A.1 Motivation: Moving Beyond the Limitations of a Single Surrogate

In complex sequential decision-making tasks such as Bayesian Optimization (BO), the quality of the surrogate model is the key determinant of search efficiency. Traditional pipelines typically rely on a single surrogate model whose performance is tightly constrained by the inductive bias of that model. Once this solitary model exhibits systematic deviations when characterizing the fitness landscape, the search procedure readily falls into suboptimal regions, causing the exploration-exploitation balance to break down.

Our proposed multi-view collaborative decision framework essentially functions as an adaptive controller for the search policy. The framework replaces the single surrogate with a committee composed of heterogeneous experts. Unlike traditional ensemble learning, whose goal is mainly to reduce variance through model averaging, each expert in our framework performs independent inference and coordinates decisions under a principled dynamic trust mechanism so as to maximize the potential value of the next exploration step.

A.1.1 Online Expert Combination

To formalize this collaborative mechanism, we cast multi-view collaborative search as a Prediction with Expert Advice problem in online learning.

During the t-th iteration of BO, the procedure is as follows:

Input state: The algorithm receives the search space $\mathcal{X}$, the historical evaluation data $\mathcal{D}_{t-1} = \{(x_i, y_i)\}_{i=1}^{n}$, and a set of N heterogeneous experts $\mathcal{E} = \{E_1, \ldots, E_N\}$.

Expert suggestions: Each expert E_i uses its internal surrogate to compute an acquisition score $\alpha_i(x)$ for any candidate architecture $x \in \mathcal{X}$.

Collaborative decision: Rather than a simple linear weighted fusion, we adopt a quota-based proposal mechanism. Theoretically this is equivalent to sampling experts from a dynamic weight distribution $w^{(t)}$ to guide the search; for stability we implement a deterministic quota rule:

1. **Per-view scoring**: Compute the acquisition score $\alpha_i(x)$ of every candidate architecture x under each view.

2. **Quota assignment**: According to the per-view weights $w_i^{(t)}$, allocate the batch size k into expert-specific quotas $q_i = \lfloor k \cdot w_i^{(t)} \rfloor$.
3. **Within-view selection**: Each expert E_i nominates the Top-q_i candidates by ranking its own scores $\alpha_i(x)$.
4. **Cross-view completion**: Take the union of all proposals. If the union contains fewer than k candidates, fill the remaining slots according to the aggregated ranking that combines per-view acquisition scores.

Performance feedback and loss computation: After the ground-truth evaluations, we denote the resulting batch by $\mathcal{B}_t = \{(x_j, y_j)\}_{j=1}^{k}$. The mean absolute error of expert i on this batch is $\mathrm{MAE}_i(\mathcal{B}_t)$, and the Spearman correlation coefficient $\rho_i(\mathcal{B}_t)$ measures ranking consistency. To make the two metrics comparable across views, we apply the following dimensionless scaling:

$$\tilde{\mathrm{MAE}}_i^{(t)} = \frac{\mathrm{MAE}_i(\mathcal{B}_t)}{\max_j \mathrm{MAE}_j(\mathcal{B}_t) + \varepsilon}, \qquad s_i^{(t)} = \frac{\rho_i(\mathcal{B}_t) + 1}{2} \tag{1}$$

Here $\varepsilon > 0$ is a stabilizer and $s_i^{(t)} \in [0, 1]$ shifts and scales the correlation coefficient into a unified interval. We then combine the two terms with a weight $\gamma \in [0, 1]$ to obtain the single loss

$$L_i^{(t)} = \gamma \cdot \tilde{\mathrm{MAE}}_i^{(t)} + (1 - \gamma) \cdot \left(1 - s_i^{(t)}\right) \tag{2}$$

This definition explicitly balances numerical accuracy (the first term, which is better when smaller) and ranking consistency (the second term, which is better when larger).

A.1.2 Enhanced Weight Updates

To accommodate NAS tasks where evaluations are expensive and noisy, we design a log-space exponentiated weighting rule together with momentum smoothing.

A.1.2.1 Core Update Module and Theoretical Connections

After each decision round, we first compute the instantaneous weight. To improve numerical stability and handle loss values that span multiple orders of magnitude, we adopt an inverse power form, where the exponent p is set externally according to task requirements:

$$w_{i,\mathrm{inst}}^{(t+1)} = \frac{(\varepsilon + L_{i,\mathrm{enh}}^{(t)})^{-p}}{\sum_j (\varepsilon + L_{j,\mathrm{enh}}^{(t)})^{-p}} \tag{3}$$

Here $L_{i,\text{enh}}^{(t)}$ denotes the loss estimate after passing through the enhancement module (see A.1.2.2).

Theoretical interpretation: gradient-style updates in log space

To justify the effectiveness of the rule above, we expose its deep links to classical online-learning algorithms. Using the identity $x^{-p} = \exp(-p \ln x)$, Eq. (A.1.2-1) can be rewritten in a Softmax form:

$$w_{i,\text{inst}}^{(t+1)} \propto \exp\left(-p \cdot \underbrace{\ln\left(\varepsilon + L_{i,\text{enh}}^{(t)}\right)}_{\text{Potential } \Phi_i^{(t)}} \right) \tag{4}$$

Here $\Phi_i^{(t)}$ can be viewed as a potential defined in log space. Unlike standard EWA, which uses cumulative loss, our framework feeds a smoothed loss estimate $L_{i,\text{enh}}^{(t)}$ as input.

Examining the ratio between consecutive rounds gives the incremental law

$$\frac{w_{i,\text{inst}}^{(t+1)}}{w_{i,\text{inst}}^{(t)}} = \frac{(\varepsilon + L_{i,\text{enh}}^{(t)})^{-p}}{(\varepsilon + L_{i,\text{enh}}^{(t-1)})^{-p}} \cdot \frac{\sum_j (\varepsilon + L_{j,\text{enh}}^{(t-1)})^{-p}}{\sum_j (\varepsilon + L_{j,\text{enh}}^{(t)})^{-p}} \tag{5}$$

The first fraction depends only on expert i and reflects its own loss fluctuation, whereas the second fraction is a global normalization factor shared by all experts. Observed in the log domain, we obtain

$$w_{i,\text{inst}}^{(t+1)} = w_{i,\text{inst}}^{(t)} \cdot \exp\left(-p \cdot \delta \ell_i^{(t)}\right) \cdot \exp\left(-p \cdot \Delta\Phi^{(t)}\right) \tag{6}$$

where the single-step loss increment is

$$\delta \ell_i^{(t)} = \ln\left(\varepsilon + L_{i,\text{enh}}^{(t)}\right) - \ln\left(\varepsilon + L_{i,\text{enh}}^{(t-1)}\right) \tag{7}$$

and

$$\Delta\Phi^{(t)} = -\frac{1}{p}\left[\ln \sum_j (\varepsilon + L_{j,\text{enh}}^{(t)})^{-p} - \ln \sum_j (\varepsilon + L_{j,\text{enh}}^{(t-1)})^{-p}\right] \tag{8}$$

is the global potential difference that is independent of any specific expert. Consequently, each expert is updated by combining its relative loss change $\delta \ell_i^{(t)}$ with the normalization adjustment $\Delta\Phi^{(t)}$.

To prevent large oscillations caused by noisy single evaluations, we introduce a weight-momentum mechanism:

$$w^{(t+1)} = (1 - \beta^{(t)}) \cdot w_{\text{inst}}^{(t+1)} + \beta^{(t)} w^{(t)} \tag{9}$$

This convex combination smooths the trajectory while retaining responsiveness to consistent trends, ensuring that a single feedback signal does not cause dramatic weight swings.

A.1.2.2 Enhancement Module

To further boost robustness, we insert multiple enhancement stages between the raw loss $L_i^{(t)}$ and the final weights:

1. **Time-weighted error estimation**: Compute a finite-window weighted average of recent losses. A forgetting factor emphasizes recent history, enabling the algorithm to focus on the most recent performance.
2. **Saliency amplification**: Compare each expert's windowed composite loss with its local median. Sustained improvement triggers a fixed-factor shrinkage, whereas sustained degradation inflates the loss. This heuristic effectively rescales $L_{i,\text{enh}}^{(t)}$ before the inverse-power mapping to highlight significant differences; the power p itself remains governed by the preset hyperparameter.
3. **Dynamic smoothing**: The smoothing coefficient anneals with iteration progress. Early iterations almost fully trust the instantaneous weight ($\beta^{(t)} \approx 0$), while later iterations assign more mass to historical weights (larger $\beta^{(t)}$) to stabilize decisions during convergence.

A.1.3 Theoretical Analysis and Guarantees

This section employs regret analysis from online learning to show how the framework adapts to evolving search environments.

A.1.3.1 Theoretical Assumptions

To ensure that the regret bounds hold, we rely on the following boundedness assumptions, all of which are satisfied by design:

- **Bounded log loss**: Because $L \in [0, 1]$ and $\varepsilon > 0$, the surrogate loss $\ell \propto \ln(\varepsilon + L)$ stays bounded.
- **Controlled parameters**: The learning rate p and momentum coefficient β are restricted to predefined intervals.

A.1.3.2 Strongly Adaptive Regret

NAS operates in a nonstationary environment where the best expert can change between early exploration and late fine-tuning. We therefore measure performance using the Strongly Adaptive Regret (SA-Regret), which compares the algorithm against the best expert over any interval $I = [s, e]$.

Daniely et al. introduced the Strongly Adaptive Online Learner (SAOL), an expert-combination scheme built on multi-scale memory, local EWA, and smoothed switching. It maintains windows of varying lengths, applies exponentiated weighting within each window to favor recently strong experts, and uses restarts or momentum to smooth transitions. If the base EWA enjoys an $O(T^{\alpha})$ regret over T rounds, SAOL achieves $\tilde{O}(|I|^{\alpha})$ SA-Regret on any interval I.

Our multi-view scheduler approximates SAOL at the task level. Although we do not explicitly materialize all scales, we preserve its three key ingredients:

1. **Finite-window memory**: $L_{i,\mathrm{enh}}^{(t)}$ aggregates only the most recent K evaluations, so stale data decays quickly. Functionally this mirrors the sliding windows in SAOL and allows the weights to follow the locally optimal expert within a few iterations.
2. **Local Softmax updates**: The saliency amplification distinguishes particularly good or bad experts within each window, after which $(\varepsilon + L)^{-p}$ maps the composite loss to probabilities, akin to running EWA using the composite loss as the potential.
3. **Momentum-based smoothing**: The update $w^{(t+1)} = (1 - \beta^{(t)}) w_{\mathrm{inst}}^{(t+1)} + \beta^{(t)} w^{(t)}$ with a growing $\beta^{(t)}$ plays the role of SAOL's restart/smoothing step, preventing sharp oscillations during phase transitions.

Together these modules inherit SAOL's finite memory, local EWA, and smooth control, providing the theoretical motivation for robust dynamic scheduling in NAS.

A.1.3.3 Risk Diversification

The framework's effectiveness stems from integrating multiple information sources whose biases are uncorrelated. This yields a decision system that is substantially more robust than any single source.

Let $S_i \subset \mathcal{X}$ denote the high-error region (blind spot) of expert E_i, namely the subset where the expert exhibits significant systematic deviation.

- **Single-expert risk**: If the global optimum $x^* \in S_i$, the probability of search failure increases dramatically.
- **Multi-expert risk**: Systematic failure occurs only if $x^* \in \bigcap_{i=1}^{N} S_i$.

Because the framework fuses representations with different inductive biases, each expert is encouraged to inspect a different side of the architecture manifold. When expert biases are mutually independent or weakly correlated, the blind spots S_i become misaligned, and the expected volume of their intersection is vastly smaller than any single region. Under the idealized independence assumption, we have

$$\mathbb{E}\big[\mathrm{Vol}(\bigcap_{i=1}^{N} S_i)\big] \leq \prod_{i=1}^{N} \mathbb{E}\big[\mathrm{Vol}(S_i)\big]. \tag{10}$$

Intuitively, the probability space for a joint failure (every expert misjudges simultaneously) shrinks exponentially, allowing the blind spot of one view to be compensated by the others.

A.2 Additional Experimental Configurations

For reproducibility, this section supplements the search-space definition, dataset initialization, surrogate-model construction, and the template used to summarize the best architectures.

A.2.1 Search Space Definition

The main paper already describes inter-layer skip connections and hierarchical aggregation. Here we provide the full list of node-aggregation operators used in experiments. Each node aggregation module in the three-layer backbone selects one operator from a catalog of over thirty graph primitives that cover both spectral convolutions and attention/message-passing families: standard GCN; Chebyshev polynomial convolution with orders $k = 1$ through 7; SGC with explicit $k = 1/2/3$ filters; TAGConv variants corresponding to $k = 2/3/5$ hops; the steady-state ARMAConv; residually enhanced GCNII; PyG GraphConv (additive aggregation baseline); ResGated GraphConv; multi-head self-attention such as GAT and GATv2; TransformerConv tailored to Transformer-style message passing; GENIEPath with layer-wise adaptive path aggregation; AGNNConv with global vector attention; GraphSAGE with sum and max aggregators; GIN with multiple MLP stacks offset from -2 to $+2$; and a pure MLP variant. Together they offer frequency-domain filtering, adaptive neighborhood aggregation, mean- and max-style preferences, and single-node preservation, enabling the search to explore widely under the same depth budget.

For the inductive PPI setting we lightly prune the list by removing operators that depend heavily on transductive training, while other datasets use the full catalog. Inter-layer skip connections toggle between on/off states and work in tandem with Jumping Knowledge pooling to decide whether information bypasses layers. Hierarchical aggregation offers max pooling, feature concatenation, and LSTM-based aggregation (PubMed disables the LSTM option for stability). Aside from operator and structure choices, training hyperparameters such as activation, normalization, and dropout remain fixed so that the search budget is focused on node operators and cross-layer connections.

A.2.2 Dataset Initialization

All benchmarks use a homogeneous three-layer GNN backbone with Jumping Knowledge pooling and batch normalization. Only the hidden dimension, regularization strength, and optimization strategy vary slightly across datasets:

- **CiteSeer**: Hidden dimension 256, dropout 0.55, 400 training epochs. Optimizer: AdamW with learning rate 1.5e-3 and weight decay 7e-4. Scheduler: cosine annealing over 400 epochs with a minimum learning rate of 1e-3. Gradient norms are clipped at 5.
- **Cora**: Hidden dimension 64, dropout 0.60, 400 epochs. Optimizer: AdamW with learning rate 5e-3 and weight decay 5e-4, paired with the same cosine schedule and a gradient clip of 5.
- **PubMed**: Hidden dimension 64, dropout 0.50, 400 epochs. Optimizer: Adam with learning rate 5e-3 and weight decay 1e-3, combined with a 500-epoch cosine scheduler. ELU activation is used throughout.
- **CoraFull**: Hidden dimension 512, dropout 0.50, 500 epochs. Optimizer: AdamW with learning rate 1e-3 and weight decay 1.5e-3. Scheduler: cosine annealing over 500 epochs with a minimum learning rate of 5e-4.
- **PPI**: Hidden dimension 256, no dropout, 600 epochs. Optimizer: Adam with learning rate 5e-3 and no weight decay, combined with ReduceLROnPlateau in maximization mode (decay factor 0.5, patience 10). The training batch size is 2, and multi-process data loading is enabled to compute Micro-F1.

Smaller datasets such as CiteSeer and Cora employ a long-patience early-stopping strategy to mitigate single-batch noise.

A.2.3 Surrogate-Model Details

- **Statistical view (Gaussian Process)**: After encoding each genotype into a feature vector, a pretrained variational autoencoder compresses it into a low-dimensional latent vector. The observed performance, standardized to be dimensionless, is concatenated with the latent vector and fed into a Gaussian process regressor with a Matern kernel. We filter outliers, perform multiple random initializations, and retain the state with the highest marginal likelihood. During prediction, new genotypes undergo the same encoding and are fed into the GP to obtain the posterior mean and standard deviation.
- **Topological view (Graph Neural Network)**: Each genotype is parsed as a DAG whose nodes correspond to operators and whose edges denote information flow, augmented by operator category and layer-position features. The surrogate consists of three graph-convolution layers followed by a linear regression head optimized with a weighted sum of mean squared error and a ranking loss. We train it using Adam, supporting both training from scratch and short re-fitting rounds that warm-start from prior weights.
- **Discrete view (Random Forest + XGBoost)**: Discrete choices over aggregation operators, skip connections, and hierarchical aggregators are one-hot encoded and concatenated into a fixed-length vector. This view hosts two lightweight models: a 200-tree random forest with maximum depth 20 whose leaf means yield predictions and whose across-tree standard deviation measures uncertainty, and an ensemble of five XGBoost models (each with 100 trees of depth 6, learning rate 0.1, subsampling 0.8). Their average serves as the prediction, and the inter-model standard deviation quantifies uncertainty. We keep both because the ensemble is more stable yet still lightweight enough to retrain after each search iteration.

A.2.4 Representative Best Architectures

We record the best genotypes discovered on each dataset using textual descriptors:

Citeseer:

[chebconv_5||graphconv||gin_trainable||skip||none||lstm]

Cora:

[tagconv_5||tagconv_3||gat_generalized_linear||skip||none||lstm]

PubMed:

[chebconv_4||geniepath||gat||skip||skip||concat]

CoraFull:

[tagconv_2||tagconv||mlp||skip||none||max]

PPI:

[resgated||chebconv_7||gin_2||skip||skip||lstm]

References

Daniely, O., Gonen, A., Shalev-Shwartz, S. (2015). Strongly Adaptive Online Learning. In *International Conference on Machine Learning (ICML)*.