Hyperparameter Optimization Machines

Martin Wistuba
Information Systems and
Machine Learning Lab
University of Hildesheim
Universitätsplatz 1,
31141 Hildesheim,
Germany
wistuba@ismll.
uni-hildesheim.de

Nicolas Schilling
Information Systems and
Machine Learning Lab
University of Hildesheim
Universitätsplatz 1,
31141 Hildesheim,
Germany
schilling@ismll.
uni-hildesheim.de

Lars Schmidt-Thieme
Information Systems and
Machine Learning Lab
University of Hildesheim
Universitätsplatz 1,
31141 Hildesheim,
Germany
schmidt-thieme@ismll.
uni-hildesheim.de

***Abstract*—Algorithm selection and hyperparameter tuning are omnipresent problems for researchers and practitioners. Hence, it is not surprising that the efforts in automatizing this process using various meta-learning approaches have been increased. Sequential model-based optimization (SMBO) is one of the most popular frameworks for finding optimal hyperparameter configurations. Originally designed for black-box optimization, researchers have contributed different meta-learning approaches to speed up the optimization process. We create a generalized framework of SMBO and its recent additions which gives access to adaptive hyperparameter transfer learning with simple surrogates (AHT), a new class of hyperparameter optimization strategies. AHT provides less time-overhead for the optimization process by replacing time- and space-consuming transfer surrogate models with simple surrogates that employ adaptive transfer learning. In an empirical comparison on two different meta-data sets, we can show that AHT outperforms various instances of the SMBO framework in the scenarios of hyperparameter tuning and algorithm selection.**

I. Introduction

Algorithm selection and hyperparameter tuning are omnipresent problems for researchers and practitioners. The selection of an algorithm for a specific problem and furthermore the respective hyperparameter configuration has a vital impact on the quality of the final predictions. The most conventional method for selecting the algorithm is usually based on the practitioner's past experience, the hyperparameters are then usually tuned using a combination of manual search and grid or random search. This has two drawbacks. First, inexperienced researchers will have difficulties in choosing the right combination of algorithm and hyperparameter configuration. Second, finding the best hyperparameter configuration by using a grid search will be a time-consuming task. For bigger data sets and more advanced algorithms, only few hyperparameter evaluations are feasible with respect to the whole search space.

Recent research proposes automatic algorithm selection and hyperparameter tuning as the solution for these problems. It has been shown that this task can be solved in less computational time and additionally finds hyperparameter configurations that are better than those found by human domain experts [2], [16]. Algorithm selection is a well-studied problem that is not limited to machine learning but also finds application in artificial intelligence and operations research [20]. Recently, a program for combined algorithm selection and hyperparameter tuning was published for the well-known data mining tool WEKA [23]. The current direction of research tries to mimic the tuning behavior of human experts. The information of past tuning processes is transferred to current tuning processes either by initializing the tuning process by trying configurations that performed well on previous experiments [18], [8], [25] or by using specific machine learning models that predict the performance of an algorithm and hyperparameter configuration on the current problem [1], [21], [26], [19].

Our contributions in this work are twofold: We propose hyperparameter optimization machines (HOM), a hyperparameter tuning framework that is a generalization of SMBO and its recent meta-learning additions. Second, we propose a novel instance of HOM. Its advantage is less space and time requirements in comparison to transfer surrogate models and adaptivity within the first trials in comparison to initialization techniques. In extensive experiments on two different meta-data sets created from 50 data sets, our proposed approach is able to outperform all seven state of the art hyperparameter tuning methods we compared to.

II. Related Work

Sequential model-based optimization (SMBO) [14] was proposed to be used for hyperparameter optimization and has proven to be effective [2]. Different surrogate models were proposed [13] as well as the use of meta-knowledge. Some work proposes to make use of meta-knowledge with initializations [8], [25], others propose transfer surrogate models [1], [21], [26], [19]. Furthermore, SMBO has also been applied for combined algorithm and hyperparameter selection [23]. In this work we want to summarize all these contributions within the hyperparameter optimization machine (HOM). This generalization allows a new class of tuning strategies which will be investigated in this work.

Besides SMBO, hyperparameter optimization strategies based on optimization techniques from artificial intelligence such as tabu search [4], particle swarm optimization [10]

and evolutionary algorithms [9] exist. Bandit optimization techniques were recently proposed for automatic machine learning [12]. There is plenty of work focusing on algorithm selection [20], some work tries to adapt this to hyperparameter tuning as well [15].

III. Hyperparameter Optimization Problem

A machine learning algorithm $\mathcal{A}_{\boldsymbol{\lambda}}$ can be understood as a mapping $\mathcal{A}_{\boldsymbol{\lambda}} : \mathcal{D} \rightarrow \mathcal{M}$ where $\mathcal{D}$ is the set of all data sets, $\mathcal{M}$ is the space of all models and $\boldsymbol{\lambda} \in \Lambda$ is the chosen hyperparameter configuration with $\Lambda = \Lambda_1 \times \ldots \times \Lambda_p$ being the p-dimensional hyperparameter space of algorithm $\mathcal{A}$. The learning algorithm estimates a model $M_{\boldsymbol{\lambda}} \in \mathcal{M}$ that minimizes a loss function $\mathcal{L}$ (e.g. misclassification rate) with its model regularization term $\mathcal{R}$:

$$\mathcal{A}_{\boldsymbol{\lambda}}\left(D^{(\text{train})}\right) := \underset{M_{\boldsymbol{\lambda}} \in \mathcal{M}}{\arg\min}\ \mathcal{L}\left(M_{\boldsymbol{\lambda}}, D^{(\text{train})}\right) + \mathcal{R}\left(M_{\boldsymbol{\lambda}}, \boldsymbol{\lambda}\right). \tag{1}$$

Then, the task of *hyperparameter optimization* is to find the optimal hyperparameter configuration $\boldsymbol{\lambda}^*$ using a validation set i.e.

$$\boldsymbol{\lambda}^* := \underset{\boldsymbol{\lambda} \in \Lambda}{\arg\min}\ \mathcal{L}\left(\mathcal{A}_{\boldsymbol{\lambda}}\left(D^{(\text{train})}\right), D^{(\text{valid})}\right) =: f_D\left(\boldsymbol{\lambda}\right). \tag{2}$$

For demonstration purposes, in the remaining sections we consider the problem of tuning the hyperparameters of a classifier. Thus, the hyperparameter response function f_D will be the misclassification rate. This is no limitation but shall help the reader to understand the concepts at a concrete example.

Algorithm 1 Hyperparameter Optimization Machines

Input: Hyperparameter space Λ, observation history $\mathcal{H}$, transfer function $\mathfrak{T}$, acquisition function a, surrogate model Ψ, trade-off parameter $\boldsymbol{\alpha}$, hyperparameter response function f to be minimized, total number of HOM iterations T.
Output: Best hyperparameter configuration found.

1: $\Lambda_0 \leftarrow \emptyset$, $f_{\text{best}} \leftarrow \infty$
2: **for all** $t = 1 \ldots T$ **do**
3: Fit Ψ to $\mathcal{H}$ (Update)
4: $\boldsymbol{\lambda} \leftarrow \arg\min_{\boldsymbol{\lambda}' \in \Lambda} (1 - \boldsymbol{\alpha}_t)\, \mathfrak{T}\left(\boldsymbol{\lambda}', \Lambda_{t-1}\right) - \boldsymbol{\alpha}_t a\left(\boldsymbol{\lambda}', \Psi\right)$ (Predict)
5: $\Lambda_t \leftarrow \Lambda_{t-1} \cup \{\boldsymbol{\lambda}\}$
6: Evaluate $f\left(\boldsymbol{\lambda}\right)$
7: $\mathcal{H} \leftarrow \mathcal{H} \cup \{(\boldsymbol{\lambda}, f\left(\boldsymbol{\lambda}\right))\}$
8: **if** $f\left(\boldsymbol{\lambda}\right) < f_{\text{best}}$ **then**
9: $\boldsymbol{\lambda}_{\text{best}},\ f_{\text{best}} \leftarrow \boldsymbol{\lambda},\ f\left(\boldsymbol{\lambda}\right)$
10: **return** $\boldsymbol{\lambda}_{\text{best}}$

IV. Hyperparameter Optimization Machines

Sequential model-based optimization (SMBO) [14] was originally proposed for black-box optimization. It was designed for finding an optimum of a function f which is expensive to evaluate. A surrogate model Ψ is used that tries to approximate f but has the advantage of being less time-consuming to evaluate. This surrogate model is combined with an acquisition function a [14] to tackle the exploitation-exploration dilemma. SMBO has been applied to the problem of hyperparameter optimization [21] by minimizing the function f_D defined in Equation 2 and has proven to be very effective. Since then, researchers have adapted and specialized SMBO for the hyperparameter optimization problem.

A specific property of hyperparameter optimization beyond mere black-box optimization is that the hyperparameter response function of a specific algorithm behaves similar on similar data sets. Furthermore, some hyperparameter configurations provide decent results on average which are usually proposed as default hyperparameter configurations. Hence, information about the data sets and information about previous experiments can be used by practitioners to find good initial hyperparameter configurations. This important difference between arbitrary black-box optimization and hyperparameter optimization and the fact that meta-knowledge already helped for other hyperparameter optimization techniques [18], gave rise to various transfer learning approaches based on initialization [8], [25], transfer surrogate models [1], [19], [22], [26] and pruning [24] and has empirically proven its usefulness. Finally, the sequential hyperparameter evaluation was adapted to be capable of searching in parallel to exploit the hardware available in our days [21].

For all these reasons, the name *sequential model-based optimization* has become outdated and does no longer seem appropriate in the context of hyperparameter optimization. Therefore, we want to propose hyperparameter optimization machines as a generalization of SMBO which covers the recent contributions. Algorithm 1 outlines our aforementioned generalization. Like the SMBO, it consists of a surrogate model Ψ and an acquisition function a. Iteratively, Ψ is updated using all observations $\mathcal{H}$ and then the next hyperparameter configuration is selected (Line 3 and 4). In contrast to the state of the art, this is done by using a linear combination of the acquisition function a and a new component, the transfer function $\mathfrak{T}$. We consider the negative value of the acquisition function a because a higher value means better values and we want to minimize the combined term. As will be detailed in the following, this covers SMBO with initialization as well as our proposed tuning strategy proposed in Section V. The parameter $\boldsymbol{\alpha}$ is a meta-hyperparameter that controls the influence of transferred meta-knowledge and the knowledge gathered about the new data set so far. A reasonable $\boldsymbol{\alpha}$ is one that is close to 0 for small $t \in \mathbb{N}$ and then increases over time to 1. Thus, meta-knowledge plays a major role for the selection of the first hyperparameter configurations and becomes irrelevant as soon as enough knowledge is gathered about the new data set. After selecting the most promising hyperparameter configuration, it is evaluated and the result is added to $\mathcal{H}$. This is repeated until T configurations are evaluated.

A. Relationship between HOM and SMBO

In the last section we introduced hyperparameter optimization machines (HOM) with the goal to generalize across the contributions that have been done to SMBO for hyperparam-

eter optimization. Hence, we will now map most relevant hyperparameter optimization methods related to the SMBO framework to HOM.

SMBO itself is obviously a specialization of HOM in the case that $\boldsymbol{\alpha} = \mathbf{1}$ such that the transfer function $\mathfrak{T}$ will not be considered or for arbitrary $\boldsymbol{\alpha}$ if $\mathfrak{T}$ is a constant function. In the following, we will ignore that $\mathfrak{T}$ can be a constant function and assume that it somehow reflects the meta-knowledge, meaning that $\mathfrak{T}(\Lambda_t)$ is lower if $\Lambda_t \subset \Lambda$ contains hyperparameter configurations that performed well in previous experiments and vice versa. It is important to notice, that HOM is not an instance of SMBO with a specific acquisition function. The expression in Line 4 of Algorithm 1 and an acquisition function have in common that they acquire the next hyperparameter configuration but the acquisition function of SMBO depends only on the hyperparameter configuration and its predicted performance but is independent of the time and previously selected hyperparameter configurations.

The work that is proposing a specific surrogate model for SMBO [1], [13], [19], [21], [22], [26] is hence also an instance of HOM. Choosing an appropriate transfer function, SMBO with I initialization steps [8], [25] is a special case of HOM where $\boldsymbol{\alpha}_t = 0$ if $t \leq I$ and 1 otherwise.

Finally, even random search [3] and grid search can be considered as an instance of HOM with a very specific acquisition function and with no need for a surrogate model.

V. Adaptive Hyperparameter Transfer Learning with Simple Surrogates

So far two different ways of exploiting meta-knowledge in the SMBO framework are common. One option is to use an initialization [8], [25] and combine it with *simple surrogate models*, models that are learned only on observations of the current data set. The advantage in this case is that it does not need any additional run time during the optimization process compared to the plain SMBO framework. Otherwise, the initialization sequence is static and does not consider the first trials. In contrast, *transfer surrogate models* [1], [19], [22], [26] (machine learning models that are learned on the observations on the current data and on observations of past experiments on other data sets) adaptively consider the meta-knowledge but they are costly in terms of space and time. Finally, applying meta-knowledge by initialization or transfer surrogate models achieves similar performances [25].

In this section we propose our main contribution, a new instance of hyperparameter optimization machines which we call *Adaptive Hyperparameter Transfer Learning with Simple Surrogates* (AHT). The idea is to make use of the newly introduced transfer function $\mathfrak{T}$ and combine it with simple surrogates while letting $\boldsymbol{\alpha}$ adopt arbitrary values between 0 and 1. This will lead to a new hyperparameter optimization strategy that tries to combine the advantages of both, initialization and transfer surrogate models, and reduce their drawbacks. In the following sections, we will derive a transfer function $\mathfrak{T}$ that tries to minimize a meta-loss and theoretically investigate the asymptotic space and time requirements compared to simple and transfer surrogates, respectively.

A. Meta-Loss

Many machine learning problems are solved by learning parameters of a model by minimizing a predefined loss function. Analogously, Wistuba et al. proposed a meta-loss for hyperparameter optimization [25]. This is based on the idea that the goal is to minimize the distance to the global minimum $f_D^{\min}$ (DTM) after trying a subset of hyperparameter configurations $\Lambda_T \subset \Lambda$:

$$\mathrm{DTM}(\Lambda_T, D) := \min_{\boldsymbol{\lambda} \in \Lambda_T} f_D(\boldsymbol{\lambda}) - f_D^{\min}. \tag{3}$$

The computation of the DTM over a set of data sets should not be computed by simply averaging all DTMs because the scales and offsets of the hyperparameter performance of different data sets can vary much and hence provide unequal influence of each individual data set. Thus, the average DTM is achieved by averaging after scaling the hyperparameter performance to the interval $[0, 1]$:

$$\mathrm{ADTM}(\Lambda_T, \mathcal{D}) := \frac{1}{|\mathcal{D}|} \sum_{D \in \mathcal{D}} \min_{\boldsymbol{\lambda} \in \Lambda_T} \frac{f_D(\boldsymbol{\lambda}) - f_D^{\min}}{f_D^{\max} - f_D^{\min}}. \tag{4}$$

In the remainder of this paper whenever f is used, the scaled version of it is meant. We want to highlight that the true minimum and maximum is unknown and will be approximated using the extreme values as plug-in estimates.

B. Evaluating Hyperparameter Configurations based on Meta-Knowledge

So far the evaluation of a hyperparameter configuration is based on an acquisition function that takes the predicted value and uncertainty into account. During this selection, meta-information can be used indirectly with transfer surrogate models. HOMs allow to introduce our new approach AHT that uses simple surrogates and still accelerates the optimization process as well as decreases the needed memory. The use of meta-knowledge is solely based on the transfer function $\mathfrak{T}$. Then Line 4 of Algorithm 1 combines the knowledge about the current data set and the knowledge about past experiments. Yet, it is unclear what properties are required for a good transfer function. We define two requirements: First, hyperparameter configurations that performed well on previous data set should be rated higher than others. Second, with increasing information about the new data set, the influence of the meta-information should vanish. The second requirement can also be achieved by choosing $\boldsymbol{\alpha}$ accordingly but this would inflate the number of meta-hyperparameters for the HOM which we want to avoid.

Since our ultimate goal is to minimize the DTM on the new data set, we use the ADTM on the previous data sets as a proxy for rating the hyperparameter configurations. Thus, the transfer function can be derived from Equation 4 and we define it as

$$\mathfrak{T}(\boldsymbol{\lambda}, \Lambda_{t-1}) := \frac{1}{|\mathcal{D}|} \sum_{D \in \mathcal{D}} \min_{\boldsymbol{\lambda}' \in \Lambda_{t-1} \cup \{\boldsymbol{\lambda}\}} f_D(\boldsymbol{\lambda}') \tag{5}$$

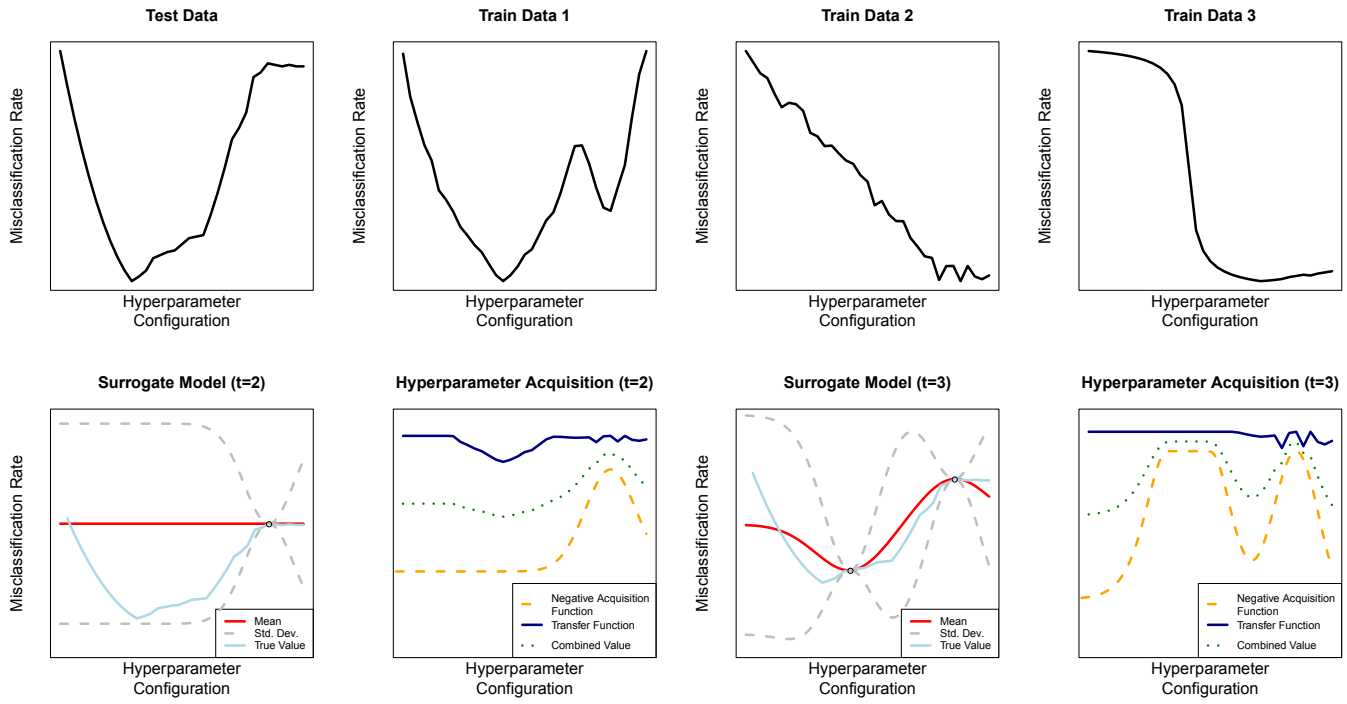

Fig. 1. First row: Hyperparameter response surfaces of the current data set where we want to find the best hyperparameter configurations and of three data sets which have been investigated before (meta-data). Second row: Sequential process of AHT. One can clearly see the positive impact of the transfer function on the hyperparameter configuration selection in unexplored areas. In all plots: the lower the better.

where Λ_{t-1} is the set of evaluated hyperparameter configurations after $t-1$ trials. The hyperparameter configuration $\boldsymbol{\lambda}$ that minimizes Equation 5 is that one that has reduced the misclassification rate the most on previous data sets if the hyperparameter configurations in Λ_{t-1} have been tried already.

One problem remains. The functions f_D are only partially observed, hence Equation 5 cannot be computed for arbitrary hyperparameter configurations $\boldsymbol{\lambda}$. To achieve this, we replace f_D with plug-in estimators Ψ_D that are trained on the available observations of data set D and approximate the true f_D. These observations are part of the meta-data set and hence do not involve any further evaluations of f_D. Similarly, the surrogate model Ψ is trained only on the observations of the new data set.

The effect of the transfer function for $0 < \boldsymbol{\alpha}_t < 1$ will be twofold. First, it will serve as some kind of soft initialization where likely good hyperparameters are preferred by taking into account the little knowledge that has been gathered on the new data set. Second, it fulfills the criterion that the influence of the meta-knowledge is no longer used as soon as enough trials on the new data set have been performed. This will be achieved by the minimization term in Equation 5 that will ensure that the transfer function loses ground over time on the hyperparameter configuration selection decision. *The resulting side effect is that we will choose the same value for all* $\boldsymbol{\alpha}_t$.

We will explain and illustrate the impact of our proposed combination of the acquisition function with the transfer function in Figure 1. In the first row of Figure 1 the misclassification rate of a classifier with a one-dimensional hyperparameter space for four different data sets are shown. The task is now to estimate the hyperparameter configuration for the test data (upper left plot) with smallest error using the information gathered on the other three data sets.

For illustration purposes we start at $t = 2$ that means after already evaluating the model with one hyperparameter configuration. We are using a Gaussian process as a surrogate model and expected improvement [14] as acquisition function. As transfer function we use the function that we derived in Equation 5. The surrogate model got updated using the single observation (Figure 1; "Surrogate Model (t=2)") and the values of the acquisition function and the transfer function can be estimated (Figure 1; "Hyperparameter Acquisition (t=2)"). The standard SMBO is depending only on the acquisition function (dashed **orange** line) for selecting the next hyperparameter configuration and not directly on the meta-knowledge. Hence, it would choose a hyperparameter configuration at the left border of the hyperparameter search space. Computing the transfer function (solid **blue** line), we can see that the transfer function has lower values in which we can expect lower (that means better) function values. Even though data sets 1 and 2 indicate that the right region of the hyperparameter space is good for some data, the transfer function has high values here. The reason for this is that we already evaluated a hyperparameter configuration in that region such that the improvement in these regions likely is small.

Finally, the addition of the transfer function to the Hyperpa-

TABLE I

COMPARISON OF TIME AND SPACE REQUIREMENTS. THE MEMORY REQUIREMENTS OF GP WITH TRANSFER FUNCTION IS ONLY LINEAR IN THE NUMBER OF DATA SETS d AND THE UPDATE TIME IS INDEPENDENT OF THE SIZE OF THE META-DATA SET ASSUMING THAT IN EACH PREVIOUS EXPERIMENT n OBSERVATIONS ARE GATHERED AND ON THE NEW DATA SET t ARE GATHERED SO FAR.

	Simple GP [21], [8], [25]	Transfer GP [1], [22], [26]	GP with transfer function (this work)
Training (offline)	-	$\mathcal{O}\left(d^3n^3\right)$	$\mathcal{O}\left(dn^3\right)$
Update (online)	$\mathcal{O}\left(t^2\right)$	$\mathcal{O}\left(t^2 + d^2n^2 + dnt\right)$	$\mathcal{O}\left(t^2\right)$
Prediction (online)	$\mathcal{O}\left(t\right)$	$\mathcal{O}\left(t + dn\right)$	$\mathcal{O}\left(t + dn\right)$
Space (online)	$\mathcal{O}\left(t^2\right)$	$\mathcal{O}\left(t^2 + d^2n^2 + dnt\right)$	$\mathcal{O}\left(t^2 + dn^2\right)$

rameter Optimization Machine framework allows us not only to find a balance between exploration and exploitation on the current data set but also on the usage of meta-knowledge by adding some weight on regions that have been good on other data sets (dotted **green** line). Based on the smallest value of the combination of acquisition and transfer function, the next hyperparameter configuration is chosen. Again the surrogate model is updated and the next hyperparameter configuration can be estimated. In this simple example with low-dimensional hyperparameter configurations and little meta-data, one can see already now that the transfer function loses influence over time. As explained before, this is a wanted effect because at some point the knowledge on the current data set is sufficient.

This simple example motivates the improvement of AHT over plain SMBO without meta-knowledge. If meta-knowledge is used there are so far two options. One is to use initializations. Compared to AHT this will lead to a fixed number of initial trials no matter, if we already know that this is a bad hyperparameter region or not. AHT will not make this mistake since it is using the information of previous trials. The advantage of AHT over transfer surrogates cannot be shown in a simple one-dimensional example but we will try to explain it here and prove it empirically in the course of the paper. As we have seen in the example, AHT is using the meta-knowledge in a way that the meta-knowledge is losing its influence over time. Transfer surrogates can handle this only to a certain degree. This will be an important issue for transfer surrogates, if they are applied to a data set where hyperparameters behave completely different to the majority of data sets in the meta-data set. At any point of time the transfer surrogate is more biased to hyperparameter configurations that have been good on the meta-data set while AHT will at some point ignore the meta-data completely and rely on the data collected of the new data set only.

C. Space and Time Requirements

In the following we discuss the time and space complexity of the three different approaches of using meta-knowledge in HOM: 1) initialization combined with a simple surrogate, 2) transfer surrogate model and 3) adaptive hyperparameter transfer learning with simple surrogates (AHT; Section V). This discussion is led for the case that the surrogate model is based on a Gaussian process (GP), the most widespread approach [1], [22], [26].

Assuming the meta-knowledge of d data sets is available and for each of these data sets, for simplicity, n observations are available while t is the number of observations of the new, unknown data set. We distinguish between the three most time-consuming operations in the HOM. One is the *training* operation, this includes all operations that can be done before the actual optimization process. For transfer surrogate models that includes estimating the parameters of the surrogate model on the meta-data, for initialization techniques estimating the initial hyperparameter sequence and our approach will estimate the plug-in estimators during this step. In comparison to the other operations, this can be done offline and hence the time needed for this operation is of less interest for us. The second operation is *update* (see Algorithm 1). This operation has to be done once for each evaluation of a hyperparameter configuration. It updates the surrogate model such that the newly gathered information is considered. The last operation is *prediction.* This is the operation that evaluates the quality of a single hyperparameter configuration. Table I summarizes the space and time complexity for the different operations. The transfer surrogate models [1], [22], [26] need more time for offline and online computations than the combination of a GP with transfer function, respectively. Gaussian processes need to store the kernel matrix. Hence, the space complexity of transfer surrogates is quadratic in the number of meta-data sets instead of only linear. Obviously, the simple GP is beneficial in terms of space and time complexity. No offline training is needed in cases without initialization [21] but investing time to use the meta-knowledge pays off in general [8], [25].

VI. EXPERIMENTAL EVALUATION

Our proposed instance of hyperparameter optimization machines AHT will be compared to up to seven different competitor strategies on two different meta-data sets. First, we compare the methods in the scenario of hyperparameter tuning only. This is carried out on a meta-data set generated on 50 different data sets with the LIBSVM library [5]. This smaller meta-data set allows the comparison to transfer surrogate models that are based on Gaussian processes. Finally, we apply our method in the scenario of combined algorithm selection and hyperparameter tuning on a meta-data set generated by using 19 different classifiers of WEKA [11] on 59 different data sets.

A. Tuning Strategies

In our empirical evaluation we will compare representatives of the following five types of hyperparameter optimization machines. Those that do not use a surrogate model at all, those

that use a simple surrogate model and no meta-knowledge, those that use an initialization to employ meta-knowledge combined with simple surrogates, and finally our proposed method that makes use of the transfer function. We want to remark that we are comparing to the most recent work in our field we are aware of, including work that has been published on last year's ECML and DSAA.

1) No Surrogate Model: Random Search (Random) is selecting hyperparameters at random and hence does not use a surrogate model. Bergstra and Bengio [3] have shown that a random hyperparameter search is able to outperform a grid search in cases with hyperparameters with low effective dimensionality.

2) Simple Surrogates: Instead of using the name proposed by the original authors, we decided to give following tuning strategies a different name, to make their relationship to SMBO/HOM clear.

Independent Gaussian Process (I-GP): This tuning strategy is better known as Spearmint [21]. It makes use of a Gaussian process with squared-exponential kernel with automatic relevance determination (SE-ARD) as a surrogate model. No knowledge from previous experiments is considered.

Independent Random Forest (I-RF): Sequential Model-based Algorithm Configuration or for short SMAC is another popular approach for tuning hyperparameters [13]. In contrast to Spearmint (I-GP) it makes use of random forests instead of a Gaussian process. Again, no knowledge from previous experiments is employed. I-RF is used in AutoWEKA [23].

3) Simple Surrogates with Initialization: Simple surrogates can be improved by employing meta-initializations. We decided to make use of Learning Initialization which has shown to outperform simpler meta-initializations [25]. We abbreviate these variations of I-GP and I-RF with I-GP (init) and I-RF (init), respectively. I-RF with a meta-initialization is used in Auto-SKLearn [7].

4) Transfer Surrogate Strategies: Transfer surrogate strategies make use of the meta-knowledge employing a special surrogate model. We are comparing to following three strategies in this category.

Surrogate Collaborative Tuning (SCoT): Bardenet et al. propose to employ meta-knowledge in two steps [1]. First, an SVMRank is learned over the whole meta-data. The predictions are used to replace the labels of the meta-instances. The authors argue that this solves the problem of different scales on different data sets. On this transformed meta-data set, a Gaussian process with SE-ARD kernel is trained. In the original work, it was proposed to use an RBF kernel for SVMRank. For reasons of computational complexity, we follow the lead of Yogatama and Mann [26] and use a linear kernel instead.

Gaussian Process with Multi-Kernel Learning (MKL-GP): Yogatama and Mann [26] propose to employ a Gaussian process trained on the whole meta-data set. To overcome the problem of different scales on different data sets, the meta-data is normalized. Furthermore, they propose the use of a new kernel which is a linear combination of an SE-ARD kernel and a kernel modeling the distance between data sets.

TABLE II
THE LIST OF ALL META-FEATURES USED BY US.

Meta-Features	
Number of Classes	Class Probability Max
Number of Instances	Class Probability Mean
Log Number of Instances	Class Probability Std. Dev.
Number of Features	Kurtosis Min
Log Number of Features	Kurtosis Max
Data Set Dimensionality	Kurtosis Mean
Log Data Set Dimensionality	Kurtosis Standard Deviation
Inverse Data Set Dimensionality	Skewness Min
Log Inverse Data Set Dimensionality	Skewness Max
Class Cross Entropy	Skewness Mean
Class Probability Min	Skewness Standard Deviation

Factorized Multilayer Perceptron (FMLP): FMLP [19] is the most recent transfer surrogate model we are aware of. Published on last year's ECML PKDD, it is using a specific neural network to learn the similarity between the new data set and those from previous ones implicitly in a latent representation.

5) Our Proposed Strategy: Adaptive hyperparameter transfer learning with simple surrogates (AHT; Section V) is an instance of hyperparameter optimization machines proposed by us. In the experiments, we will consider AHT with two different simple surrogates: a Gaussian process (AHT-GP) and a random forest (AHT-RF).

All Gaussian processes use a squared exponential kernel with automatic relevance determination (SE-ARD), the kernel parameters are estimated by maximizing the marginal likelihood [17]. Hyperparameters of the optimization strategies are estimated using a grid search on the meta-data using leave-one-data-set-out cross-validation.

B. Meta-Data Sets

For the creation of the support vector machine (SVM) meta-data set, we made use of 50 classification data sets chosen at random from the UCI repository. Existing train/test splits were merged, shuffled and split into 80% train and 20% test. We added the 22 meta-features described in Table II.

We trained an SVM [5] for a linear, polynomial and Gaussian kernel. The resulting hyperparameters are kernel indicator variables, the trade-off parameter C, the degree of the polynomial kernel d and the width γ of the Gaussian kernel. If a hyperparameter was not involved, its value was set to 0. We precomputed the misclassification rate on the grid $C \in \left\{2^{-5}, \ldots, 2^{6}\right\}$, $d \in \{2, \ldots, 10\}$ and $\gamma \in \left\{10^{-4}, 10^{-3}, 10^{-2}, 0.05, 0.1, 0.5, 1, 2, 5, 10, 20, 50, 10^{2}, 10^{3}\right\}$. The creation of this meta-data set was finished in about 160 CPU hours.

For the problem of combined algorithm and hyperparameter configuration problem, we chose 59 small classification data sets from the UCI repository. The misclassification error was precomputed on a grid for 19 different Weka classifiers [11] in more than 891 CPU hours.

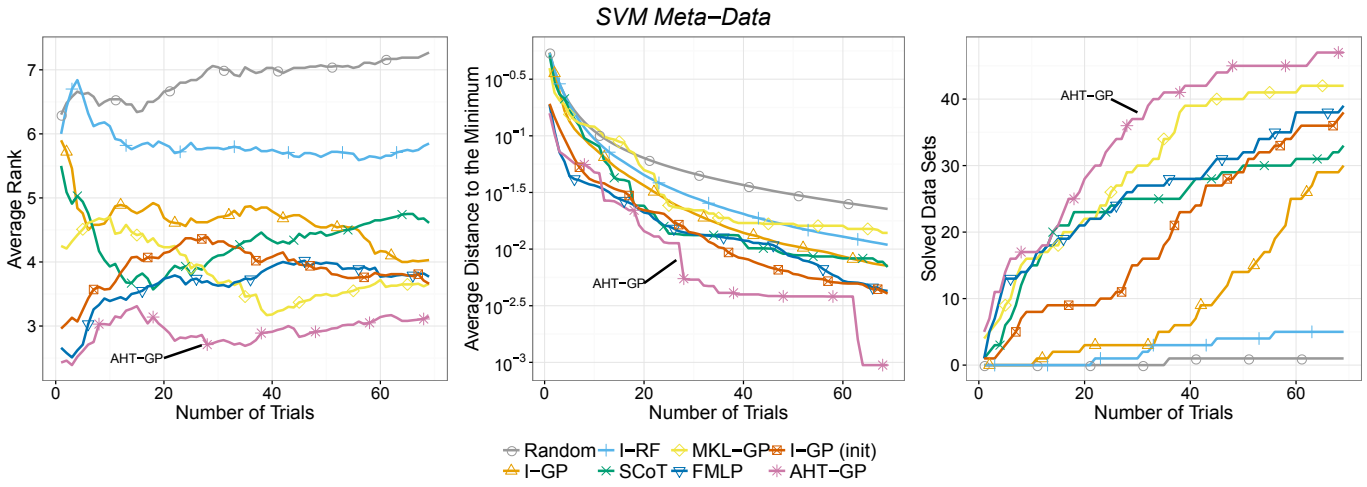

Fig. 2. Our proposed method AHT outperforms seven competitor methods with respect to all three evaluation metrics.

C. Evaluation Metrics

The quality of the tuning methods is compared with respect to three evaluation metrics. For computing the average rank, the methods are ranked with respect to the best hyperparameter configuration found so far, ties are resolved by giving them the average rank (Example: four tuning strategies have found hyperparameter configurations that achieve a misclassification rate of 0.01, 0.05, 0.05 and 0.1, respectively. Then the ranking is 1, 2.5, 2.5, 4.). The average distance to the global minimum was defined in Equation 4 and solved data sets counts the number of data sets where the method has achieved already a DTM of 0. Results reported are estimated using a leave-one-data-set-out cross-validation and averaged over at least ten repetitions (for strategies being more affected by random effects (Random, I-RF, I-GP) we committed up to 10,000 repetitions).

D. Hyperparameter Tuning

The task of hyperparameter tuning is to find the best hyperparameter configuration for a given algorithm. This is the typical scenario for researchers tuning baselines or their own new algorithm or for practitioners that have already decided which algorithm is best for their problem. This problem is putatively easier because the search space is smaller. Nevertheless, it is an important problem in practice.

MKL-GP tends to provide good results on most of the data sets and is able to find the best hyperparameter configuration in few trials (see Figure 2 right) but fails to find any good hyperparameter configuration on few data sets which leads to a comparable bad ADTM that is sometimes worse than random (see Figure 2 middle). Remarkable are also the good results of I-GP using an initialization, being competitive to the best transfer surrogate models published so far. Our proposed method AHT outperforms the competitor methods with respect to all three evaluation metrics on the SVM meta-data set. While the improvement within the first ten trials exists but it is small, a considerable improvement is developed soon afterwards. The reason for this is that AHT is based mainly on the meta-knowledge as the competitor methods. At the point where the meta-knowledge can no longer be exploited for guiding the search, AHT's special mechanism is used to expand its leading position.

E. Combined Algorithm Selection and Hyperparameter Tuning

In this section we want to empirically investigate the performance of the different hyperparameter optimization strategies in the scenario of combined algorithm selection and hyperparameter tuning. This problem leads to larger meta-data sets (about 1.4 million meta-instances) which did not allow us to commit these experiments for the transfer surrogates that are based on Gaussian processes. Nevertheless, we still compare to the transfer surrogate FMLP which has proven to be the best among the transfer surrogates [19].

In the previous experiments we always combined AHT with a Gaussian process. Since it was previously reported that the random forest as a surrogate model provides better results for problems with high-dimensional and discrete hyperparameter spaces [6] which is the case on the Weka meta-data set (more than 60% are indicator variables), we also provide results for AHT combined with a random forest. For us, the most promising advantage of a random forest is the shorter training time compared to a Gaussian process. We also committed these experiments on the SVM meta-data set but the results were worse than the combination with a Gaussian process and thus we omitted them to avoid overcrowded figures.

Figure 3 summarizes the results. Surprisingly, initialization (I-GP (init), I-RF (init)) did not provide good results for this meta-data set. We also tried to use HOM with the transfer function proposed in Equation 5 and set α such that an initialization effect was achieved but it still did not provide useful results. This is another indication that stresses that the soft and adaptive initialization effect of AHT is better than a hard initialization. Hence, the transfer surrogate (FMLP) is

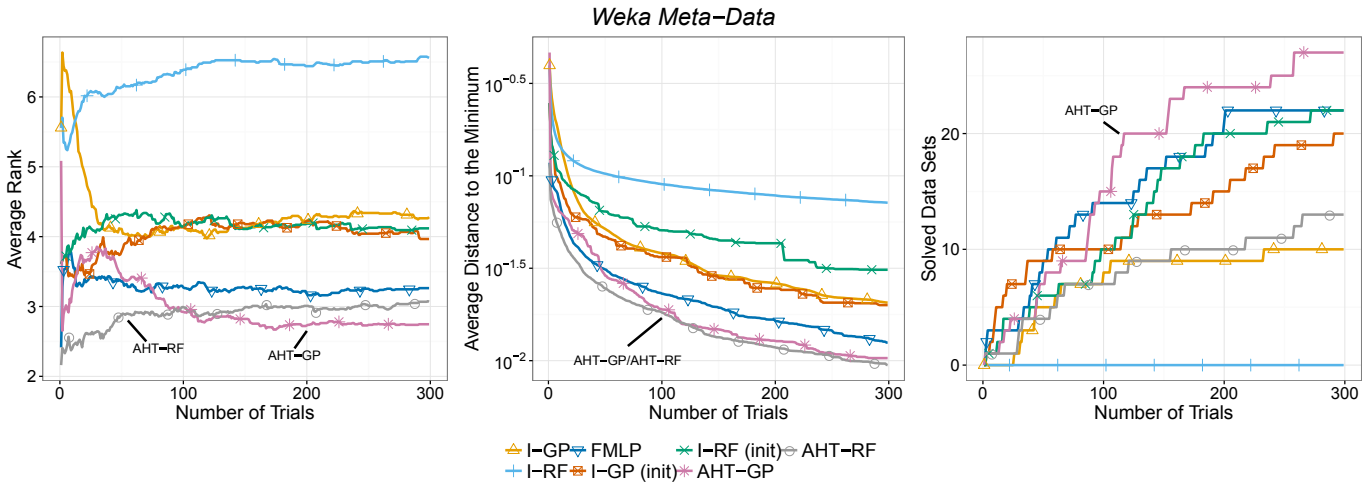

Fig. 3. AHT with two different surrogate models achieves the best ADTM on the Weka meta-data set but the combination with a Gaussian process leads to finding optimal hyperparameter configurations in more cases.

for this meta-data set clearly better than an initialization but in the end, AHT still provides the best results. The combination of AHT with a Gaussian process (AHT-GP) achieves similar results with respect to the ADTM compared to AHT-RF but finds the optimum on many data sets faster than AHT-RF.

F. Run Time

Finally, an important problem for us to tackle was to reduce the time overhead introduced by the different tuning strategies. Figure 4 summarizes these results. Unsurprisingly, strategies relying on simple surrogate models (I-GP, I-RF) are the fastest. We have seen that combining simple surrogates with an initialization (that does not result in an time overhead during the optimization) achieved good results for hyperparameter tuning but they showed less convincing results in the setting of combined algorithm selection and hyperparameter tuning. Transfer surrogates (FMLP) are by orders of magnitudes the slowest approach but have found good hyperparameter configurations in both scenarios. Finally, we achieved our goal of combining the advantages of both approaches. AHT provides the best results in both scenarios but has less time overhead than transfer surrogates for finding good hyperparameter configurations on average.

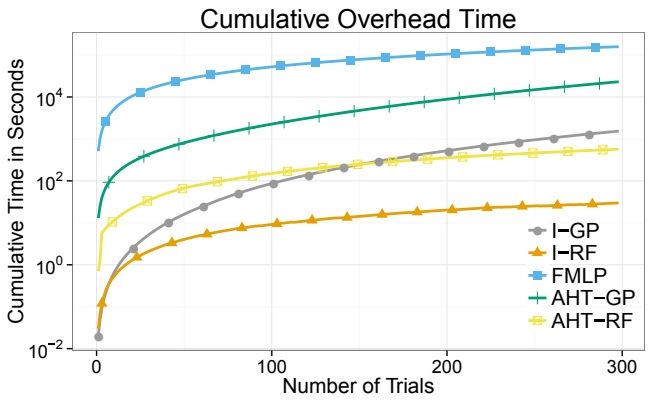

Fig. 4. Strategies based on transfer surrogates are the slowest among all investigated methods. AHT provides the best performance for a reasonable time overhead.

G. Case Study

For an in-depth understanding of how the different hyperparameter optimization strategies work, we select one data set for a deeper analysis. We select the *banana* data set because none of the tuning strategies was able to find the optimal hyperparameter configurations within 300 trials. Figure 5 shows the hyperparameter performance distribution for the different algorithms giving first insight why this data set is actually that difficult to optimize for. There are many different algorithms achieving small misclassification rates and hence it is difficult to narrow down the search to just few algorithms. Figure 6 gives insight with what frequency a tuning strategy has selected a specific algorithm for evaluating a hyperparameter configuration. Our Weka meta-data set is limited to hyperparameter configurations that we have evaluated beforehand to make this experiment possible. Because the different algorithms have a different number of hyperparameters, the number of test meta-instances varies between the different algorithms. The uniform distribution shows the fraction of test meta-instances per algorithm and hence can be used as an indication whether a tuning strategy prefers an algorithm or not. If the value of the uniform distribution is higher than the value of the tuning strategy, the tuning strategy does not believe that the optimum can be here and vice versa. Thus, it can be seen that the tuning strategies are capable of identifying that a multilayer perceptron does not achieve good performance on this specific data set while k-nearest neighbors (IBK) does.

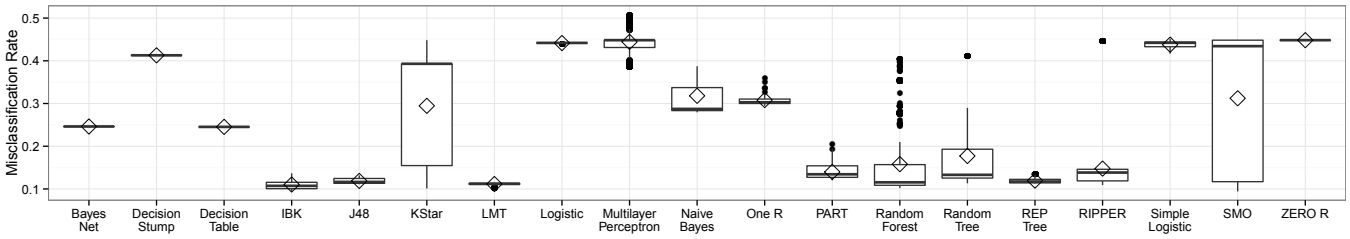

Fig. 5. Distribution over all hyperparameter configurations for different algorithms of the data set *banana*.

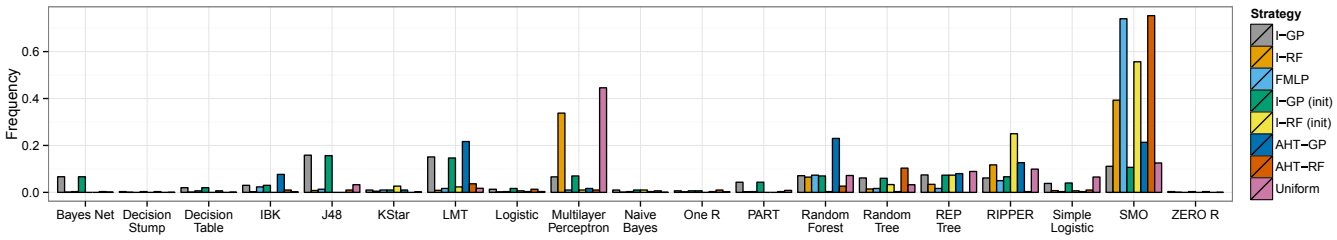

Fig. 6. Selection frequency of evaluating the performance of a hyperparameter configuration for a specific hyperparameter configuration. If the value is higher than the uniform distribution, this algorithm was preferred by the tuning strategy.

VII. Conclusion

We propose hyperparameter optimization machines as a generalization of sequential model-based optimization that includes current meta-learning extensions for the use in the hyperparameter optimization context. This generalization allows us to focus on the new hyperparameter optimization strategy AHT which uses meta-knowledge in an adaptive fashion and combines it with time- and space-efficient simple surrogate models. In experiments on two different meta-data sets for the problem of hyperparameter tuning as well as combined algorithm selection and hyperparameter tuning, the advantage of AHT compared to various other hyperparameter optimization strategies is shown empirically. We are able to show that AHT produces less time-overhead for the optimization than transfer surrogates by outperforming all competitor methods. However, we acknowledge that simple surrogates using an initialization are still the method with least overhead but this approach does not achieve good results in the scenario of combined algorithm selection and hyperparameter tuning.

Acknowledgment

The authors gratefully acknowledge the co-funding of their work by the German Research Foundation (Deutsche Forschungsgesellschaft) under grant SCHM 2583/6-1.

References

[1] R. Bardenet, M. Brendel, B. Kégl, and M. Sebag. Collaborative hyperparameter tuning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 199–207, 2013.

[2] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 2546–2554, 2011.

[3] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, Feb. 2012.

[4] G. C. Cawley. Model selection for support vector machines via adaptive step-size tabu search. In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms*, 2001.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[6] K. Eggensperger, M. Feurer, F. Hutter, J. Bergstra, J. Snoek, H. Hoos, and K. Leyton-Brown. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In *NIPS workshop on Bayesian Optimization in Theory and Practice*, 2013.

[7] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2962–2970, 2015.

[8] M. Feurer, J. T. Springenberg, and F. Hutter. Initializing bayesian hyperparameter optimization via meta-learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 1128–1135, 2015.

[9] F. Friedrichs and C. Igel. Evolutionary tuning of multiple svm parameters. *Neurocomput.*, 64:107–117, Mar. 2005.

[10] X. C. Guo, J. H. Yang, C. G. Wu, C. Y. Wang, and Y. C. Liang. A novel ls-svms hyper-parameter selection based on particle swarm optimization. *Neurocomput.*, 71(16-18):3211–3215, Oct. 2008.

[11] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[12] M. D. Hoffman, B. Shahriari, and N. de Freitas. On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, pages 365–374, 2014.

[13] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the*

5th International Conference on Learning and Intelligent Optimization, LION'05, pages 507–523, Berlin, Heidelberg, 2011. Springer-Verlag.

[14] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4):455–492, Dec. 1998.

[15] R. Leite, P. Brazdil, and J. Vanschoren. Selecting classification algorithms with active testing. In *Machine Learning and Data Mining in Pattern Recognition - 8th International Conference, MLDM 2012, Berlin, Germany, July 13-20, 2012. Proceedings*, pages 117–131, 2012.

[16] N. Pinto, D. Doukhan, J. J. DiCarlo, and D. D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Computational Biology*, 5(11):e1000579, 2009. PMID: 19956750.

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

[18] M. Reif, F. Shafait, and A. Dengel. Meta-learning for evolutionary parameter optimization of classifiers. *Machine Learning*, 87(3):357–380, 2012.

[19] N. Schilling, M. Wistuba, L. Drumond, and L. Schmidt-Thieme. Hyperparameter optimization with factorized multilayer perceptrons. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015. Proceedings, Part II*, 2015.

[20] K. A. Smith-Miles. Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Comput. Surv.*, 41(1):6:1–6:25, Jan. 2009.

[21] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 2960–2968, 2012.

[22] K. Swersky, J. Snoek, and R. P. Adams. Multi-task bayesian optimization. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 2004–2012, 2013.

[23] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 847–855, New York, NY, USA, 2013. ACM.

[24] M. Wistuba, N. Schilling, and L. Schmidt-Thieme. Hyperparameter search space pruning - a new component for sequential model-based hyperparameter optimization. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015. Proceedings, Part II*, 2015.

[25] M. Wistuba, N. Schilling, and L. Schmidt-Thieme. Learning hyperparameter optimization initializations. In *International Conference on Data Science and Advanced Analytics, DSAA 2015, Paris, France, October 19 - 21, 2015*, 2015.

[26] D. Yogatama and G. Mann. Efficient transfer learning method for automatic hyperparameter tuning. In *International Conference on Artificial Intelligence and Statistics (AISTATS 2014)*, 2014.