Predicting from Strings:

Language Model Embeddings for Bayesian Optimization

Yilong Chen

Department of Statistics, University of Chicago

Nov, 2024

1. Preliminaries
2. Regressor
3. Pretraining and Inference
4. Attention Mechanism
5. Experiment

Blackbox Optimization Problem

Goal: Maximize a real-valued function f over a search space $\mathcal{X}$:

$$x^* = \arg\max_{x \in \mathcal{X}} f(x) \tag{1}$$

Challenges:

- f is a blackbox function, i.e., no analytical expression or gradients.
- Evaluation of f is expensive or limited.
- Requires balance between **exploration** and **exploitation**.

Definition of a Regressor

Regressor: A prediction model that outputs the distribution of $f(\cdot)$ values over a query point x, given the history of evaluations:

$$\{(x_s, y_s)\}_{s=1}^{t}$$

where $y_s = f(x_s)$ for $s = 1, 2, \ldots, t$.

Key Properties:

- *Learnability:* The regressor can be trained using offline or online data.
- *Distributional Output:* Provides mean and uncertainty estimates for predictions.

Acquisition Function for Exploration and Exploitation

Definition: The regressor is transformed into an acquisition function:

$$a_{t+1}(x) : \mathcal{X} \to \mathbb{R}$$

which guides the optimization process.

Next Proposal:

$$x_{t+1} = \arg\max_{x \in \mathcal{X}} a_{t+1}(x) \tag{2}$$

Acquisition Optimizer:

- Samples $x \in \mathcal{X}$ efficiently, using zeroth-order or evolutionary algorithms.
- Balances exploration (high uncertainty) and exploitation (high mean).

Bayesian Optimization Loop

Procedure:

1. Initialize a set of observations $\{(x_s, y_s)\}_{s=1}^{t}$.
2. Train a regressor on the given data.
3. Compute the acquisition function $a_{t+1}(x)$ based on the regressor's predictions.
4. Find the next proposal:

$$x_{t+1} = \arg\max_{x \in \mathcal{X}} a_{t+1}(x)$$

5. Evaluate $f(x_{t+1})$ and update the history.
6. Repeat until a stopping criterion is met.

Outcome: Approximate the global maximum of $f(x)$.

1. Preliminaries
2. **Regressor**
3. Pretraining and Inference
4. Attention Mechanism
5. Experiment

Embedding-Based Regressor: Concept

Purpose: Convert input $x \in \mathcal{X}$ into a fixed-length representation for regression.

Embedding Process:

- The embedder $\phi : \mathcal{X} \to \mathbb{R}^d$ maps x to a vector $\bar{x} \in \mathbb{R}^d$.
- Input x is represented as a **string**, which is processed by a language model (e.g., T5 encoder).
- A forward pass through the model produces token logits in $\mathbb{R}^{L \times d}$.

Insert Graph: Diagram of embedding process:

- Input string $\rightarrow$ Tokenization $\rightarrow$ Language Model $\rightarrow$ Average Pooling $\rightarrow$ $\bar{x}$.

Embedding-Based Regressor: Final Representation

Final Step:

- Perform **average pooling** across the token axis to generate a fixed-length embedding in $\mathbb{R}^d$.
- Output: $\bar{x} \in \mathbb{R}^d$.

Benefits:

- Handles arbitrary input types (e.g., JSON, categorical).
- Generates consistent, fixed-length feature vectors for regression tasks.

In-context Regression Transformer: Input Sequence

Input Sequence:

$$(\bar{x}_1 \oplus \bar{y}_1), \ldots, (\bar{x}_t \oplus \bar{y}_t) \tag{3}$$

where:

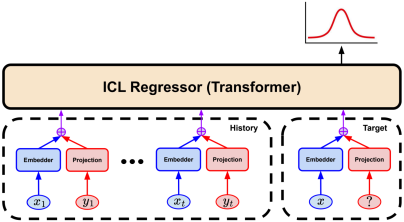
- $\bar{x}_i \in \mathbb{R}^d$: Embedding of trial x_i.
- $\bar{y}_i \in \mathbb{R}^d$: Feature representation of observed value y_i, obtained via a trainable projection.
- $\oplus$: Concatenation operator.

In-context Regression Transformer: Prediction

Prediction Process:

- Append query $(\bar{x} \oplus \bar{0})$, where $\bar{0}$ is a placeholder.
- Perform a forward pass through the Transformer.
- The output feature corresponding to $t + 1$ predicts:

$$\mathcal{N}(\mu_{t+1}(x), \sigma^2_{t+1}(x)) \tag{4}$$



Additional Techniques

Parallel Predictions:

- Simultaneously predict over a set of k target points:

$$(\bar{x}_{t+1} \oplus \bar{0}), \ldots, (\bar{x}_{t+k} \oplus \bar{0}) \tag{5}$$

- Custom attention pattern allows tokens to attend to history but not to targets.

Additional Techniques

y-Normalization:

- Shift objectives to have zero mean and scale by standard deviation.
- Transform y values into $[0, 1]$ for numerical stability.

Encoding Metadata:

- Include task-specific metadata m as part of input:

$$\bar{x} \to (\bar{x} \oplus m) \tag{6}$$

- Useful for providing additional context about the objective or search space.

Pretraining and Inference: Overview

Task Definition:

- A task $\mathcal{T} = (f, \mathcal{X})$ represents a specific objective function f over a search space $\mathcal{X}$.

Pretraining:

- Assumes a collection of offline **training tasks** $\{\mathcal{T}_1, \mathcal{T}_2, \dots\}$.
- Each task has its own evaluated trials $\{(x_s, y_s)\}_{s=1}^{T}$.
- T: Offline trajectory length (task-specific).

Purpose:

- Learn from historical offline data to generalize across tasks.

Pretraining: Loss Function

Frozen Embedder:

- During pretraining, the weights θ of the ICL regression Transformer are optimized.
- The embedder remains **frozen**.

Training Example:

- History: $\{(x_s, y_s)\}_{s=1}^{t'}$ for some $t' \in [0, T]$.
- Targets: $\{(x_{t'+i}, y_{t'+i})\}_{i=1}^{T-t'}$.

Loss Function:

$$\sum_{i=1}^{T-t'} \ell_\theta(x_{t'+i}, y_{t'+i}; \{(x_s, y_s)\}_{s=1}^{t'}), \tag{7}$$

where:

- ℓ_θ is the negative log-likelihood of the Gaussian distribution.

Inference: Overview

At Inference:

- The Transformer predicts the mean $\mu_{t+1}(x)$ and standard deviation $\sigma_{t+1}(x)$ for the target.
- The acquisition function is defined as:

$$a_{t+1}(x) = \mu_{t+1}(x) + \sqrt{\beta} \cdot \sigma_{t+1}(x), \tag{8}$$

where $\sqrt{\beta}$ is a problem-dependent constant.

Optimization:

- Uses a zeroth-order optimizer (e.g., evolutionary search) to maximize $a_{t+1}(x)$.
- Requires only **forward passes** (no gradient-based optimization).

Handling Distributional Shifts

Challenges:

- Distributional shifts may occur between pretraining and inference.

Solutions:

- **Data Augmentation:** Randomize parameter names during pretraining.
- **Search Space Transformation:** Adjust search space at inference to align with pretraining conditions.

1. Preliminaries
2. Regressor
3. Pretraining and Inference
4. **Attention Mechanism**
5. Experiment

Attention in Transformer for $\bar{x} \oplus \bar{0}$

Goal: Predict a distribution $\mathcal{N}(\mu_{t+1}(\bar{x}), \sigma^2_{t+1}(\bar{x}))$ for a query point $\bar{x} \oplus \bar{0}$.

Input Sequence:

$$\mathcal{S} = \{(\bar{x}_1 \oplus \bar{y}_1), (\bar{x}_2 \oplus \bar{y}_2), \ldots, (\bar{x}_t \oplus \bar{y}_t), (\bar{x} \oplus \bar{0})\}$$

Attention Formula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V$$

Key Idea: Query $\bar{x} \oplus \bar{0}$ computes attention weights over historical points $\{(\bar{x}_i \oplus \bar{y}_i)\}_{i=1}^t$.

How Attention Works for $\bar{x} \oplus \bar{0}$

Steps:

- Compute attention weights:

$$\alpha_{t+1,i} = \mathrm{softmax}\left(\frac{K(\bar{x} \oplus \bar{0}) \cdot Q(\bar{x}_i \oplus \bar{y}_i)}{\sqrt{d_k}}\right)$$

- Aggregate historical information:

$$z_{t+1} = \sum_{i=1}^{t} \alpha_{t+1,i} V_i(\bar{x}_i \oplus \bar{y}_i)$$

  where V_i is the value vector of $(\bar{x}_i \oplus \bar{y}_i)$.

- Use z_{t+1} to predict $\mu_{t+1}(\bar{x})$ and $\sigma^2_{t+1}(\bar{x})$.

Output:

$$\mathcal{N}(\mu_{t+1}(\bar{x}), \sigma^2_{t+1}(\bar{x}))$$

Synthetic Optimization: Overview

Definition:

- In common optimization scenarios, the search space is modeled as a flat Cartesian product of:
  - **Float parameters**: Continuous-valued parameters.
  - **Categorical parameters**: Discrete-valued parameters.
- The goal is to evaluate the performance of **Embed-then-Regress** on such tasks.

Key Representation:

- Each x (input) is represented as a *JSON string*.
- Example:

$$\{"p0" : 0.3, "p1" : 4\}$$

- Where:
  - $p0$: Continuous parameter.
  - $p1$: Integer parameter.

Benchmark: Blackbox Optimization Benchmarking (BBOB)

Setup:

- Uses the **BBOB suite** (ElHara et al., 2019), one of the most widely used synthetic function benchmarks.
- Contains 24 objectives over **continuous search spaces**.

Training-Test Split:

- Original functions are divided into **training** and **test sets**.
- Apply transformations (e.g., shifting, rotating, discretizing, etc.) to generate diverse functions.
- Induce non-continuous search spaces with categorical parameters.

Goal: Evaluate the generalization of the trained model on unseen tasks.

Comparison with GP-Bandit

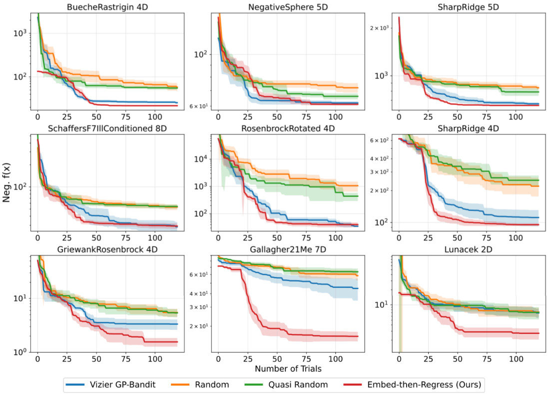
Baseline:

- The traditional baseline is **GP-Bandit** (Song et al., 2024c), a UCB-based Bayesian Optimization method.
- Uses the same acquisition optimizer (**Firefly**) as Embed-then-Regress.

Findings:

- Embed-then-Regress is generally **comparable** to GP-Bandit in performance.
- In some cases, it **significantly outperforms** GP-Bandit, especially on:
  - Non-continuous search spaces.
  - Test functions requiring task generalization.

Performance



Thank You