that our approach is significantly different from (Wu et al., 2017). Wu et al. (2017) trained neural networks to directly predict a configuration given a target image. This requires generation of tens of

[2] All datasets are available at the UCI repository: https://archive.ics.uci.edu/ml/index.php exception the Barley, which can be found at https://datamarket.com

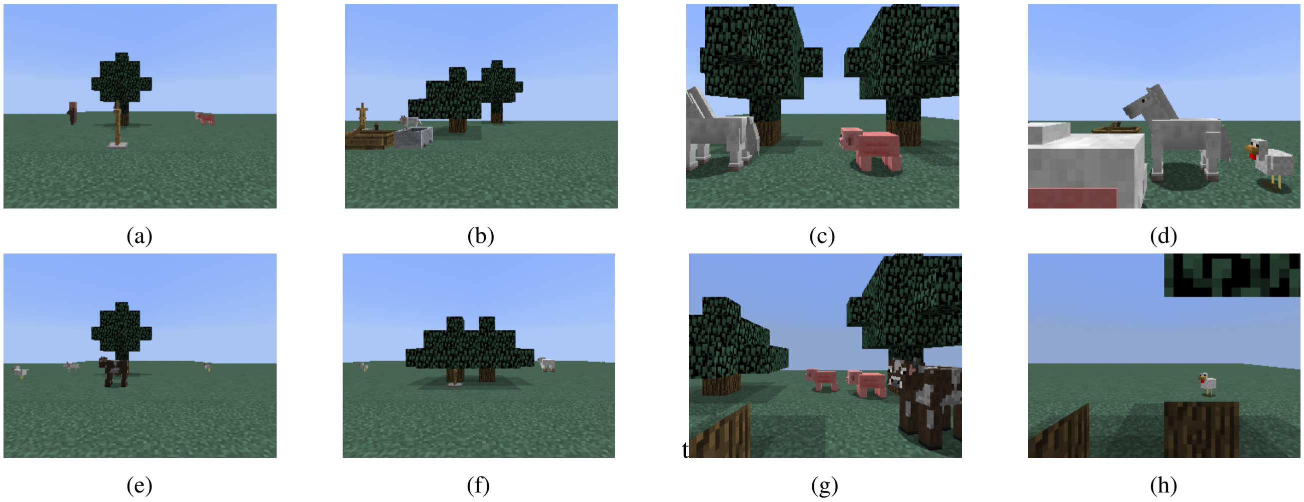

Figure 5. Use SVO to search for a XML configuration of the "Minecraft" engine to reproduce three target images (a), (b) and (c). The best configuration found by SVO are (e), (f) and (g) respectively. Images (d) and (h) were selected randomly to illustrate the complexity of the problem.

thousands images from the rendering engine for training the neural networks. The number of required training images can easily go a few magnitudes higher if more objects or attributes are considered. SVO is a meta-approach for this problem, where we *do not* learn a model to directly analyse images. Instead, we build a SG-VAE for configurations and search in the latent space of configurations for a matching explanation of the target image. Our approach only requires to generate a few images to guide the search.

4. Conclusions and future work

We have presented a new algorithm, SVO, for performing optimization in highly structured spaces. The idea of SVO is to learn a probabilistic low-dimensional latent space in which the solution of some high dimensional optimization problem can be easily found and mapped back to the original problem. This process is not possible without extra assumptions, which in our case are imposed by means of a context-free grammar. The SG-VAE, a new variational auto-encoder, explicitly uses the structure of the problem to generate feasible solutions. Bayesian optimization ideas are used in the latent space learned by the SG-VAE. The uncertainty in the latent space is propagated into the optimization using variational inference and a Gaussian process with uncertain inputs. The results obtained in the experimental section are competitive. However, we believe that there is still a large room for improvement in this research direction. For example, we see a limitation in the use of VAE as it is not well suited to learn the dimension of the latent space from the data, a GP-LVM could be a possible alternative. Other limitations of the SG-VAE are related smoothness and structure of the problem. Even if the dimensionality of the original space is large the method is expected to work if it is highly structured. However, in low dimensional spaces with not mush structure the SG-VAE is not expected to be beneficial (information cannot be compressed). It is also interesting to investigate how previous knowledge can be embedded in the search. In this work we used a context-free grammar to favour certain solutions of the input space, similarly to the idea of using transfer learning when similar problems have been solved before. Further combinations of this ideas will be explored in the future.

References

Abbati, G., Tosi, A., Osborne, M. A., and Flaxman, S. Adageo: Adaptive geometric learning for optimization and sampling. In *NIPS workshop in Approximate inference*, 2017.

Box, G. E. P. and Jenkins, G. *Time Series Analysis, Forecasting and Control*. Holden-Day, Incorporated, 1990. ISBN 0816211043.

Dai, Z., Damianou, A., González, J., and Lawrence, N. Variational auto-encoded deep Gaussian processes. *International Conference on Learning Representations (ICLR)*, 2016.

Damianou, A. C., Titsias, M. K., and Lawrence, N. D. Variational inference for latent variables and uncertain inputs in Gaussian processes. *Journal of Machine Learning Research*, 17(42):1–62, 2016.

Duvenaud, D., Lloyd, J. R., Grosse, R., Tenenbaum, J. B., and Ghahramani, Z. Structure discovery in nonparametric regression through compositional kernel search. *arXiv preprint arXiv:1302.4922*, 2013.

Gonzalez, J., Longworth, J., James, D. C. J., and Lawrence, N. D. Bayesian optimization for synthetic gene design. In *NIPS workshop in Bayesian optimization*, 2014.

Grosse, R., Salakhutdinov, R. R., Freeman, W. T., and Tenenbaum, J. B. Exploiting compositionality to explore a large space of model structures. *arXiv preprint arXiv:1210.4856*, 2012.

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. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276, 2018. doi: 10.1021/acscentsci.7b00572.

Hopcroft, J. E., Motwani, R., and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006. ISBN 0321455363.

Hwang, Y. and Choi, J. The automatic statistician: A relational perspective. *CoRR*, abs/1511.08343, 2015.

Jenatton, R., Archambeau, C., González, J., and Seeger, M. Bayesian optimization with tree-structured dependencies. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1655–1664, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Kim, H. and Teh, Y. W. Scalable structure discovery for regression using gaussian processes. *AutoML 2016 Proceedings, Journal of Machine Learning Research Workshop and Conference Proceedings*, 2016.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*, 2017.

Malkomes, G., Schaff, C., and Garnett, R. Bayesian optimization for automated model selection. In *Advances in Neural Information Processing Systems*, pp. 2900–2908, 2016.

Mockus, J. On bayesian methods for seeking the extremum and their application. In *IFIP Congress*, pp. 195–200, 1977.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104 (1):148–175, 2016.

Swersky, K., Duvenaud, D., Snoek, J., Hutter, F., and Osborne, M. A. Raiders of the lost architecture: Kernels for Bayesian optimization in conditional parameter spaces. In *NIPS workshop on Bayesian Optimization in theory and practice (BayesOpt?13)*, 2013.

Wang, Z., Hutter, F., Zoghi, M., Matheson, D., and De Freitas, N. Bayesian optimization in a billion dimensions via random embeddings. *J. Artif. Int. Res.*, 55(1):361–387, January 2016. ISSN 1076-9757.

Wu, J., Tenenbaum, J. B., and Kohli, P. Neural scene derendering. In *CVPR*, 2017.