A Additional experimental details

A.1 Synthetic experiments

We pretrain the model for 2000 iterations with 128 synthetic functions at each iteration. We randomize the length scale parameter $\ell \sim \mathcal{U}[5.0, 10.0]$ and function scale parameter $\sigma_y \sim \mathcal{U}[1.0, 10.0]$ of the RBF kernel to increase pretraining data diversity. For each function generated, we sample 228 data points that we separate into 100 context points and 128 target points and train the model using the loss function in (2). Each input x is a 32-dimensional vector, and each dimension is sampled from a uniform distribution $\mathcal{U}[-3, 3]$.

For each test function, we sample a large dataset of 20000 data points. We then randomly select 100 samples from the data points with function values lower than the 20th percentile as the few-shot data. We condition the model on this few-shot dataset and the maximal value $y^\star$ in the large dataset to generate 256 candidates and report the best score achieved among these candidates. We normalize the score to $[0, 1]$ using the worst and the best value in the large dataset.

A.2 ExPT pretraining details

Architectural details In all experiments, we use the same ExPT architecture. Before feeding to the Transformer encoder, we embed the (y, x) context pairs with a 1-layer MLP and embed the target $y's$ with another 1-layer MLP. The transformer encoder has 4 layers with a hidden dimension of 128, 4 attention heads, `GELU` activation, and a dropout rate of 0.1. For the VAE model, we use a standard isotropic Gaussian distribution as the prior. Both the VAE encoder and VAE decoder have 4 layers with a hidden dimension of 512, and the latent variable z has a dimension of 32.

Optimization details In Design-Bench experiments, we pretrain ExPT for 10,000 iterations with 128 synthetic functions in each iteration. We use AdamW optimizer [31, 40] with a learning rate of $5e-4$ and $(\beta_1, \beta_2) = (0.9, 0.99)$. We use a linear warmup schedule for 1000 steps, followed by a cosine-annealing schedule for 9000 steps.

A.3 Construction of new D'Kitty and Ant tasks

This section details how we constructed new objectives from the original D'Kitty and Ant that we used to evaluate ExPT in Section 3.2.1. For each new objective, we apply the corresponding oracle to the inputs $x's$ in the original dataset to create the dataset for the objective.

Ant tasks In Ant, the original goal is to design a morphology that allows the Ant robot to run as fast as possible in the x (horizontal) direction. The objective function is the sum of rewards in 100 time steps, where the reward R at each time step is defined as:

$$R = \text{Forward reward} + \text{Survival reward} - \text{Control cost} - \text{Contact cost}, \tag{5}$$

where Forward reward $= (x_t - x_{t-1})/dt$ is the velocity of the Ant in the x direction.

In Ant-v_y, the reward at each time step is similar, except that Forward reward $= (y_t - y_{t-1})/dt$ is the velocity of the Ant in the y (vertical) direction. In other words, we aim to design morphologies that allow the robot to run fast in the y direction.

In Ant-Energy, the reward at each time step is:

$$R = 1 + \text{Survival reward} - \text{Control cost} - \text{Contact cost}, \tag{6}$$

which means we incentivize the robot to conserve energy instead of running fast.

D'Kitty tasks In D'Kitty, the goal is to design a morphology that allows the D'Kitty robot to reach a fixed target location, and the objective function f is the Euclidean distance to the target. In the original D'Kitty task, the target location is on the vertical line from the starting point. In the two new tasks D'Kitty-45 and D'Kitty-60, the target locations are 45 deg and 60 deg away from the original target, respectively.

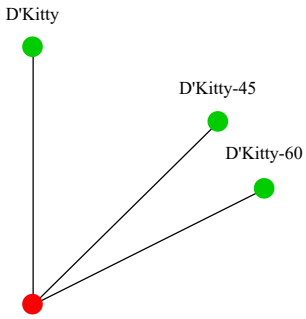

Figure 5: Different D'Kitty tasks. The red dot denotes the starting location, and the green dots are the target locations.

B Excluded Design-Bench tasks

B.1 Superconductor

We found the approximate oracle provided by Design-Bench not accurate enough to provide a reliable comparison of optimization methods on this task. Figure 6 plots the score values in the dataset against the score values predicted by the approximate oracle, which shows a weak correlation between these two values.

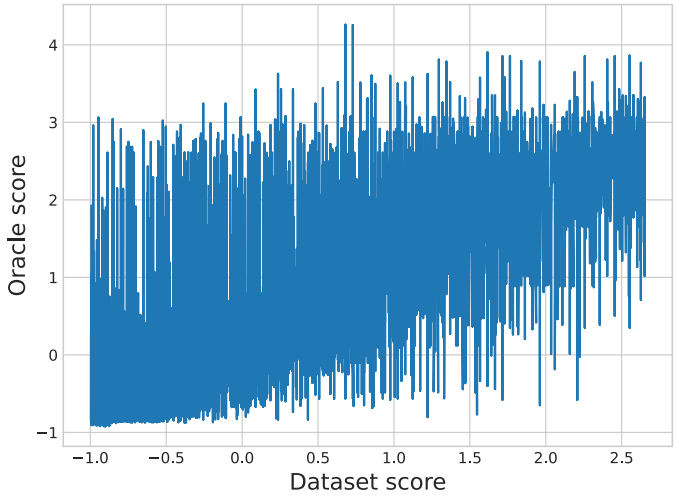

Figure 6: The correlation between the score values in the dataset (x-axis) and the score values predicted by the approximate oracle (y-axis) in Superconductor.

B.2 Hopper

As noted in previous works that use Design-Bench [34], the oracle provided for the Hopper task is inconsistent with the true-dataset values. The outputs of the oracle on the dataset are skewed heavily towards low-function values, which makes it an unreliable task for evaluation.

B.3 ChEMBL

As observed in previous works [58, 34], all methods produced nearly the same results on the ChEMBL task, so we excluded it in our experiments.

C Additional ablation and analysis

C.1 Effects of GP hyperparameters

We empirically examine the impact of two GP hyperparameters, the variance σ and the length scale ℓ, on the performance of ExPT. Specifically, we evaluate the performance of ExPT on D'Kitty and Ant when σ is too small (ExPT-small-σ) or too large (ExPT-large-σ), and when ℓ is too small (ExPT-small-ℓ) or too large (ExPT-large-ℓ). In ExPT-small-σ and ExPT-large-σ, we sample σ from $\mathcal{U}[0.01, 0.1]$ and $\mathcal{U}[100, 200]$, respectively. In ExPT-small-ℓ and ExPT-large-ℓ, we sample ℓ from $\mathcal{U}[0.1, 1.0]$ and $\mathcal{U}[100, 200]$, respectively.

Table 6: Impact of σ and ℓ on ExPT performance on Ant and D'Kitty in `random` (left) and `poorest` (right) settings. We average the performance across 3 seeds.

	Baseline	D'Kitty	Ant
	$\mathcal{D}_{\text{few-shot}}$(best)	0.883	0.563
Median	ExPT	0.902 ± 0.006	$\mathbf{0.705 \pm 0.018}$
	ExPT-small-σ	$\mathbf{0.915 \pm 0.006}$	0.661 ± 0.111
	ExPT-large-σ	0.797 ± 0.000	0.471 ± 0.012
	ExPT-small-ℓ	0.793 ± 0.004	0.459 ± 0.005
	ExPT-large-ℓ	0.795 ± 0.003	0.460 ± 0.003
Mean	ExPT	0.865 ± 0.016	$\mathbf{0.639 \pm 0.026}$
	ExPT-small-σ	$\mathbf{0.896 \pm 0.016}$	0.630 ± 0.089
	ExPT-large-σ	0.752 ± 0.013	0.534 ± 0.015
	ExPT-small-ℓ	0.726 ± 0.018	0.518 ± 0.018
	ExPT-large-ℓ	0.725 ± 0.016	0.528 ± 0.006

	Baseline	D'Kitty	Ant
	$\mathcal{D}_{\text{few-shot}}$(best)	0.307	0.124
Median	ExPT	$\mathbf{0.922 \pm 0.009}$	$\mathbf{0.686 \pm 0.090}$
	ExPT-small-σ	0.862 ± 0.064	0.656 ± 0.098
	ExPT-large-σ	0.792 ± 0.004	0.489 ± 0.019
	ExPT-small-ℓ	0.792 ± 0.006	0.462 ± 0.004
	ExPT-large-ℓ	0.795 ± 0.003	0.460 ± 0.004
Mean	ExPT	$\mathbf{0.871 \pm 0.018}$	$\mathbf{0.646 \pm 0.061}$
	ExPT-small-σ	0.755 ± 0.085	0.606 ± 0.077
	ExPT-large-σ	0.726 ± 0.016	0.547 ± 0.012
	ExPT-small-ℓ	0.725 ± 0.019	0.529 ± 0.014
	ExPT-large-ℓ	0.722 ± 0.014	0.530 ± 0.011

The results in Table 6 show that overall, suboptimal values of σ and ℓ lead to a substantial drop in the performance of ExPT on both tasks. It is also noticeable that ℓ has a more significant influence on the performance than σ. In other words, the shape of the synthetic functions has a more critical impact on downstream performances than the magnitudes of the function values. A too small ℓ or large ℓ results in synthetic functions that exhibit either excessive oscillations or excessive smoothness, leading to poor generalization to downstream functions.

C.2 ExPT with different pretraining data distributions

We perform an ablation study where we pretrain ExPT on different data distributions, including different GP kernels (GP-Cosine, GP-Linear, GP-Periodic), randomly initialized 1-layer neural networks (Random MLP), and neural network checkpoints trained on the few-shot data (Trained MLP). For each network used to generate data in Random MLP and Trained MLP, we randomly select the initialization method in {uniform, normal, xavier uniform, xavier normal, kaiming uniform, kaiming normal}, the hidden size in {16, 32, 64, 128, 256, 512, 1024}, and the depth in {2, 3, 4, 5, 6}. Each network in Random MLP is randomly initialized, while each network in Trained MLP is trained on the few-shot data.

Table 7: Performance of ExPT with different pretraining data distributions on the random setting

	Pretraining data	D'Kitty	Ant	TF8	TF10	Mean score
Median	GP-RBF	0.902 ± 0.006	0.705 ± 0.018	0.473 ± 0.014	0.477 ± 0.014	0.639 ± 0.013
	GP-Cosine	0.795 ± 0.006	0.463 ± 0.003	0.379 ± 0.013	0.456 ± 0.006	0.523 ± 0.007
	GP-Linear	0.900 ± 0.002	0.686 ± 0.013	0.377 ± 0.009	0.468 ± 0.010	0.608 ± 0.009
	GP-Periodic	0.902 ± 0.003	0.655 ± 0.029	0.452 ± 0.013	0.467 ± 0.006	0.619 ± 0.013
	Random MLP	0.906 ± 0.004	0.520 ± 0.123	0.480 ± 0.021	0.487 ± 0.015	0.598 ± 0.041
	Trained MLP	0.914 ± 0.007	0.691 ± 0.003	0.446 ± 0.021	0.482 ± 0.029	0.633 ± 0.015
Max	GP-RBF	0.973 ± 0.005	0.970 ± 0.004	0.933 ± 0.036	0.677 ± 0.048	0.888 ± 0.023
	GP-Cosine	0.955 ± 0.008	0.963 ± 0.011	0.906 ± 0.079	0.709 ± 0.068	0.883 ± 0.042
	GP-Linear	0.972 ± 0.001	0.965 ± 0.016	0.899 ± 0.095	0.654 ± 0.033	0.872 ± 0.036
	GP-Periodic	0.971 ± 0.005	0.966 ± 0.005	0.875 ± 0.022	0.646 ± 0.026	0.864 ± 0.014
	Random MLP	0.973 ± 0.001	0.953 ± 0.013	0.938 ± 0.068	0.653 ± 0.004	0.879 ± 0.022
	Trained MLP	0.974 ± 0.005	0.935 ± 0.022	0.879 ± 0.039	0.660 ± 0.003	0.862 ± 0.017
Mean	GP-RBF	0.865 ± 0.016	0.639 ± 0.026	0.476 ± 0.010	0.474 ± 0.015	0.614 ± 0.017
	GP-Cosine	0.725 ± 0.022	0.534 ± 0.011	0.385 ± 0.007	0.455 ± 0.004	0.525 ± 0.011
	GP-Linear	0.866 ± 0.001	0.633 ± 0.017	0.397 ± 0.013	0.465 ± 0.010	0.590 ± 0.010
	GP-Periodic	0.865 ± 0.008	0.594 ± 0.010	0.464 ± 0.008	0.469 ± 0.008	0.598 ± 0.009
	Random MLP	0.883 ± 0.011	0.516 ± 0.074	0.481 ± 0.016	0.485 ± 0.016	0.591 ± 0.029
	Trained MLP	0.910 ± 0.008	0.660 ± 0.003	0.451 ± 0.019	0.478 ± 0.026	0.625 ± 0.014

Table 8: Performance of ExPT with different pretraining data distributions on the poor setting

	Pretraining data	D'Kitty	Ant	TF8	TF10	Mean score
Median	GP-RBF	0.922 ± 0.009	0.686 ± 0.090	0.552 ± 0.042	0.489 ± 0.013	0.662 ± 0.039
	GP-Cosine	0.795 ± 0.005	0.463 ± 0.003	0.379 ± 0.013	0.456 ± 0.006	0.524 ± 0.007
	GP-Linear	0.918 ± 0.009	0.675 ± 0.065	0.380 ± 0.013	0.450 ± 0.004	0.606 ± 0.023
	GP-Periodic	0.928 ± 0.006	0.689 ± 0.037	0.487 ± 0.089	0.498 ± 0.013	0.651 ± 0.036
	Random MLP	0.902 ± 0.012	0.446 ± 0.004	0.499 ± 0.010	0.495 ± 0.005	0.586 ± 0.008
	Trained MLP	0.909 ± 0.006	0.733 ± 0.039	0.431 ± 0.043	0.482 ± 0.028	0.639 ± 0.029
Max	GP-RBF	0.946 ± 0.018	0.965 ± 0.004	0.873 ± 0.035	0.615 ± 0.022	0.850 ± 0.020
	GP-Cosine	0.961 ± 0.004	0.951 ± 0.027	0.906 ± 0.079	0.709 ± 0.068	0.872 ± 0.045
	GP-Linear	0.976 ± 0.003	0.971 ± 0.008	0.896 ± 0.012	0.623 ± 0.030	0.867 ± 0.013
	GP-Periodic	0.975 ± 0.004	0.969 ± 0.001	0.709 ± 0.086	0.641 ± 0.061	0.824 ± 0.038
	Random MLP	0.975 ± 0.003	0.970 ± 0.007	0.797 ± 0.050	0.629 ± 0.018	0.843 ± 0.020
	Trained MLP	0.975 ± 0.003	0.905 ± 0.033	0.716 ± 0.094	0.578 ± 0.023	0.794 ± 0.038
Mean	GP-RBF	0.871 ± 0.018	0.646 ± 0.061	0.549 ± 0.032	0.488 ± 0.011	0.639 ± 0.031
	GP-Cosine	0.728 ± 0.021	0.528 ± 0.010	0.385 ± 0.007	0.455 ± 0.004	0.524 ± 0.010
	GP-Linear	0.872 ± 0.025	0.624 ± 0.031	0.397 ± 0.009	0.447 ± 0.004	0.585 ± 0.017
	GP-Periodic	0.887 ± 0.047	0.634 ± 0.015	0.511 ± 0.069	0.496 ± 0.011	0.634 ± 0.036
	Random MLP	0.790 ± 0.048	0.522 ± 0.042	0.499 ± 0.012	0.489 ± 0.006	0.575 ± 0.027
	Trained MLP	0.869 ± 0.012	0.684 ± 0.043	0.447 ± 0.057	0.476 ± 0.027	0.619 ± 0.022

Tables 7 and 8 show the performance of ExPT on the few-shot random and few-shot poor settings when pretrained with different data distributions. Overall, the model achieves good performance across different data distributions, with GP-RBF being the best in most settings. This ablation study shows the robustness of ExPT to the pretraining data distribution.

C.3 ExPT with different decoder architectures

In addition to the pretraining data distribution, we also conducted an ablation study on the architecture of ExPT, in which we replaced the VAE model with a diffusion model (ExPT-Diffusion). We take the diffusion architecture from [35].

Table 9: Performance of ExPT with different decoder architectures on the random setting

	Decoder architecture	D'Kitty	Ant	TF8	TF10	Mean score
Median	VAE	0.902 ± 0.006	0.705 ± 0.018	0.473 ± 0.014	0.477 ± 0.014	0.639 ± 0.013
	Diffusion	0.816 ± 0.028	0.642 ± 0.018	0.457 ± 0.116	0.489 ± 0.019	0.601 ± 0.045
Max	VAE	0.973 ± 0.005	0.970 ± 0.004	0.933 ± 0.036	0.677 ± 0.048	0.888 ± 0.023
	Diffusion	0.966 ± 0.007	0.967 ± 0.006	0.868 ± 0.150	0.628 ± 0.014	0.857 ± 0.044
Mean	VAE	0.865 ± 0.016	0.639 ± 0.026	0.476 ± 0.010	0.474 ± 0.015	0.614 ± 0.017
	Diffusion	0.741 ± 0.013	0.603 ± 0.016	0.468 ± 0.115	0.486 ± 0.016	0.575 ± 0.040

Table 10: Performance of ExPT with different decoder architectures on the poor setting

	Decoder architecture	D'Kitty	Ant	TF8	TF10	Mean score
Median	VAE	0.922 ± 0.009	0.686 ± 0.090	0.552 ± 0.042	0.489 ± 0.013	0.662 ± 0.039
	Diffusion	0.821 ± 0.038	0.638 ± 0.011	0.295 ± 0.010	0.421 ± 0.007	0.544 ± 0.017
Max	VAE	0.946 ± 0.018	0.965 ± 0.004	0.873 ± 0.035	0.615 ± 0.022	0.850 ± 0.020
	Diffusion	0.974 ± 0.003	0.956 ± 0.008	0.677 ± 0.007	0.593 ± 0.026	0.800 ± 0.011
Mean	VAE	0.871 ± 0.018	0.646 ± 0.061	0.549 ± 0.032	0.488 ± 0.011	0.639 ± 0.031
	Diffusion	0.731 ± 0.035	0.600 ± 0.014	0.311 ± 0.011	0.415 ± 0.013	0.514 ± 0.018

Tables 9 and 10 show that ExPT + VAE outperforms ExPT + Diffusion in all tasks and settings. We hypothesize that ExPT with a too powerful decoder may learn only to model the distribution over the target $x's$ and ignore the conditioning variables (context $x's$, context $y's$, and target y), which consequently hurts the generalization of the model.

C.4 Effects of $|\mathcal{D}_{\text{unlabeled}}|$

We empirically examine the effects of the size of $\mathcal{D}_{\text{unlabeled}}$ on the downstream performance of ExPT. Specifically, we subsample the $x's$ in the `public` dataset with a ratio $r \in \{0.01, 0.1, 0.2, 0.5, 1.0\}$. Adaptation and evaluation are the same as in Section 3.

Figure 7 shows the median and mean performance of ExPT on Dkitty and Ant in both `random` and `poorest` settings with respect to the ratio r. In the `random` setting, ExPT is able to reach or

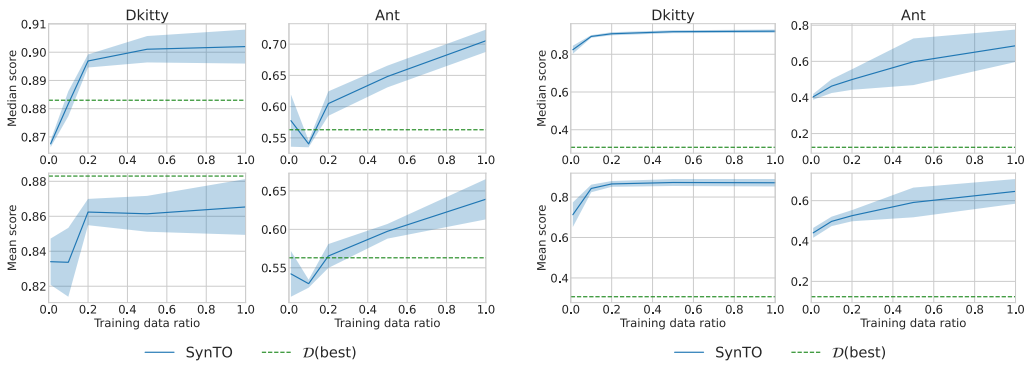

Figure 7: The performance of ExPT on Dkitty and Ant in the `random` (left) and `poorest` (right) setting when we vary the training data ration r. We average the performance across 3 seeds.

surpass the best data point in the few-shot dataset by using as few as 0.2 of the pretraining data. In the `poorest` setting, ExPT performs better than the best dataset point with only 0.01 of the pretraining data. Moreover, the performance improves consistently as the pretraining data size increases, suggesting that we can achieve even better performance by simply using more unlabeled data for pretraining. This result highlights the unique capability of ExPT of learning from unlabeled data, providing new opportunities for solving challenging optimization problems where unlabeled data is plentiful but labeled data is scarce.

C.5 Sorting context and target points

In the main experiments in Section 3, for each generated function during pretrainnig, we sample 228 points that we divide randomly into 100 context points and 128 target points. However, at adaptation, we condition on target output values that are likely to be higher than the best input value in the context set. Therefore, it is natural to sort the context points and target points during pretraining, so that the target inputs always have higher values than the context inputs. We denote this pretraining mechanism as ExPT-sorted.

Table 11: Comparison of pretraining on randomly divided context and target points (ExPT) versus sorted context and target points (ExPT-sorted) on Ant and D'Kitty in `random` (left) and `poorest` (right) settings. We average the performance across 3 seeds.

	Baseline	D'Kitty	Ant
	$\mathcal{D}_{\text{few-shot}}$(best)	0.883	0.563
Median	ExPT	$\mathbf{0.902 \pm 0.006}$	$\mathbf{0.705 \pm 0.018}$
	ExPT-Sorted	0.811 ± 0.019	0.631 ± 0.015
Mean	ExPT	$\mathbf{0.865 \pm 0.016}$	$\mathbf{0.639 \pm 0.026}$
	ExPT-Sorted	0.794 ± 0.020	0.590 ± 0.014

	Baseline	D'Kitty	Ant
	$\mathcal{D}_{\text{few-shot}}$(best)	0.307	0.124
Median	ExPT	$\mathbf{0.922 \pm 0.009}$	$\mathbf{0.686 \pm 0.090}$
	ExPT-Sorted	0.911 ± 0.003	0.685 ± 0.044
Mean	ExPT	0.871 ± 0.018	0.646 ± 0.061
	ExPT-Sorted	$\mathbf{0.900 \pm 0.003}$	$\mathbf{0.642 \pm 0.035}$

Table 11 shows that ExPT-sorted underperforms ExPT in the `random` setting, while performing very similarly in the `poorest` setting. This indicates that learning to predict any points provides a better and more general pretraining objective than only learning to predict points with high values.

C.6 Comparisons with more baselines

In addition to the baselines in Section 3, we compare ExPT with 3 variants of Gradient Ascent, a method that was considered in previous works [58, 36, 57, 34]. The Grad. Asc baseline simply learns a forward model and finds an optimal x^* by taking 200 gradient-ascent steps to improve an existing input x. The two variants Grad. Min and Grad. Mean create ensembles of forward models and perform gradient ascent using the min and mean ensemble predictions.

Tables 12 and 13 show the performance of ExPT and all baselines in the `random` and `poorest` settings. We see that while the gradient ascent methods perform well on certain tasks, with good

Table 12: Comparison of ExPT and the baselines on the few-shot `random` setting of 4 Design-Bench tasks. We report median, max, and mean performance across 3 random seeds. Higher scores are better. Blue denotes the best entry in the column, and Violet denotes the second best.

	Baseline	D'Kitty	Ant	TF Bind 8	TF Bind 10	Mean score (↑)
	$\mathcal{D}_{\text{few-shot}}$(best)	0.883	0.563	0.439	0.466	—
Median	MINs	**0.859 ± 0.014**	0.485 ± 0.152	0.416 ± 0.019	0.468 ± 0.014	0.557 ± 0.050
	COMs	0.752 ± 0.007	0.411 ± 0.012	0.371 ± 0.001	0.468 ± 0.000	0.501 ± 0.005
	BONET	0.852 ± 0.013	**0.597 ± 0.119**	0.441 ± 0.003	0.483 ± 0.009	**0.593 ± 0.036**
	BDI	0.592 ± 0.020	0.396 ± 0.018	**0.540 ± 0.032**	0.438 ± 0.034	0.492 ± 0.026
	GP-qEI	0.842 ± 0.058	0.550 ± 0.007	0.439 ± 0.000	0.467 ± 0.000	0.575 ± 0.016
	Grad. Asc	0.403 ± 0.134	0.088 ± 0.017	0.492 ± 0.017	**0.492 ± 0.018**	0.369 ± 0.0465
	Grad. Min	0.712 ± 0.028	0.220 ± 0.035	0.504 ± 0.025	0.465 ± 0.008	0.475 ± 0.024
	Grad. Mean	0.437 ± 0.180	0.150 ± 0.037	**0.551 ± 0.029**	**0.485 ± 0.018**	0.406 ± 0.066
	ExPT	**0.902 ± 0.006**	**0.705 ± 0.018**	0.473 ± 0.014	0.477 ± 0.014	**0.639 ± 0.013**
Max	MINs	**0.930 ± 0.010**	**0.890 ± 0.017**	0.814 ± 0.030	0.639 ± 0.017	0.818 ± 0.019
	COMs	0.920 ± 0.010	0.841 ± 0.044	0.686 ± 0.152	0.656 ± 0.020	0.776 ± 0.057
	BONET	0.909 ± 0.012	0.888 ± 0.024	0.887 ± 0.053	**0.702 ± 0.006**	**0.847 ± 0.024**
	BDI	0.918 ± 0.006	0.806 ± 0.094	0.906 ± 0.074	0.532 ± 0.023	0.791 ± 0.049
	GP-qEI	0.896 ± 0.000	0.887 ± 0.000	0.513 ± 0.104	0.647 ± 0.011	0.736 ± 0.029
	Grad. Asc	0.775 ± 0.032	0.240 ± 0.032	0.923 ± 0.005	0.675 ± 0.017	0.653 ± 0.0215
	Grad. Min	0.822 ± 0.053	0.434 ± 0.092	**0.960 ± 0.002**	0.632 ± 0.009	0.712 ± 0.039
	Grad. Mean	0.829 ± 0.009	0.337 ± 0.063	**0.957 ± 0.010**	0.668 ± 0.034	0.698 ± 0.029
	ExPT	**0.973 ± 0.005**	**0.970 ± 0.004**	0.933 ± 0.036	**0.677 ± 0.048**	**0.888 ± 0.023**
Mean	MINs	0.624 ± 0.025	0.009 ± 0.013	0.415 ± 0.030	0.465 ± 0.015	0.378 ± 0.021
	COMs	0.515 ± 0.050	0.020 ± 0.006	0.369 ± 0.003	0.471 ± 0.004	0.344 ± 0.016
	BONET	**0.837 ± 0.023**	**0.579 ± 0.024**	0.448 ± 0.011	**0.484 ± 0.009**	**0.587 ± 0.017**
	BDI	0.570 ± 0.032	0.385 ± 0.012	**0.536 ± 0.032**	0.444 ± 0.027	0.484 ± 0.026
	GP-qEI	0.505 ± 0.006	0.019 ± 0.001	0.439 ± 0.001	0.473 ± 0.002	0.359 ± 0.003
	Grad. Asc	0.400 ± 0.073	0.090 ± 0.018	0.513 ± 0.014	**0.492 ± 0.017**	0.374 ± 0.031
	Grad. Min	0.599 ± 0.068	0.221 ± 0.034	0.531 ± 0.015	0.462 ± 0.009	0.453 ± 0.032
	Grad. Mean	0.527 ± 0.079	0.150 ± 0.038	**0.569 ± 0.028**	0.438 ± 0.017	0.421 ± 0.041
	ExPT	**0.865 ± 0.016**	**0.639 ± 0.026**	0.476 ± 0.010	0.474 ± 0.015	**0.614 ± 0.017**

Table 13: Comparison of ExPT and the baselines on the few-shot `poorest` setting of 4 Design-Bench tasks. We report the median, max, and mean performance across 3 random seeds. Higher scores are better. Blue denotes the best entry in the column, and Violet denotes the second best.

	Baseline	D'Kitty	Ant	TF Bind 8	TF Bind 10	Mean score (↑)
	$\mathcal{D}_{\text{few-shot}}$(best)	0.307	0.124	0.124	0.000	—
Median	MINs	0.480 ± 0.156	0.316 ± 0.040	0.437 ± 0.007	0.463 ± 0.003	0.424 ± 0.052
	COMs	0.733 ± 0.023	0.401 ± 0.026	0.111 ± 0.000	0.459 ± 0.006	0.426 ± 0.014
	BONET	0.310 ± 0.000	0.236 ± 0.047	0.319 ± 0.018	0.461 ± 0.017*	0.332 ± 0.021
	BDI	0.309 ± 0.000	0.192 ± 0.012	0.365 ± 0.000	0.454 ± 0.017	0.330 ± 0.007
	GP-qEI	**0.883 ± 0.000**	**0.565 ± 0.001**	**0.439 ± 0.000**	**0.467 ± 0.000**	**0.589 ± 0.000**
	Grad. Asc	0.741 ± 0.026	0.321 ± 0.012	0.425 ± 0.064	0.419 ± 0.073	0.477 ± 0.044
	Grad. Min	0.806 ± 0.004	0.454 ± 0.061	0.357 ± 0.040	0.376 ± 0.079	0.498 ± 0.046
	Grad. Mean	0.742 ± 0.054	0.472 ± 0.066	0.350 ± 0.014	0.395 ± 0.019	0.489 ± 0.038
	ExPT	**0.922 ± 0.009**	**0.686 ± 0.090**	**0.552 ± 0.042**	**0.489 ± 0.013**	**0.662 ± 0.039**
Max	MINs	0.841 ± 0.014	0.721 ± 0.031	**0.962 ± 0.019**	**0.648 ± 0.025**	**0.793 ± 0.022**
	COMs	0.931 ± 0.022	0.843 ± 0.020	0.124 ± 0.000	**0.739 ± 0.057**	0.659 ± 0.025
	BONET	0.929 ± 0.031	0.557 ± 0.118	0.809 ± 0.038	0.519 ± 0.039*	0.704 ± 0.057
	BDI	**0.939 ± 0.002**	0.693 ± 0.109	**0.913 ± 0.000**	0.596 ± 0.020	0.785 ± 0.033
	GP-qEI	0.896 ± 0.000	**0.887 ± 0.000**	0.439 ± 0.000	0.645 ± 0.021	0.717 ± 0.005
	Grad. Asc	0.837 ± 0.038	0.684 ± 0.071	0.821 ± 0.077	0.568 ± 0.019	0.728 ± 0.052
	Grad. Min	0.910 ± 0.009	0.801 ± 0.029	0.842 ± 0.066	0.555 ± 0.028	0.777 ± 0.033
	Grad. Mean	0.882 ± 0.028	0.807 ± 0.046	0.747 ± 0.055	0.542 ± 0.039	0.745 ± 0.042
	ExPT	**0.946 ± 0.018**	**0.965 ± 0.004**	0.873 ± 0.035	0.615 ± 0.022	**0.850 ± 0.020**
Mean	MINs	0.623 ± 0.051	0.015 ± 0.017	**0.464 ± 0.009**	0.463 ± 0.002	0.391 ± 0.020
	COMs	0.607 ± 0.021	0.033 ± 0.003	0.109 ± 0.001	0.454 ± 0.004	0.301 ± 0.007
	BONET	0.490 ± 0.023	0.234 ± 0.052	0.318 ± 0.018	0.459 ± 0.018	0.375 ± 0.028
	BDI	0.364 ± 0.004	0.215 ± 0.021	0.369 ± 0.000	0.453 ± 0.018	0.350 ± 0.011
	GP-qEI	0.533 ± 0.001	0.018 ± 0.000	0.439 ± 0.000	**0.470 ± 0.002**	0.365 ± 0.001
	Grad. Asc	0.659 ± 0.069	0.334 ± 0.018	0.432 ± 0.061	0.427 ± 0.042	0.463 ± 0.048
	Grad. Min	**0.794 ± 0.003**	0.454 ± 0.051	0.374 ± 0.018	0.386 ± 0.044	**0.502 ± 0.029**
	Grad. Mean	0.702 ± 0.083	**0.467 ± 0.050**	0.356 ± 0.023	0.405 ± 0.018	0.483 ± 0.044
	ExPT	**0.871 ± 0.018**	**0.646 ± 0.061**	**0.549 ± 0.032**	**0.488 ± 0.011**	**0.639 ± 0.031**

performance on the TF-Bind8 task in particular, ExPT is still the best performing method in all settings and metrics.

D Compute

All training is done on 10 AMD EPYC 7313 CPU cores and one NVIDIA RTX A5000 GPU.

E Reproducibility

We made a strong effort to ensure that our work can be reproduced properly. In Section 2, we provide a comprehensive description of our methodology, while in Section 3 and Appendix A, we provide the specifics of our pretraining and evaluation setup, as well as our choice of hyperparameters. We compare our approach with various baseline methods from different approaches on multiple tasks in Design-Bench [58] with distinct properties. Our results are averaged over 3 seeds and we also report the standard deviation. Additionally, we conduct several ablation experiments to examine how sensitive ExPT is to different hyperparameters.

F Broader impact

The field of offline black-box optimization can have positive impacts in many spheres, including in drug-discovery, nuclear reactor design, and optimal robot design. The few-shot setting that we introduce in this work is also highly relevant to these fields which have large quantities of unlabelled data, but only a limited quantity of labelled data points. It is also worth noting however, that it is possible to use black-box optimization in general for malicious purposes such as to produce chemicals with harmful properties. Even though our work does not directly enable such use cases, this possibility should be taken into account when applying ExPT and similar frameworks to these kinds of impactful real-world scenarios.