# Efficient Automated Online Experimentation with Multi-Fidelity

**Steven Kleinegesse**[*]
University of Edinburgh
steven.kleinegesse@ed.ac.uk

**Zhenwen Dai**
Spotify
zhenwend@spotify.com

**Andreas Damianou**
Spotify
andreasd@spotify.com

**Kamil Ciosek**
Spotify
kamilc@spotify.com

**Federico Tomasi**
Spotify
federicot@spotify.com

## Abstract

Prominent online experimentation approaches in industry, such as A/B testing, are often not scalable with respect to the number of candidate models. To address this shortcoming, recent work has introduced an automated online experimentation (AOE) scheme that uses a probabilistic model of user behavior to predict online performance of candidate models. While effective, these predictions of online performance may be biased due to various unforeseen circumstances, such as user modeling bias, a shift in data distribution or an incomplete set of features. In this work, we leverage advances from multi-fidelity optimization in order to combine AOE with Bayesian optimization (BO). This mitigates the effect of biased predictions, while still retaining scalability and performance. Furthermore, our approach also allows us to optimally adjust the number of users in a test cell, which is typically kept constant for online experimentation schemes, leading to a more effective allocation of resources. Our synthetic experiments show that our method yields improved performance, when compared to AOE, BO and other baseline approaches.

## 1 Introduction

Industry practitioners of machine learning regularly have to choose which production system or configuration they should deploy to users. Classical model selection algorithms, however, only evaluate the offline performance of the machine learning model, which is not representative of the whole production system. This has led to A/B testing becoming the gold standard in industry, as it evaluates direct measurements of online performance. Unfortunately, while widely-used, A/B testing can only compare a few different production systems effectively and is therefore not scalable with respect to the number of candidate models.

Recently, Dai et al. (2020) have developed an alternative methodology, called Automated Online Experimentation (AOE), which tackles these issues. AOE works by building a probabilistic surrogate model of the immediate feedback that is observed during the online experiment, which might, for instance, be whether or not a user has clicked on a recommendation, how long they have stayed on a website, etc. This AOE surrogate model can thus be understood as a way of modeling user behavior that allows us to predict the feedback we may receive from a user. We consider the downstream online metric that we wish to optimize to be an average of this immediate feedback, e.g. click-through rate, average time spent on a website, etc. Given recommendations from a particular candidate

[*]This work was carried out during an internship at Spotify.

model $\mathbf{M}$, the AOE surrogate model can then be used to obtain an estimate of the downstream metric $v(\mathbf{M})$, as well as a corresponding measure of uncertainty. After doing this for all candidate models, we can use the estimate and its uncertainty to rank all candidates via an acquisition function. The highest-scoring candidate is then deployed and corresponding data is collected, which is immediately used to update the surrogate model and repeat the process to select a new candidate. This approach is notably different to Bayesian optimization (see Shahriari et al., 2016, for a review), as AOE builds a surrogate model of user behavior and uses that to obtain predictions of $v(\mathbf{M})$, whereas BO directly builds a surrogate model of $v(\mathbf{M})$.

The AOE approach of Dai et al. (2020) has been shown to significantly improve model selection in production systems. Their experiments, however, were based on offline data where it was assumed that it is possible to perfectly model the immediate feedback using the AOE surrogate model. This allowed them to perfectly recover the ground-truth (online) metric with AOE predictions of this metric. This should generally not be possible in a real online setting, due to the following reasons:

**Limited modeling capacity:** We would need an extremely flexible surrogate model and large amounts of data in order to fully capture complex human behavior. We often do not have the means and the resources to do this, preventing us from modeling the immediate feedback perfectly.

**Distribution shifts in data:** Humans often show non-stationary behaviour, meaning that what we have learned during some part of the online experimentation may not hold true for the remainder. Furthermore, the training data of the candidate models might be severely different to the online data because of, for instance, unknown confounding factors or different contexts (e.g. time of the day).

**Incomplete sets of features:** Before training a candidate model, we generally have to do feature engineering and select some relevant features. However, if these features are incomplete, or non-stationary (which relates to the previous point), then even at sufficient modeling capacity, our surrogate model may not be able to fully-explain the online observations.

These limitations of AOE, which are not exhaustive, mean that we should expect to see a *bias* between AOE predictions and the ground-truth metric in a true online setting. In this work, we aim to reduce such biases by using advances from multi-fidelity optimization, which concerns the trade-off of high-fidelity (expensive but accurate) and low-fidelity (cheap but inaccurate) evaluations. We make use of multi-fidelity optimization in two crucial ways. First, we supplement high fidelity observations with low-fidelity AOE predictions in order to reduce bias. Second, our notion of fidelity includes the number of users in a test cell as well, which we can then optimally adjust. This is unlike AOE and A/B testing, where the number of users in a cell is usually kept constant.

We provide background on AOE and multi-fidelity optimization in Section 2 and explain how to combine these in Section 3. We evaluate our method in Section 4 using a toy experiment where we can actively control unwanted biases and a more realistic experiment that uses image data. Finally, we conclude our findings and provide future directions in Section 5.

## 2 Background

### 2.1 Model Selection for Production Systems

We consider the problem of finding a candidate model $\mathbf{M}^*$ that, given an online budget, maximizes an online metric $v$, i.e.

$$\mathbf{M}^* = \arg\max_{\mathbf{M}\in\mathcal{M}} v(\mathbf{M}), \tag{1}$$

where $\mathcal{M}$ is the set of candidate models. In our setting, a candidate model takes a representation $\mathbf{x}$ as input, e.g. user features, and outputs decisions $\mathbf{a}$, e.g. recommendations.

We also assume that, given inputs $\mathbf{x}$ and decisions $\mathbf{a}$, we receive some immediate feedback $m$ from the user, e.g. whether or not they have clicked on the provided recommendation. We define the online, or *accumulative*, metric that we wish to maximize as the expectation of this immediate feedback, i.e.

$$v(\mathbf{M}) \approx \frac{1}{N}\sum_{j=1}^{N} m_j, \tag{2}$$

where $\{m_j\}_{j=1}^N$ is the set of immediate feedback received when deploying the candidate model $\mathbf{M}$.

When the set of candidate models $\mathcal{M}$ is small, e.g. consisting of two models, then we could simply deploy all candidate models to a random sub-set of users, as done in A/B testing or multi-variate testing, and compare the accumulative metric observations for each of them. However, this commonly-used approach corresponds to an exhaustive search over the space of candidate models, which consumes a prohibitive amount of resources when there are many candidate models.

### 2.2 Automated Online Experimentation

Following Dai et al. (2020) we formally define the sample-average in Equation 2 as follows. First, we assume that the output of a candidate model can be represented by a distribution over decisions, i.e. $p(\mathbf{a}|\mathbf{x}, \mathbf{M})$. This corresponds to a Dirac delta function in the case of deterministic mappings (such as neural networks). Second, we assume that the distribution of immediate feedback is given by $p(m|\mathbf{a}, \mathbf{x})$ which, importantly, does not depend on the candidate model (only on its provided decisions). The accumulative online metric for a candidate model is then defined as an expectation over these distributions, yielding

$$v(\mathbf{M}) = \int\int\int m\, p(m|\mathbf{a}, \mathbf{x}) p(\mathbf{a}|\mathbf{x}, \mathbf{M}) p(\mathbf{x})\, \mathrm{d}\mathbf{a}\, \mathrm{d}\mathbf{x}\, \mathrm{d}m, \tag{3}$$

where $p(\mathbf{x})$ is the distribution of input representations. Unfortunately, the above high-dimensional integral is generally not analytically or computationally tractable.

The distribution of inputs $p(\mathbf{x})$ can generally be approximated empirically and the distribution $p(\mathbf{a}|\mathbf{x}, \mathbf{M})$ of the candidate model outputs is assumed to be known. Thus, the key to computing the integral in Equation 3 is the distribution of immediate feedback $p(m|\mathbf{a}, \mathbf{x})$, which is generally unknown. Dai et al. (2020) advocated to build a Bayesian surrogate model of the distribution of immediate feedback, which should then allow us to compute the desired integral, analytically in some cases and via Monte-Carlo integration otherwise, yielding an estimate $\widehat{v}_{AOE}(\mathbf{M})$ of the metric.

The probabilistic surrogate model $\widehat{p}(m|\mathbf{a}, \mathbf{x}, \mathcal{D})$ depends on previously-gathered data $\mathcal{D}$ and is updated with newly-gathered data at every online iteration. Dai et al. (2020) use a Gaussian Process (GP) surrogate model, which allows them to analytically compute the integral in Equation 3 in some cases. Furthermore, it is possible to propagate the GP uncertainty and compute an estimate of the uncertainty in $v(\mathbf{M})$ as well (see their paper for more information). When the integral cannot be computed analytically, we can also sample from the probabilistic surrogate model and approximate the integral with a Monte-Carlo sample-average. However, we are not necessarily limited in using GPs, as other surrogate models might be similarly appropriate depending on the problem.

Once we have constructed and fitted a surrogate model $\widehat{p}(m|\mathbf{a}, \mathbf{x}, \mathcal{D})$ during an online iteration, we can use Equation 3 and the predictions from a candidate model $\mathbf{M}$ to obtain an estimate $\widehat{v}_{AOE}(\mathbf{M})$ of the online metric, as well as a measure of uncertainty. We can then use these to score the candidate model by means of an acquisition function from the BO literature, e.g. Expected Improvement (EI) or Entropy Search (ES). After doing this for all candidate models, we then deploy the candidate with the highest score and repeat the aforementioned procedure until convergence or budget exhaustion.

### 2.3 Multi-Fidelity Optimization

When trying to optimize a function $f$ we often also have access to evaluations from different levels of *fidelity* $f_1, \ldots, f_n$, where low-fidelities (e.g. $f_1$) are cheap but inaccurate and high-fidelities (e.g. $f_n$) are expensive but accurate. The field of multi-fidelity optimization advocates that observations from lower fidelities can be used to improve the fit and optimization of the exact function $f$, or higher fidelities. Prominent approaches then differ in the way that they combine different fidelities and how they weigh, or rank, them (e.g. Forrester et al., 2007; Kandasamy et al., 2016; Marco et al., 2017; Song et al., 2019; Li et al., 2020).

## 3 Methodology

In this section we introduce automated online experimentation with multi-fidelity (AOE-MF), which combines AOE and Bayesian optimization (BO; see Shahriari et al., 2016, for a review) in order to mitigate the effect of biased AOE predictions and improve performance.

### 3.1 Treating AOE predictions as low-fidelity data

There are many potential causes that may result in a mis-match, or bias, between AOE predictions and real online observations in the online setting (see Section 1). We thus propose to treat the AOE predictions as a low-fidelity source of information that is readily available, while we treat expensive observations of the online metric as a high-fidelity source of information. To do so, we supplement the high-fidelity (real) observations with the low-fidelity (simulated) observations by conditioning our belief of the online metric on AOE predictions, effectively acting as a prior.

Following standard BO approaches, we fit a Gaussian Process (GP) surrogate model to observations of the online metric, i.e.

$$v(\mathbf{M}) = f(\mathbf{M}) + \epsilon, \qquad \text{with} \quad f \sim \mathcal{GP}(\boldsymbol{\mu}(\mathbf{M}), k(\mathbf{M}, \mathbf{M}')), \tag{4}$$

where $\mathbf{M}$ is an element of the candidate model space, $\epsilon$ is Gaussian noise with standard deviation $\sigma$ and $f$ is a sample of the GP. The GP prior is defined by the mean function $\boldsymbol{\mu}(\mathbf{M})$ and the kernel function $k(\mathbf{M}, \mathbf{M}')$.

It is standard practice in BO, and GP fitting in general, to define a mean function $\boldsymbol{\mu}(\mathbf{M})$ that is zero everywhere (see Rasmussen and Williams, 2006, for a theoretical discussion on this). In our approach, however, we define the mean function to be the AOE predictions, i.e.

$$\text{BO} \quad \Longrightarrow \quad \boldsymbol{\mu}(\mathbf{M}) = \mathbf{0} \tag{5}$$

$$\text{AOE-MF} \quad \Longrightarrow \quad \boldsymbol{\mu}(\mathbf{M}) = \widehat{v}_{AOE}(\mathbf{M}). \tag{6}$$

Fortuin et al. (2020) have recently shown that using a customised, or meta-learned, GP mean function can indeed lead to an improved fit. Intuitively, by using the AOE predictions as a mean function, we effectively leverage prior knowledge of the online metric. This knowledge, or belief, can then be updated upon observing real data, reducing the risk of getting misled by biased AOE predictions.

After specifying the GP mean function to be the AOE predictions at every online iteration, we can then gather data according to a regular BO scheme. We refer to the approach explained above as AOE-MF-PM, where PM stands for prior mean. Below we explain how to take this one step further and also include the number of users within a test cell as a measure of fidelity.

### 3.2 Variable number of users as different fidelity levels

Selecting the number of users within a test cell is an important task when performing online experimentation such as A/B testing or AOE. A large number of users in the test cell lead to a more accurate and representative observation of the online metric, at an increased cost. Conversely, reducing the number of users decreases the cost, but we may not easily conclude which of our candidate models perform best. For more rigid approaches, such as A/B testing, it is common to select the maximum test cell size that fits within the budget. However, for automated online procedures, such as AOE or AOE-MF, differently-sized test cells also change the exploration-exploitation behavior (as smaller test sizes lead to more exploration) and is therefore a crucial point of consideration.

We propose to factor the test cell size into the decision process of AOE-MF by further leveraging results from multi-fidelity optimization. We assume that we have access to different fidelity levels of the observations, i.e. different number of users within a test cell. At every online iteration, we then wish to find the optimal fidelity level to use, as well as the corresponding optimal candidate model.

We here use the prominent linear-fidelity model of Kennedy and O'Hagan (2000) because of its simplicity and applicability, but we note that non-linear variations and more recent methods (e.g. Kandasamy et al., 2016; Song et al., 2019; Li et al., 2020) may be appropriate as well depending on the problem at hand. In this model, a higher fidelity function $f_l$ is modeled as a scaled sum of a lower fidelity function $f_{l-1}$ and an error term $\delta_l$, i.e.

$$f_l(\mathbf{M}) = \delta_l(\mathbf{M}) + \rho_{l-1} f_{l-1}(\mathbf{M}), \qquad \text{for} \quad l = 2, \ldots, L \tag{7}$$

where $L$ is the total number of fidelity levels and $\rho_{l-1}$ is a regression parameter that indicates the correlation between the lower and higher fidelity functions. The fidelity functions $\{f_l\}_{l=1}^{L}$ and error terms $\{\delta_l\}_{l=2}^{L}$ in Equation 7 are all taken to be separate Gaussian Processes (GPs). The fidelity functions $\{f_l\}_{l=1}^{L}$ can be expressed as a single GP with a modified kernel function (see Kennedy and O'Hagan, 2000), which allows us to jointly fit the separate GPs.

**Algorithm 1** Automated Online Experimentation with Multi-Fidelity (AOE-MF-PM)

**Input:** Candidate models $\mathbf{M} \in \mathcal{M}$, AOE surrogate model $\widehat{p}(m|\mathbf{a}, \mathbf{x})$, test cell sizes $\mathbf{c} = (c_1, \ldots, c_L)$, surrogate models for each fidelity $\{f_l\}_{l=1}^{L}$, total online budget
**Output:** Optimal candidate model $\mathbf{M}^*$, trained AOE surrogate model

```
1:  Gather initial training data for each fidelity $\{f_l\}_{l=1}^{L}$
2:  while online budget not exhausted do
3:      Fit the AOE surrogate model $\widehat{p}(m|\mathbf{a}, \mathbf{x})$ to the training data
4:      Compute AOE predictions $\widehat{v}_{AOE}(\mathbf{M})$ for all $\mathbf{M} \in \mathcal{M}$ according to Equation 3
5:      Initialize the GP surrogate models for each fidelity using $\widehat{v}_{AOE}(\mathbf{M})$ as the prior mean
6:      Fit each GP surrogate model using data gathered at the corresponding fidelity level
7:      Compute an acquisition function $\alpha^{(l)}(\mathbf{M})$ for all $l = 1, \ldots, L$ and $\mathbf{M} \in \mathcal{M}$
8:      Find the optimal fidelity $l^*$ according to Equation 8
9:      Find the current optimal candidate model $\mathbf{M}^*$ according to Equation 9
10:     Deploy candidate $\mathbf{M}^*$ at fidelity level $l^*$ and gather online data
11:     Update the training data with the newly-gathered online data
```

At every online iteration, we then use data gathered from each fidelity level $l$ to model the corresponding fidelity function using separate GPs. To decide at which fidelity level to gather data during the next online iteration, we here follow the method of Marco et al. (2017). First, using GPs fitted to each fidelity function, we compute entropy search (Hennig and Schuler, 2012) acquisition functions $\alpha_{ES}^{(l)}(\mathbf{M})$ for each of them over the whole space of candidate models. If each of the fidelity levels has an associated cost $c_l$, e.g. the number of users within a test cell, the optimal fidelity level $l^*$ is given by the maximum value of $\alpha_{ES}^{(l)}(\mathbf{M})$ per unit cost, i.e.

$$l^* = \arg\max_{l} \max_{\mathbf{M}} \left[ \alpha_{ES}^{(l)}(\mathbf{M})/c_l \right] \tag{8}$$

The optimal candidate model $\mathbf{M}^*$ to deploy is then the one that maximises the acquisition function for that optimal fidelity, i.e.

$$\mathbf{M}^* = \arg\max_{\mathbf{M}} \alpha_{ES}^{(l^*)}(\mathbf{M}). \tag{9}$$

We summarise and combine our approaches from Sections 3.1 and 3.2 in Algorithm 1.

## 4 Experiments

We evaluate our AOE-MF-PM method on a toy experiment where the candidate models are contextual bandits, as well as a more realistic experiment where the candidate models are convolutional neural networks (CNNs). Where appropriate, we also compare our approach to baseline methods such as random deployment, BO and regular AOE. We only consider a single test cell size for the contextual bandits experiment, in order to better investigate the effect of conditioning on AOE predictions as discussed in Section 3.1.

### 4.1 Contextual Bandits

As an example of a commonly-used algorithm in industry, we turn to contextual bandits. We consider 5-armed bandits with 10 possible contexts $x \in \mathbb{R}$. The immediate feedback $m$ that we observe is binary, e.g. whether or not a user has clicked on a recommendation, and the online metric $v$ is simply the average of the feedback collected from several users, e.g. a click-through rate. Given a recommended bandit arm $a$ and a context $x$, we generate immediate feedback $m \in \{0, 1\}$ using (known) ground-truth reward probabilities (see the appendix for a plot showing these). In this experiment we keep the test cell size constant to 200 users for every online iteration.

Each contextual bandit generates recommendations $a$ by means of a policy $\pi(x)$ that takes the context as an input. We here consider a simple *quadratic* policy with two hyper-parameters $\alpha \in [0, 2]$ and $\beta \in [0, 2]$, i.e.

$$\pi(x; \alpha, \beta) = \alpha(x - \beta)^2. \tag{10}$$

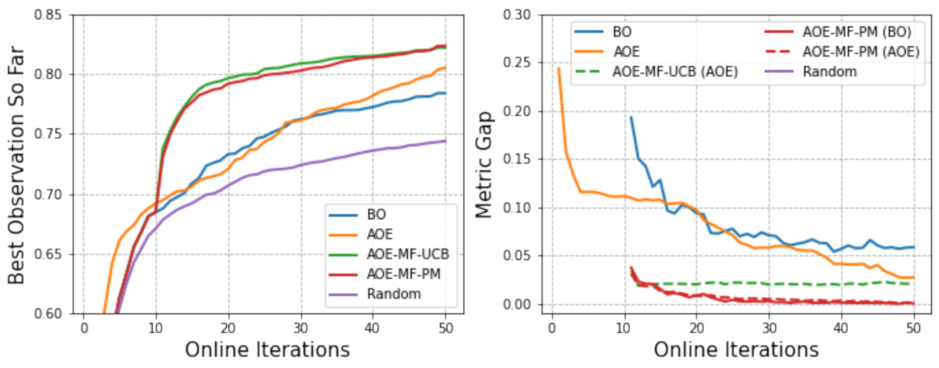

Figure 1: Low-bias results for the contextual bandit experiments. The left plot shows a comparison of our method with baseline methods using the best online observation of the metric so far (higher is better). The right plot shows a similar comparison using the metric gap, which measures how close each method is to the maximum of the ground-truth (lower is better). The dashed lines in the right plot show the AOE predictions computed as part of the respective methods. See the appendix for a corresponding plot with error bars.

The contextual bandit has a finite action space, however, and therefore we discretize the output of Equation 10 to select the appropriate bandit arm (by closeness). The set of candidate models $\mathbf{M} \in \mathcal{M}$ is defined by a $100 \times 100$ grid of $(\alpha, \beta) \in [0, 2]^2$. This means that we have over 10,000 candidate models that could be deployed to users, which is prohibitively large for standard online experimentation procedures such as A/B testing.

Following Dai et al. (2020) we use a sparse Gaussian Process (GP) as a surrogate model of the distribution of feedback $p(m|a, x)$. We vary the number of (fixed) inducing points to manually control the bias, which allows us to emulate an online setting where we might experience limited modeling capacity. By decreasing the number of inducing points we limit the sparse GP fitting capabilities, leading to an imperfect approximation of $p(m|a, x)$ and, thus, a biased estimation of the online metric according to Equation 3. Conversely, we can reduce this bias by increasing the number of (fixed) inducing points.

**Baselines** The random baseline involves a uniformly random candidate model being deployed every online iteration. For the BO baseline we use a GP surrogate model of $v(\mathbf{M})$ with a Matérn-5/2 kernel and the Expected Improvement (EI) acquisition function. The AOE baseline uses a sparse GP to approximate $p(m|a, x)$ and the EI acquisition function to decide which candidate model to deploy; these settings follow those of Dai et al. (2020). We also include a naive multi-fidelity approach which we call AOE-MF-UCB, inspired by the approach of Kandasamy et al. (2016). Here, we first compute the EI acquisition functions of BO and AOE and then we compute the minimum of those, i.e. $\alpha(\mathbf{M}) = \min(\alpha_{BO}(\mathbf{M}), \alpha_{AOE}(\mathbf{M}))$ over the space of candidate models. We then maximize this overlap to decide which candidate model to deploy.

**Metrics** We evaluate and compare the performance of our method on two different, common metrics: 1) the best metric observation up to that iteration and 2) the metric gap, which measures the difference between the maximum of the ground-truth metric and the ground-truth metric value at the optimum predicted by each respective method. See the appendix for a plot of the ground-truth of the online metric $v(\mathbf{M})$.

**Results** We present the findings for 10 inducing points, which corresponds to a relatively low bias, in Figure 1. As can be seen in both plots, our method (AOE-MF-PM) and our naive multi-fidelity baseline (AOE-MF-UCB) perform better than the remaining baseline methods. AOE-MF-PM is particularly strong on the metric gap metric (right plot), with almost no difference to the ground-truth. AOE on the other hand, takes much longer to converge, with a final value worse than that of AOE-MF-PM. We also present results for 5 inducing points in Figure 2, which corresponds to a relatively large bias. Here, AOE-MF-PM matches the performance of BO almost exactly and AOE performs particularly poorly. This is because the surrogate model approximation of $p(m|a, x)$ is

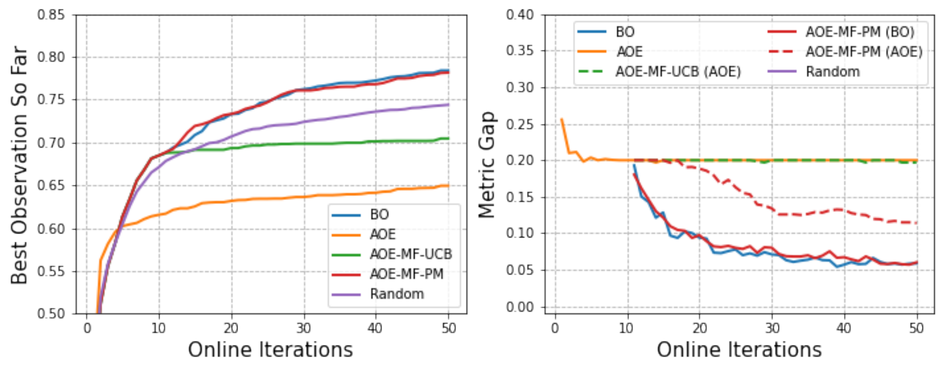

Figure 2: High-bias results for the contextual bandit experiments. The left plot shows a comparison of our method with baseline methods using the best online observation of the metric so far (higher is better). The right plot shows a similar comparison using the metric gap, which measures how close each method is to the maximum of the ground-truth (lower is better). The dashed lines in the right plot show the AOE predictions computed as part of the respective methods. See the appendix for a corresponding plot with error bars.

quite inaccurate, resulting in biased metric predictions, which means that the AOE predictions do not actually help much in our multi-fidelity approach. Nonetheless, our approach does not perform any worse than the BO baseline on both metrics. The AOE-MF-UCB baseline performs slightly better than the AOE baseline on the best observation metric, but only the same on the metric gap baseline. Interestingly, the random baseline performs better than both the AOE and AOE-MF-UCB baseline on the best observation metric, suggesting that AOE predictions may even be misleading when there is a large bias.

## 4.2 Convolutional Neural Networks

Next, we turn to convolutional neural networks (CNNs) as an expressive class of candidate models for image classification. We consider relatively small CNNs, with three convolutional layers and two dense layers, that we train on CIFAR-10 image data (Krizhevsky, 2009) with cross-entropy loss and a stochastic gradient-descent (SGD) optimizer. In this setting, the context $\mathbf{x}$ are the CIFAR-10 images and the decisions $\mathbf{a}$ are the class predictions of the CNN candidate models. The immediate feedback $m$ is binary again, with $m = 0$ when the classes are incorrectly predicted and $m = 1$ when they are correctly predicted. The online metric we wish to maximize is the average of this feedback, i.e. the test set accuracy. The caveat here, however, is that we first need to gather the test set data through online experimentation. We here also consider the case of gathering data from differently-sized test cells, specifically either from $1{,}000$ or $5{,}000$ users, as discussed in Section 3.2.[2]

The set of candidate models $\mathcal{M}$ is defined by four hyper-parameters: the learning rate $l_r$, the SGD momentum $\alpha$, the batch-size $b$ and the learning rate exponential decay parameter $\lambda$. The learning rate can take $10$ values between $10^{-5}$ and $10^{-1}$ in the log-space and the momentum can take $10$ values between $0.5$ and $1$. The batch-size can take values $b \in \{32, 64, 128, 256, 512\}$, while the learning rate decay can take values $\lambda \in \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. Altogether, this means that our set of candidate models consists of $3{,}000$ different CNNs.

Unlike for the contextual bandit experiments, we cannot easily use a sparse GP as the surrogate model to approximate $p(m|\mathbf{a}, \mathbf{x})$. The image data $\mathbf{x}$ is 3072-dimensional, which is generally too high for GPs to handle. Instead, we here use a residual neural network (ResNet) surrogate model to learn the mapping $\mathbf{x}, \mathbf{a} \to m$. To do so, we use a standard ResNet50 architecture that has $48$ convolutional layers, followed by a few fully-connected layers. We pre-train this surrogate model on the training set that was used to train the candidate models ($10^4$ CIFAR-10 images) and use the learned mapping to compute AOE predictions $\widehat{v}_{AOE}(\mathbf{M})$ with a sample-average of Equation 3. During the online experimentation, we then freeze the convolutional layers and only re-train the

[2]We note that our method does not place a restriction on the number of fidelities, i.e. different test cell sizes.

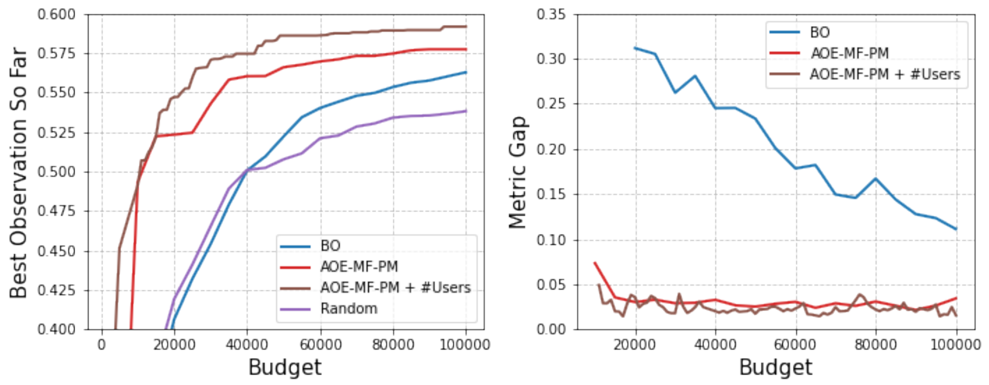

Figure 3: Results for the CNN experiments. The left plot shows a comparison of our method with baseline methods using the best online observation of the metric so far (higher is better). The right plot shows a similar comparison using the metric gap, which measures how close each method is to the maximum of the ground-truth (lower is better). See the appendix for a corresponding plot with error bars.

fully-connected layers upon gathering new data. We found that this generally improves robustness, at only a small performance cost.

**Baselines** Similarly to the contextual bandit experiment, we compare our AOE-MF-PM approach to a random and BO baseline. However, because we cannot use a sparse GP as a surrogate model, due to the high-dimensional image data, we also cannot use AOE and AOE-MF-UCB as a baseline in this experiment. Our ResNet surrogate model only provides a mean estimate of the feedback $m$ and not an uncertainty estimate, which is required by regular AOE (but not our approach).

**Results** We present our findings in Figure 3, where we evaluate our methods on the same two metrics introduced in Section 4.1. Our AOE-MF-PM method with two fidelity levels ($1{,}000$ and $5{,}000$ users) performs better than the baseline approaches and our method with a single fidelity level ($5{,}000$ users), on both metrics. When looking at the red and brown lines, it becomes clear that we are indeed more resource-efficient when taking the number of users into account.

## 5 Conclusions

In this work we have introduced automatic online experimentation with multi-fidelity (AOE-MF), which combines AOE with Bayesian optimization (BO) by means of multi-fidelity optimization. Our method, called AOE-MF-PM, allows us to mitigate the effect of biased AOE predictions by using them as a prior mean to a probabilistic surrogate model of online observations. Furthermore, our method allows us to find the optimal number of users in a test cell, unlike previous approaches. Our experiments show that our method effectively mitigates biases from inaccurate modeling of user behavior, outperforming all baseline methods. Additionally, when the bias is extremely large and AOE predictions are not particularly useful, our method does not perform worse than BO. Our experiments on high-dimensional image data show that finding the optimal number of users in a test cell further improves resource allocation. These findings suggest that multi-fidelity approaches provide an exciting new direction of research in automated online experimentation.

There are many interesting future directions of this particular work. First, we utilised a linear combination of different fidelity levels, which could be taken further by considering non-linear combinations or more advanced methods from recent work in multi-fidelity optimization (e.g. Song et al., 2019; Li et al., 2020). Second, we do not take the uncertainty of AOE predictions into account when using them as a prior mean, which could further improve efficiency and resource allocation. Lastly, it would be fruitful to evaluate our method with a real, online experiment, which could further help to understand the sources of biases and how to mitigate them.

## References

Zhenwen Dai, Praveen Chandar, Ghazal Fazelnia, Benjamin Carterette, and Mounia Lalmas. Model selection for production system via automated online experiments. In *Advances in Neural Information Processing Systems*, volume 33, pages 1106–1116. Curran Associates, Inc., 2020.

A. Forrester, A. Sobester, and A. Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463:3251 – 3269, 2007.

Vincent Fortuin, Heiko Strathmann, and Gunnar Rätsch. Meta-Learning Mean Functions for Gaussian Processes. *arXiv e-prints*, art. arXiv:1901.08098, 2020.

Philipp Hennig and Christian J. Schuler. Entropy search for information-efficient global optimization. *J. Mach. Learn. Res.*, 13(null):1809–1837, June 2012.

Kirthevasan Kandasamy, Gautam Dasarathy, Junier Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 1000–1008, Red Hook, NY, USA, 2016. Curran Associates Inc.

M. C. Kennedy and A. O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Shibo Li, Wei Xing, Robert Kirby, and Shandian Zhe. Multi-fidelity bayesian optimization via deep neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8521–8531. Curran Associates, Inc., 2020.

Alonso Marco, Felix Berkenkamp, Philipp Hennig, Angela P. Schoellig, Andreas Krause, Stefan Schaal, and Sebastian Trimpe. Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with bayesian optimization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1557–1563, 2017. doi: 10.1109/ICRA.2017.7989186.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*, volume 2. MIT Press, 2006.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2016. doi: 10.1109/JPROC.2015.2494218.

Jialin Song, Yuxin Chen, and Yisong Yue. A general framework for multi-fidelity bayesian optimization with gaussian processes. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 3158–3167. PMLR, 16–18 Apr 2019.

## A Appendix

### A.1 Contextual Bandit Experiment: Ground-Truths

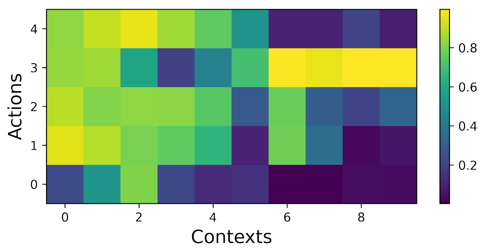

Figure 4: Ground-truth reward probabilities (indicated by the colorbar) for each combination of the possible contexts $x$ and actions $a$.

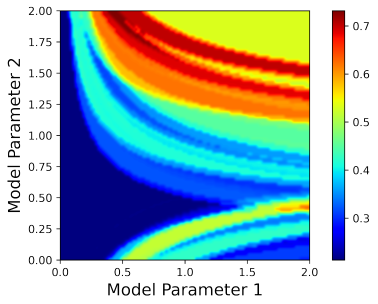

Figure 5: Ground-truth online metric $v(\mathbf{M})$ as a function of the contextual bandit policy hyper-parameters ($\alpha$ and $\beta$ in Equation 10. This was computed using the ground-truth reward probabilities in Figure 4 and Equation 3.

### A.2 Contextual Bandit Experiment: Results with Error Bars

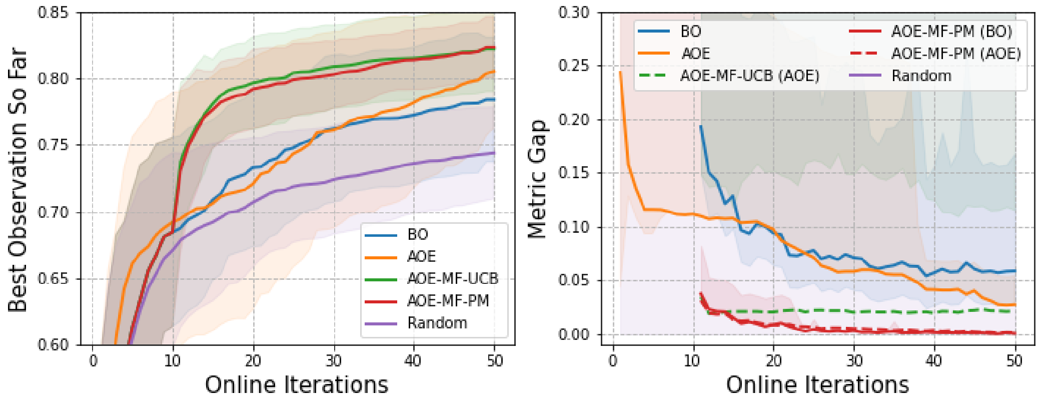

Figure 6: Low-bias results for the contextual bandit experiments with error bars, obtained through repeat runs with different random seeds, corresponding to Figure 1. Each method was repeated 50 times, with the bold curves showing the mean and the shaded regions showing the standard deviation.

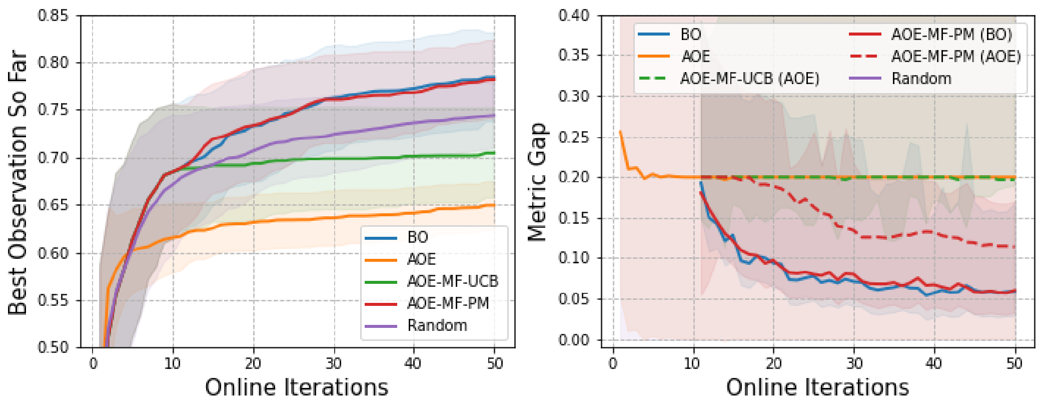

Figure 7: High-bias results for the contextual bandit experiments with error bars, obtained through repeat runs with different random seeds, corresponding to Figure 2. Each method was repeated 50 times, with the bold curves showing the mean and the shaded regions showing the standard deviation.

### A.3 CNN Experiment: Results with Error Bars

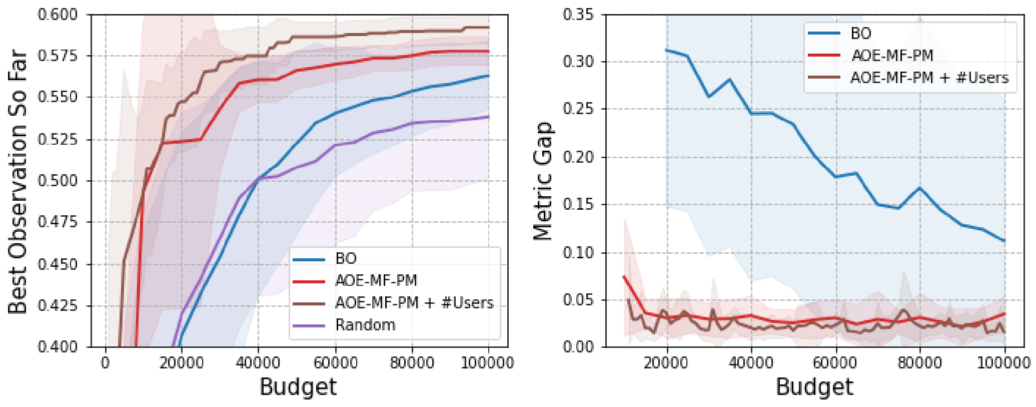

Figure 8: Results for the CNN experiments with error bars, obtained through repeat runs with different random seeds, corresponding to Figure 3. The BO and random baseline were repeated 100 times, while AOE-MF-PM and AOE-MF-PM + #Users were repeated 20 and 10 times, respectively, due to computational limitations. The solid lines show the means and the shaded regions show the standard deviation.