Multi-Objective Optimisation for Smart Energy Grids Using ResNet

Manda Kausthubh

May 2025

1 Introduction

In this project, we apply Bayesian Optimisation (BO) for multi-objective optimisation in a smart energy grid classification task. The primary goal is to jointly optimise two conflicting objectives:

- **Accuracy**: Classification accuracy of a neural network, ratio of correct classification to the total number of data points.
- **Sparsity**: A proxy for model compactness and energy efficiency, Number of parameters of the model that are zero. This metric is very important for studying explainablity of ML models. Here our choice of ML model is a ResNet with variable learning rates and varying depths.
- **Training Time**: Looking at resources consumption and employability of models.

We utilise a ResNet-based black-box model, and guide our search using BO with weighted-sum scalarization of objectives.

2 Methodology

The optimisation loop follows these steps (optimizing over hyper-parameters such as width and depth of resnets, and drop-out):

1. Sample initial points using Sobol sequences.
2. Evaluate a black-box function returning accuracy, sparsity, time, and a weighted objective.
3. Fit a `SaasFullyBayesianSingleTaskGP` model using `botorch`.
4. Optimize the acquisition function `qUpperConfidenceBound`.
5. Repeat to refine the surrogate model and locate optimal trade-offs.

The black-box function is defined as:

```
def black_box_function(x, weights):
    try:
        x_np = x.detach().cpu().numpy().reshape(-1)
        y_pred, _, model = NN_prediction(x_np, test_features, train_loader, val_loader)
        sparsity = compute_model_density(model)
        acc = sklearn.metrics.accuracy_score(y_test, y_pred)
        objectives = torch.tensor([1.0 - acc, sparsity])
        return torch.Tensor([acc, sparsity, objectives @ weights])
    except:
        print("Unexpected eval error", sys.exc_info()[0])
        return torch.tensor([1.0])
```

3 SAASBO: Sparse Axis-Aligned Subspace Bayesian Optimization

SAASBO is a Bayesian Optimization technique using a fully Bayesian GP model with a hierarchical prior that promotes sparsity in high-dimensional problems. The core idea is to assume that only a few dimensions are relevant and exploit this sparsity.

Given observations $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, the GP prior is defined over a covariance kernel:

$$k(\mathbf{x}, \mathbf{x}') = \tau^2 \exp\left(-\sum_{d=1}^{D} \lambda_d (x_d - x'_d)^2\right) \tag{1}$$

Here, λ_d is a precision parameter for the d-th input. SAASBO places a sparsity-inducing prior over $\{\lambda_d\}$:

$$\lambda_d \sim \text{Gamma}(a, b), \quad a \ll 1 \tag{2}$$

This encourages most λ_d to be close to zero, effectively removing the corresponding input dimension from the kernel. The posterior over model hyperparameters is inferred using No-U-Turn Sampling (NUTS), enabling uncertainty-aware optimization.

SAASBO is ideal for our setting where the dimensionality of $\mathbf{x}$ is high but only a few components meaningfully influence the objectives.

4 Results

The optimization yields a set of Pareto-optimal configurations that trade off the objectives. Below, we show several 2D projections of the Pareto front and supporting metrics:

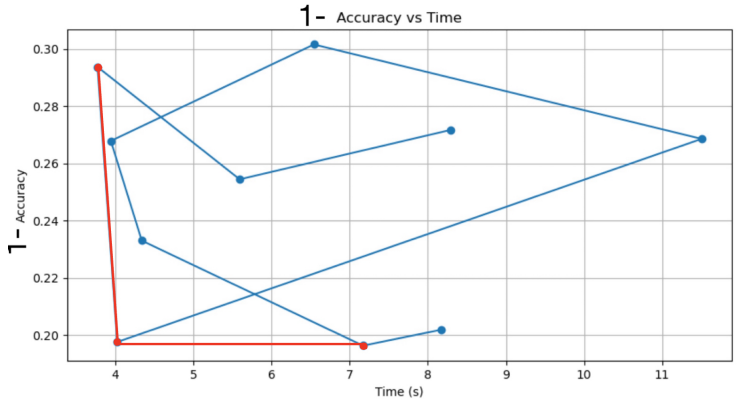

Figure 1: Accuracy vs Sparsity Pareto Front

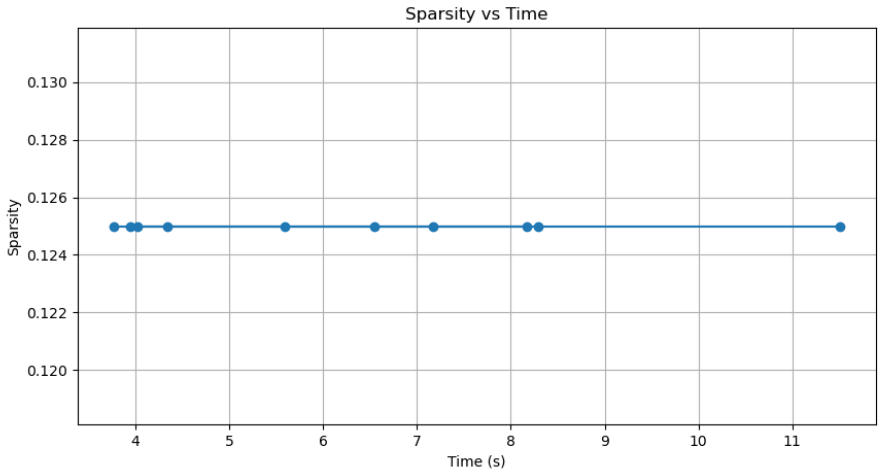

Figure 2: 1 - Accuracy vs Inference Time (Pareto Front in red)

5 Experimental Setup

The code loops through different weight vectors to scalarize the three objectives. At each step, we optimize the acquisition function using multiple restarts and raw samples. Error handling was incorporated to mitigate

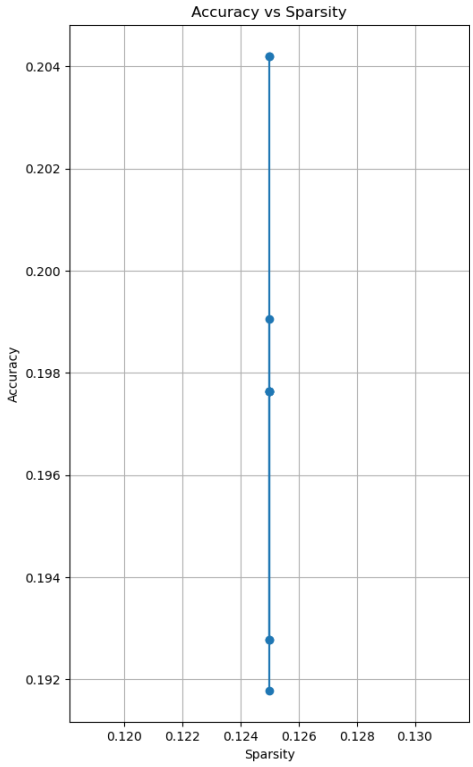

Figure 3: Sparsity vs Inference Time

'NaN' gradients and numerical instabilities.

To maintain continuity despite optimizer failures (e.g., 'OptimizationGradientError' due to NaNs), we used `try-except` blocks that catch errors during acquisition optimization and skip to the next iteration.

6 Observations

From the results, we observe:

- Sparsity remains fairly constant during optimization, suggesting its insensitivity to depth and width within the tested range.
- The Pareto front shows that significant gains in time can be made with minor losses in accuracy.
- SAASBO efficiently identifies a subspace of depth, width and drop-out that yield high-performance models.

7 Conclusion

Multi-objective optimization using SAASBO enables principled trade-offs in neural network deployment settings. The use of a sparsity-inducing prior allows efficient exploration of high-dimensional spaces by focusing on relevant dimensions.